AI e Datalab
Relatório - Grupo Bari

Augusto Antonio Kolb Schiavini

Fevereiro 2026

Materiais: Colab — Dataset

1 Introdução

Esse trabalho consiste na elaboração de uma análise exploratória de dados junto a um modelo de Machine Learning. Para isso foi disponibilizada uma base de dados fictícia com características como renda mensal, score de crédito, valor financiado, idade e outras. Primeiro faço exploração dos dados para conseguir insights significativos (2). Em seguida descrevo o modelo que fiz (4) por fim prescrevo medidas interessantes que o banco pode tomar em relação às conclusões que cheguei (8).

Para este trabalho utilizei Python, SQL tal como bibliotecas específicas como Pytorch, Pandas, MatplotLib, Numpy e Sklearn.

2 Análise Exploratória de Dados

Meu approach para encarar os dados consistiu de analisar suas relações passo a passo, partindo de informações isoladas em direção a informações mais abrangentes, que considerassem mais de uma variável.

Os primeiros passos foram:

1. Baixar o arquivo como csv;
2. Formatar coisas menores (retirar a classe clienteId, transformar as colunas de valor monetário em algo compreensível);
3. Remover quaisquer dados incongruentes (dados negativos ou por exemplo valor_financiado ¿ valor_imovel);

3 Gráficos

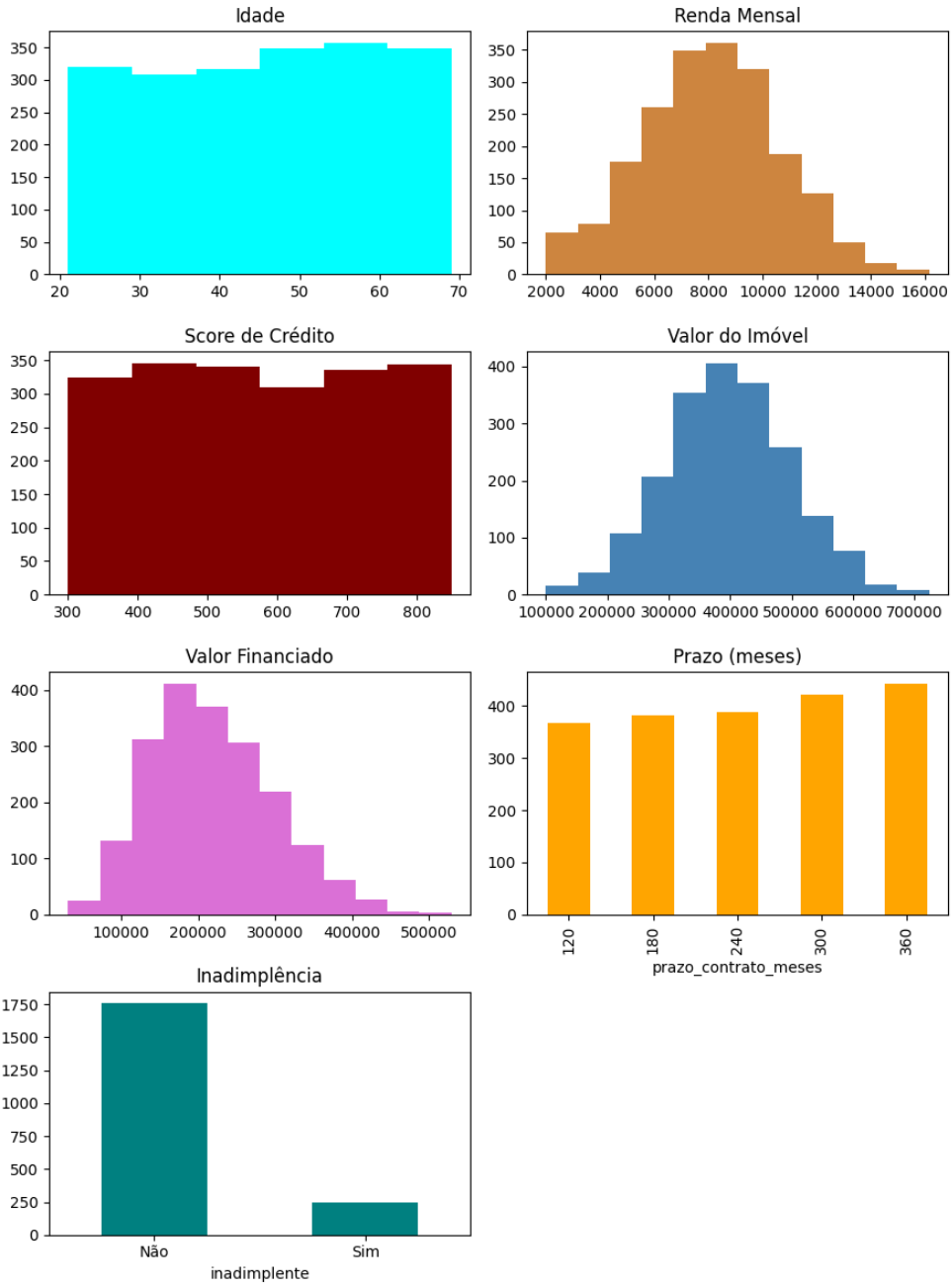
3.1 Métricas básicas

Primeiro, fui atrás de descobrir as métricas de estatística descritiva:

```
   media_idade  media_renda_mensal  media_score_credito  media_valor_imovel  \
0         45.0              8106.0                576.0            395944.0

   media_prazo_contrato_meses  media_valor_financiado  media_inadimplente
0                       246.0                218081.0                 0.0
Desvio Padrão
idade                          14.044942
renda_mensal                 2557.428903
score_credito                 159.054475
valor_imovel                101424.071322
prazo_contrato_meses           85.098463
valor_financiado            80721.383826
inadimplente                    0.326209
dtype: float64
```

Os dados mostram um público de equilibrado com renda sólida, mas com uma variação considerável entre os indivíduos. O poder aquisitivo é relevante, o que atrelado com um score de crédito apenas "regular"demanda uma certa atenção à inadimplência.

