PROJETO FINAL: ANÁLISE, LIMPEZA E TREINAMENTO DE MODELOS

Grupo 5: André Dias, João Victor Meneghelli Milanezi, Vinícius Rosa

1. Análise do Dataset

Primeiramente, foi necessário avaliar quantos dados temos de cada coluna, foi observado que com um total de 40 mil dados, mas com uma grande quantidade de localizações vazias (mais da metade), também temos outras categorias com dados vazios, mas com no máximo na casa dos mil dados. Também foi necessário observar quantos dados únicos de cada tipo o dataset apresenta, para entendermos como poderemos fazer a limpeza, nesta etapa foi possível verificar duplicidades nos dados como respostas ‘n’ e ‘no’ ou ‘y’ e ‘yes’ para representar a mesma coisa.

2. Processo de limpeza

A fim de garantir uma melhor qualidade dos dados e a precisão do modelo, excluímos colunas que não possuíam informações relevantes para treinar o algoritmo, como *id*, ou possuíam muitos dados faltantes, como *outcome* e *location*. Além disso, excluímos as linhas cujo valor de *job, education, marital* e *contact* fosse “nan” ou “unkown”. Através da análise do dataset, estabelecemos intervalos de *balance*, *age* e *previous* para assegurar que a presença de outliers não enviesasse o modelo.

Além disso, foram feitos gráficos para nos ajudar a entender melhor a distribuição dos nossos dados, nesse processo, foram observados outliers como pessoas com 150 anos de idade, pessoas com balanço superior a 15000 ou inferior a -5000, pessoas com previous maior que 30, duration maior que 2500 ou campaign maior que 20, essas remoções foram necessárias para que uma parte ínfima do dados não seja entendida como padrão.

Outro fator foi o mapeamento das strings contidas no dataset, tudo que fosse ordenado foi colocado, numa ordem condizente (por exemplo mês do ano) e tudo que fosse não ordenado foi colocado com números distribuídos aleatoriamente.

3. Treinamento inicial e avaliação

Nessa etapa, foi necessário decidir qual modelo iríamos utilizar. Para isso, utilizamos a biblioteca “lazypredict”, que treina vários modelos e nos mostra os resultados obtidos por cada um. Com isso, decidimos pelo XGB Classifier que apresentou as melhores métricas.

4. Treinamento otimizado

Após termos escolhido o XGB Classifier, era necessário otimizá-lo. O primeiro problema que encontramos foi que o nosso dataset estava desbalanceado. Apesar de a acurácia geral na fase de testes dar um valor alto, por volta de 90%, nosso modelo predizia muito mal a categoria 1, isto é, quando o cliente contrata os serviços do banco. Para resolver isso, utilizamos a técnica de oversampling, que visa aumentar o número de exemplos da classe minoritária, utilizamos um parâmetro de sampling-strategy de 0.55, pois parecia desempenhar melhor. .

O próximo problema que buscamos resolver foi que para a classe 1 o “recall” estava baixo, por volta de 47%. O modelo estava classificando vários exemplos como classe 0, o que cria vários falsos negativos e baixa o recall. Com isso, utilizamos thresholds, em que podemos estipular o quanto de “certeza” o modelo tem de ter para fazer uma classificação. Em outras palavras, se antes se o modelo avaliava que a probabilidade de ser classe 0 era 51%, ele predizia ela, agora ele necessitava de uma probabilidade maior - o que diminui os falsos negativos.

Além disso, foram buscados os hiperparâmetros que melhoraram a predição, um processo que foi feito utilizando um Grid Search programado na “mão”, devido a influência do threshold.

Para todos esses processos utilizamos validação cruzada, visando evitar overfitting e tratar de modo eficiente nossos dados.

Vale destacar também que não necessariamente o modelo que teve um melhor desempenho nos testes apresentou um melhor desempenho no kaggle, às vezes uma melhora significativa nos testes representava piora no kaggle, e vise versa. Algumas explicações para isso são: overfitting e que os dados de teste do kaggle tinham mais classes 1 que os dados de treino.

5. Decisões metodológicas

Foi pensado inicialmente em substituir os vários dados de telefone categorizados como “desconhecidos”, treinando um modelo a parte justamente para isso, porém, os dados de desconhecidos pareciam não encaixar muito bem nas duas outras categorias (telefone e celular), e os resultados disso também não foram satisfatórios, dessa forma, foi decidido em retirá-los do nosso modelo, e foi excluído do notebook para facilitar a visualização do que estava sendo útil ao projeto.

Também foi tentado substituir os dados NaN das demais categorias usando o modelo KNN e de regressão logística, porém nenhum dos dois trouxe resultados significativos positivos. Dessa forma, foi decidido manter os dados da maneira que estavam.

Outra tentativa foi relacionar os dados de loan e housing, visto que loan significa ter um empréstimo ativo, e housing significa ter um empréstimo habitacional. Assim, foi pensado que caso o cliente tenha um empréstimo habitacional, ele tem um empréstimo consequentemente, dessa forma, caso housing = 1 -> loan = 1. Porém, foi visto que isso não trouxe mudanças positivas.

Também tentamos excluir coluna a coluna do dataset e ver se resultava em algo positivo, ou trabalhando com 2 modelos ao mesmo tempo: KNN e XGB. Entretanto, nenhum deles ocasionou melhorias.