Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Estatística

Dissertação de Mestrado

Modelagem de risco de crédito via LSTM

por

Gustavo Durães Almeida

Brasília, Setembro de 2021

por

Gustavo Durães Almeida

Dissertação apresentada ao Departamento de Estatística da Universidade de Brasília, como requisito parcial para obtenção do título de Mestre em Estatística.

Orientador: Prof. Dr. Eduardo Yoshio Nakano

Brasília, Setembro de 2021

Dissertação submetida ao Programa de Pós-Graduação em Estatística do Departamento de Estatística da Universidade de Brasília como parte dos requisitos para a obtenção do grau de Mestre em Estatística.

Texto aprovado por:

Prof. Eduardo Yoshio Nakano

Orientador, EST/UnB

Prof. André Luiz Fernandes Cançado

EST/UnB

Prof. Marcelo Angelo Cirillo

UFLA

Dedico trabalho a minha namorada e a minha mãe, que com muito carinho sempre me apoiaram.

Meus sinceros agradecimentos aos professores do PPGEST/UnB, em especial ao meu orientador, Eduardo Yoshio Nakano, por ter me conduzido com extrema tranquilidade, competência e interesse. Também agradeço ao Sistema de Cooperativas de Crédito do Brasil (Sicoob) por fornecer os dados que possibilitaram a execução desse trabalho.

Resumo

A modelagem de risco de crédito utiliza frequentemente de variáveis explicativas comportamentais medidas de forma longitudinal afim de estimar a probabilidade de não pagamento de uma operação de crédito. Usualmente, as informações longitudinais são sintetizadas atraváes de estatísticas resumos e em seguida são utilizadas como preditoras em modelos lineares de classificação binária. Este trabalho avalia a utilização e performance de uma rede neural recorrente (Long short-term memory), que é capaz de processar integralmente as variv́eis explicativas longitudinais, como uma alternativa à regressão logística na modelagem de risco de crédito utilizando dados reais da instituição financeira Sicoob.

Abstract

Credit risk modelling usually utilizes longitudinal behavioural data as explanatory variables in default classification tasks. Usually, the aforementioned longitudinal data is summarized using domain appropriate summary statistics in order to transform the longitudinal data in linear data so that the information can be used as input for a logistic regression. This work assess the usability and performance of a recurrent neural network, that is capable of consuming the longitudinal data directly, as and alternative for a logistic regression model in the credit risk context using real data from a financial institution named Sicoob.

Sumário

Abreviações e Siglas

AUC	área sobre a curva característica de operação do receptor
EAD	Exposição caso haja default (*Exposure at default*)
FN	Falso negativo
FP	Falso positivo
KS	Kolmogorov-Smirnov
PE	Perda esperada
PD	Probabilidade de default
LGD	Perda esperada dado que houve default (*Loss given default*)
MCC	Coeficiente de correlação de Matthews.
LSTM	Long Short-term Memory
MLG	Modelos Lineares Generalizados
R.L.	Regressão logística
RNN	Rede neural recorrente (*Recurrent Neural Network*)
SHAP	SHapley Additive exPlanations
VIF	Fator de inflação da variância (*Variance inflation factor at default*)
VN	Verdadeiro negativo
VP	Verdadeiro positivo

Capítulo 1

Introdução

1.1 Considerações iniciais

No contexto do mercado financeiro, é usual a utilização de Modelos Lineares Generalizados para estimação do risco em operações de crédito. A proeminência dessa técnica não se dá por acaso. Os MLGs fornecem uma estimativa de risco estatisticamente sólida e seus parâmetros são facilmente interpretados e monitorados. A possibilidade de monitorar e explicar a relação entre as covariáveis e a variável resposta é normativamente necessária por requisição dos acordos da Basiléia. Outro fator muito importante é o fato da regressão binomial ter como resultado direto uma medida de probabilidade, o que nos permite criar uma função de custo adequada para auxiliar na tomada de decisão da concessão de crédito.

De forma geral, a maioria das variáveis explicativas utilizadas na modelagem de risco de crédito são obtidas através da síntese de dados longitudinais. Usualmente, é utilizada uma medida resumo (por exemplo a mediana de gastos no cartão de crédito nos últimos 12 meses, o máximo do endividamento nos últimos 12 meses) para permitir o uso dessas informações como variáveis explicativas na regressão logística.

Este trabalho propõe uma forma de se utilizar o histórico completo de informações relevantes através de uma aplicação de Redes Neurais Recorrentes denominada Long short-term

memory (LSTM) (Hochreiter e Schmidhuber, 1997).

Uma das dificuldades ao considerar variáveis explicativas longitudinais é a dificuldade em medir a contribuição dessas variáveis na modelagem. Todavia, a abordagem através dos valores SHAP (SHapley Additive exPlanations) (Lundberg e Lee, 2017), proporciona uma forma de mensurar a contribuição marginal de cada variável explicativa no valor da predição da variável resposta propiciando uma forma de mensurar a importância das covariáveis dos modelos citados acima.

Neste contexto, o objetivos deste trabalho é conduzir um estudo comparativo entre a qualidade do ajuste, usabilidade e interpretabilidade de modelos lineares generalizados com resposta binária e um modelo de rede neural artificial que utilizada uma camada Long short-term memory e, além disso, mensurar a relação entre as variáveis explicativas e a variável resposta dentro de cada modelo através dos valores SHAP. Busca-se, com os valores SHAP, uma interpretação mais direta sobre influência das variáveis preditoras no valor de saída dos modelos.

Além disso, também serão comentadas e discutidas as diferenças no processo de modelagem envolvido na utilização das diferentes técnicas.

É importante ressaltar que ajustar o melhor modelo possível não está no escopo deste trabalho. A natureza da pesquisa consiste em mensurar as diferenças de ajuste, ganho de interpretabilidade através dos valores SHAP e usabilidade dos diferentes modelos propostos, considerando a mesma disponibilidade de informação.

Esse estudo foi conduzido utilizando um conjunto de dados de 30 mil clientes da instituição financeira Sicoob e todas as análises foram feitas utilizando o software Python.

Capítulo 2

Revisão Conceitual

2.1 Conceitos Básicos em risco de crédito

O objetivo dos modelos de risco de crédito é quantificar o risco a que a instituição financeira está sujeita quando inicia um operação de crédito com um cliente. Essa quantificação pode variar de acordo com cada instituição financeira. Neste trabalho consideraremos que a perda esperada em caso de não pagamento se dá pela fórmula:

$$PE = PD \cdot LDG \cdot EAD,$$

onde

- PE É a perda esperada.
- PD É a probabilidade de default.
- LGD É a proporção de perda esperada dado que houve default.
- EAD É o valor de exposição caso aconteça o default

No desenvolvimento que se dará nos capítulos posteriores serão testado modelos para estimação da probabilidade de default em crédito para pessoas físicas.

2.2 Modelos

A seguir serão discutidos vários aspectos dos modelos citados acima. Ambos os modelos podem ser construídos utilizando o logito como função objetivo. As observações tem resposta dicotomizada em default e não default e portanto, serão utilizados modelos que se adequam a essa estrutura.

A implementação e utilização de todos os modelos a seguir se dá de forma parecida. Os modelos serão ajustados em uma extensa etapa de treinamento e posteriormente serão feitos testes em frações externas ao treinamento afim de validar a capacidade de predição do modelo. Após verificado que os modelos estão devidamente ajustados e que mantêm sua performance nas frações de testes o modelo é utilizado em produção e quaisquer atualizações ou reajuste dos coeficientes são feitos seguindo o mesmo procedimento da etapa de ajuste inicial. Neste trabalho não está sendo avaliada a viabilidade da implementação de modelos ditos *on-line*, que são constantemente atualizados com a incrementação de novas observações à medida que estas se mostram disponíveis.