3.2 Histogramas



Aqui é interessante que, com exceção da Inadimplência, todas as variáveis se comportam de uma maneira previsível. A idade, os prazos e o score de crédito são heterogêneos, indicando não só um público bem amplo como também um histórico bem amplo.

Já a renda mensal, o valor do imóvel e o valor financiado se comportam como curvas de sino. Os valores extremos são bem raros, e em específico no valor financiado há uma inclinação a esquerda, indicando (junto com o prazo) que pessoas preferem empréstimos de valores menores mas com mais parcelas.

Por fim, vale resaltar que há um número até considerável de inadimplentes, dado o tamanho reduzido da base de dados. Devemos então olhar como as outras informações se relacionam com a inadimplência.

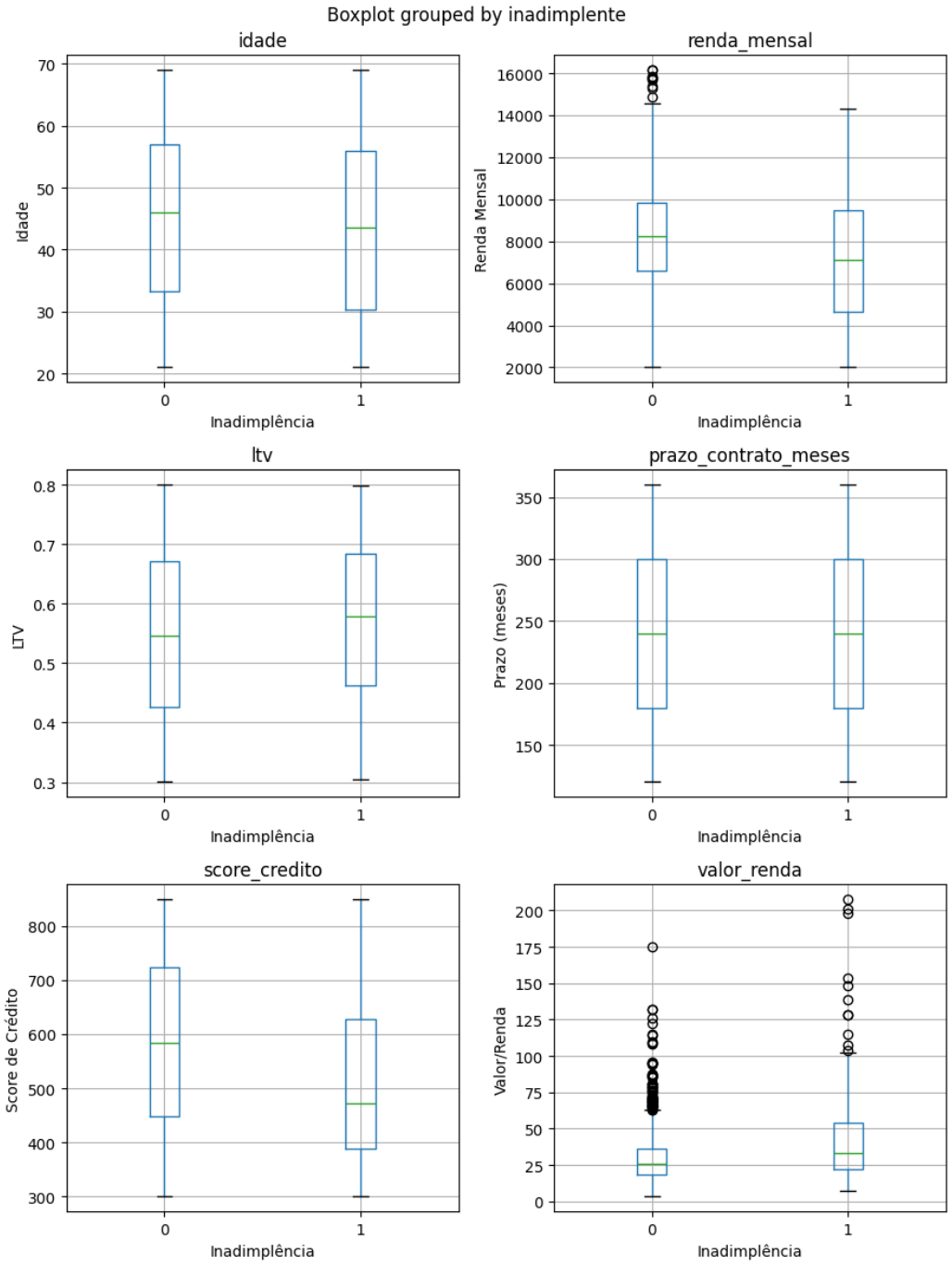
3.2.1 Insights

1. Pouquíssimos extremos;
2. Distribuição heterogênea;
3. Pessoas preferem empréstimos de valores menores com mais parcelas.

3.3 Boxplots

Antes vimos a distribuição geral das variáveis. Agora com o Boxplot podemos ver a dispersão, os outliers e como certas variáveis se relacionam com a Inadimplência.

Para isso, transformei as variáveis valor_financiado e valor_imovel em ltv (Loan-To-Value, a razão entre o valor financiado e o que deve ser pago). Criei também valor_renda, a razão entre o valor do imóvel e a renda do cliente, de forma que sabemos quantos aportes ele precisa para pagar tudo.



Com os Boxplots, já temos uma noção melhor das coisas. Começando pela

renda mensal, vemos que há muitos outliers adimplentes, talvez indicando que pessoas com alta renda dificilmente caem em inadimplência. Além disso é perceptível que a caixa de Inadimplentes é se alonga para baixo, isto é, a inadimplência se concentra em quem ganha menos.

O mais assustador é o score de crédito. A caixa da inadimplência é bem projetada para baixo, tanto seus quartis até sua mediana. Um score abaixo de 500 já um indicador forte de inadimplência.

No valor_renda vemos que há uma dispersão maior de outliers na inadimplência: quem financia imóveis muito mais caros que sua renda pode cair na inadimplência.

