FATEC RUBENS LARA

CIÊNCIA DE DADOS – 6° CICLO

Detecção de Fraudes: Comparando Modelos de Aprendizado de Máquina para o Reconhecimento de Transações Potencialmente Fraudulentas

Juliana C. Monteiro

Alexandre Garcia de Oliveira

Santos – SP

Junho/2024

Resumo

Este projeto dedica-se à investigação e identificação de padrões de transações fraudulentas através do uso de algoritmos de aprendizado de máquina sob uma abordagem bayesiana. A inferência bayesiana, de acordo com Polli (2019), permite a incorporação de conhecimentos a priori no modelo e seus parâmetros são dados por distribuições de probabilidade, diferentemente da inferência frequentista que possui valores fixos para os parâmetros. Em 23 de maio de 2023, o Banco Central do Brasil (BACEN) e o Conselho Monetário Nacional (CMN) publicaram a Resolução Conjunta nº 6, que estabelece novas obrigações para as instituições financeiras com o objetivo de prevenir fraudes no Sistema Financeiro Nacional. Esse cenário marca o início de uma nova era para o aprimoramento dos modelos de prevenção a fraudes, impulsionada pelo crescente volume de dados nessa área.

Palavras-chave: Ciência de dados; Detecção de Fraude; Aprendizado de Máquina.

Abstract

This project is dedicated to investigating and identifying patterns of fraudulent transactions through the use of machine learning algorithms under a Bayesian approach. Bayesian inference, according to Polli (2019), allows the incorporation of a priori knowledge into the model and its parameters are given by probability distributions, unlike frequentist inference which has fixed values for the parameters. On May 23, 2023, the Central Bank of Brazil (BACEN) and the National Monetary Council (CMN) published Joint Resolution No. 6, which establishes new obligations for financial institutions with the aim of preventing fraud in the National Financial System. This scenario marks the beginning of a new era for improving fraud prevention models, driven by the growing volume of data in this area.

Key words: Data science; Fraud Detection; Machine Learning.

1 Introdução

Em um cenário onde as fraudes financeiras estão em constante evolução, as instituições vêm intensificando seus esforços na busca por técnicas cada vez mais avançadas de detecção de fraudes, visando assegurar a integridade de suas operações. Segundo a Cisor Advisor (2023), plataforma dedicada à transmissão de informações de segurança cibernética, a estimativa de que 20% da receita online dos bancos na América Latina seja perdida devido a fraudes destaca a urgência de aprimorar os sistemas de proteção. Notavelmente, o Brasil lidera esse desafio, registrando prejuízos anuais da ordem de R$60 bilhões em decorrência de crimes relacionados ao roubo de identidade.

Os modelos de aprendizado de máquina, comumente conhecidos como *machine learning*, têm se destacado como ferramentas promissoras na prevenção de diversas formas de atividades fraudulentas. No entanto, a natureza sensível das informações pessoais demanda cuidados específicos. As bases de dados utilizadas para o treinamento desses modelos permanecem centralizadas nas empresas, sendo disponibilizadas apenas sob demanda e mediante autorização para empresas credenciadas.

Entretanto, uma mudança significativa ocorreu em 1º de novembro de 2023, com a entrada em vigor da Resolução Conjunta no 6 do Banco Central (Bacen), em conjunto com o Conselho Monetário Nacional (CMN). Essa resolução estabelece critérios para o compartilhamento de dados e informações relacionadas a indícios de fraudes e é voltada às instituições financeiras, instituições de pagamento e outras entidades autorizadas pelo Banco Central do Brasil. Essa medida abre novos horizontes para o aprimoramento das soluções antifraude, uma vez que possibilita a construção de bases de dados mais robustas e completas. Com o compartilhamento de informações entre as instituições autorizadas, os modelos de aprendizado de máquina terão acesso a dados mais variados.

Diante desse panorama, o presente artigo propõe uma comparação de modelos para

prever e identificar padrões suspeitos, assim prevenindo a fraude. O conjunto de dados utilizado foi anonimizado para proteger a privacidade dos titulares.

2 Fundamentação Teórica

2.1 Transações Fraudulentas de Cartão de Crédito

O cartão de crédito é amplamente utilizado pela população como uma ferramenta de uso ao crédito pessoal. Além de oferecer crédito, ele também oferta diversos benefícios com o objetivo de atrair e fidelizar seus usuários. Com os progressos tecnológicos, o uso do cartão de crédito tornou-se mais acessível e prático para os consumidores, especialmente com o crescimento das opções de compras online (*e-commerce*). Contudo, de acordo com Figueiredo (2003), essa conveniência também atraiu a atenção de fraudadores especializados, com a clonagem de cartões sendo a prática ilegal mais comum.

