Relatório do Projeto de Machine Learning, Classificador Binário

Desenvolvedor: **Lucca de Sena Barbosa**

Curso: **Ciências da Computação – 3º período**

João Pessoa, Paraíba

1º Semestre de 2025

RESUMO

O artigo tem como objetivo documentar as técnicas utilizadas para o desenvolvimento de um algoritmo de Machine Learning capaz de prever se um registro ganha uma renda superior ou inferior (ou igual) a U$50.000 anuais. Para isto, foi realizado uma análise exploratória dos dados com objetivo de compreender a sua distribuição e com isto desenvolver um tipo de tratamento. Logo após aplicar o tratamento, foi necessário preparar os dados para os algoritmos transformando-os em dados numéricos e por fim treinar os algoritmos e conferir sua capacidade de classificar registros.

Palavras-Chave: Aprendizado de Máquina, prever, base de dados

SUMÁRIO

1. INTRODUÇÃO

Este projeto surgiu a partir de uma iniciativa pessoal do desenvolvedor a fim de aprimorar seus conhecimentos técnicos na área de aprendizado de máquina. Neste projeto, foi utilizada uma base de dados encontrada na *UCI Machine Learning Repository,* criado por *Barry Becker* em 1994 e por se tratar de um problema de classificação binária, ou seja, um problema onde possua dois tipos de classes é necessário desenvolver algumas etapas para que o problema seja solucionado.

Para o desenvolvimento desse projeto, foi necessário dividi-lo em três etapas com objetivo de que os modelos consigam extrair o máximo de seu desempenho. Na primeira etapa, foi realizada uma análise exploratória dos dados com a finalidade de identificar e tratar inconsistências. Na segunda etapa, foi aplicado técnicas de pré-processamento como uma maneira de preparar os dados para os modelos de aprendizado de máquina. E na terceira e última etapa com dados nunca vistos pelos modelos, foi realizado um teste para medir a capacidade de classificação dos Algoritmos. Este documento também detalha as justificativas para cada técnica utilizada na construção dos modelos.

2. ANÁLISE EXPLORATÓRIA DOS DADOS

Após a aplicação da análise exploratória dos dados, percebeu-se que a base de dados apresentava algumas inconsistências que precisariam ser tratadas para que o desenvolvimento do algoritmo seja bem-sucedido. Sobre os atributos categóricos, observou-se uma quantidade de valores faltantes nos atributos “workclass”, “occupation” e “native-country”. Por serem atributos qualitativos nominais, optou-se por preenchê-los com a moda de cada coluna.

Quanto as colunas numéricas, notou-se a existência de valores com intervalos muitos maiores do que o comum, o que pode ser um indicativo de outlier. Para determinar se há outliers ou não, foi utilizado um cálculo estatístico chamado *Intervalo Entre Quartis (IQR)* que estabelece um limite máximo e mínimo do que pode ser considerado um dado normal. Entretanto, após utilizar essa técnica foi descoberto que 992 registros na classe “fnlwgt” e 9008 registros na classe “hours-per-week” estavam fora do intervalo estabelecido pelo IQR. Curiosamente, após o treinamento dos modelos, constatou-se que o desempenho dos algoritmos era melhor quando os outliers não eram tratados. Por essa razão, optou-se por não realizar o tratamento desses “outliers” na versão final do modelo. Além disso, notou-se que as colunas numéricas apresentavam distribuições não normais, o que deve ser considerado na etapa de pré-processamento, especialmente no momento do escalonamento dos dados.

Em relação ao atributo classe, identificou-se um desbalanceamento, pois os registros que ganham mais que US$50.000 representavam 24.08% da base de dados enquanto os registros que ganham menos ou igual a US$50.000 representam 75.92%.

Considerando que a base foi criada em 1994, é de se esperar que os dados estejam distribuídos dessa forma, no entanto isso pode se tornar um problema para os algoritmos de previsão, uma vez que isso pode ocasionar uma generalização excessiva para a classe majoritária. Em resumo, esta análise exploratória foi conduzida com o objetivo de garantir que os dados fossem adequadamente tratados na próxima etapa do projeto.

