**ML: Credit Scoring**

A Cadeia de crédito é composta por 4 pilares: **prospecção**, **concessão**, **gestão de risco** e **recuperação**.

Os principais participantes da cadeia de crédito são os **poupadores**, os **tomadores** e as **instituições financeiras** que intermediam essa relação.

Quando um tomador solicita um crédito, a instituição financeira avalia os dados cadastrais e financeiros através de um **modelo de decisão** chamado **Credit Scoring**.

O Credit Scoring **estima a probabilidade** de um tomador se tornar inadimplente, classificando-os como **adimplente** ou **inadimplente**. Essa probabilidade está relacionada ao **risco de crédito**.

O risco de crédito é a probabilidade de a instituição financeira não receber o valor emprestado de volta no prazo acordado. Parte dos juros cobrados está embutida nesse risco.

Portanto, a **concessão de crédito** **está intrinsecamente** **relacionada ao risco**, que é um fator crucial nas decisões das instituições financeiras.

**O que é o Credit Scoring?**

O credit scoring é um **modelo estatístico multivariado** que estima a probabilidade de um cliente ser adimplente ou inadimplente. Do ponto de vista do banco, quando um cliente solicita um crédito, o banco analisa essa solicitação usando um modelo de credit scoring.

Alguns algoritmos comumentes usados para credit scoring são a **regressão logística**, **Random forest**, **Naive Bayes** e **support vector machine**.

A regressão logística é uma boa opção porque tem alto comprometimento com a explicabilidade, permitindo acompanhar bem as variáveis ao longo do tempo, e exige pouco poder computacional, o que é importante dado o tamanho das bases de dados de bancos.

* Como funciona uma solicitação de crédito;
* O que é credit scoring;
* Calcular o total de linhas e colunas;
* Identificar dados duplicados;
* Identificar dados nulos.

Regressão Logística

Base de dados para treino e teste

É importante ter uma variável resposta binária e variáveis independentes conhecidas em nossa base de dados. A variável resposta, chamada default, é analisada junto com variáveis categóricas e discretas.

O primeiro passo é separar a base de dados em duas partes: X, que contém as variáveis independentes, e Y, que é a variável resposta. Utilizamos o método ‘drop’ para criar a base X, excluindo a coluna default.

Em seguida, dividimos as bases X e Y em conjuntos de treino e teste usando a função ‘train\_test\_split’ do Scikit-learn. Definimos o tamanho do conjunto de teste (test\_size) e um estado aleatório (Random\_state) para garantir que a divisão seja replicável. Por fim, verificamos o número de elementos em cada conjunto, confirmando que a divisão foi realizada.

Transformação de categorias

Aprendemos sobre a transformação de variáveis categóricas em valores numéricos para que possamos utilizar a regressão logística. O processo envolve os seguintes passos:

**Identificação de Variáveis Categóricas:** Primeiro identificamos quais colunas da nossa base de dados contêm strings que representam categorias, como a coluna “conta\_corrente”.

**Verificação de Valores Únicos:** Utilizamos o método ‘unique()’ para descobrir quais são os diferentes valores que essa variável pode assumir.

**Criação de um Dicionário:** Em seguida, criamos um dicionário onde cada categoria é mapeada para um número.

**Aplicação do Mapeamento**: Usamos o método ‘map()’ para substituir as strings na coluna pela representação numérica correspondente, transformando assim a variável categórica em uma variável numérica.