2.2 Modelos de Previsão para Detecção de Fraudes

Os modelos de previsão utilizam algoritmos de *machine learning* para analisar padrões nos dados transacionais e identificar comportamentos suspeitos. Alguns dos métodos comuns incluem:

- **Modelos Supervisionados:** Conforme o conceito exposto por Barber (2012), nos modelos supervisionados, existe uma base de dados com as características x e classificação y. Neste processo, o algoritmo irá identificar quais características presentes em x são suficientes para chegar à classificação de y.
- **Modelos Não Supervisionados:** Esta abordagem visa identificar padrões em bases de dados, mas sem a existência de uma resposta prévia para a classificação que será identificada (JUNIOR, 2019). Eles identificam agrupamentos incomuns ou anômalos nos dados, o que pode indicar a presença de fraudes.

- **Modelos Semi-Supervisionados:** No aprendizado semi-supervisionado, utiliza-se os dados não rotulados para tentar criar um classificador melhor do que o criado com base apenas nos dados rotulados (JUNIOR, 2019).

3 Procedimentos Metodológicos

3.1 Sobre a Base de Dados

Dado que a disponibilidade de conjuntos de dados autênticos relacionados a serviços financeiros é restrita, a amostra de estudo desse projeto foi retirada do *Kaggle*, plataforma de competição de ciência de dados da Google LLC. Intitulada de *Credit Card Fraud*, os dados da base abrangem transações realizadas durante duas horas com cartões de crédito em setembro de 2013 por titulares europeus. As variáveis são:

- Time: tempo decorrido em segundos por transação;
- V1-V28: Variáveis anonimizadas através da técnica *Principal Components Analisys (PCA)* representando diversos atributos das transações;
- *Amount*: Valor da transação;
- *Class*: Variável resposta indicando se a transação é fraudulenta ou não (0 ou 1).

3.2 Ferramentas de Análise

As análises serão criadas usando a linguagem de programação *Python* por ser uma linguagem *open-source*, ou seja, pode ser usada de forma gratuita e sua manutenção é mantida pela comunidade de usuários, de fácil aprendizado e otimizada para várias áreas da computação, como inteligência artificial, banco de dados, biotecnologia, entre outras

(MENEZES, 2019). As bibliotecas ou pacotes das linguagens de programação agem como submódulos que caregam funções criadas para determinado objetivo (COODESH, 2023). As bibliotecas de destaque usadas nesse projeto foram *Pandas*, *Numpy* e, finalmente, a biblioteca *Pymc*. A biblioteca *Numpy* é a base para diversas outras bibliotecas, pois lida com diversas estruras e objetos matemáticos (SAPRE; VARTAK, 2020). Já o pacote Pandas é otimizado para lidar com a estrutura tabular dos dados (SAPRE; VARTAK, 2020), gerando os *Dataframes* que iremos manipular. Por fim, o *Pymc* é um pacote probabilístico que fornece ferramentas para a contrução de modelos bayesianos, sua versatilidade permite a implementação de diversos modelos simples e complexos como regressão linear hierárquica generalizada, classificação, séries temporais, equações diferenciais ordinárias e modelos não paramétricos, como processos gaussianos (ABRIL-PLA et al., 2023).

3.3 Análise Exploratória dos Dados

A análise começa com uma exploração dos dados, onde são examinadas as características e estatísticas resumidas do conjunto. A base conta com 283.726 registros únicos de transações. A variável resposta *"Class"*, de natureza binária, possui uma proporção de somente 0,0172% de fraudes, ou seja, altamente desbalanceada. Como os dados já estavam com o PCA aplicado, optou-se por medir a influência entre as variáveis independentes e a variável dependente para a detectar as colunas mais relevantes e reduzir mais ainda a dimensionalidade. Com isso, usou-se o cálculo da correlação de *Pearson*. O coeficiente de correlação de *Pearson* é uma medida de associação linear entre variáveis (FIGUEIREDO FILHO; JÚNIOR, 2010). Sua fórmula é a seguinte:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{1}$$

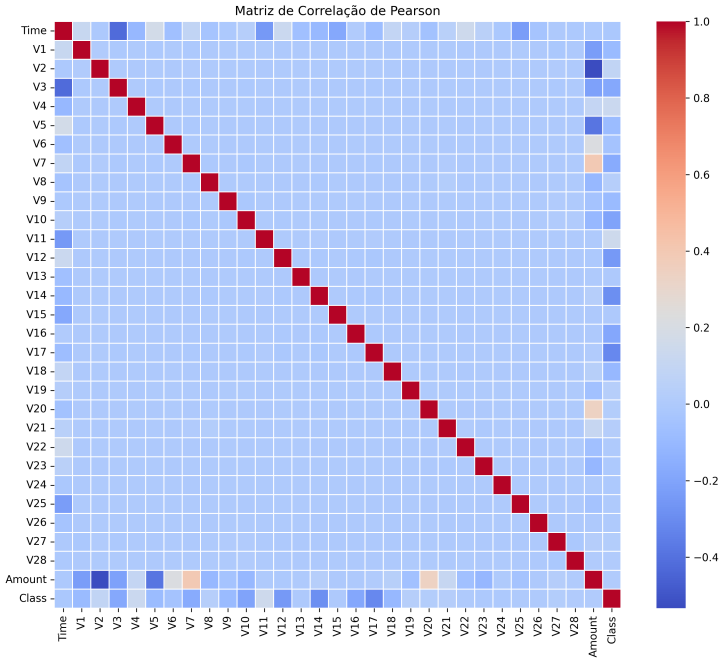
Onde:

- r é o coeficiente de correlação de Pearson.
- x_i e y_i são os valores individuais das variáveis x e y, respectivamente.
- $\bar{x}$ e $\bar{y}$ representam as médias das variáveis x e y, respectivamente.
- n é o número total de pares de observações.
- O numerador, $\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$, é a soma dos produtos das diferenças entre cada valor e sua média.
- O denominador, $\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}$, é o produto dos desvios padrão das variáveis x e y.