No ltv, quem dá entradas menores de financiamento tende a ser ligeiramnte mais inadimplente.

Por fim, questões como idade, ltv e prazo ficaram razoavelmente estáveis.

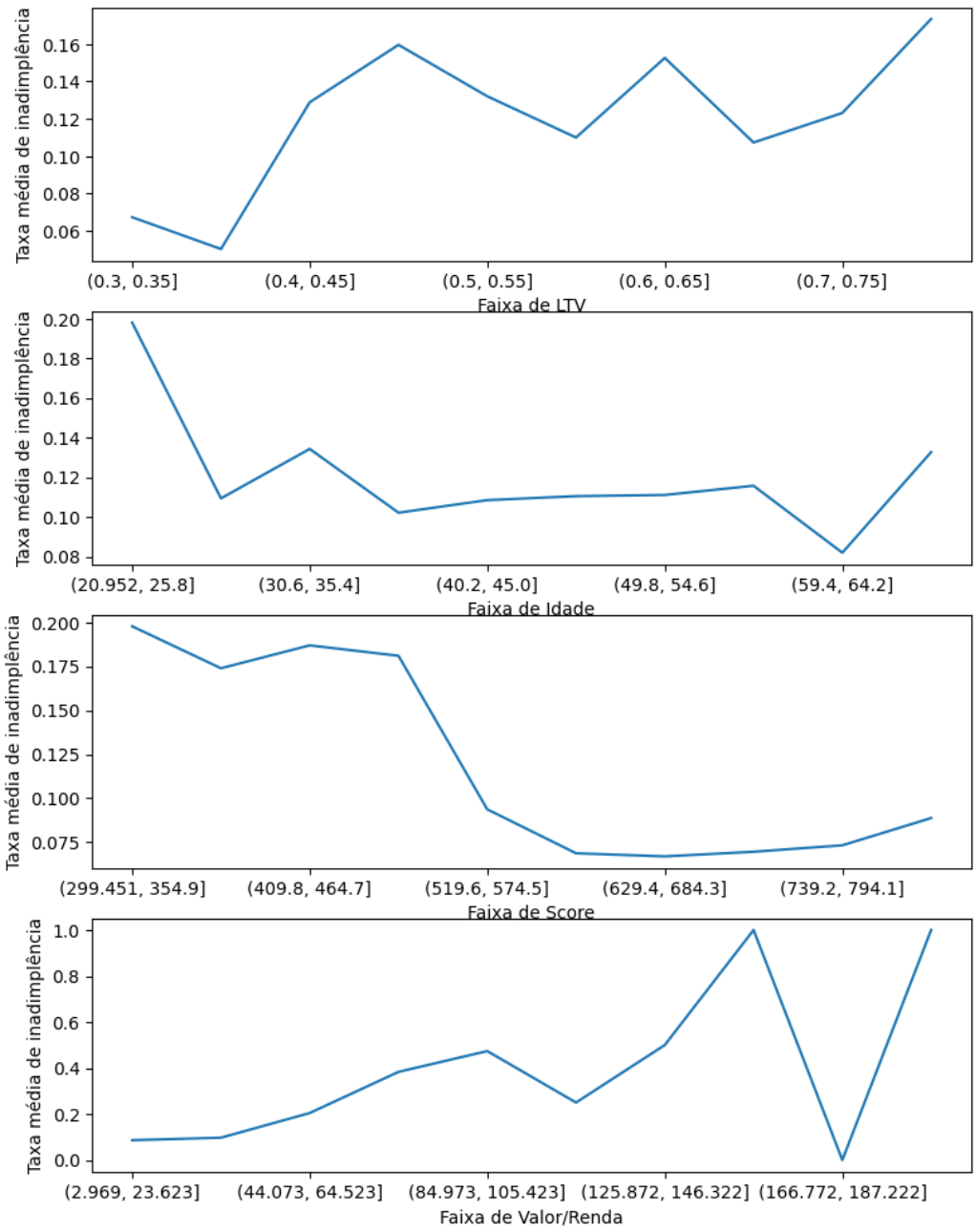
3.3.1 Insights

1. Mediana de Adimplentes perto de 600 enquanto Inadimplentes perto de 500;

2. Entradas menores = mais inadimplência;

3.4 Faixas por Inadimplência

Com base nos Boxplots, decide então mensurar as variáveis que afetaram a inadimplência em gráficos de faixa para ver seu comportamento.



O gráfico confirma que quanto menos dinheiro o cliente dá de entrada, maior o risco. Clientes com LTV baixo (0.3 a 0.35) têm risco de apenas 6%.

Clientes que financiam mais de 75% do valor do imóvel (LTV ¿ 0.75) veem o risco subir para mais de 17%.

Já a idade, que antes não parecia mudar muito, agora da sinais mais claros: há um pico de inadimplência na faixa de 21-25 anos, caindo pela metade por volta dos 30 anos em diante.

O score continua sendo uma variável importantíssima, e agora temos mais detalhes: clientes com score abaixo de 464 tem 18% a 20% de inadimplência, valor que cai mais da metade a partir de 519, e chegando a 8% a partir de 574.

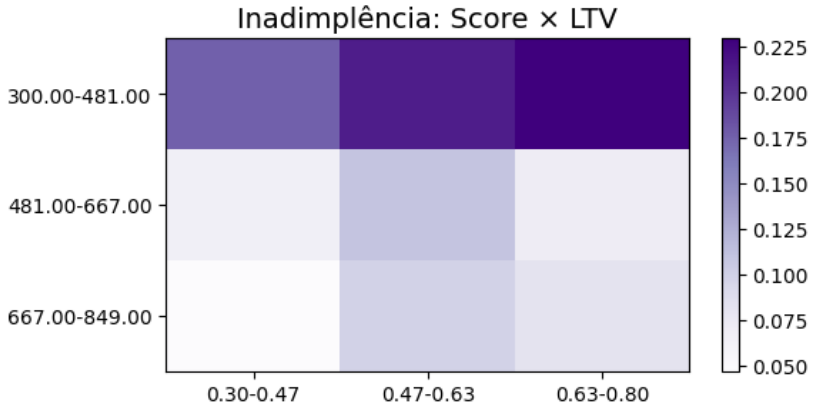
Por fim, vemos que a inadimplência cresce de forma consistente quanto mais cresce a razão valor/renda, reforçando a ideia de que há um risco cada vez maior a quanto mais a pessoa tem dificuldade de pagar o imóvel com sua renda. Há picos enormes no fim do gráfico que, apesar de representarem outliers com valores extremos, demonstram um risco quase certo que deve ser levado em conta.