É importante mencionar que existem outras alternativas aos modelos propostos neste trabalho. Todavia, qualquer outro tipo de técnica utilizada deverá retornar uma probabilidade como resposta e não apenas um rótulo de classificação, pois a probabilidade de default é uma medida utilizada para calcular a provisão[1] nas operações de crédito.

[1] A provisão é um percentual relativo ao montante da operação de crédito, determinada pela probabilidade de defaul e indica o valor que deve ser guardado afim de manter a saúde financeira da instituição financeira em caso de default.

2.2.1 Modelo Logístico

Tratando-se de reposta binária, é usual assumir uma distribuição binomial para o componente aleatório do modelo linear generalizado. Utilizaremos, pelo simples motivo de conveniência, a função de ligação natural (logito) para o ajuste, resultando num modelo da forma:

$$log\left(\frac{\pi(x_i)}{1-\pi(x_i)}\right) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}, \tag{2.1}$$

onde

- $\pi(x_i)$ é a probabilidade de default do i-ésimo indivíduo.
- x_i contém os valores observados das variáveis explicativas em um vetor da forma $(0, x_{i1}, x_{i2}, \ldots, x_{ip})$.
- o vetor $\tilde{\beta} = (\beta_0, \ldots, \beta_p)$ é estimado por $\hat{\beta} = (\hat{\beta}_0, \ldots, \hat{\beta}_p)$ que por sua vez é obtido por um processo iterativo de mínimos quadrados ponderados.

Isso implica que a probabilidade de default para uma observação é dada por

$$\pi(x_i) = \frac{\exp\left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}\right)}{1 + \exp\left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}\right)} \tag{2.2}$$

Uma outra grande vantagem do modelo logístico é a simplicidade na obtenção de estimativas intervalares e a fundamentação estatística em seus resultados. No caso dos modelos de Machine Learning, toda a estrutura e escolha de covariáveis se baseia em tentativa e erro e acurácia. Além disso, a obtenção de estatísticas intervalares em modelos de machine learning não é trivial como na regressão logística.

A maior desvantagem do modelo logístico em relação ao modelo que será discutido nas seções posteriores é que a relação entre os coeficientes β_i e o valor de saída do modelo é sempre linear e, na análise comportamental no contexto bancário, esse pressuposto nem sempre é

satisfeito, tendo em vista que fatores como a renda, idade e padrões de consumo interagem de forma complexa e muitas vezes não linear com o default.

Outro ponto a ser considerado é que o fator de inflação da variância (VIF) tem crescimento diretamente proporcional ao aumento do número de covariáveis no modelo logístico e, considerando o contexto observacional da análise e a grande disponibilidade de variáveis preditoras candidatas, isso se torna um problema caso não sejam usadas técnicas de regularização [2].

[2]Entede-se como regularização a Regressão Logística com penalidade L1 (LASSO), Regressão Logística com penalidade L2 (RIDGE) e também a "Elastic net", que combina as duas penalizações (**lsso**)

2.2.2 Long short-term memory

O Long short-term memory (LSTM) (Hochreiter e Schmidhuber, 1997) é uma arquitetura de redes neurais artificiais recorrentes (RNN) que tem como diferencial a capacidade de processar sequências longas de informações, o que se torna útil no contexto do problema apresentado neste trabalho. É comum as variáveis explicativas utilizadas na modelagem de risco de crédito variarem no tempo, geralmente sofrendo atualizações mensais, então, essa técnica permite que seja utilizado de forma completa o histórico financeiro de cada cliente.

O LSTM tem dois parâmetros chave que precisam ser determinados antes do treinamento do modelo. Um deles é o número de *timesteps*, que é o número de unidades temporais que serão observadas. No presente trabalho, são observados 12 meses de histórico das variáveis explicativas selecionadas, logo, utilizam-se obrigatoriamente 12 *timesteps*. O segundo parâmetro chave é chamado de *units* (ou *hidden units*), que determina vários aspectos da camada LSTM como a dimensão do vetor de saída da camada LSTM, a dimensão do *cell state* e *hidden state* e a dimensão dos vetores de saída das estruturas chave da células LSTM. O vetor de saída da camada LSTM pode ser interpretado como o resultado da redução de dimensionalidade de toda a matriz de dados longitudinais, encontrada através do ajuste de pesos de forma que se minimize o erro de classificação. Não existe uma metologia que determine a dimensão ótima deste vetor, tornando a tarefa de busca deste parâmetro direcionada pela otimização das métricas de ajuste do modelo e não por um cálculo pré-definido.

Uma *timestep* de uma camada LSTM é composta por 5 estruturas:

- O *Cell State* é a estrutura responsável por memorizar informações importantes entre as *timesteps*.
- O *Forget gate*, que produz um vetor com valores entre 0 e 1 e determina quais informações do *Cell State* serão eliminadas ou reduzidas.
- O conjunto de estruturas *Input gate e Candidate Values* determina quais informações serão adicionadas ao *Cell State*. A estrutura do *Input Gate* é idêntica ao *Forget Gate* porém interage de forma diferente com o *Cell State*. O *Input gate* determina quais informações serão adicionadas e os *Candidate Values* determinam a dimensão dessa atualização.
- O *Output gate* controla quais informações serão transformadas na atualização do *hidden state*.
- O *Hidden state* é o vetor de saída de cada *timestep* e consequentemente na última *timestep* é o vetor de saída da cama LSTM como um todo. Em

Essas estruturas interagem da seguinte forma:

1. O vetor de variáveis explicativas x_t no tempo é concatenado ao vetor *hidden state* da timestep anterior (h_{t-1}).
2. O vetor *hidden state* é processado pelo *forget gate*. O vetor resultante desse processamento é multiplicado ao *cell state* podendo retirar informações quando algum dos valores de deus elementos for inferior a 1 ou manter as informações do *cell state* caso todos os valores do vetor sejam 1.
3. O vetor *hidden state* é processado pelo *input gate* de forma análoga ao processo executado no *forget gate*, porém, o vetor resultante dessa estrutura vai determinar quais informações serão adicionadas ao *cell state*.

4. O vetor *hidden state* é processado pelo *Candidate values* resultando em um vetor com elementos nos intervalo [-1,1] o vetor resultante do *Candidate values* é multiplicado pelo vetor resultante do *input gate* e em seguida somado ao *cell state* agregando novas informações a esse vetor.

5. O vetor *hidden state* é processado pelo *Candidate values* resultando em um vetor com elementos nos intervalo [-1,1] o vetor resultante do *Candidate values* é multiplicado pelo vetor resultante do *input gate* e em seguida somado ao *cell state* agregando novas informações a esse vetor.

6. Os elementos do vetor *cell state*, neste momento já atualizados pelo *forget gate* e *input gate*, passam por uma transformação linear feita pela função tangente hiperbólica. Em seguida o vetor *hidden state* é processado pelo *forget gate* e em sequência é multiplicado pelo vetor *cell state* já transformado pela função tangente hiperbólica. O resultado dessa multiplicação resulta na atualização do *hidden state* daquela *timestep*. Caso estejamos tratando da última *timestep* da camada LSTM o vetor *hidden state* atualizado será a saída da camada, caso contrário, repete-se a etapa 1.

Na figura 2.1, tem-se a representação de uma única *timestep* de uma camada LSTM com 3 *units*. Neste exemplo, são utilizadas 2 variáveis explicativas, $F1$ e $F2$, observadas em 3 meses (resultando em 3 *timesteps*). O valor de saída da célula LSTM é o *hidden state* $\tilde{h}_t$ da última *timestep*.