O coeficiente r varia entre -1 e 1. Valores próximos a 1 indicam uma forte correlação positiva, enquanto valores próximos a -1 indicam uma forte correlação negativa. Valores próximos a 0 indicam ausência de correlação linear.

Para uma melhor visualização, a matriz resultante do cálculo de correlação foi plotada em um gráfico:

Gráfico da Matriz de Correlação das Variáveis da Base



Fonte: próprio autor.

3.4 Pré-processamento

A preparação dos dados passou pelas etapas de normalização, seleção de variáveis relevantes e separação dos conjuntos de treino e teste. Para otimizar o tempo de convergência dos modelos, optou-se pela técnica de normalização dos dados, pois permite que a escala dos dados seja ajustada de forma que todos os valores de uma variável fiquem entre o intervalo de 0 e 1 (AVELINO, 2024). Após a análise do gráfico de correlação (imagem X), as colunas selecionadas para serem usadas em todos os modelos foram as variáveis ”V12”, ”V14” e ”V17” por possuirem correlação moderada com a variável resposta e não

possuirem correlação notável entre si, caso contrário, suas capacidades de previsão seriam irrelevantes (MORETTIN; SINGER, 2022). Após a normalização e seleção de variáveis, os dados foram separados em conjutos de treino, onde 70% da amostra foi reservada para o treino dos modelos e conjunto de teste, onde os 30% restantes foram separados para a validação dos modelos.

3.5 Modelo 1 - Regressão Logística

A regressão logística estima a probabilidade de ocorrência de um evento, com base em um determinado conjunto de dados de variáveis independentes (IBM, 2023). Conforme elucidado por Gonzalez (2018), o modelo é expresso pela função logit, que é o logaritmo da razão de chances (odds) de a variável dependente ser. A fórmula do modelo de regressão logística é:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n \tag{2}$$

Onde:

- p é a probabilidade de sucesso (ou presença) da variável dependente.
- β_0 é a interceptação.
- $\beta_1, \beta_2, \ldots, \beta_n$ são os coeficientes de regressão associados às variáveis independentes $x_1, x_2, \ldots, x_n$, respectivamente.

Ao aplicar a função logística (ou sigmóide) em ambos os lados da equação, pode-se resolver para a probabilidade de sucesso p:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n)}} \tag{3}$$

Onde e é a base do logaritmo natural. Essa função logística transforma a regressão linear em uma curva que varia entre 0 e 1, adequada para modelar probabilidades.

A variável dependente Y na regressão logística é frequentemente binária, logo, nestes casos ela segue a distribuição de Bernoulli (BELFIORE, 2015), tendo uma probabilidade desconhecida p. Lembrando que a distribuição de Bernoulli é apenas um caso especial da distribuição binomial, onde $n = 1$ (considera a realização de um único experimento).

$$Y = \begin{cases} 1, & sucesso \\ 0, & fracasso \end{cases}$$

A probabilidade de sucesso é $0 \leq p \leq 1$ e a probabilidade de fracasso é $q = 1 - p$. Na regressão logística, é feita a estimação da probabilidade desconhecida p, dado uma combinação linear de variáveis independentes.

Levando em consideração o uso de 70% dos dados em sua construção, o primeiro modelo (Modelo 1), seguiu as seguintes distribuições:

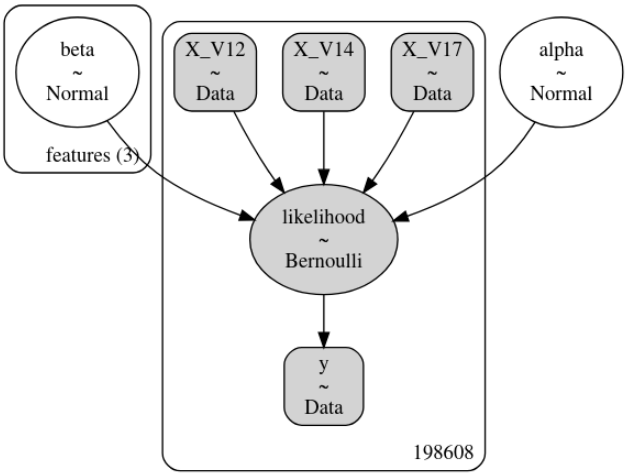
Código em Python do Modelo 1

```
[ ] with pm.Model(coords=coords_corr) as model_1:
      betas = pm.Normal("beta",0,1,dims="features")
      alpha = pm.Normal("alpha",0,1)

      X = [
        pm.MutableData("X_"+column,X_train[column].values.ravel())
        for column in X_train.columns ]
      y = pm.MutableData("y",y_train.values.ravel())

      mu = alpha + tt.dot(betas,X)
      likelihood = pm.Bernoulli("likelihood",pm.math.invlogit(mu),observed=y)
```

Fonte: próprio autor.