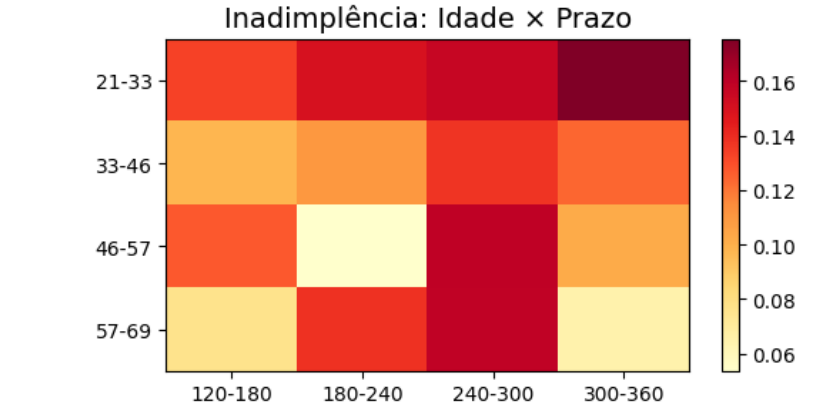
3.4.1 Insights

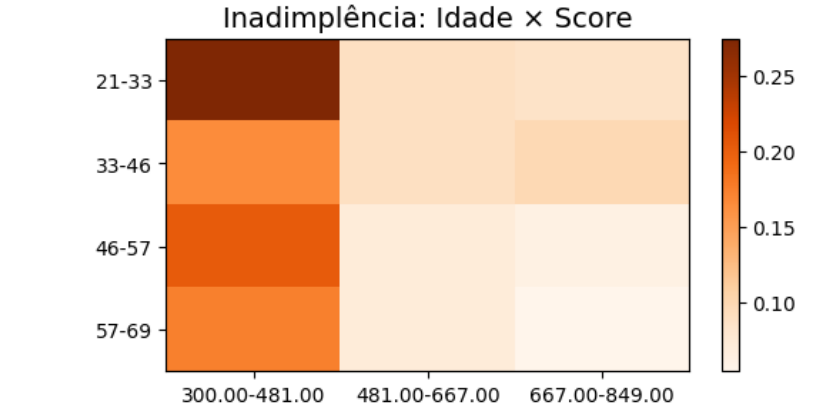
1. Score mínimo bastante seguro: 574;
2. Inadimplência dobra para scores abaixo de 519;
3. Maior relação valor/renda = mais Inadimplente

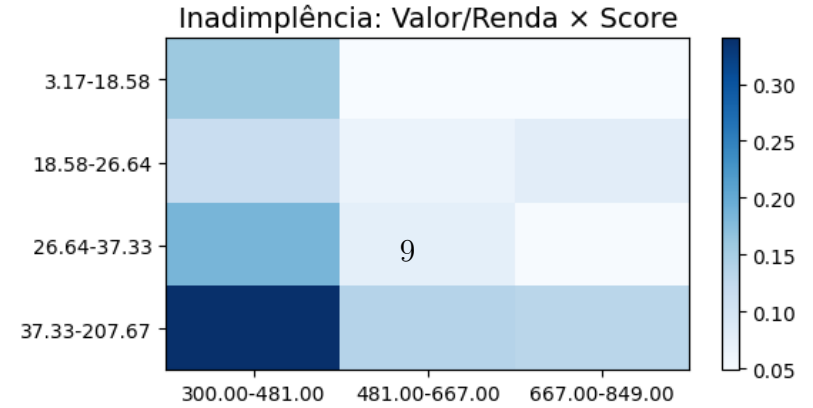
3.5 Heatmaps

Depois de relacionar variáveis específicas com a inadimplência, resta saber como essas variáveis se comportam junto a outras variáveis para conseguir insights melhores. Para isso usei heatmaps:


Inadimplência: Score × LTV
300.00-481.00
481.00-667.00
667.00-849.00
0.30-0.47
0.47-0.63
0.63-0.80
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050


Inadimplência: Idade × Prazo
21-33
33-46
46-57
57-69
120-180
180-240
240-300
300-360
0.16
0.14
0.12
0.10
0.08
0.06


Inadimplência: Idade × Score
21-33
33-46
46-57
57-69
300.00-481.00
481.00-667.00
667.00-849.00
0.25
0.20
0.15
0.10


Inadimplência: Valor/Renda × Score
3.17-18.58
18.58-26.64
26.64-37.33
37.33-207.67
300.00-481.00
481.00-667.00
667.00-849.00
0.30
0.25
0.20
0.15
0.10
0.05

Começando pelo heatmap Score x LTV temos o seguinte: Um score muito baixo (481 ou menos) corre um risco de pelo menos 15% de inadimplência independentemente do quão baixo for o ltv. No sentido oposto, um score alto (667 ou mais) entregará no máximo 8-9% de risco. Interessante notar também que para scores altos e médios (481 ou mais) o risco é menor para um ltv ou grande (0.63 ou mais) ou pequeno (0.47 ou menos) enquanto o meio é ligeiramente mais arriscado. Se antes haviamos colocado uma linha de score segura de 574, com um ltv bem alto ou bem baixo pode-se haver algum negócio se o score for pelo menos 481.