3. PRÉ-PROCESSAMENTO

Após os dados serem tratados, seguimos com a etapa de pré-processamento. Nesta etapa é importante destacar que seu objetivo é preparar a base de dados para os algoritmos de machine learning, uma vez que são compostos por atributos numéricos e qualitativos. Se faz necessário esta etapa, uma vez que os modelos são puramente cálculos matemáticos e não é coerente entregar uma base de dados onde possuam dados categóricos.

Para essa etapa, foi dividido a base de dados entre atributos previsores representado pela variável *x_features* e pelo atributo classe representado pela variável *y_class*. Após isso, foi aplicado um método de codificação que possui o objetivo de transformar os dados em valores numéricos. Quanto ao método de codificação, foi utilizado uma função do scikit-learn chamada LabelEncoder que transforma os dados na base de dados em valores numéricos inteiros.

No entanto, esse método pode proporcionar enviesamento para os modelos, uma vez que eles entendem um valor alto como prioridade e inferior com menos prioridade, na qual nem sempre isso ocorre. Para evitar este problema, iremos utilizar um método de escalonamento, onde esses dados serão aproximados de tal maneira que os modelos não se tornem enviesados. Quanto ao método de escalonamento, foi utilizado uma função do scikit-learn chamada MinMaxScaler apenas nas colunas numéricas assumidas previamente.

Para solucionar o problema de desbalanceamento nas classes, considerou-se o uso da técnica over-sampling. Geralmente, esse método gera *novos exemplos sintéticos* (por exemplo, com SMOTE), dessa forma teoricamente o algoritmo conseguirá classificar coerentemente e sem generalizações excessivas as duas classes.

- Antes do pré-processamento:

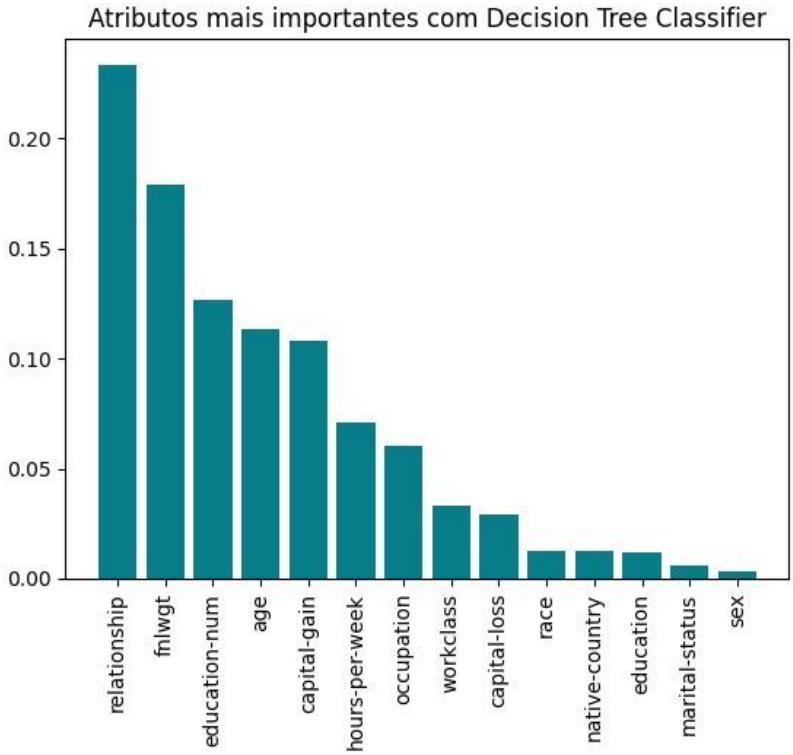
```
array([[39, 'State-gov', 77516, ..., 0, 40, 'United-States'],
       [50, 'Self-emp-not-inc', 83311, ..., 0, 13, 'United-States'],
       [38, 'Private', 215646, ..., 0, 40, 'United-States'],
       ...,
       [40, 'Private', 154374, ..., 0, 40, 'United-States'],
       [58, 'Private', 151910, ..., 0, 40, 'United-States'],
       [22, 'Private', 201490, ..., 0, 20, 'United-States']],
      shape=(32560, 14), dtype=object)

array([0, 0, 0, ..., 1, 0, 0], shape=(32560,))
```