O Modelo 1 também pode ser representado pelo grafo abaixo:

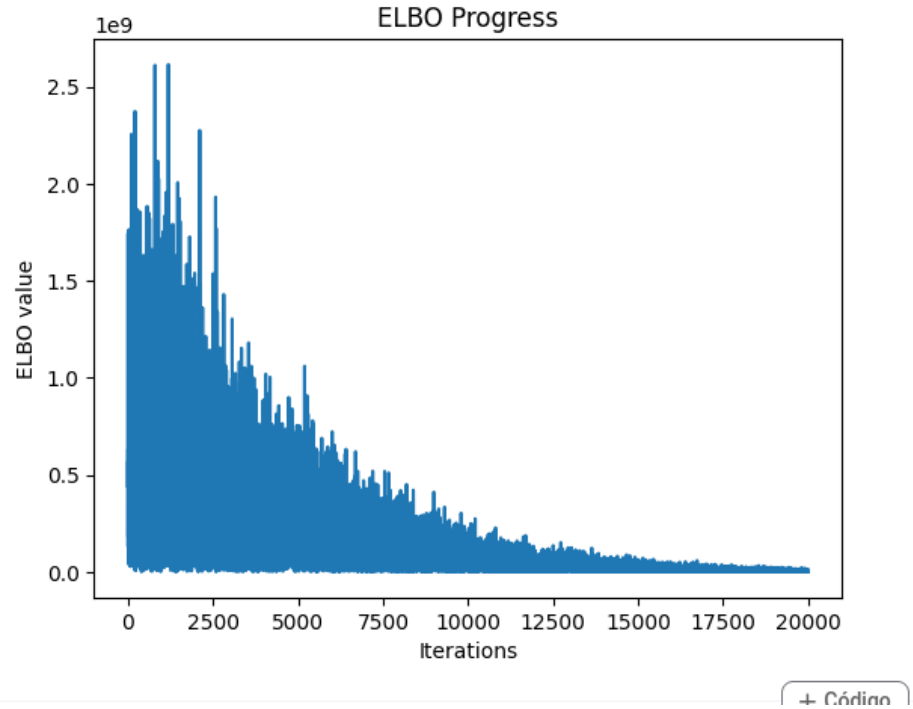
Gráfo do Modelo 1



Fonte: próprio autor.

O Evidence Lower Bound (ELBO) é uma métrica usada em métodos bayesianos variacionais para estimar o logaritmo da probabilidade de dados observados. Ele serve como um limite inferior para a probabilidade real, permitindo que se avalie a qualidade de ajustes em modelos probabilísticos (KINGMA; WELLING, 2019). Para verificar se houve convergência numa aproximação dos parâmetros a priori do Modelo 1, analisou-se a evolução do valor da função ELBO ao longo das iterações do algoritmo de inferência variacional.

Gráfico da Função ELBO da Convergência dos Parâmetros a Priori



Fonte: próprio autor.

Conforme os picos vão diminuindo e a distribuição ficando linear, podemos concluir que os parâmetros a priori convergiram no modelo.

Com o objetivo de avaliar o desempenho do modelo, a acurácia é uma das métricas mais comuns e amplamente utilizadas. Ela é calculada como a proporção de previsões corretas (tanto positivas quanto negativas) em relação ao total de previsões realizadas (MARIO FILHO, 2023). A fórmula para calcular a acurácia é dada por:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

Onde: - VP representa os Verdadeiros Positivos, - VN representa os Verdadeiros Negativos, - FP representa os Falsos Positivos, - FN representa os Falsos Negativos.

Ao final, o Modelo 1 apresentou acurácia de 99,9%.

Acurácia	0.9985
Erros	131

Entretanto, tratando-se de um problema de classificação binária com uma base altamente desbalanceada, a acurácia não se mostra a melhor métrica de avaliação. Isso ocorre porque, dado que 99% da base possui marcação de ”não-fraude (0)”, é natural que a acurácia aponte uma grande proporção de acertos. Para lidar com esse desafio, existem outras métricas que avaliam o modelo de forma mais adequada, como a precisão, o *recall* e o *F1-Score*.