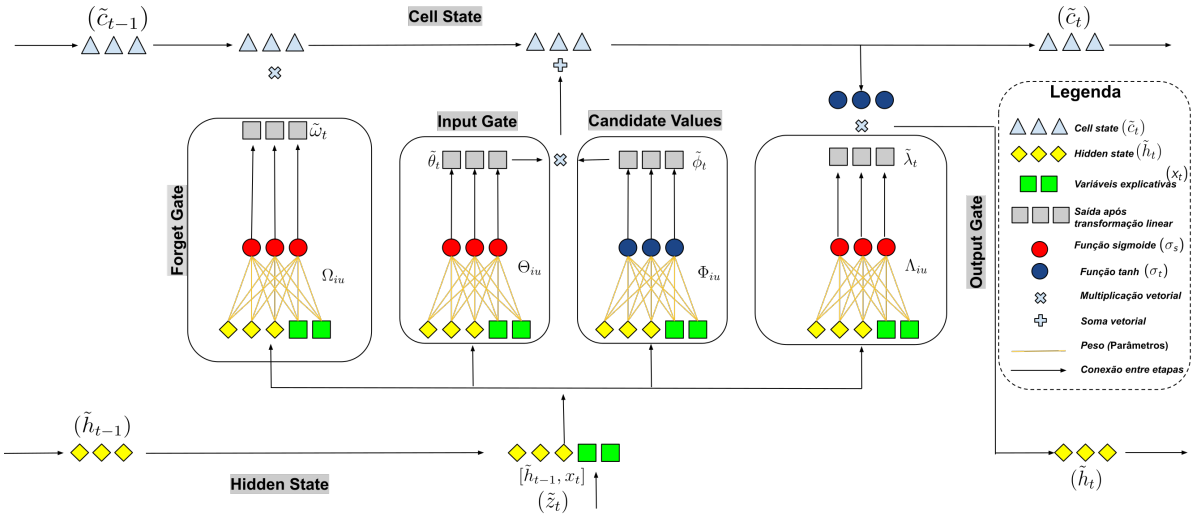

Figura 2.1: Representação de uma única *timestep* de uma camada LSTM.

Fonte: Elaborada pelo autor.

As equações que definem as operações e os vetores resultantes em cada parte de uma célula LSTM com 3 *units* são dadas por

$$Forget\ gate: \tilde{\omega}_t = \left[\sigma_s\Big(\sum_{i=1}^{p} \Omega_{iu_1} \cdot z_i^t + b_{\Omega_{u_1}}\Big), \ldots, \sigma_s\Big(\sum_{i=1}^{p} \Omega_{iu_h} \cdot z_i^t + b_{\Omega_{u_h}}\Big)\right] \tag{2.3}$$

$$Input\ gate: \tilde{\theta}_t = \left[\sigma_s\Big(\sum_{i=1}^{p} \Theta_{iu_1} \cdot z_i^t + b_{\Theta_{u_1}}\Big), \ldots, \sigma_s\Big(\sum_{i=1}^{p} \Theta_{iu_h} \cdot z_i^t + b_{\Theta_{u_h}}\Big)\right] \tag{2.4}$$

$$Candidate\ values: \tilde{\phi}_t = \left[\sigma_t\Big(\sum_{i=1}^{p} \Phi_{iu_1} \cdot z_i^t + b_{\Phi_{u_1}}\Big), \ldots, \sigma_t\Big(\sum_{i=1}^{p} \Phi_{iu_h} \cdot z_i^t + b_{\Phi_{u_h}}\Big)\right] \tag{2.5}$$

$$Cell\ state: \tilde{c}_t = \tilde{\omega}_t \circ \tilde{c}_{t-1} + \tilde{\theta}_t \circ \tilde{c}_t \tag{2.6}$$

$$Output\ gate: \tilde{\lambda}_t = \left[\sigma_s\Big(\sum_{i=1}^{p} \Lambda iu_1 \cdot z_i^t + b_{\Lambda u_1}\Big), \ldots, \sigma_s\Big(\sum_{i=1}^{p} \Lambda iu_h \cdot z_i^t + b_{\Lambda_{u_h}}\Big)\right] \tag{2.7}$$

$$Hidden\ state: \tilde{h}_t = \tilde{c}_t \circ [\sigma_t(\lambda_{t_u}), \ldots, \sigma_t(\lambda_{t_u})] \tag{2.8}$$

onde

- $\sigma_s(x)$ é a função de ativação (sigmoide) de uma *unit* definida por $\sigma_s(x) = \frac{1}{1+e^{-x}}$
- $\sigma_t(x)$ é a função de ativação (tangente hiperbólica) de uma *unit* definida por $\sigma_t(x) = \frac{e^{(x)}-e^{(-x)}}{e^{(x)}+e^{(-x)}}$
- O vetor que indica a quantidade de *units*, definido por u, tem dimensão h. A dimensão do vetor u determinara a dimensão de todos os outros elementos da camada LSTM.
- $\tilde{h}_{t-1}$ corresponde ao vetor do *hidden state* com dimensão h na *timestep* anterior.

  x_{it} corresponde ao valor das variáveis preditoras no tempo t, com dimensão p, que é relativa a quantidade de variáveis explicativas utilizadas no modelo.

  $[h_{it-1}, x_{it}]$, abreviado como z_i^t, corresponde a concatenação desses dois vetores, que tem dimensão final $h + p$.
- $\Omega_{iu}, \Theta_{iu}, \Phi_{iu}, \Lambda_{iu}$ são os parâmetros (pesos) que associam o i-ésimo valor do vetor z_i^t a u-ésima unidade correspondente no *forget gate*,*input gate*,*candidate values* e *output gate* respectivamente. Esses pesos são compartilhados entre todas as *timesteps*.
- $b_{\Omega_u}, b_{\Theta_u}, b_{\Phi_u}, b_{\Lambda_u}$ correspondem aos interceptos (*biases*) presentes em cada *unit* das estruturas *forget gate*,*input gate*,*candidate values* e *output gate* respectivamente.
- A operação "$\circ$"define a multiplicação matricial elemento a elemento, também conhecida como produto de Hadamard.

Os parâmetros das células LSTM são compartilhados em todas as *timesteps*, reduzindo o consumo de memória e otimizando o tempo de processamento, além disso, utilizando o compartilhamento de parâmetros ao invés de parâmetros independentes em cada *timestep* a camada LSTM fica menos suscetível ao *overfitting*.

A arquitetura utilizada é composta por uma camada LSTM, uma etapa de concatenação do vetor de saída dessa camada com o vetor de variáveis de perfil seguido por uma função de ativação sigmoide. Para mensurar o erro de classificação foi utilizada uma função chamada *binary cross-entropy* ou *logloss*, definida por:

$$Loglossy(\pi(y)) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(\pi\left(y_i\right)\right) + \left(1 - y_i\right) \cdot \log\left(1 - \pi\left(y_i\right)\right) \tag{2.9}$$

onde

- N é o tamanho da amostra.
- y_i é o rótulo (0 ou 1) da i-ésima observação.
- $p(y_i)$ é a probabilidade de *default* predita pelo modelo para a i-ésima observação

Por exemplo, uma rede com 10 *units* é composta por 640 parâmetros, obtidos a partir da seguinte forma:

- No *Forget gate* o vetor z_i^t tem dimensão 15 e u tem dimensão 10, logo, Ω_{iu} é composto por 150 parâmetros de peso. Cada *unit* tem 10 parâmetros de viés ($b_{\Omega_{u_1}}, \ldots, b_{\Omega_{u_1}}$), totalizando 160 parâmetros nessa camada.
- No *Input gate* o vetor z_i^t tem dimensão 15 e u tem dimensão 10, logo, Θ_{iu} é composto por 150 parâmetros de peso. Cada *unit* tem 10 parâmetros de viés ($b_{\Theta_{u_1}}, \ldots, b_{\Theta_{u_1}}$), totalizando 160 parâmetros nessa camada.
- No *Candidate calues* o vetor z_i^t tem dimensão 15 e u tem dimensão 10, logo, Φ_{iu} é composto por 150 parâmetros de peso. Cada *unit* tem 10 parâmetros de viés ($b_{\Phi_{u_1}}, \ldots, b_{\Phi_{u_1}}$), totalizando 160 parâmetros nessa camada.
- No *Output gate* o vetor z_i^t tem dimensão 15 e u tem dimensão 10, logo, Λ_{iu} é composto por 150 parâmetros de peso. Cada *unit* tem 10 parâmetros de viés ($b_{\Lambda_{u_1}}, \ldots, b_{\Lambda_{u_1}}$), totalizando 160 parâmetros nessa camada.