- Depois do pré-processamento:

```
(array([[ 0.30136986,  0.0443019 ,  0.8       , ...,  4.        ,
          1.        , 38.        ],
       [ 0.45205479,  0.0482376 ,  0.8       , ...,  4.        ,
          1.        , 38.        ],
       [ 0.28767123,  0.13811345,  0.53333333, ...,  4.        ,
          1.        , 38.        ],
       ...,
       [ 0.31506849,  0.09650032,  0.53333333, ...,  4.        ,
          1.        , 38.        ],
       [ 0.56164384,  0.09482688,  0.53333333, ...,  4.        ,
          0.        , 38.        ],
       [ 0.06849315,  0.12849934,  0.53333333, ...,  4.        ,
          1.        , 38.        ]], shape=(32560, 14)),
 array([0, 0, 0, ..., 1, 0, 0], shape=(32560,)))
```

4. SELEÇÃO DE ATRIBUTOS

Os atributos em uma base de dados são essenciais para qualquer tipo de treinamento para modelos de machine learning, pois com eles os algoritmos conseguirão aprimorar sua capacidade de previsão. No entanto, dependendo do objetivo de um projeto, nem sempre todos eles serão representativos para o problema em questão porque muitos deles podem atrapalhar o desempenho do algoritmo. Tendo esse problema em vista, para este projeto adotamos um método para selecionar as melhores features em uma base de dados que consiste em utilizar um modelo de machine learning chamado Decision Tree Classifier para a seleção. No entanto, após aplicar esse método foi retornado os seguintes atributos:



Para determinar quais atributos utilizar, decidiu-se selecionar os atributos desde “relationship” até “captain-gain”, uma vez que esses atributos demonstraram certa relevância nos gráficos. No entanto, após essa seleção e a preparação dos dados para que os modelos recebessem informações desses atributos, os resultados do treinamento dos modelos não apresentaram melhora no desempenho. No próximo tópico, serão apresentados os resultados do treinamento de vários algoritmos com todos os atributos da base de dados. Esses modelos demonstraram desempenho superior em relação à abordagem anterior, justificando a decisão de utilizar todos os atributos da base no treinamento final.

5. Treinamento dos Modelos:

Para escolher o melhor modelo para o nosso projeto de Machine Learning é essencial deixar claro o objetivo que é o desenvolvimento de um modelo capaz de classificar coerentemente registros que ganham mais ou menos ou igual a US$50.000 por ano. Se o objetivo é classificar coerentemente os registros com as duas classes, o algoritmo deve demonstrar uma capacidade de classificação igualitária entre os rótulos. Como a base possui um desbalanceamento de atributos classe, é normal que os algoritmos consigam identificar melhor registros que ganham menos ou igual a US$50.000. Contudo, os algoritmos que demonstraram um desempenho equivalente foi o Balanced Random Forest e o Tradicional Random Forest. Logo abaixo, terá os resultados dos algoritmos:

Logistic Regression:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **82.33%**.
- O modelo identificou corretamente **3236** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **588** registros com o rótulo '>50'.

Support Vector Machine:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **85.73%**.
- O modelo identificou corretamente **3243** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **739** registros com o rótulo '>50'.

Random Forest Classifier:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **88.29%**.
- O modelo identificou corretamente **3246** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **855** registros com o rótulo '>50'.

Gradient Boosting Classifier:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **87.34%**.
- O modelo identificou corretamente **3265** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **792** registros com o rótulo '>50'.