Precisão (*Precision*): A precisão é uma métrica que mede o quanto as previsões positivas feitas pelo modelo são realmente corretas, ou seja, o quão confiável é o modelo quando prevê um positivo (MARIO FILHO, 2023). É a proporção de verdadeiros positivos em relação ao total de positivos previstos (verdadeiros positivos + falsos positivos). Sua fórmula é dada por:

$$Precisão = \frac{VP}{VP + FP}$$

Recall: O *recall* mede o quanto o modelo consegue identificar corretamente os verdadeiros positivos, ou seja, o quão eficaz ele é em encontrar todos os casos positivos reais (MARIO FILHO, 2023). É a proporção de verdadeiros positivos em relação ao total de positivos reais (verdadeiros positivos + falsos negativos). Sua fórmula é:

$$Recall = \frac{VP}{VP + FN}$$

F1-Score: Por fim, o *F1-Score* combina a precisão e o *recall* em uma única métrica que avalia o equilíbrio entre essas duas métricas (MARIO FILHO, 2023). É calculado como a média harmônica da precisão e do *recall*, sendo especialmente útil em casos de desequilíbrio de classes (quando uma classe ocorre muito mais frequentemente que outra). A fórmula do *F1-Score* é:

$$F1 = 2 \times \frac{Precisão \times Recall}{Precisão + Recall}$$

Com essas métricas, é possível obter uma visão mais balanceada do desempenho do modelo, especialmente em cenários desbalanceados, tendo o Modelo 1 obtidos as seguintes métricas:

Precisão	1.0
Recall	0.030
F1-Score	0.058

A precisão, o *recall* e o *F1-Score* apresentam métricas baixas, com precisão perfeita, mas *recall* e *F1-Score* muito baixos, o que é comum em casos de desequilíbrio entre as classes.

3.6 Modelo 2 - Regressão Logística com LASSO

Uma das estratégias para lidar com bases desbalanceadas, onde a regressão logística simples pode não ter um alto desempenho devido ao desequilíbrio entre as classes, é a implementação da técnica de *Least Absolute Shrinkage and Selection Operator* (LASSO) no modelo. Isso ocorre porque a aplicação do LASSO em uma regressão logística ajuda a regularizar o modelo, aplicando uma penalização sobre os coeficientes da regressão. Essa penalização favorece a seleção de variáveis relevantes, enquanto reduz a influência das variáveis menos importantes (WANG; XU; ZHOU, 2015).

Em cenários com dados desbalanceados, o LASSO pode ser particularmente útil, pois a penalização aplicada a esses coeficientes auxilia na melhoria da generalização do modelo e na prevenção de sobreajuste, especialmente à classe majoritária. Isso resulta em um modelo mais equilibrado, com melhor capacidade de identificar a classe minoritária. A função de custo da regressão logística com Lasso pode ser escrita como:

$$\mathcal{L}(\beta) = \sum_{i=1}^{n} [-y_i \log(p_i) - (1 - y_i) \log(1 - p_i)] + \lambda \sum_{j=1}^{p} |\beta_j|$$

Onde:

- $\mathcal{L}(\beta)$ é a função de custo que o modelo busca minimizar,
- y_i é a variável alvo (classe),
- p_i é a probabilidade prevista para a classe positiva,
- λ é o parâmetro de regularização (controle da intensidade da penalização),
- β_j são os coeficientes da regressão.

Com isso, o Modelo 2 seguiu as seguintes distribuições:

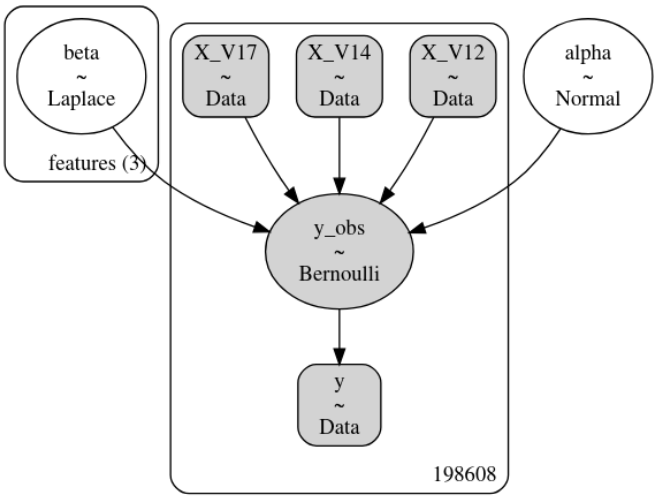
Código em Python com o Modelo 2

```
with pm.Model(coords=coords_corr) as model_2:
    betas = pm.Laplace('beta', mu=0, b=1,dims="features")
    alpha = pm.Normal('alpha', mu=0, sigma=10)

    X = [
    pm.MutableData("X_"+column,X_train[column].values)
    for column in X_train.columns ]
    y = pm.MutableData("y",y_train.values.ravel())


    mu = alpha + tt.dot(betas,X)

    y_obs = pm.Bernoulli('y_obs', pm.math.invlogit(mu), observed=y)
```

Fonte: próprio autor.

Também podendo ser representado pelo grafo abaixo:

Gráfo do Modelo 2



Fonte: próprio autor.

Por fim, as métricas de avaliação do modelo foram as seguintes:

Precisão	1.0
Recall	0.037
F1-Score	0.071

Para o caso da amostra em estudo, o Modelo 2 teve uma melhor performance em relação ao Modelo 1, mas o aumento não foi considerável.