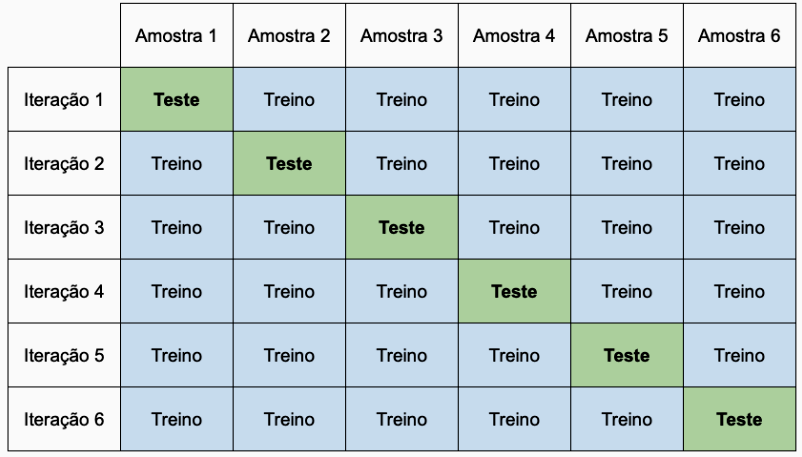
Após fazer a transformação em outras categorias, foi realizado o train test split para dividir os dados em conjuntos de treino e teste, com uma semente definida (SEED = 77) para garantir a reprodutibilidade. Após a divisão, foi impresso o número de elementos em cada base, mostrando que a base de treino tinha 700 elementos e a base de teste, 300.

Por fim, criei o modelo de regressão logística e ajustei aos dados de treino, e a acurácia do modelo foi calculada, resultando em um valor de 0.74, que representa a precisão do modelo em prever corretamente os dados de treino. Essa é a primeira métrica estatística a ser avaliada;

Cross-Validation

O cross-validation é uma técnica para saber quão em o seu modelo está sendo generalizado pelo modelo supervisionado (a regressão logística é um desses modelos) produzido. A separação que fizemos da base em teste já cumpre essa função também. Entretanto o cross-validation cria diferentes amostras da base de dados para treino e assim, podemos ter certeza da performance. A técnica que vamos focar é a K-Fold.

Esta técnica consiste em dividir sua base K vezes, realizar o treino com a maior parte destas amostras e depois testar a última parte, geralmente.

Ex:  
  


No exemplo, a base de dados foi dividida em 6 amostras diferentes e, para cada uma das iterações, o modelo será treinado com 5 partes e a última parte será usara para teste (n\_splits = 5).

O custo operacional será bem maior, de fato, mas para conjuntos de dados limitados é uma ótima saída para ter certeza se o seu modelo está sendo ajustado da maneira que se espera.

* Converter variáveis categóricas em texto para numéricas;
* Separar os dados em variáveis independentes e dependentes, x e y respectivamente;
* Definir um random\_state;
* Dividir os dados em base de treino e de teste;
* Aplicar a regressão logística.

Acurácia de teste

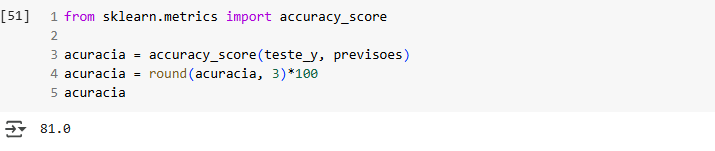
É uma métrica estatística importante para avaliar modelos de classificação. Iniciamos revisando a função .score() e a necessidade de utilizar mais de uma métrica para escolher o modelo para concessão de crédito no ByteBank.

Nos concentramos em como calcular a acurácia de teste usando a função ‘accuracy\_score’ da biblioteca Scikit-learn. Para isso, utilizamos o método ‘.predict()’ do modelo para obter previsões a partir da base de teste. Em seguida, calculamos a acurácia comparando essas previsões com os valores reais da base de teste.

A screenshot of a computer code

Description automatically generated

Por fim, discutimos a formatação do resultado da acurácia, apresentando-a em percentual.



Plotagem Matriz de Confusão

Na aula sobre métricas e estatísticas, foi abordada a matriz de confusão, que é uma ferramenta importante para valiar o desempenho de modelos de classificação. A matriz é composta por quatro quadrantes:

Verdadeiros Positivos (VP): Casos em que o modelo acerta ao classificar um bom pagador como bom pagador.