Neural Network Classifier:

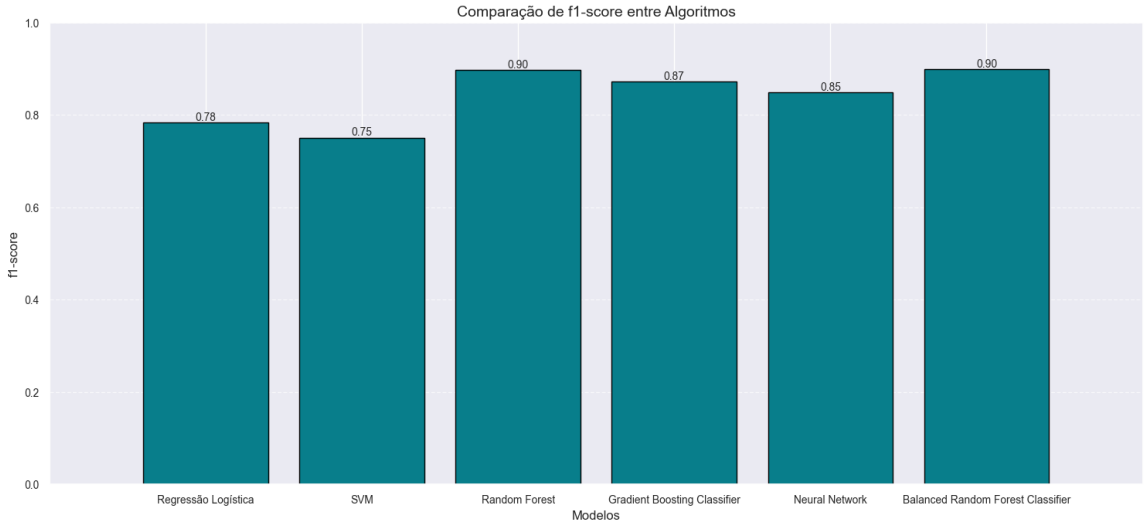
- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **86.18%**.
- O modelo identificou corretamente **3136** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **867** registros com o rótulo '>50'.

Balanced Random Forest Classifier:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **86.14%**.
- O modelo identificou corretamente **3000** registros com o rótulo '<=50K'.

- O modelo identificou corretamente **1001** registros com o rótulo '>50'.

Logo abaixo, será mostrado um gráfico visando a comparação de desempenho entre os modelos. Será utilizado a métrica f1-score, pois ela é uma média harmônica entre Precision e Recall, o que nesse contexto indica um modelo capaz de identificar e classificar corretamente os dois tipos de classe.



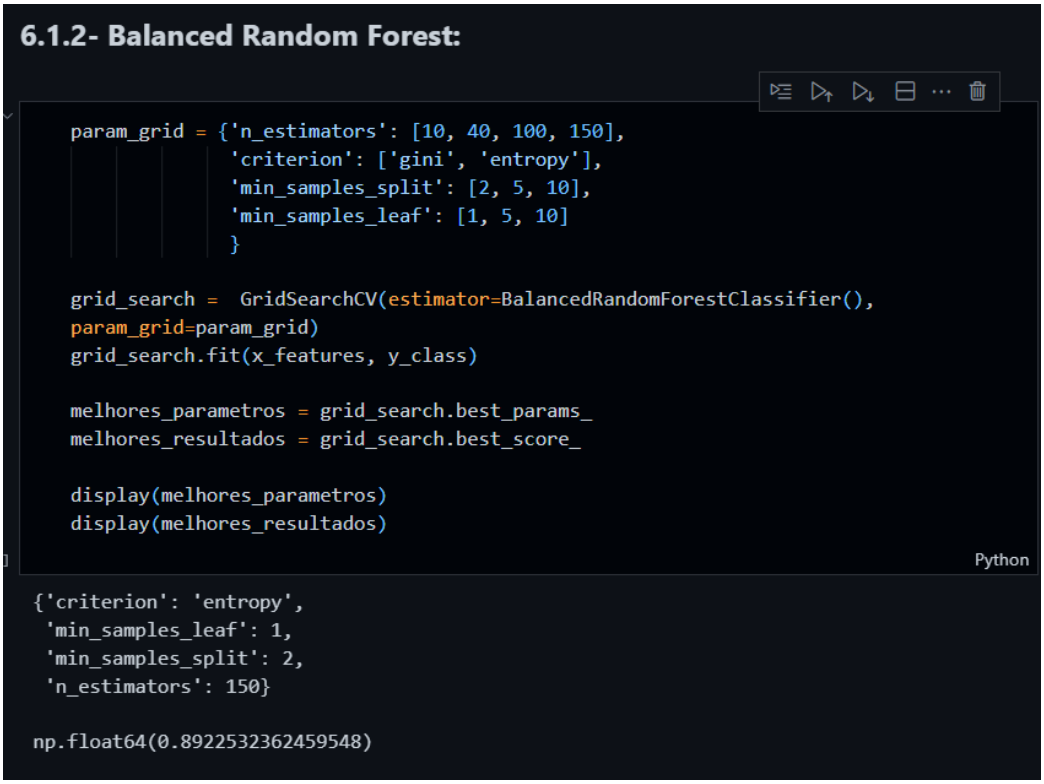
6. Otimização dos Modelos:

Como visto anteriormente os algoritmos que tiveram um melhor desempenho foi o Random Forest da biblioteca scikit-learn e o Balanced Random Forest da biblioteca imblearn.

No entanto, para definir qual será o algoritmo definitivo do projeto será realizado uma otimização dos dois modelos e aquele que demonstrar um melhor desempenho, será o escolhido. Para definir qual técnica de otimização utilizar, iremos utilizar uma funcionalidade do scikit-learn chamada GridSearch que consiste em após assumir um conjunto de parâmetros, retornar qual deles proporcionará uma melhor acurácia para o modelo. Após aplicar este método, o algoritmo que teve um melhor desempenho foi o Random Forest Classifier, mantendo a acurácia do treinamento.

Balanced Random Forest Classifier:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **89.22%.**
- O modelo identificou corretamente **2986** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **991** registros com o rótulo '>50'.



Random Forest Classifier:

- A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **88.27%.**
- O modelo identificou corretamente **3245** registros com o rótulo '<=50K'.
- O modelo identificou corretamente **855** registros com o rótulo '>50'.

No entanto, após a otimização de parâmetros com o GridSearch, notamos que o Random Forest foi o algoritmo mais capacitado em classificar registros referentes a sua renda, equilibrando nas métricas Recall e Precision.

- Métricas finais do modelo Random Forest:
  - A acurácia geral do modelo em relação a todos os dados foi de aproximadamente **89.37%.**
  - O modelo identificou corretamente **3254** registros com o rótulo '<=50K'.
  - O modelo identificou corretamente **3375** registros com o rótulo '>50'.

	precision	recall	f1-score	support
0	0.91	0.87	0.89	3731
1	0.88	0.92	0.90	3685
accuracy			0.89	7416
macro avg	0.89	0.89	0.89	7416
weighted avg	0.89	0.89	0.89	7416

REFERÊNCIAS

Livro: O'REILLY, Aurélien Géron. *Mãos à Obra: Aprendizado de Máquina com Scikit-learn, Keras & TensorFlow.* 2. ed. São Paulo: Novatec, 2021.

- **Grid Search: p. 61-63.**
- **Cálculo de Desempenho: p. 71-78.**

IMBALANCED-LEARN. UnderSampling. Disponível em: https://imbalancedlearn.org/stable/references/generated/imblearn.under_sampling.TomekLinks.h tml

IMBALANCED-LEARN. BalancedRandomForestClassifier. Disponível em: https://imbalancedlearn.org/stable/references/generated/imblearn.ensemble.BalancedRandomFo restClassifier.html

Oracle. IQR (Intervalo entre quartis). Disponível em: https://docs.oracle.com/cloud/help/pt_BR/pbcs_common/PFUSU/insights_metrics_IQR.htm#PFUSU-GUID-CF37CAEA-730B-4346-801E-64612719FF6B

Vídeo no YouTube: YOUTUBE. *Feature Engineering para Machine Learning.* Disponível em: https://www.youtube.com/watch?v=HQmqdY3UOtQ.

Curso Online: UDEMY. *Machine Learning e Data Science com Python.* Disponível em: https://www.udemy.com/course/machine-learning-e-data-science-compython-y/learn/lecture/26272186?start=330#overview.

Curso Online: UDEMY. *Formação Cientista de Dados: O Curso Completo.* Disponível: https://www.udemy.com/share/101Xys3@nC0C7B9-AyMuf124Zjq26fjAtdqaTA0vefRXmGrQlrdWAcoYnRvuennvubEBKByQOA==/