3.7 Modelo 3 - Regressão Logística com LASSO e *Oversampling*

Outra técnica aplicada em bases desbalanceadas é o metodo de *Oversampling*. A técnica de *Oversampling* é uma técnica usada para lidar com dados desbalanceados, aumentando a quantidade de amostras da classe minoritária para balancear a base de dados. A técnica mais conhecida e utilizada nesse projeto é o *SMOTE* (Synthetic Minority Over-sampling Technique), que gera exemplos sintéticos, criando novos pontos de dados entre os exemplos existentes da classe minoritária (CHAWLA; BOWYER; HALL; KEGELMEYER, 2002). Isso permite que o modelo aprenda melhor as características dessa classe, melhorando sua performance.

O Modelo 3 adicionou ao modelo de regressão logística com LASSO uma base de dados reamostrada através do SMOTE, o que duplicou a quantidade de registros em relação a base original, fazendo com que a variável ”Class” ficasse proporcialmente com a mesma quantidade de classes (fraudes e não-fraudes).

Código em Python com a Aplicação da Reamostragem da Base

```
smote = SMOTE(random_state=42)

X_resampled, y_resampled = smote.fit_resample(x, y)

print(f"Tamanho original (classe 0, classe 1): {y.value_counts()}")
print(f"Tamanho após oversampling (classe 0, classe 1): {y_resampled.value_counts()}")
```

```
Tamanho original (classe 0, classe 1): Class
0.0    283253
1.0       473
Name: count, dtype: int64
Tamanho após oversampling (classe 0, classe 1): Class
0.0    283253
1.0    283253
```

Fonte: próprio autor.

Com isso, o Modelo 3 seguiu as seguintes distribuições:

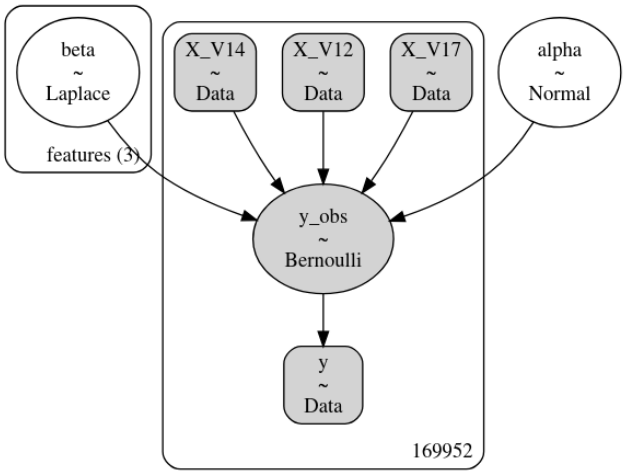
Código em Python com o Modelo 3

```
with pm.Model(coords=coords_corr) as model_3:
    betas = pm.Laplace('beta', mu=0, b=1,dims="features")
    alpha = pm.Normal('alpha', mu=0, sigma=10)

    X = [
    pm.MutableData("X_"+column,X_train_r[column].values)
    for column in X_train_r.columns ]
    y = pm.MutableData("y",y_train_r)


    mu = alpha + tt.dot(betas,X)


    y_obs = pm.Bernoulli('y_obs', pm.math.invlogit(mu), observed=y)
```

Fonte: próprio autor.

O gráfo do Modelo 3 possui os seguintes parâmetros:

Gráfo do Modelo 3



Fonte: próprio autor.

Por conta do balanceamento da base, usou-se somente a acurária para validar a quantidade de acertos do modelo, que retornou com 87,4%:

Acurácia	0.874
Erros	21476

3.8 Modelo 4 - Processo Gaussiano Classificador

Processos Gaussianos ou *Gaussian Process* (GP) são modelos probabilísticos que definem uma distribuição sobre funções. A principal característica é que qualquer subconjunto finito de variáveis aleatórias associadas a esse processo segue uma distribuição normal multivariada. Diferentemente de modelos paramétricos, como a regressão linear, GPs não assumem uma forma fixa para a função que mapeia entradas para saídas, permitindo que os dados determinem essa forma. A função de covariância (ou kernel) é essencial, pois descreve como os valores de saída são correlacionados em diferentes pontos de entrada, in-

fluenciando propriedades como suavidade e periodicidade da função modelada (SCHULZ; SPEEKENBRINK; KRAUSE, 2018).

Os hiperparâmetros da função de covariância incluem o comprimento de escala (l), que controla como as variáveis de entrada afetam as correlações, e a variância, que determina a magnitude esperada das flutuações. No Modelo 4, usaremos o kernel Matérn 3/2, que é otimizado para lidar com dados mais irregulares, ajustando suavidade e variações rápidas com seus hiperparâmetros.

Em tarefas de classificação binária, GPs modelam incerteza ao utilizar uma função latente contínua, que é transformada em uma probabilidade de classe por meio de uma função de ligação, como a função softmax(LIU et al., 2021). Para um vetor de entrada $\mathbf{z} = [z_1, z_2, \ldots, z_n]$, a função softmax é definida como:

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

Onde:

- z_i: O valor original do i-ésimo elemento.
- e^{z_i}: A exponenciação do i-ésimo valor.
- $\sum_{j=1}^{n} e^{z_j}$: A soma das exponenciais de todos os valores no vetor.