Falsos Positivos (FP): Casos em que o modelo erra ao classificar um mau pagador como bom pagador.

Falsos Negativos (FN): Casos em que o modelo erra ao classificar um bom pagador como mau pagador.

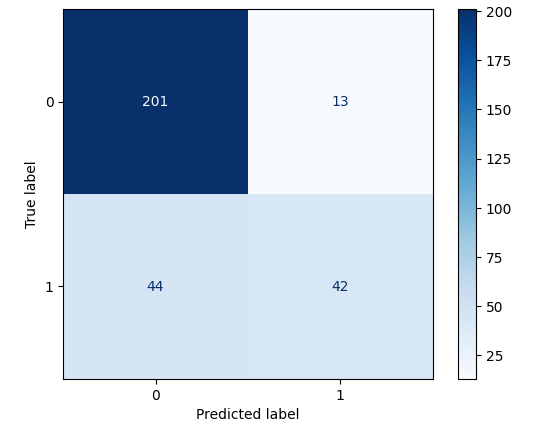
Verdadeiros Negativos (VN): Casos em que o modelo acerta ao classificar um mau pagador como mau pagador.

No projeto, temos:

Verdadeiros Positivos: 201; Falso Positivo: 13; Falso Negativo: 44; Verdadeiros Negativos: 42

A close-up of a computer screen

Description automatically generated



Isso significa que o nosso modelo teve 201 + 42 (VP + VN) = 243 predições corretas e 44 + 13 (FP + FN) = 57 predições erradas.

Classification Report

Aprendemos sobre métricas estatísticas importantes para avaliar modelos de classificação, focando no classification report da biblioteca Scikit-Learn.

**Matriz de Confusão**: É uma métrica fundamental que nos ajuda a entender o desempenho do classificador.

**Classification Report**: Uma função que compila várias métricas derivadas da matriz de confusão.

**Acurácia:** Percentual de acertos do modelo em relação ao total de previsões.

**Precisão:** Proporção de predições corretas de uma categoria em relação a todas as precições dessa categoria. É calculada como: Precisão = VP / (VP+FP).

**Recall:** Proporção de predições corretas da categoria alvo, calculada como: Recall = VP / (VP + FN).

**F1-Score:** Média harmônica entre precisão e recall, dada por: F1-score = 2 \* (Precisão \* Recall) / (Precisão + Recall).

**Support:** Quantidade absoluta de casos encontrados em cada categoria.

Essas métricas ajudam a avaliar a eficácia do modelo, especialmente em problemas de classificação binária.

Plotando a curva ROC

Aprendemos sobre a importância da curva ROC (Receiver Operating Characteristic) como uma métrica estatística para avaliar modelos de classificação, especialmente no contexto de concessão de crédito. A curva ROC nos ajuda a entender a calibragem do modelo de regressão logística.

Os principais passos abordados incluem:

**Importação das funções:** Utilizamos as funções ‘roc\_curve’ e ‘roc\_auc\_score’ da biblioteca Scikit-learn.

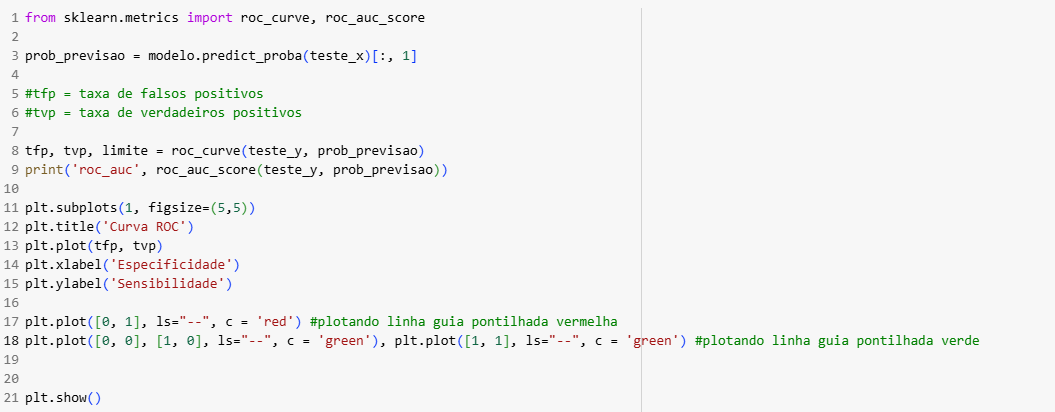
**Previsão de Probabilidades**: Calculamos as probabilidades de previsão do modelo com ‘modelo.predict\_proba(teste\_x)’.