Somando os parâmetros acima, chega-se ao valor de 640 parâmetros otimizáveis mencionado anteriormente.

Seguindo o exemplo, os parâmetros da camada LSTM são otimizados de acordo com as seguintes etapas:

1. Os 640 parâmetros são iniciados de forma aleatória.

2. A amostra de treino é fracionada em sub amostras de tamanho pré determinado, denominadas *batches*.

3. Cada uma das *batches* é processada ao longo da rede em todas as $timesteps$, de acordo com as equações (2.1), (2.2), (2.3), (2.4), (2.5). Ao final da rede, após o processamento da saída da última $timestep$ por uma função sigmoide, são geradas probabilidades preditas e em seguida o erro é calculado de acordo com (2.7).

4. A partir do erro, são feitas as derivadas parciais do erro em relação aos pesos. As derivadas finais dos pesos são compostas pela soma das derivadas em todas as *timesteps*.

5. De acordo com o resultado das derivadas parciais (gradiente) os pesos são ajustados a fim de minimizar o valor da função de erro. Usualmente, o gradiente é controlado por um otimizador que reduz a dimensão do gradiente a fim de melhorar a busca pelo mínimo global, diminuindo o risco do gradiente desaparecer ou explodir (Hochreiter, 1998).

6. As etapas de 1 a 4 são repetidas de acordo com o número de iterações (*epochs*), que são pré definidas na etapa de desenvolvimento do modelo.

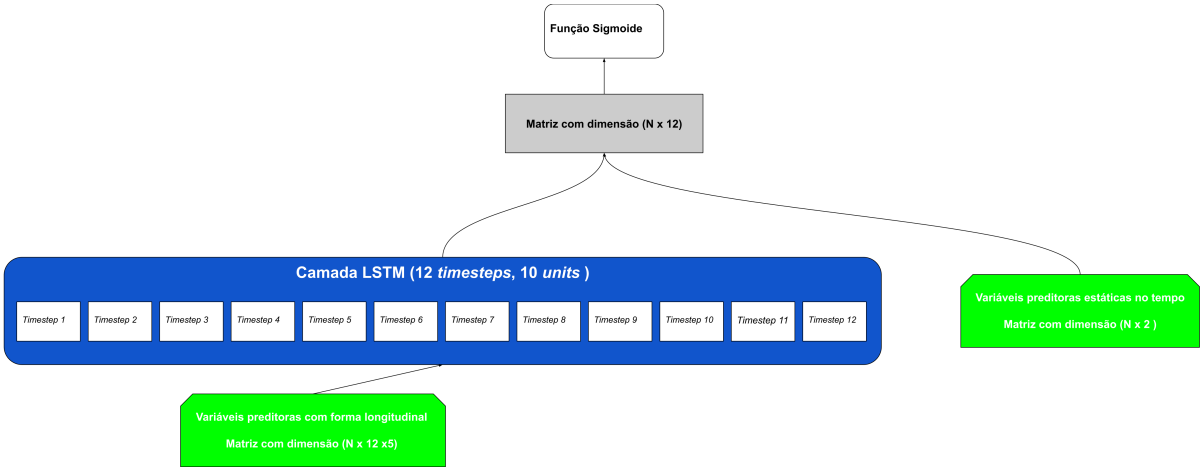

Figura 2.2: Exemplo da arquitetura utilizada com 8 *units*

Fonte: Elaborada pelo autor.

A rede neural ilustrada na figura 2.2 foi implementada usando a API Keras, (Chollet et al., 2015). Seja h a dimensão do vetor de *hidden units* e p o número de variáveis explicativas de comportamento, o número de parâmetros da arquitetura completa se dá por:

$$\Psi = 4 \cdot ((h + p) \cdot h + h) + (h + p + 1) \tag{2.10}$$

onde

- $4 \cdot ((h + p) \cdot h + h$ representa a quantidade de pesos otimizáveis a camada LSTM
- $(h + p + 1)$ representam os pesos otimizáveis entre a cama LSTM e a função sigmoide de saída da rede neural.

Ao final de todo o processo de ajuste do modelo, os pesos $\Omega_{iu}, \Theta_{iu}, \Phi_{iu}, \Lambda_{iu}$ ótimos são obtidos e a rede é capaz de gerar predições da probabilidade de *default* a partir de novas observações.

2.3 Ferramentas adicionais

2.3.1 *SHapley Additive exPlanation (SHAP) Values*

A necessidade da utilização de uma ferramenta externa aos modelos preditivos para de fato entender como as metodologias utilizam as covariáveis para criar a predição final se dá principalmente pela da natureza do estudo proposto ser observacional e não experimental, resultando na perda do controle do efeito cruzado entre covariáveis e da influência de fatores externos não analisados. No contexto dos algoritmos de machine learning, a utilização de um recurso adicional para explicar a relação das covariáveis com a resposta predita se faz ainda mais necessária pois a relação supracitada entre variáveis preditoras e variável resposta não é trivialmente recuperada devido à grande quantidade de parâmetros internos nos modelos, gerando inclusive a alcunha de "caixas pretas" para essa classe de modelos.

Mesmo quando é utilizada a regressão logística, onde temos os efeitos (coeficientes) associados a cada covariável bem especificados e estatisticamente testados, o problema de correlação entre as covariáveis persiste e, por definição, a implementação da regressão logística exige independência entre as covariáveis, resultando em um problema na interpretabilidade correta de cada covariável na probabilidade predita. O recurso a seguir nos fornece uma alternativa interessante para resolver esse problema.

Os valores SHAP (Lundberg e Lee, 2017) são uma forma de analisar a contribuição justa de cada covariável para a predição de um modelo, a partir da utilização dos valores shapley (Shapley, 1953) de uma função da esperança condicional do modelo original.

Dado algum banco de dados com P covariáveis e n observações, seja $\tilde{\mathbf{f}}$ o vetor contendo todas as P covariáveis. $\mathbf{f}'$ é o conjunto de todas as possíveis combinações dos elementos do vetor $\tilde{\mathbf{f}}$ e $\mathbf{f}'_p$ é um subconjunto de $\mathbf{f}'$ composto por todas as combinações do vetor $\mathbf{f}'$ que contém a covariável f_p. Seja x_i uma observação e $\pi_{\{\mathbf{f}'\}}(x_i)$ a predição de um modelo preditivo ajustado com $\mathbf{f}'$ covariáveis[3] acerca da observação x_i.