Devido a complexidade do modelo seus cálculos matriciais, foi necessário retirar uma amostra estratificada de 0,5% da base, mantendo a proporção das classes da base original, ou seja, o alto desbalanceamento das classes também foi replicado na base de amostra.

Código em Python com Aplicação da Amostra do Modelo 4

```
amostra_proporcao = 0.005

df_amostra, _ = train_test_split(
    df_normalizado,
    test_size=1 - amostra_proporcao,  # Porcentagem de dados na amostra
    stratify=df_normalizado['Class'],
    random_state=42
)

print('Proporção de classes base amostrada:\n', df_amostra['Class'].value_counts(normalize=True))
print('Proporção de classes base original:\n', df_normalizado['Class'].value_counts(normalize=True))
```

```
Proporção de classes base amostrada:
 Class
0.0    0.99859
1.0    0.00141
Name: proportion, dtype: float64
Proporção de classes base original:
 Class
0.0    0.998333
1.0    0.001667
```

Fonte: próprio autor.

Dessa forma, o Modelo 4 foi construindo com os seguintes hiperparâmetros:

Código em Python com Modelo 4

```
with pm.Model() as model_4:
    l = pm.HalfNormal("l0", 1)
    cov = pm.gp.cov.Matern32(3, ls=l)
    gp = pm.gp.Latent(cov_func=cov)

    s = pm.HalfNormal("s0", sigma=1, shape=3)

    l1 = pm.HalfNormal("l1", 1)
    cov1 = pm.gp.cov.Matern32(3, ls=l1)
    gp1 = pm.gp.Latent(cov_func=cov1)

    f = gp.prior("f", X=X_train_amostra.values)
    g = gp1.prior("g", X=X_train_amostra.values)

    fs = pm.math.stack([f, g], axis=1)
    j = pm.Deterministic("j", fs)
    p = pm.Deterministic("p", pm.math.softmax(fs,axis=1))

    y_obs = pm.Categorical("y_obs", p=p, observed=y_train_amostra)
```

Fonte: próprio autor.

Por fim, o Processo Gaussiano Classificador atigiu os seguintes valores de avaliação.

Acurácia	0.999
Precisão	0.995
Recall	0.998
F1-Score	0.999

Com isso, é possível concluir que o Modelo 4, apesar dos desafios do desbalanceamento da base e dos poucos dados amostrais oferecidos ao treinamento, foi capaz de aprender o padrão dos dados e se saiu muito bem na predição de registros de transações fraudulentas.

4 Considerações Finais e Próximos Desafios

Este estudo abordou diferentes estratégias para a classificação de transações fraudulentas em bases desbalanceadas, utilizando uma variedade de modelos e técnicas, como a regressão logística, LASSO, SMOTE e Processos Gaussianos. Os resultados indicaram que, enquanto a regressão logística simples oferece uma boa acurácia, ela é insuficiente para lidar com o desbalanceamento das classes, pois outras métricas, como precisão, recall e F1-Score, são comprometidas. A inclusão de técnicas de regularização e balanceamento de classes, como o LASSO e o SMOTE, contribuiu para a melhoria do modelo, mas ainda apresentou limitações no desempenho, principalmente no que se refere ao recall e F1-Score.

O Processo Gaussiano, por sua vez, se destacou ao lidar com a incerteza nos dados e à sua capacidade de modelar o padrão das transações fraudulentas, mesmo com uma amostra reduzida e a persistência do desbalanceamento das classes. Isso sugere que modelos mais complexos e não paramétricos podem oferecer uma solução eficaz para esse tipo de problema, especialmente quando combinados com técnicas de otimização e regularização.

Entretanto, algumas questões ainda precisam ser endereçadas. Um dos principais desafios observados foi o desbalanceamento das classes, que ainda afeta significativamente as métricas de avaliação, mesmo com a aplicação de técnicas como SMOTE. Outra dificuldade foi a necessidade de reduzir a amostra para o uso de Processos Gaussianos, o que

pode comprometer a generalização do modelo. Além disso, a interpretação dos modelos, especialmente os não paramétricos, representa um desafio, já que eles oferecem menos transparência do que modelos tradicionais.

Referências

ABRIL-PLA, Oriol et al. PyMC: a modern, and comprehensive probabilistic programming framework in python. Peerj Computer Science. Boston, p. 2-25. 6 jul. 2023.

ANDRÉ LUIZ CAMPOS DAS NEVES RIBEIRO (Mato Grosso). Oab/Mt - Comissão de Direito Bancário. Fraudes Bancárias. 2023. Cartilha. Disponível em: https://www.oabmt.org.br/Admin2/Arquivos/Documentos/202305/PDF57753.pdf. Acesso em: 14 nov. 2023.

AVELINO, Caio. Quando normalizar ou padronizar seus dados em Machine Learning? 2024. Disponível em: https://www.datadrivenschool.com/blog/quando-normalizar-ou-padronizar-seus-dados. Acesso em: 16 out. 2024.