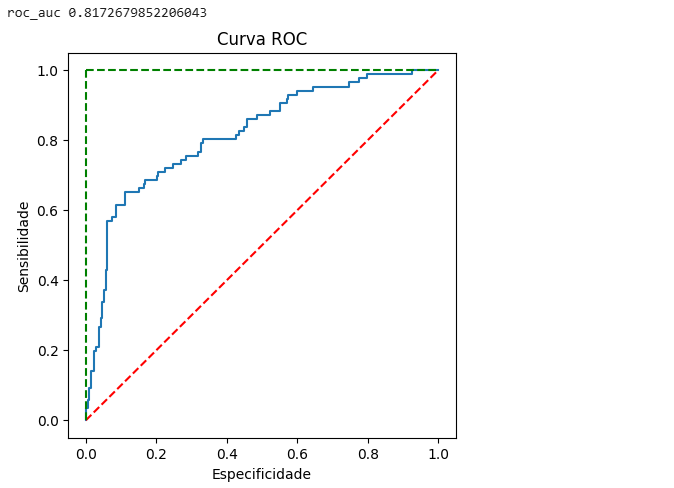
**Cálculo da curve ROC:** Definimos as taxas de falsos positivos (ftp) e verdadeiros positivos (tvp) usando a função ‘roc\_curve’.

**Cálculo da área sob a curva:** Usamos ‘roc\_auc\_score’ para calcular a área sob a curva ROC, que indica a performance do modelo.

**Plotagem da curva:** Utilizamos o Matplotlib para visualizar a curva ROC, incluindo linhas guias que ajudam na interpretação do gráfico.

A área sob a curva (AUC) é um indicador importante, onde valores acima de 0.8 indicam uma excelente discriminação do modelo.





Entendendo a curva ROC

Aprendemos sobre métricas estatísticas, focando na curva ROC, que avalia o desempenho de um modelo com base na relação entre a taxa de verdadeiros positivos (sensibilidade) e a taxa de falsos positivos (especificidade) para diferente limites.

A sensibilidade é a taxa de casos em que a categoria alvo (categoria 1) é classificada corretamente, enquanto a especificidade mede a taxa em que a categoria 0 é incorretamente classificada como 1. A curva ROC é representada graficamente, onde a linha vermelha indica um modelo aleatório e a linha verde um modelo perfeito. O ideal é que a curva ROC esteja o mais próximo possível do canto superior esquerdo, maximizando a taxa de verdadeiros positivos e minimizando a taxa de falsos positivos.

Além disso, discutimos a área sob a curva ROC (AUC), que quantifica a eficácia do modelo. A convenção estatística para a AUC classifica a discriminação do modelo em diferentes níveis: baixa, aceitável, excelente e excepcional, dependendo do valor da AUC.

* O que é e como construir uma matriz de confusão;
* O que são as métricas precision, recall, f1-score e support;
* O que é a curva ROC, como construí-la e interpretá-la.

**Análise e visualização de dados**

Análise Descritiva

Aprendemos sobre a análise descritiva de dados no contexto de um modelo de Machine Learning para crédito. O primeiro passo é entender o comportamento das variáveis na base de dados. Utilizamos a função ‘.describe()’ do Pandas para obter métricas estatísticas resumidas, como contagem, média, desvio padrão, mínimo, máximo e percentis para variáveis numéricas.