Sobre o heatmap de Idade x Prazo: O gráfico é consideravelmente caótico. O insight mais perceptível é que não é vantajoso emprétimos muito longos para pessoas muito jovens. O risco chega a 16

Sobre o heatmap Idade x Score vemos que independentemente da idade, um score baixo (481 ou menos) é bem arriscado (pelo menos 16%). Quando passamos para um score médio para cima, vemos que para pessoas até 46 anos o risco chega a no máximo 10% e diminui bastante o quão mais velho a pessoa fica. Uma pessoa de até 46 anos com um score médio pode ser um bom cliente se seu ltv for baixo.

Por fim sobre Valor/Renda x Score temos que o score novamente força que tensiona a inadimplência. Um score baixo tem um risco de pelo menos 14% qualquer que seja a relação valor do imóvel com renda. Agora para scores médios para cima, a inadimplência não passa de 8% para financiamento de até 37 vezes, com um subida até que significativa do risco quando passa desse valor. Isso reforça o seguinte ponto: cliente com scores medianos demonstram baixo risco se tiverem um ltv alto ou uma relação valor/renda aceitável (menor que 37 vezes).

3.5.1 Insights

1. Score médio: 481 - 574;
2. Score médio com ltv bem baixo ou bem alto ou Valor/renda até 37 não

passa de 8% de inadimplência.

3. 10% risco máximo para pessoas de até 46 anos com score médio

4 Modelo de Predição de Inadimplência

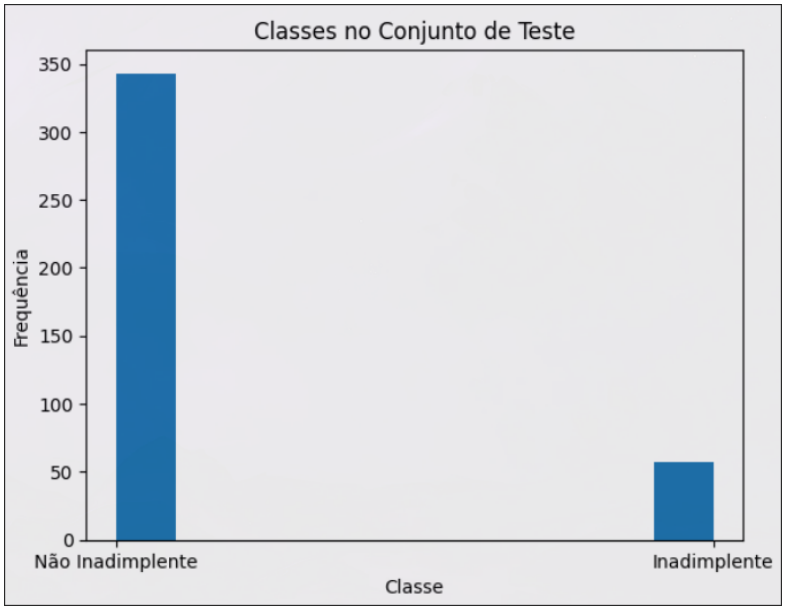
Foi solicitado um modelo preditivo (Machine Learning) capaz de prever se um cliente terá o comportamento de pagamento pontural ou inadimplente com base nos dados entregues.

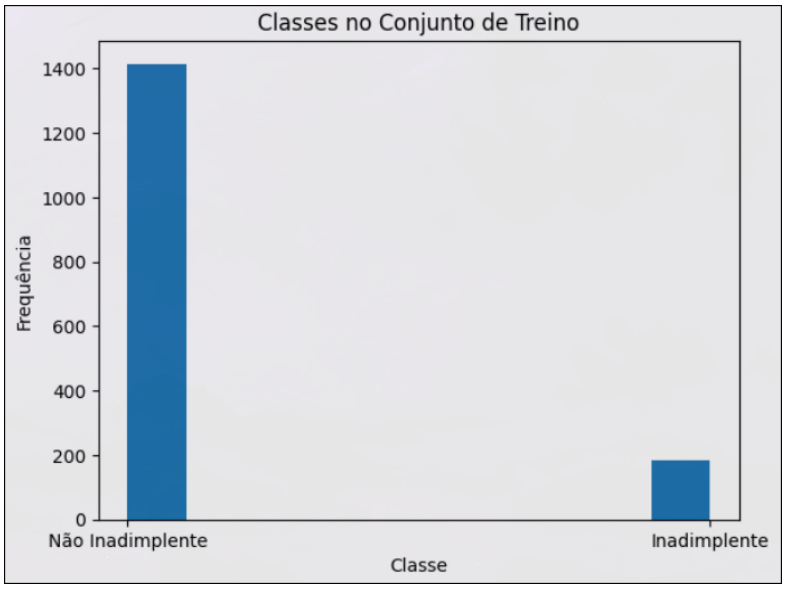
Para resolver esse problema, montei uma Rede Neural MLP (Multilayer Perceptron) em Python e uma rodada de treinamento e avaliação. O fluxo da rodada é o seguinte: filtramos os dados, dividimos a base em treino e teste e inserimos os dados na rede neural. As seções 5, 6 e 7 tratam dos pormenores da implementação, dos desafios encontrados e soluções tomadas.

5 Base de Dados

De cara, fiz o seguinte para tratar os dados:

1. Pente fino para ver se não havia nenhum registro com NULL;
2. Remoção da coluna cliente_Id, dado que não serve;
3. Dividi a base em 80% para treino e 20% para teste;
4. Normalizei a base de treino com o StandardScalar para ajudar o modelo (números de magnitudes muito distintas pioram a performance);
5. Coloquei os dados em batches de 64 para acelerar o treinamento do modelo.





5.1 Desbalanceamento

Quanto fiz esses gráficos já percebi que havia algo de errado no dataset: a classe 0 (Não Inadimplente) tem muito mais amostras que a classe

1(Inadimplente). Sei por experiência que isso gera distorções na avaliação do modelo, mas resolvi continuar dessa forma por uma questão de didática. As consequências desse problema e as maneiras que encontrei para resolver se encontram na seção 6.