Podemos definir o valor SHAP por:

$$\phi(f_1, x_i) = \sum_{f'_1 \subseteq f' \subseteq \mathbf{f}} \frac{\pi_{f'_1}(x_i) - \pi_{(\mathbf{f}'_1 \setminus f_1)}(x_i)}{|\mathbf{f}'_1| \binom{|\mathbf{f}|}{|\mathbf{f}'_1|}} \tag{2.11}$$

Por exemplo, para calcular o valor SHAP para a variável f_1 da observação (x_1) de um banco de dados contendo 3 covariáveis ($\mathbf{f} = (f_1, f_2, f_3)$) teríamos que ajustar $\sum_{k=1}^{3} \binom{3}{k} = 2^3 = 8$ modelos diferentes. Após o ajuste dos modelos a fórmula expandida para o cálculo se dá por:

$$\begin{aligned} \phi(f_1, x_1) = &\frac{\pi_{\{(f_1)\}}(x_i) - \pi_{\{E(y_i)\}}(x_i)}{1 \cdot \binom{3}{1}} + \\ &\frac{\pi_{\{(f_1,f_2)\}}(x_i) - \pi_{\{f_2\}}(x_i)}{2 \cdot \binom{3}{2}} + \\ &\frac{\pi_{\{(f_1,f_3)\}}(x_i) - \pi_{\{f_3\}}(x_i)}{2 \cdot \binom{3}{2}} + \\ &\frac{\pi_{\{(f_1,f_2,f_3)\}}(x_i) - \pi_{\{f_2,f_3\}}(x_i)}{3 \cdot \binom{3}{3}} \end{aligned} \tag{2.12}$$

É notório que, na prática, é utilizado uma grande quantidade de covariáveis e a estimação dos valores SHAP por esse método seria inviável computacionalmente, tendo em vista o crescimento exponencial do número de modelos à medida que aumenta-se o número de covariáveis.

Tendo isso em vista, (Lundberg e Lee, 2017) demonstram que os valores SHAP podem ser estimados através de um algoritmo criado por eles chamado Kernel SHAP.

O algoritmo não necessita do ajuste de outros modelos além do original e funciona da seguinte forma. Sejam

[3] O "modelo" ajustado sem nenhuma covariável é definido por $E(y_i)$, que é simplesmente a esperança da variável resposta.

$$z'_k \in \{0,1\}^P, k \in 1,2,\ldots,2^P \tag{2.13}$$

$$h_x(z'_k) = z, \;\; h_x : \{0,1\}^P \rightarrow \mathbb{R}^P \tag{2.14}$$

$$g(z_k)' = \phi_0 + \sum_{j=1}^{P} \phi_j z'_k \tag{2.15}$$

A equação 2.16 representa uma ponderação das amostras de acordo com a quantidade de elementos não nulos do vetor z'_k.

$$\omega(\mathbf{z}'_k) = \frac{P-1}{\binom{P}{\sum\limits_{i \in z'_k} i}(\sum\limits_{i \in z'_k} i)(P - \sum\limits_{i \in z'_k} i)}, \;\; \forall \sum_{i \in z'_k} i \notin \{0, P\} \tag{2.16}$$

$$L(f,g,\omega) = \sum_{z'_k \in Z} \left[f(h_x(z'_k)) - g(z'_k) \right]^2 \cdot \omega(\mathbf{z}'_k) \tag{2.17}$$

Assumindo independência das covariáveis, a equação 2.17 pode ser reescrita como:

$$L(f,g,\omega) = \sum_{z'_k \in Z} \left[E_{Z_S}[f(z)] - g(z'_k) \right]^2 \cdot \omega(\mathbf{z}'_k) \tag{2.18}$$

Onde S são os valores diferentes de zero em z'_k.

A equação 2.18 é resolvida utilizando uma regressão linear, ponderada pela equação 2.16, resultando na estimação dos valores SHAP.

Existem algoritmos para estimar os valores SHAP que visam aumentar a eficiência do processo estimação e tentar estimar relações de dependências baseando-se na arquitetura dos algoritmos utilizados. Os algoritmos de estimação específicos são: Linear SHAP, otimizado para modelos lineares e Deep SHAP, que é otimizado para algortimos de Deep Learning (Lundberg e Lee, 2017) .

2.4 Métricas de avaliação para modelos de classificação

Afim de comparar os dois modelos serão utilizadas as métricas definidas nessa seção.

2.4.1 Área sobre a Curva Característica de Operação do Receptor

A curva característica de operação do receptor (curva ROC) é representada graficamente pelo conjunto dos valores da Sensibilidade e de 1-Especificidade para todos os possíveis pontos de corte utilizados para enquadrar as probabilidades preditas nos rótulos da classificação.

Para definirmos a Sensibilidade e Especificidade é útil definirmos antes a matriz de confusão, que é definida por:

		Classe Predita		
		Não *default*	*Default*	
Classe Real	Não *Default*	VN	FP	$VN + FP$
	Default	FN	VP	$FN + VP$
		$VN + FN$	$FP + VP$	

- VP são os "verdadeiros positivos", observações de *default* que de fato foram classificadas como *default*.
- FP são os "falsos positivos", observações de não *default* que de foram classificadas como *default*.
- FN são os "falsos negativos", observações de *default* que de foram classificadas como não *default*.
- VN são os "verdadeiros negativos", observações de não *default* que de foram classificadas como não *default*.

A sensibilidade e Especificidade são definidas como:

- Sensibilidade: $\frac{\text{VP}}{\text{VP+FN}}$

- Especificidade: $\frac{\text{VN}}{\text{VN+FP}}$

Podemos interpretar a Sensibilidade como a probabilidade do modelo prever um *default* quando de fato houve um *default* e podemos interpretar a Especificidade como a probabilidade do modelo prever um não *default* quando de fato não houve um *default*.

A área sobre a curva característica de operação do receptor (AUC) é definida como um valor escalar que mede o desempenho de um classificador binário (Hanley e McNeil, 1982). O AUC é definido no intervalo $[0.5, 1]$ onde o valor mínimo representa a performance de um classificador baseado num chute aleatório e o valor máximo representa um classificador perfeito. O valor do AUC pode ser interpretado como a probabilidade do modelo ordenar corretamente as observações

2.4.2 J de Youden

A estatística J de Youden é definida por:

$$J = \text{Sensibilidade} + \text{Especificidade} - 1 \tag{2.19}$$

Essa métrica é utilizada em conjunto com a curva ROC como uma forma de avaliar o desempenho de um modelo de classificação binária (Youden, 1950). A estatística assume valores no intervalo [0,1]. Quando a métrica tem valor 0 indica-se que o modelo atribui a mesma proporção de resultados positivos para verdadeiros positivos e faltos positivos. Um valor de 1 indica que o modelo não produz falsos positivos ou falsos negativos.

2.4.3 Coeficiente de correlação de Matthews

O coeficiente de correlação de Matthews (MCC) é utilizado para medir a correlação entre as classes preditas e observadas de um classificador binário (Matthews, 1975).

O MCC é definido por

$$MCC = \frac{VP \cdot VN - FP \cdot FN}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}} \tag{2.20}$$

Por ser um coeficiente de correlação, o coeficiente retorna valores no intervalo [-1,1]. Quando o coeficiente tem valor 1 indica-se que o modelo faz uma predição perfeita. O valor de -1 indica total não concordância entre as classes preditas e observadas. O coeficiente tem valor 0 quando o modelo de classificação tem o mesmo desempenho de uma classificação totalmente aleatória.

2.4.4 Brier *Score*

O Brier *Score* mede a acurácia das probabilidades preditas em um modelo de classificação binária, e é definido como o erro médio quadrático entre a probabilidade predita e o rótulo real (BRIER, 1950). Seja $\pi(x_i)$ a probabilidade de default predita para uma i-ésima obsevação, y_i a variável resposta com valor 0 em caso de não *default* e 1 em caso de *default* e N o tamanho da amostra, o Brier *Score* é definido por:

$$Brier\ Score = \frac{1}{N}\sum_{i=1}^{N}(\pi(x_i) - y_i))^2 \tag{2.21}$$

Onde:

- N é o total de observações.
- y_i é o rótulo (0 ou 1) da i-ésima observação.
- $\pi(y_i)$ é a probabilidade de default predita pelo modelo para a i-ésima observação.