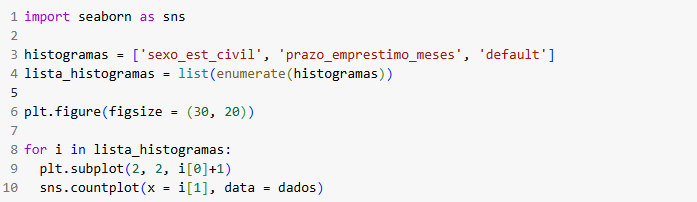
Foi destacado que a aplicação do ‘.describe()’ em variáveis categóricas e binárias não é adequada, pois não fornece informações úteis. Em seguida, foi sugerido um exercício para aplicar essa função em outras variáveis numéricas da base,

Além disso, abordamos como contar as categorias de variáveis categóricas usando ‘.value\_counts()’, o que nos permitiu entender a distribuição de indivíduos em cada categoria. Observamos que a base de dados é desbalanceada, com mais créditos negados do que concedidos, o que é importante considerar ao avaliar o desempenho do modelo, pois a acurácia pode ser enganosa em casos de desbalanceamento.

Histogramas

Aprendi sobre a importância da análise e visualização de dados para entender a distribuição de variáveis em um modelo de Machine Learning, especificamente no contexto de Credit Scoring. A visualização é fundamental para que possamos interpretar as dados e decidir os próximos passos na construção do modelo.

Utilizamos a biblioteca Seaborn para criar histogramas, que são gráficos que mostram a distruibição de variáveis. Começamos importando a biblioteca e, em seguida, criamos uma lista com as variáveis que desejamos visualizar: ‘sexo\_est\_civil’, ‘prazo\_emprestimo\_meses’ e ‘default’.



Depois, enumeramos essa lista para facilitar a plotagem dos gráficos. Utilizamos um loop para gerar múltiplos histogramas em um único gráfico, organizando-os em um layout de 2 linhas e 2 colunas. Por fim, discutimos a interpretação dos histogramas gerados, observando a distribuição de cada variável e identificando possíveis desbalanceamentos nos dados.

A graph of a bar

Description automatically generated with medium confidence

A graph of blue bars

Description automatically generated with medium confidence

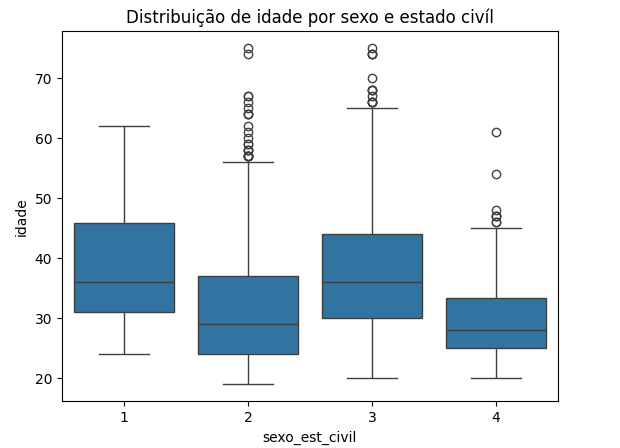
A blue rectangular object with white text

Description automatically generated

Boxplot e Gráfico de Barras

Aprendemos sobre a análise e visualização de dados utilizando gráficos, especificamente o boxplot e o gráfico de barras.

Começamos com o boxplot, que é uma ferramenta poderosa para visualizar a distribuição de dados. Utilizamos a função ‘sns.boxplot()’ do Seaborn para comparar a idade em relação ao sexo e estado civil. O gráfico nos permite identificar a mediana, os quartis e valores discrepantes, proporcionando uma visão clara da distribuição das variáveis.