6 Rede Neural e Treinamento

```
class MLP(torch.nn.Module):
  def __init__(self, input_size, hidden_sizes, output_size):
    super(MLP, self).__init__()

    #Registra as camadas numa lista para rastrear melhor
    #os parâmetros
    self.layers = torch.nn.ModuleList()

    in_size = input_size
    for hi_size in hidden_sizes:
      self.layers.append(torch.nn.Linear(in_size, hi_size))
      self.layers.append(torch.nn.ReLU()) #ReLU é eficiente
      #para classificação binária
      #self.layers.append(nn.Dropout(0.3)) #Desligando
      #alguns neurônios para evitar
      #Overfitting:Não foi necessário
      in_size = hi_size

    self.output_layer = torch.nn.Linear(in_size, output_size)

  def forward(self, x):
    for layer in self.layers:
      x = layer(x)

    out = self.output_layer(x)
    return out
```

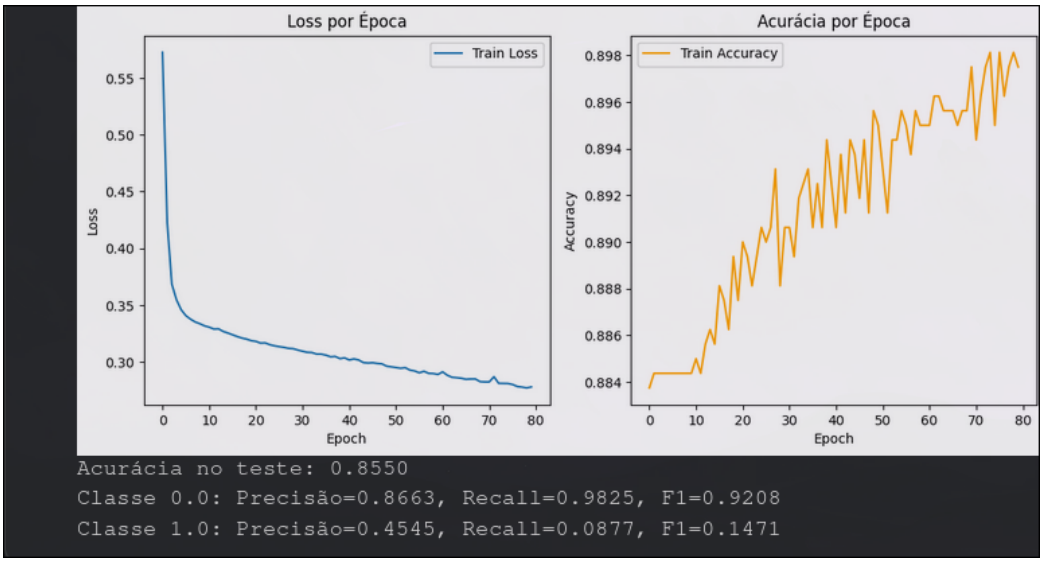
Esta é a rede neural para classificação binária. Definimos por parâmetros quantas variáveis vamos usar, quantas camadas de neurônios serão necessárias (e quantos neurônios em cada camada) e a saída(0 — 1). A arquitetura da rede se encarrega de organizar as camadas de neurônios em uma lista e aplica

as funções matemáticas necessárias em cada. Entre as camadas internas usa-se a função ReLU, muito comum em Machine Learning para classificação binária.

Além disso para o treinamento utilizamos:

1. Estrutura de [128,64,32] a [256, 128, 64, 32] neurônios a depender do volume de dados (ver em 6.4);
2. Função de Loss BCEwithLogitsLoss que "ensina"a rede ao ajustar os pesos por meio do backpropagation. Além disso aplica uma função Sigmoid na saída da rede, comum para classificação de 0 ou 1;
3. Taxa de Aprendizado de 0,001: A mais comum e estável possível;
4. Weight Decay de 1e-3, que ajuda a generalizar o modelo;
5. 100 Épocas.

6.1 Primeiro resultado



A curva de Loss está descendo corretamente e a Acurácia do treino subiu apesar das suas oscilações, o o que ajudou numa acurácia de teste de 0.8550. Porém essa acurácia é enganosa. Perceba que o Recall da classe 1 está 0.087. Um Recall tão baixo e uma precisão mediana como essa (o F1 é a relação dos dois! E também é baixo por isso) indicam que o modelo identificou pouquíssimos casos como classe 1, ou seja, ele não aprendeu a reconhecê-la. A acurácia está está alta porque o modelo aprendeu que pode chutar a classe 0 e acertar facilmente, pois a esmagadora maioria dos casos pertence a essa classe.

6.1.1 Ajuda Externa: Outro Modelo

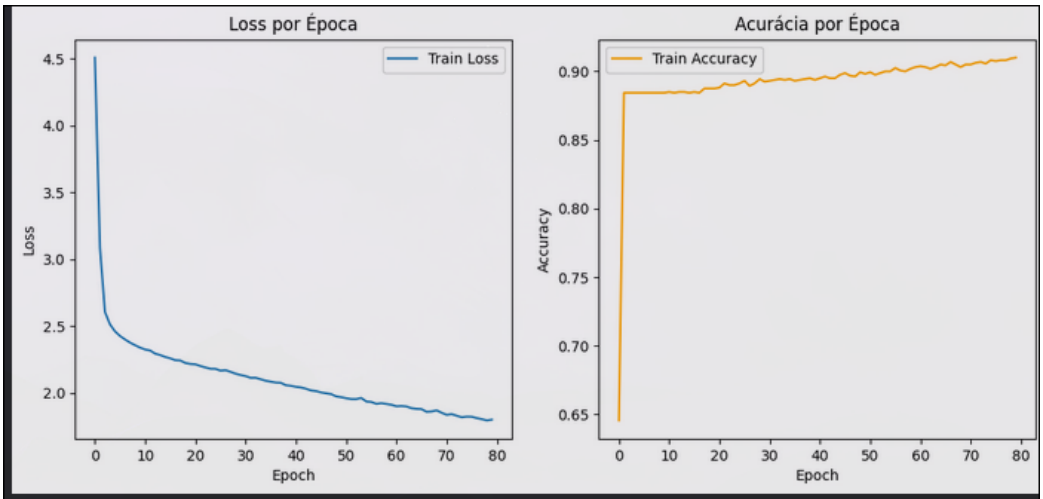
```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

rf = RandomForestClassifier(class_weight='balanced', random_state=42)
rf.fit(X_train_scaled, y_train)
print(classification_report(y_test, rf.predict(X_test_scaled)))
```

```
              precision    recall  f1-score   support

           0       0.86      0.99      0.92       343
           1       0.29      0.04      0.06        57

    accuracy                           0.85       400
   macro avg       0.57      0.51      0.49       400
weighted avg       0.78      0.85      0.80       400
```

Nesse sentido, decidi testar três estratégias para alterar essa desproporção entre as classes.

