UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VINICIUS CASIMIRO

DEPLOY DE CLASSIFICADOR DE APROVAÇÃO DE EMPRÉSTIMOS COM STREAMLIT

BAURU
NOVEMBRO/2024

VINICIUS CASIMIRO

DEPLOY DE CLASSIFICADOR DE APROVAÇÃO DE EMPRÉSTIMOS COM STREAMLIT

Relatório da disciplina Inteligência Artificial do curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Docente: Prof. Dr. Clayton Pereira

BAURU
NOVEMBRO/2024

Sumário

1 Introdução

A análise de crédito é um componente essencial do setor financeiro, permitindo que instituições avaliem a viabilidade de conceder empréstimos a indivíduos ou empresas. Com o crescimento da ciência de dados e do aprendizado de máquina, tornou-se possível automatizar esse processo com maior precisão e agilidade. Este trabalho aborda o desenvolvimento e o deploy de um modelo classificador para previsão de aprovação de empréstimos utilizando a ferramenta Streamlit.

O modelo proposto utiliza uma base de dados que inclui variáveis socioeconômicas e relacionadas ao histórico financeiro dos solicitantes, como idade, renda anual, experiência profissional, valor solicitado, taxa de juros, entre outras. A abordagem emprega técnicas de pré-processamento para tratar dados categóricos e numéricos, além de algoritmos de aprendizado supervisionado para realizar as previsões.

O uso do Streamlit facilita a criação de aplicações web interativas diretamente em Python, permitindo que usuários finais interajam com o modelo de classificação de forma intuitiva. Este documento detalha as etapas de construção, treinamento, exportação e deploy do modelo, com ênfase em boas práticas de processamento de dados e integração com interfaces web.

1.1 Aprendizado de Máquina em Classificação de Empréstimos

Modelos de aprendizado de máquina, como Regressão Logística e Florestas Aleatórias, são amplamente utilizados para classificação binária. Eles permitem prever a aprovação ou rejeição de um empréstimo com base em atributos como renda, histórico de crédito e montante solicitado. O sucesso do modelo depende da qualidade dos dados e de técnicas como:

Preparação dos Dados: Limpeza, tratamento de valores ausentes e codificação de variáveis categóricas.

Normalização: Ajuste das variáveis para que tenham escalas comparáveis.

Métricas de Avaliação: Acurácia, precisão, recall e F1-score.

1.2 Deploy de Modelos

O deploy é a etapa em que o modelo é disponibilizado para uso em produção. Requisitos fundamentais incluem:

- Consistência no Processamento: Garantir que os dados recebidos em produção passem pelas mesmas etapas de processamento aplicadas durante o treinamento.
- Eficiência: O modelo deve ser capaz de realizar previsões em tempo real.
- Interface Amigável: Facilitar o uso por não-especialistas através de uma interface intuitiva.

1.2.1 Streamlit

Streamlit é uma biblioteca de código aberto que permite criar interfaces web em Python de forma rápida. Além de uma documentação simples e direta, também conta com muito apoio da comunidade, que fornece ajuda, criando widgets personalizados, atualizando constantemente aspectos do framework e participando de fóruns que fomentam o conhecimento cruzado. Além disso, a plataforma conta com widgets interativos, como botões, seletores, gráficos animados, mapas e muitos outros componentes. Também permite integração direta com outras grandes bibliotecas da linguagem, permitindo visualizações e previsões com baixo nível de acoplamento, fazendo com que o usuário não se preocupe com integração. Por fim, a plataforma permite um deploy simplificado, com serviços de hospedagem como Streamlit Cloud.

Todas essas vantagens, tornam o processo de codificação simples, onde o desenvolvedor não tem que se preocupar com realizar integrações, além da facilidade para gerar uma imagem na internet.

2 Metodologia

Para treinamento dos modelos de inteligência artificial foi utilizado o banco de dados disponível no Kaggle (LO, 2024) contendo informações demográficas e financeiras de pessoas. Dessa forma, o banco de dados conta com 14 colunas distintas, onde, cada uma representa um aspecto de uma pessoa, entre essas propriedades podem ser destacadas:

- Idade da pessoa;
- Sexo da pessoa;
- Grau de formação;
- Quantidade do valor para empréstimo;
- Os ganhos anuais e
- O score de crédito da pessoa.

Essas, são colunas que tem envolvimento direto e podem implicar num resultado negativo, além dessas, a coluna em português, pontuação de empréstimo, ou *loan score*, como é chamada no banco de dados, rotula binariamente cada pessoa, sendo assim, caso esse campo tenha valor 0, essa pessoa não deve ter um empréstimo liberado, mas caso contrário, o empréstimo pode acontecer sem problemas.

Para que o treinamento possa ser realizado de forma a ter o maior rendimento possível, medidas de segurança devem ser tomadas, dessa forma, entender os dados, o tipo de relacionamento entre eles e como valores específicos podem alterar resultados. Portanto, durante este processo, a fase de análise exploratória dos dados desempenha um papel fundamental, pois garante a qualidade do modelo ao permitir a detecção de inconsistências, identificação de padrões relevantes e compreensão aprofundada das relações entre as variáveis.