Essa métrica é importante no contexto desse trabalho pois a acurácia nas probabilidades preditas é desejável, tendo em vista que as probabilidades previstas são utilizas para calcular o valor a ser provisionado por cada operação de crédito.

Essa métrica funciona como um complemento para outras métricas que observam apenas o acerto de classificação. Por se tratar de uma medida de erro, valores próximos 0 indicam que as probabilidades preditas são acuradas, ou seja, quanto menor o Brier *Score* mais acuradas são as probabilidades.

2.4.5 F1 *Score*

O F1 *Score* (Murphy, 2013) é definido como a média harmônica entre a precisão e a revocação. Define-se precisão e revocação por

- Precisão: $\frac{VP}{VP+FP}$
- Revocação: $\frac{VP}{VP+FN}$

Por fim, define-se o F1 *Score* por

$F1$ *Score*

$$2 \cdot \frac{\text{Precisão} \cdot \text{Revocação}}{\text{Precisão} + \text{Revocação}}$$

O F1 *Score* representa o equilíbrio entre a precisão e revocação. Seu maior valor possível é 1, representando um modelo perfeito.

2.4.6 Gráfico Kolmogorov-Smirnov

O Gráfico Kolmogorov-Smirnov (KS) (*SAS Institute Inc*) é utilizado para medir a performance de um classificador binário. A métrica é calculada empiricamente, dividindo as probabilidades preditas em nove decis e em seguida calculando a frequência acumulada de observações de default e não default com probabilidade pertencente a cada decil.

A métrica mede o grau de separação máximo entre as distribuições acumuladas das classes positivas e negativas. Os valores do K-S estão no intervalo $[0, 100]$. Caso o KS seja 0 conclui-se que o modelos não consegue diferenciar entre as duas classes e caso o K-S seja 100 conclui-se que o modelo separa perfeitamente as duas classes.

2.4.7 *Logloss*

A *Logloss* (Murphy, 2013), assim como o Brier Score, é utilizada para mensurar a capacidade do modelo de classificação binária de prever probabilidades acuradas. Valores baixos dessa métrica indicam que o modelo é capaz de atribuir probabilidades altas para observações da classe positiva e valores altos indicam o contrário.

Define-se a Logloss como

$$Loglossy(\pi(y)) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(\pi\left(y_i\right)\right) + \left(1 - y_i\right) \cdot \log\left(1 - \pi\left(y_i\right)\right), \tag{2.22}$$

onde

- N é o total de observações.
- y_i é o rótulo(0 ou 1) da i-ésima observação.
- $\pi(y_i)$ é a probabilidade de default predita pelo modelo para a i-ésima observação.

Capítulo 3

Resultados

3.1 Dados

Os dados utilizados para conduzir as análises provêm de informações de clientes da instituição financeira Sicoob. As informações são compostas por dados do Sistema Financeiro Nacional, Bureaus de crédito e informações internas do Sicoob.

Informações acerca do público estudado, nomes de variáveis e quaisquer informação que possam violar a segurança institucional do Sicoob Confederação serão descaracterizadas.

Tomam-se dois pré-supostos em relação a base de dados para a elaboração dos modelos em questão. Primeiramente, como estamos tratando de vários tomadores de crédito diferentes tomando crédito possivelmente, e suas respectivas operações de crédito tem montante, taxa de juros e prazos diferentes, é preciso assumir que todas as operações de crédito foram concedidas de forma ótima em relação à renda e peculiaridades de cada cliente, de forma independente do processo de modelagem de riscos, para que assim seja possível considerar que cada unidade observacional é igualmente distribuída. Caso as informações acerca do montante, taxa de juros, prazo e outas condições de pagamento fossem consideradas no modelo, é possível que esses valores possam ser exaustivamente alterados afim de se obter uma probabilidade predita desejada, invalidando o modelo.

Variável explicativa	Descrição	Grupo
X_1	Quantitativa contínua	Variável de perfil
X_2	Quantitativa contínua	Variável de perfil
X_3	Quantitativa discreta	Variável de comportamento
X_4	Quantitativa contínua	Variável de comportamento
X_5	Quantitativa contínua	Variável de comportamento
X_6	Quantitativa contínua	Variável de comportamento
X_7	Quantitativa contínua	Variável de comportamento

Tabela 3.1: Variáveis explicativas.

Além disso, será selecionada apenas uma operação de crédito por cliente. Esse procedimento em conjunto da hipótese de que o comportamentos dos indivíduos é independente entre si, corrobora o pré-suposto de independência das unidades observacionais.

O banco de dados a ser utilizado é composto por 30000 observações e 7 variáveis explicativas que foram anonimizadas e padronizadas de forma que apresentem média 0 e variância 1, de acordo com a tabela 3.1. Das 7 variáveis utilizadas, 5 são observadas no período de 12 meses antes do momento da concessão de crédito e duas delas são valores fixos que não variam temporalmente. Denominaremos estes dois grupos como variáveis de comportamento e variáveis de perfil respectivamente.[1] Todas as variáveis explicativas utilizadas foram padronizadas. A padronização das variáveis explicativas acontece com dois objetivos: primeiramente, ela é fundamental para a anonimização dos dados. Além disso, a padronização aumenta a velocidade de convergência e desempenho em redes neurais pequenas (Shanker, Hu e Hung, 1996).

A variável reposta a ser utilizada é chamada de *default*. O rótulo de *default* é atribuído para operações de crédito com atraso superior a 90 dias até 12 meses após a concessão de crédito.

[1]Na implementação da regressão logística, as variáveis de comportamento serão sumarizadas através da média afim de não obter observações repetidas para o mesmo usuário.

Na implementação da rede neural os dados serão utilizados integralmente, as variáveis comportamentais serão processadas utilizando a camada *Long-short term memory* e e as variáveis de perfil serão concatenadas ao vetor resultante nessa camada.

ou operações que foram renegociadas e que tenham 15 ou mais dias de atraso também até 12 meses após a concessão de crédito.

Nos dados utilizados existem 28026 operações marcadas como não *default* e 1974 marcadas como *default*. Apesar de existir um desequilíbrio entre as duas classes, não se utilizou nenhum mecanismo de subamostragem ou *oversampling* afim de gerar um equilibro artificial entre as classes.

A aplicação de todas as técnicas apresentadas neste trabalho foi feita utilizando o software Python.

Para o ajuste dos dois modelos a base de dados foi dividida em duas partes, 70% dos dados foram utilizados para o treino e 30% para teste. Na aplicação do LSTM, a base de treino foi subdividida em 90% para efetivamente ser usada para o treinamento do modelo e 10% para validação interna. Essa subdivisão é necessária pois algoritmos de redes neurais são mais sensíveis ao sobreajuste, requisitando um monitoramento adicional para garantir que o modelo terá uma boa capacidade de generalização.

Os dois modelos foram ajustados utilizando pesos para mitigar o efeito do desbalanceamento entre as duas classes. A base de treino, que foi comumente utilizada pelos dois modelos tem uma proporção de 14.1:1 observações de não *default* para observações com *default*, logo, optou-se por adicionar um peso de $\frac{14.1}{2}$ para as observações de *default* e $\frac{1}{2}$ para as observações de não *default*. O ajuste de pesos é necessário para melhorar a performance do LSTM e é desejável na aplicação da regressão logística para podermos executar uma comparação justa entre as métricas das duas técnicas.

Fração da base	Observações de não *default*	Observações de *default*	Total
Treino	19610 (93,38%)	1390 (6.62%)	21000
Teste	8416 (93,51%)	584 (6.49%)	9000

Tabela 3.2: Distribuição da variável resposta nas frações de treino e teste.