A close-up of a computer screen

Description automatically generated

Em seguida, exploramos o gráfico de barras com a função ‘plt.bar()’, onde analisamos a relação entre o prazo do empréstimo e o valor do empréstimo. Também aprendemos a personalizar os gráficos, como definir o tamanho da figura e adicionar títulos e rótulos aos eixos.

A graph of blue lines

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

Por fim, discutimos a importância de diferentes tipos de gráficos para visualizar dados de maneiras distintas, como o histograma, que mostra a frequência de uma variável, e o gráfico de dispersão, que pode ser explorado em exercícios futuros.

Balanceamento dos dados

Em problemas de classificação vão existir casos em que podemos ter que trabalhar com bases de dados não balanceadas. Ou seja, bases de dados com um número pequeno de exemplos de uma classe em relação às demais classes. Isto pode ocorrer devido a características do problema, a forma como os dados foram coletados ou erros durante a coleta. A presença de um número de exemplos diferentes em cada uma das classes do problema é comum. Porém, devemos nos preocupar quando essa diferença se torna muito grande.

Um dos problemas que podemos ter quando trabalhamos com dados não balanceados está na acurácia. Se tivermos 80% de exemplos de clientes que não obtiveram concessão de crédito como exemplo e 20% que obtiveram crédito podemos ter uma acurácia de 80%. Entretanto, o modelo pode estar acertando 80% das vezes simplesmente porque “percebeu” que escolhendo a classe 1 para todas as tentativas teria uma porcentagem maior de acerto.

Para lidar com este tipo de problema existem algumas técnicas. Uma delas é a que estamos adotando no curso que é a utilização de outras métricas, como a **matriz de confusão**, **precisão**, **recall**, o **F1 Score** e a **curva ROC**. Essas nos permitem entender melhor o que está acontecendo com o modelo.

Outra estratégia é a remostarem dos dados. Caso a base de dados seja grande o suficiente é possível dividi-la considerando os exemplos em excesso de uma das classes. Em alguns casos também é possível testar a sobreamostragem. Na sobreamostragem são adicionadas cópias de exemplos da classe que está pouco representada na base de dados. Ainda em relação aos dados é possível gerar exemplos sintéticos. Para esse objetivo existem algoritmos que realizam uma amostragem randômica dos atributos dos exemplos da classe pouco representada na base de dados.

Trabalhar o que o modelo pode fazer também é uma opção. Alguns algoritmos conseguem impor uma penalização ao modelo durante o treino. Assim, ele irá prestar mais atenção na classe com menos exemplos.

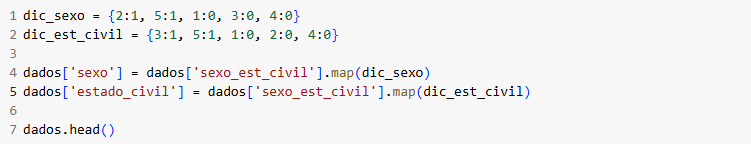
* Realizar uma análise descritiva de dados numéricos, categóricos e binários;
* Aplicação e finalidade de histogramas, gráficos de barras e boxplots;
* Identificação de uma base de dados desbalanceada através do histograma da variável resposta.

**Comparação e escolha do modelo**

*Separação de variáveis*

Aprendemos sobre a separação de variáveis em um modelo de crédito, focando na importância de não utilizar informações sensíveis, como o sexo, de acordo com as regulamentações do Banco Central do Brasil. O estado civil é uma informação relevante, então foi necessário dissociar as informações de sexo e estado civil.

Foi apresentado um método para criar novas variáveis a partir de uma variável original que continha informações combinadas. Utilizamos dicionários para mapear as categorias de sexo e estado civil, transformando-as em variáveis que podem ser utilizadas no modelo sem infringir as regras.



A screenshot of a number

Description automatically generated

Ao final, aprendi a importância de conhecer bem a base de dados e realizar manipulações adequadas para garantir que o modelo respeite as regulamentações e seja coerente com a realidade.

*Ajuste do Modelo*