6.2 Weight Balancing

Weight Balancing consiste justamente de alertar o modelo de que uma classe é minoritária, e que portanto ao errar a predição dessa classe ele deve ajustar seus pesos de forma mais severa. Isso é feito colocando uma variável pos_weight na função de Loss que indica a desproporção entre as duas classes (no nosso caso 7 vezes). E os resultados foram:



E para o RandomForest:

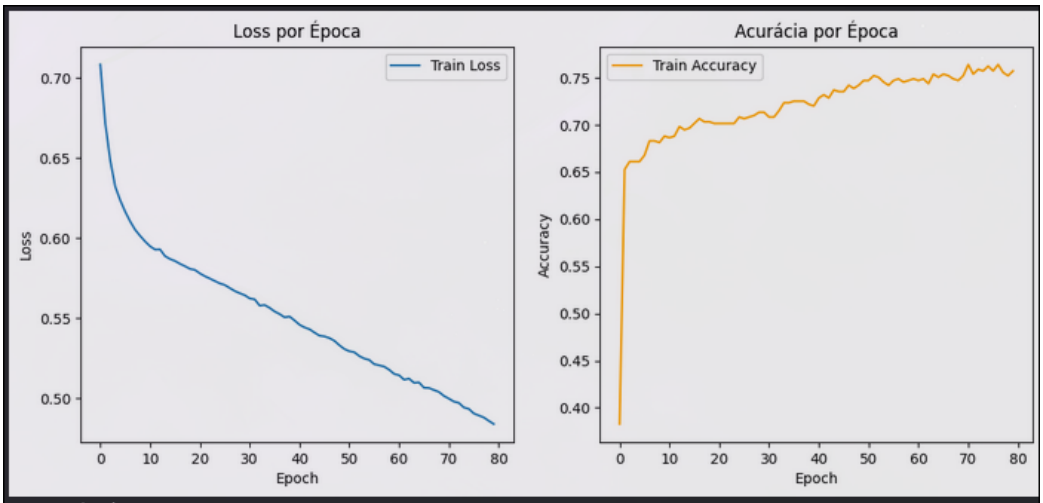
	precision	recall	f1-score	support
0	0.86	0.99	0.92	343
1	0.20	0.02	0.03	57
accuracy			0.85	400
macro avg	0.53	0.50	0.48	400
weighted avg	0.76	0.85	0.79	400

Em termos práticos, o Weight Balencing não surtiu nenhum efeito signifi-

cativo. Tentei reduzir e aumentar o pos_weight mas isso não trouxe nenhuma mudança vantajosa.

6.3 UnderSampling

O UnderSampling consiste em reduzir a quantidade de casos que sejam classe maioritária para que o modelo não "vicie"nas suas qualidades e se torne incapaz de identificar outras classes. Reduzi a classe 0 de 1400 para 500, enquanto a classe 1 seguia com 200.




```
Acurácia no teste: 0.7315
Classe 0.0: Precisão=0.7982, Recall=0.8426, F1=0.8198
Classe 1.0: Precisão=0.5143, Recall=0.4390, F1=0.4737
```

E para o RandomForest:

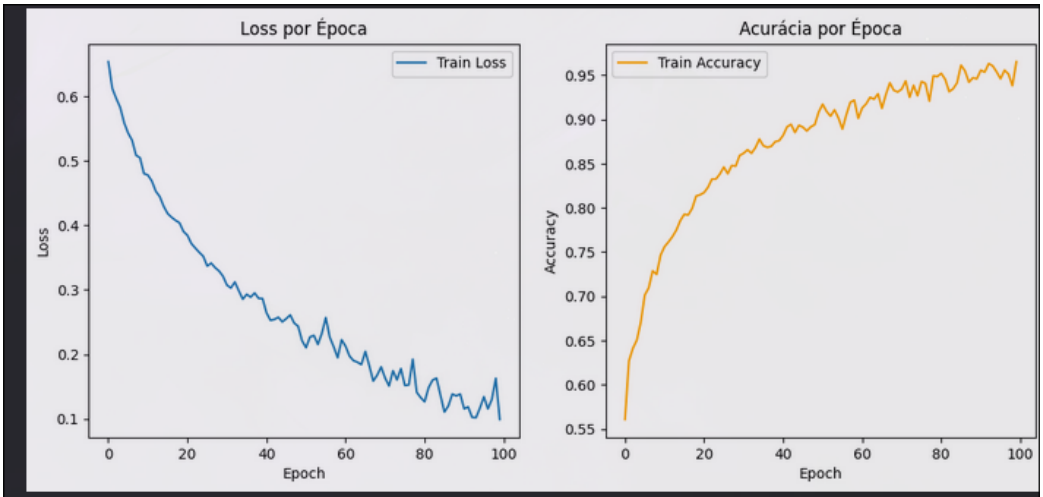
	precision	recall	f1-score	support
0	0.80	0.86	0.83	108
1	0.55	0.44	0.49	41
accuracy			0.74	149
macro avg	0.67	0.65	0.66	149
weighted avg	0.73	0.74	0.74	149

Agora temos algo interessante: o Recall da classe 1 subiu bastante, o que indica que o modelo agora consegue identificar a classe. E essa acurácia de 0.73 parece mais confiável. Reduzi a classe 0 para outros valores como 300, 400 e 800 mas nenhum desses casos produziu um desempenho como esse.