3.2 Regressão Logística

A regressão logística ajustada não utilizou nenhum tipo de regularização e sua otimização foi feita através do método de Newton. Como mencionado anteriormente, foram utilizados pesos proporcionais ao desbalanceamento das classes. Neste trabalho a regressão logística foi aplicada utilizando a biblioteca scikit-learn do software Python (Pedregosa et al., 2011).

Os seguintes coeficientes foram obtidos:

Variável	Coeficiente Estimado
Intercepto	-0.5855
X_1	-0.3852
X_2	-1.3176
X_3	-0.0222
X_4	0.2505
X_5	0.2197
X_6	0.0612
X_7	-0.3245

Tabela 3.3: Coeficientes estimados na regressão logística

Em seguida, tem-se a matriz de confusão e as métricas de performance do modelo logístico na fração de testes da base de dados.

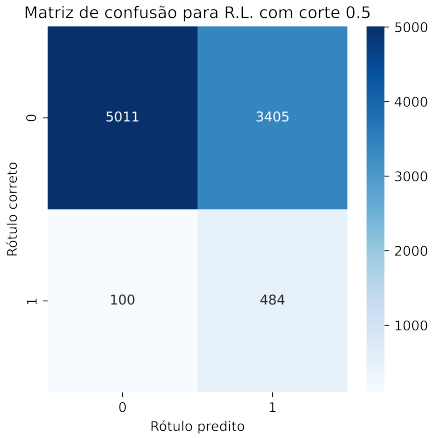

Figura 3.1: Matriz de confusão para a regressão logística.

A partir da figura 3.1, temos que o percentual total de acertos da regressão logística foi de 61.06%, o percentual de falsos positivos foi de 37.83% e o percentual de falsos negativos foi de 1.11%

A tabela 3.4 abaixo apresenta as métricas de performance obtidas na regressão logística.

Métrica	Valor
AUC	0.7506
Brier Score	0.2127
F1 Score	0.2164
J	0.4242
KS	42.9
Log loss	13.4512
MCC	0.2109

Tabela 3.4: Métricas de performance para a regressão logística.

3.3 LSTM

Para o ajuste da rede LSTM foram utilizados os seguintes parâmetros:

Parâmetro	Valor	Tipo
batch size	455	Fixo
Máximo de *epochs*	100	Fixo[2]
Critério de parada	5 *epochs* sem diminuição da função de perda na fração de validação	Fixo
Otimizador	*adam*	Fixo
batch size	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 75, 100, 200, 500\}$	Variável

Tabela 3.5: Hiperparâmetros utilizados no LSTM.

onde,

- *batch size* é o tamanho da subamostra da base de treino utilizada para a execução de cada passo do método do gradiente. Essa abordagem é mais adequada do que treinar o modelo utilizando todo o conjunto de dados de uma só vez ou utilizar uma observação por vez (Ruder, 2016a). O tamanho ótimo de cada *batch* não pode ser analiticamente definido. Tendo em vista que o problema atual apresenta uma classe relativamente rara (cerca de 6.62%), é interessante escolhermos um tamanho de *batch* que resulte numa representatividade suficiente da classe mais rara. Sendo assim, utilizou-se um *batch size* de 455 para ser obter em média 30 observações da classe rara por passo do método do gradiente. Não é necessária uma extensa etapa de experimentação para a busca do "valor ótimo" do *batch size*, a otimização deste parâmetros não tem relação com a qualidade do modelo mas sim com a velocidade de convergência.

- A quatidades de *epochs* indica a quantidade de vezes que o algoritmo de otimização dos parâmetros vai percorrer toda a base de treino. Afim de evitar o sobreajuste, também foi utilizado um critério de parada paralelo ao número de *epochs*. Esse critério de parada é ativado quando se observa uma estagnação no decrescimento da função perda na base de validação por 5 *epochs*.

- No contexto de redes neurais a escolha de um otimizador para o método do gradiente é crucial para a performance do modelo segundo (Ruder, 2016a). O otimizador *adam* (*Adaptative moment estimation*) é uma das melhores alternativas entre os otimizadores existentes Ruder, 2016b. Foram utilizados os valores dos hiperparâmetros do otimizados sugeridos por (Kingma e Ba, 2014) na rede LSTM implementada neste trabalho.

- A quantidade de *units* não pode ser analiticamente determinada. Neste trabalho foi utilizado um método exaustivo que consiste em ajustar várias redes com os parâmetros fixos descritos anteriormente e variar a quantidade de *units*. O valor final será escolhido de acordo com o número de *units* que apresentar o melhor conjunto de métricas.

Na tabela 3.6, podemos observar o resultado do experimento de testes com o número de *units*. É interessante notar que apesar do número de *units* implicar no aumento do número de parâmetros do modelo não existe uma relação linear entre o número de *units* e a melhora na qualidade do LSTM nesse conjunto de dados. Os resultados são muito próximos mas optou-se pelo número de 9 *units* pois nessa opção o modelo apresentou o maior AUC e também uma Log loss razoavelmente baixa.

Units	AUC	Brier Score	F1 Score	J	KS	Log loss	MCC
1	0.7601	0.2068	0.2163	0.4115	41.4	12.9869	0.2057
2	0.7624	0.2052	0.2148	0.4094	41.8	13.1557	0.2043
3	0.7811	0.2094	0.2227	0.4285	43.1	12.5916	0.2151
4	0.7595	0.2108	0.2148	0.4145	42.0	13.3515	0.2064
5	0.7646	0.2041	0.2320	0.4285	42.0	11.3059	0.2197
6	0.7685	0.2058	0.2246	0.4305	43.2	12.3728	0.2167
7	0.7762	0.2001	0.2271	0.4289	42.6	11.9622	0.2173
8	0.7700	0.2082	0.2184	0.4216	43.1	13.0214	0.2106
9	0.7825	0.1971	0.2360	0.4468	43.7	11.3827	0.2284
10	0.7581	0.2030	0.2151	0.3991	40.6	12.7182	0.2003
20	0.7817	0.1945	0.2327	0.4265	42.6	11.1639	0.2194
50	0.7763	0.1822	0.2368	0.4137	41.3	10.2889	0.2175
75	0.7765	0.1879	0.2343	0.4158	41.6	10.6343	0.2166
100	0.7724	0.1669	0.2558	0.3960	42.4	8.0822	0.2254
200	0.7693	0.1860	0.2339	0.4134	40.4	10.6075	0.2156
500	0.7578	0.1425	0.2555	0.3433	41.5	6.8004	0.2116

Tabela 3.6: Métricas de qualidade do ajuste da rede LSTM de acordo com o número de *units*.

A partir da figura 3.2, temos que o percentual total de acertos do LSTM foi de 66.7%, o percentual de falsos positivos foi de 31.86% e o percentual de falsos negativos foi de 1.43%

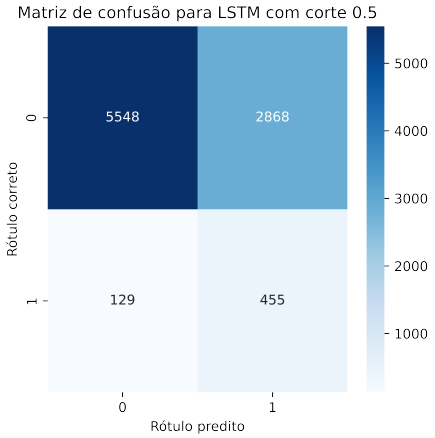

Figura 3.2: Matriz de confusão para a rede LSTM.