2.1 Análise exploratória dos dados

Para que os dados trabalhados sejam consistentes e que todos os campos tenham valores, inicialmente, duas medidas são tomadas:

```
df.info()
```

Onde df é o banco de dados. Retornando a imagem a seguir:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45000 entries, 0 to 44999
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   person_age                      45000 non-null  float64
 1   person_gender                   45000 non-null  object
 2   person_education                45000 non-null  object
 3   person_income                   45000 non-null  float64
 4   person_emp_exp                  45000 non-null  int64
 5   person_home_ownership           45000 non-null  object
 6   loan_amnt                       45000 non-null  float64
 7   loan_intent                     45000 non-null  object
 8   loan_int_rate                   45000 non-null  float64
 9   loan_percent_income             45000 non-null  float64
 10  cb_person_cred_hist_length      45000 non-null  float64
 11  credit_score                    45000 non-null  int64
 12  previous_loan_defaults_on_file  45000 non-null  object
 13  loan_status                     45000 non-null  int64
dtypes: float64(6), int64(3), object(5)
memory usage: 4.8+ MB
```

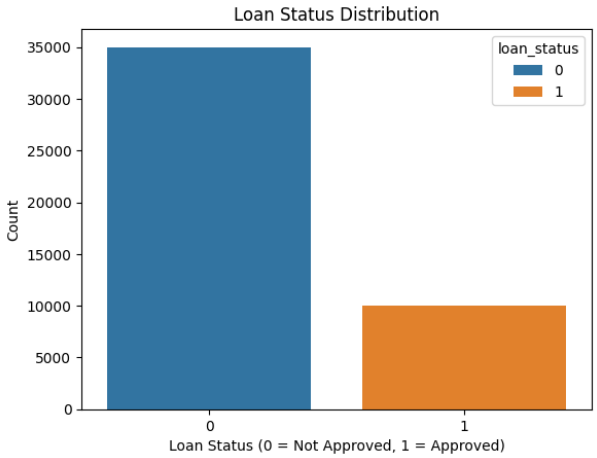
Figura 1 – Saída do comando df.info()

Mostrando todas as colunas que serão trabalhadas, quantidade de tuplas e o tipo de cada dado. Nesse exemplo, uma grande quantidade de dados categóricos são utilizados, dessa forma, um método de codificação deverá ser adotado para que esse dado tenha sentido numérico, permitindo que possa participar de cálculos matemáticos, dessa maneira, nenhum dado será perdido, e o máximo de contexto poderá ser utilizado para classificação.

Outro fator importante é a distribuição das classes dentro do banco de dados, para visualização, a biblioteca **matplotlib** foi aplicada:

```
sns.countplot(x='loan_status', data=df, hue="loan_status")
plt.title('Loan Status Distribution')
plt.xlabel('Loan Status (0 = Not Approved, 1 = Approved)')
plt.ylabel('Count')
plt.show()
```

Gerando o gráfico a seguir:



Nesse caso, é bem perceptível que as classes estão desbalanceadas, tendo 34000 entradas classificadas como 0 e apenas 9000 como 1, então, o algoritmo de balanceamento

SMOTE (técnica de sobreamostragem minoritária sintética) foi aplicado com o intuito de gerar dados mais balanceados, para isso, o algoritmo verifica a vizinhança dos pontos que estão em menor número e encontra um padrão, assim, novos pontos que seguem o padrão estimado, são gerados, aumentando o número de amostras e diminuindo o desbalanceamento. Após o processamento com o SMOTE, o número de amostras cresce, se igualando ao outro grupo, assim, tendo uma base de dados perfeitamente balanceada.

Por fim, é necessário realizar a detecção de valores fora do comum (Outliers), esses atrapalham os modelos, pois apresentam dados surreais, ou situações muito específicas, dessa forma, outliers, não são interessantes neste problema. Portanto, para mitigar esse problema, o calculo do z-score 2.1 foi realizado.

$$z = \frac{x - \mu}{\sigma} \tag{2.1}$$

Quando z $>$ 3, significa, que esse dado é maior que 3 vezes o desvio padrão, ou seja, está muito fora da média dos outros dados, portanto, nessa aplicação, qualquer dado com z-score $>$ 3, será removido.

Com os dados tratados e normalizados, o treinamento pode ter início, de forma que, o máximo de rendimento será extraído do modelo.

2.2 Treinamento

Utilizamos um modelo de Regressão Logística, treinado com um pipeline que inclui, pois ele é simples de interpretar, eficaz para classificações binárias e possui baixo custo computacional, tornando-o ideal para problemas como este. Além disso, o processo de exportação para deploy é simplificado e pode ser salvado em apenas um arquivo. Este modelo estima a probabilidade de um evento ocorrer com base em variáveis independentes, utilizando a função logística para mapear valores preditivos para uma escala entre 0 e 1.

Como existe apenas um banco de dados rotulado, os dados foram divididos, 80% destinados para treinamento e 20% para validação.

2.2.1 Exportação do modelo

Para que o modelo fosse exportado para o servidor de deploy, a biblioteca **joblib** salvou os dados de pipeline de pré-processamento e os próprios modelos de inteligência artificial.

2.3 Implementação com Streamlit

Uma interface foi criada para que o classificador pudesse ser acessado via navegador de internet, sendo capaz de receber dados dos usuários e retornar a previsão do modelo.

3 Testes e resultados

O treinamento do modelo de regressão logística, apresentou acurácia de 88.57%. Além disso, outras métricas, como precisão, *recall* e *f1-score*, foram utilizadas, afim de ter um diagnóstico melhor do resultado. Dessa forma, segue abaixo uma tabela com os resultados:

	precision	recall	f1-score	support
0	0.93	0.84	0.88	6748
1	0.85	0.93	0.89	6748
accuracy			0.89	13496
macro avg	0.89	0.89	0.89	13496
weighted avg	0.89	0.89	0.89	13496

O modelo se mostrou preciso e mostra que apenas um bom tratamento dos dados pode, trazer resultados melhores do que o esperado. Além disso, o site é eficiente, juntado facilidade e rapidez com intuitividade, tornando a navegação fluída e simples.

Referências

LO, T.-w. *Loan approval classification dataset*. 2024. Disponível em: <https://www.kaggle.com/datasets/taweilo/loan-approval-classification-data>.