A desvantagem de usar UnderSampling é que reduzimos muito os dados disponíveis para treinar o modelo, o que por um lado ajuda a rede a entender certas diferenças entre as classes, mas não todas. Se tivéssemos mais dados, o modelo certamente teria uma performance melhor.

6.4 OverSampling

Por fim, decidi fazer o oposto: O Oversampling consiste em aumentar a quantidade de exemplares da classe minoritária. Quem faz isso é o SMOTE, um algoritmo que cria exemplos sintéticos a partir da classe minoritária, principalmente por interpolação com vizinhos próximos. O intuito do Oversampling é garantir que as duas classes tenham um bom número de exemplares para que o modelo possa aprender e classificar. Dado que estou lidando com mais dados, aumentei a rede neural de [128, 64, 32] camadas de neurônios para [264, 128, 64, 32] camadas.




```
Acurácia no teste: 0.8366
Classe 0.0: Precisão=0.8906, Recall=0.7876, F1=0.8359
Classe 1.0: Precisão=0.7893, Recall=0.8916, F1=0.8373
```

E para o RandomForest:

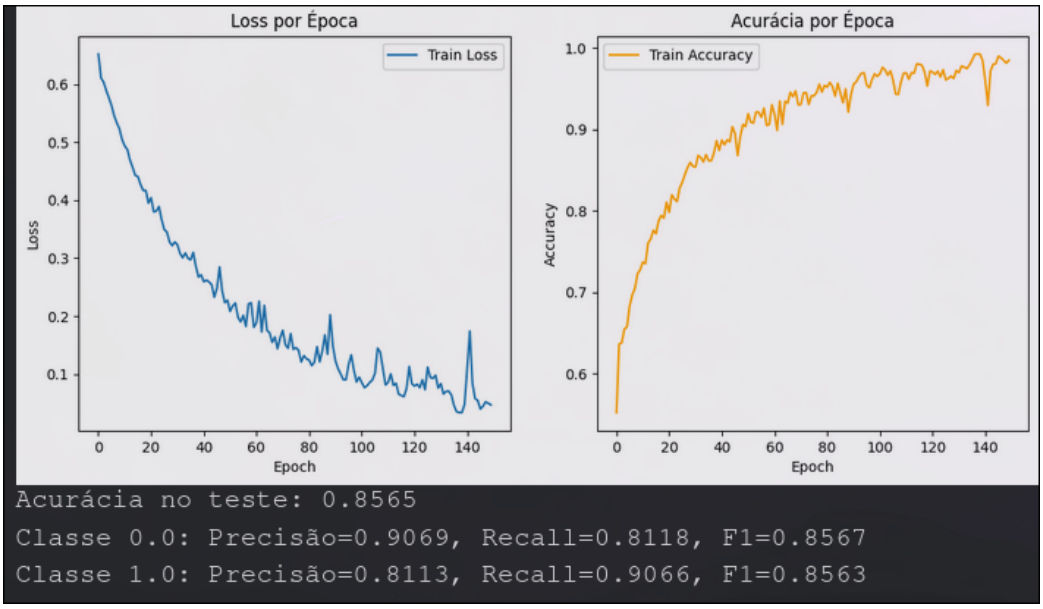
	precision	recall	f1-score	support
0	0.91	0.94	0.93	372
1	0.93	0.89	0.91	332
accuracy			0.92	704
macro avg	0.92	0.92	0.92	704
weighted avg	0.92	0.92	0.92	704

6.4.1 Por que não SMOTE sempre?

Eu poderia muito bem simplesmente ter feito Oversampling com o SMOTE e citar apenas o seu resultado, mas acho que há uma questão nisso: o SMOTE cria dados que não existem, e cabe a instiuição do banco dizer se isso é cabíbel ou não. Por essa razão dei outros caminhos para melhorar o modelo.

6.5 Bônus: Ensemble de Modelos

Por fim, peguei esse modelo com o SMOTE aplicado e fiz um Ensemble com o RandomForest. Basicamente, na fase de teste pegamos a saída probabilística dos dois modelos e fazemos uma média, de forma que possamos aproveitar nuances de cada modelo e melhorar o nosso.



7 Resultados práticos

Pegando o último modelo, temos algumas considerações importantes:

1. Forte para triagem inicial: A acurácia em 0.85 e F1 de 0.85 para ambas as classes demonstra que o modelo pode muito bem funcionar como um filtro automático, poupando trabalho humano e diminuindo o custo operacional. É um bom alicerce também para uma análise de um profissional experiente.

2. Bom para encontrar os inadimplentes: De todos os inadimplentes da

amostra, o modelo é capaz de acertar 90%(Recall). Isso pode evitar muitas concessões indevidas de crédito.

3. Janela de oportunidade: A precisão da classe de Inadimplentes é um pouco menor (0.81). Porém, com auxílio de outras ferramentas podemos encontrar supostos "Inadimplentes"que na verdade conseguem pagar suas concessões de crédito.

8 Prescrições

Com base nas análises e no modelo preditivo, o banco pode adotar o modelo como triagem inicial num processo automatizado. Caso o modelo classifique o cliente como Adimplente, é suficiente checar se seu score de crédito está dentro da linha do aceitável (481 para cima).

Caso dê negativo, pode-se analisar o score de crédito junto ao ltv, a relação valor/renda e idade. Assumindo um score médio (pois baixo seria negado na hora), pode-se utilizar mais de um valor fora da faixa desejada como critério de verdade: tiver mais de 47 anos, ou tiver valor/renda maior que 37 vezes ou ltv mediano. Cumprindo pelo menos duas variáveis, dá pra considerá-lo de alto risco de forma automática. Caso dê apenas um, é possível encaminhar os dados do cliente para uma área de precificação de risco, alterando coisas como prazos, quantidade de crédito, solicitação de entradas maiores e outras.