Observa-se a partir das tabelas 3.4 e 3.6 que o resultado do LSTM com 9 units foi superior á regressão logística. Para facilitar a comparação, a tabela abaixo contém as métricas de performance compiladas dos dois modelos.

Métrica	Logistica	LSTM 9 units
AUC	0.7506	0.7825
Brier Score	0.2127	0.1971
F1 Score	0.2164	0.2360
J	0.4242	0.4468
KS	42.9	43.7
Log loss	13.4512	11.3827
MCC	0.2109	0.2284
Acurácia	61.06%	66.7%
Falsos Positivos	37.83%	31.86%
Falsos Negativos	1.11%	1.43%

Tabela 3.7: Comparação das métricas de performance entre os dois modelos.

A rede LSTM teve performance superior em quase todas as métricas, com exceção dos falsos negativos. Dado o contexto do problema essa melhora poderia significar uma ganho financeiro para a instituição financeira caso essa rede fosse utilizada ao invés da regressão logística.

3.4 Avaliação da importância das variáveis explicativas

Para medir a importância de cada variável explicativa para a geração das predições nos dois modelos, foi utilizada a média do módulos dos valores SHAP para cada variável. Dessa forma, podemos mensurar o impacto que cada covariável tem no módulo probabilidade predita .

Para estimar os valores SHAP da regressão logística foi utilizado o método *Linear Explainer* proposto por (Lundberg e Lee, 2017). O método retorna uma matriz de dimensão 9000×7 contendo a contribuição marginal de cada variável explicativa para a probabilidade predita de cada uma das 9000 observações da base de testes. Em seguida, podemos calcular a média do módulo dos valores SHAP para obter a medida de importância de cada covariável.

Para estimar os valores SHAP da rede LSTM foi utilizado o método *Deep Explainer* proposto por (Lundberg e Lee, 2017) . A estrutura de dados de entrada do LSTM é composta de uma matriz de dimensão $9000 \times 12 \times 5$, referente aos dados comportamentais, e uma matriz de dimensão 9000×2, referente aos dados de perfil, então, as estimativas dos valores SHAP são obtidas também nessas dimensões. Podemos utilizar a propriedade de consistência, demonstrada por (Lundberg e Lee, 2017), para somar o módulo dos valores SHAP dos dados e perfil e obter uma matriz de valores SHAP de dimensão 9000×5 e concatená-la com o módulo da matriz de dados de perfil, resultando em uma matriz com dimensão 9000×7 no mesmo formato da matriz obtida para a regressão logística . A partir disso, também podemos calcular a média do módulo dos valores SHAP para obter a medida de importância de cada covariável.

A comparação do importância das covariáveis a partir da média dos módulos dos valores SHAP pode ser observada na tabela 3.8. Note que os valores diferem muito em algumas covariáveis, isso provalmente se dá pelo fato da rede LSTM ser capaz de estimar relações não lineares entre as covariáveis e a variável reposta. Essa métrica pode ser interpretada como a variação percentual média no módulo da probabilidade predita segundo (Molnar, 2019).

Média do módulo do valor SHAP	Reg. Logística	LSTM
X_1	0.096	0.026
X_2	0.832	0.178
X_3	0.05	0.012
X_4	0.048	0.698
X_5	0.058	0.051
X_6	0.001	0.119
X_7	0.1	0.619

Tabela 3.8: Comparação da média dos módulos dos valores SHAP entre a regressão logística e o LSTM.

Podemos notar, através diferença no valor da média dos módulos dos valores SHAP, que o LSTM mensura a relação das variáveis explicativas de uma forma diferente da regressão logística.

Note que a média dos módulos dos valores SHAP no caso da regressão logística tem a mesma ordenação dos módulos dos coeficientes da regressão logística na tabela 3.4, o que é esperado dada a definição dos valores SHAP. A modelagem de riscos de crédito exige a presença de algumas covariáveis que são correlacionadas, impactando nos valores dos coeficientes estimados. Nesse contexto, o SHAP é uma alternativa para mensurarmos o impacto das covariáveis nas probabilidades preditas e também é uma forma de mensurar os impactos das covariáveis com medidas repetidas e também pode ser utilizado como um critério de seleção de variáveis.

Capítulo 4

Considerações finais

Tendo em vista que todas as métricas de performance do modelo LSTM foram estritamente melhores, independente do número de *units* selecionadas, existe um indício que o LSTM é uma forma mais adequada para a modelagem de risco de crédito com covariáveis longitudinais quando o objetivo é a obtenção um modelo com maior capacidade de predição.

É importante ressaltar que a implementação da rede neural recorrente utilizada neste trabalho tem potencial para ser aprimorada através da inclusão de de camadas densamente conectadas à camada LSTM ou até mesmo o uso de outras arquiteturas recorrentes que consigam processar informações longitudinais de forma eficiente. Optou-se por uma implementação mais modesta do método por conta do foco na comparação do novo método de modelagem com o método padrão da indústria (regressão logística), caso fosse utilizada uma arquitetura contendo, por exemplo, empilhamentos de camadas LSTM ou camadas profundas densamente conectadas seria difícil isolar o efeito do processamento das covariáveis longitudinais pela camada LSTM na presença de outras estruturas capazes de mensurar relações não lineares nos dados.

Nesse contexto a utilização do SHAP também é interessante como um critério de seleção de variáveis explicativas, tendo em vista que na prática é utilizado um maior numero variáveis explicativas na modelagem, tanto para a regressão logística como para a arquitetura LSTM.

Ademais, o SHAP é uma forma interessante para mensurar o impacto de variáveis com medidas repetidas, gerando um valor de contribuição marginal para cada medida repetida de cada covariável.

Referências Bibliográficas

BRIER, GLENN W. (1950). "VERIFICATION OF FORECASTS EXPRESSED IN TERMS OF PROBABILITY". *Monthly Weather Review* 78.1, pp. 1 –3.

Chollet, François et al. (2015). *Keras*. `https://keras.io`.

Hanley, J A e McNeil, B J (1982). "The meaning and use of the area under a receiver operating characteristic (ROC) curve." *Radiology* 143.1, pp. 29–36.

Hochreiter, Sepp (abr. de 1998). "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions". *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, pp. 107–116.

Hochreiter, Sepp e Schmidhuber, Jurgen (dez. de 1997). "Long Short-term Memory". *Neural computation* 9, pp. 1735–80.

Kingma, Diederik e Ba, Jimmy (dez. de 2014). "Adam: A Method for Stochastic Optimization". *International Conference on Learning Representations*.

Lundberg, Scott M e Lee, Su-In (2017). "A Unified Approach to Interpreting Model Predictions". Ed. por I. Guyon et al., pp. 4765–4774.

Matthews, B.W. (1975). "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2, pp. 442–451. ISSN: 0005-2795.

Molnar, Christoph (2019). *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. `https://christophm.github.io/interpretable-ml-book/`. Cap. 8.6.6.

Murphy, K. (2013). “Machine learning: a probabilistic perspective”.

Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. *Journal of Machine Learning Research* 12, pp. 2825–2830.

Procedure”, SAS STAT 14.1 “User Guide The NPAR1WAY. *SAS Institute Inc*.

Ruder, Sebastian (2016b). “An overview of gradient descent optimization algorithms”. *ArXiv*.

— (2016a). “An overview of gradient descent optimization algorithms”. *ArXiv* abs/1609.04747.

Shanker, M., Hu, M.Y. e Hung, M.S. (1996). “Effect of data standardization on neural network training”. *Omega* 24.4, pp. 385–397. ISSN: 0305-0483.

Shapley, Lloyd S. (1953). “A value for n-person games”. *Contributions to the Theory of Games*, pp. 307–317.

Youden, W. J. (1950). “Index for rating diagnostic tests”. *Cancer* 3.1, pp. 32–35.