BARBER, D. Bayesian reasoning and machine learning. [S.l.]: Cambridge University Press, 2012.

BELFIORE, P. Estatística: aplicada a administração, contabilidade e economia com Excel e SPSS. 1. ed. Rio de Janeiro: Elsevier, 2015.

“Banco Central do Brasil publica Resolução Conjunta sobre compartilhamento de dados relativos a indícios de fraudes”. 2023. Elaborada pela Redação. Disponível em: https://www.mattosfilho.com.br/unico/resolucao-dados-fraudes/. Acesso em: 14 nov. 2023.

CHAWLA, Nitesh V.; BOWYER, Kevin W.; HALL, Lawrence O.; KEGELMEYER, W. Philip. SMOTE: synthetic minority over-sampling technique. Journal Of Artificial Intelligence Research. El Segundo, p. 321-357. jun. 2002.

COODESH. O que é biblioteca? 2023. Disponível em: https://coodesh.com/blog/dicionario/o-que-e-biblioteca/. Acesso em: 05 dez. 2024.

FIGUEIREDO, Alcio Manoel de Sousa. Cartão de Crédito: questões controvertidas. Curitiba: Juruá, 2003.

FIGUEIREDO FILHO, Dalson Britto; SILVA JÚNIOR, José Alexandre da. Desvendando os Mistérios do Coeficiente de Correlação de Pearson. Revista Política Hoje, Pernambuco, v. 18, n. 1, p. 115-146, dez. 2010. Disponível em: https://periodicos.ufpe.br/revistas/index.php/politicahoje/article/view/3848/3152. Acesso em: 28 nov. 2024.

GONZALEZ, Leandro de Azevedo. Regressão Logística e suas Aplicações. 2018. 46 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2018. Disponível em: https://monografias.ufma.br/jspui/bitstream/123456789/3572/1/LEANDRO-GONZALEZ.pdf. Acesso em: 05 jun. 2024.

KINGMA, Diederik P.; WELLING, Max. An Introduction to Variational Autoencoders. Boston: Now The Essence Knowledge, 2019. 102 p.

MARIO FILHO. O Que É Acurácia Em Machine Learning? 2023. Disponível em: https://mariofilho.com/o-que-e-acuracia-em-machine-learning/. Acesso em: 07 jun. 2024.

MARIO FILHO. Precisão, Recall e F1 Score Em Machine Learning. 2023. Disponível em: https://mariofilho.com/precisao-recall-e-f1-score-em-machine-learning/. Acesso em: 07 jun. 2024.

MENEZES, Nilo Ney Coutinho. Introdução a Programação com Python: algoritmos e lógica de programação para iniciantes. 3. ed. São Paulo: Novatec, 2019. 326 p.

MÉTODO DE SIMULAÇÃO E ESCOLHA DE FATORES NA ANÁLISE DOS PRINCIPAIS COMPONENTES. São Paulo: Faculdade de Saúde Pública da USP., v. 32, 1998. Disponível em: https://www.scielo.br/j/rsp/a/zWdnX5bLGcwXGkcyjQjSG6C/?format=pdflang=pt.

Acesso em: 05 jun. 2024.

MORETTIN, Pedro A.; SINGER, Julio M. Estatística e Ciência de Dados. São Paulo: Ltc, 2022. 464 p.

NEOWAY (ed.). Conheça as principais fraudes bancárias e as responsabilidades das instituições financeiras quando ocorrem. 2023. Disponível em: https://blog.neoway.com.br/fraude-bancaria/. Acesso em: 06 maio 2024.

PACHECO JUNIOR, João Carlos. Modelos para Detecção de Fraudes Utilizando Técnicas de Aprendizado de Máquina. 2019. 103 f. Dissertação (Mestrado) - Curso de Economia, Fundação Getulio Vargas, São Paulo, 2019. Disponível em: https://repositorio.fgv.br/server/api/core/bitstreams/b4d9367d-712f-496e-8b15-ecb349671723/content. Acesso em: 06 maio 2024.

POLLI, Démerson André. Introdução à Inferência Bayesiana. Brasil, 2019. 75 slides, color. Disponível em: https://repositorio.enap.gov.br/bitstream/1/4765/2/Aulas%201%20a%203%20-%20Polli-bayesian-econometrics.pdf. Acesso em: 06 maio 2024.

RAUBER, T. W. Redes neurais artificiais. Universidade Federal do Espírito Santo, p. 29, 2005.

SAPRE, Atharva; VARTAK, Shubham. Scientific Computing and Data Analysis using NumPy and Pandas. International Research Journal Of Engineering And Technology (Irjet). Nova Delhi, p. 1334-1346. dez. 2020.

SCHULZ, Eric; SPEEKENBRINK, Maarten; KRAUSE, Andreas. A tutorial on Gaussian process regression: modelling, exploring, and exploiting functions. Journal Of Mathematical Psychology. Oldemburgo, p. 1-16. ago. 2018.