Relatório do Projeto Final - Turing Academy 2024

Análise, Limpeza e Treinamento de Modelos

Caetano, Lie e Thainara

05/08/2024

1. Introdução

O presente relatório busca explicar as hipóteses e decisões metodológicas tomadas para a análise e treinamento via machine learning de um dataset público do kaggle, cujo objetivo final é construir um modelo capaz de realizar predições adequadas acerca dos casos futuros que venham a ocorrer.

Tendo o objetivo claro em mente, iniciou-se a etapa de análise do banco de dados, segunda à descrição do projeto, temos as seguintes informações sobre o dataset:

"O conjunto de dados aqui tratado diz respeito a campanhas de marketing direto de uma instituição bancária portuguesa. As campanhas tiveram como base chamadas telefônicas (telemarketing). Frequentemente, mais de um contato com o mesmo cliente foi necessário para determinar se o produto (depósito bancário na instituição) teria sido ("yes") ou não ("no") contratado.

O dataset apresenta as seguintes features:

- **id** - Número de identificação do cliente;
- **age** - Idade do cliente;
- **job** - Tipo de emprego do cliente;
- **marital** - Estado Civil;
- **education** - Grau de Educação;
- **location** - Localidade;
- **default** - Se o cliente possui crédito de inadimplência;
- **balance** - Balanço médio anual, em euros;
- **housing** - Se possui ou não empréstimo habitacional;
- **loan** - Se possui um empréstimo ativo;
- **contact** - Tipo de comunicação realizada (no último contato);
- **day** - Dia do último contato;
- **month** - Mês do último contato;
- **duration** - Duração do último contato, em minutos;

- **campanha** - Número de contatos realizados com o cliente durante a campanha;
- **pdays** - Número de dias passados desde o contato com o/a cliente em uma campanha anterior, -1 caso o cliente nunca houvesse sido contatado;
- **previous** - Número de contatos realizados antes desta campanha;
- **poutcome** - Resultado da campanha de marketing anterior;

  Além da target:
- **y** - Contratação ou não do serviço do banco pelo cliente (Sucesso da campanha)"

2. Análise e Limpeza de dados

Após sabermos com quais features e targets estamos trabalhando, iniciou-se o processo de Análise e Limpeza de dados.

Primeiramente, como forma de entender melhor os dados que estamos utilizando, obtemos um perfil geral dos dados, adquirindo informações sobre índice, colunas, valores não-nulos e tipo de dados. Assim, pudemos perceber que as features ***age, job, marital, location, balance, contact, campaign e previous*** possuem valores faltantes. Além disso, obteve-se três tipos de dados, como inteiros (**int64**), flutuantes (**float64**) e objetos (**object**), de forma que pudemos entender quais dados são categóricos e quais dados são numéricos.

A fim de compreender ainda mais os dados, obteve-se um resumo estatístico para as features, como por exemplo, para os dados categóricos, obter para cada coluna o número de entradas não nulas, número de valores únicos, valor mais comum e sua frequência. Para os dados numéricos obter para cada coluna o número de entradas não nulas, média, desvio padrão, maior valor, menor valor, número de valores únicos, entre outros. Dessa maneira, na distribuição das colunas categóricas pudemos perceber um número limitado de valores únicos e compreender a distribuição predominante por meio dos valores mais frequentes. Já para a distribuição das colunas numéricas, nota-se que a feature ***balance*** possui uma alta variação, com

muitos valores únicos, indicando possíveis outliers, enquanto a feature ***age*** tem um valor máximo de 150, indicando um erro na entrada de dados ou outliers.

Para prosseguir na análise de maneira mais simples, o grupo optou por transformar os dados categóricos de valores binário em dados numéricos, ou seja, para as features ***default, housing, loan*** e o ***target,*** valores como "y" e "yes" tornaram-se 1 e valores como "n" e "no" tornaram-se 0, de forma que pudemos entender melhor a frequência de ocorrência de cada valor. Por fim, verificamos a correlação dos dados categóricos com o target, obtendo alguns gráficos para melhor compreensão do comportamento dos dados de maiores correlações em relação ao target.

Após o melhor entendimento dos dados que estamos trabalhando, iniciamos o processo de limpeza. A primeira coisa feita foi excluir a coluna ***id*** do nosso dataset, uma vez que ele não possui nenhuma correlação com o target, como esperado e então analisar o número de dados faltantes em cada coluna. Por meio dessa descoberta e de uma análise de correlação, verificamos que a feature ***location*** não se correlaciona com o target, optando por excluí-la. Já para a feature marital, que possui muito pouco dado faltante, decidimos excluir apenas essas linhas.

Em seguida, tratamos os outliers, removendo-os, pois, como observado anteriormente, a feature ***balance*** possui alta variação, indicando possíveis outliers, assim como a presença do valor 150 na coluna ***age.*** Ainda, features como ***duration, campaign, previous e pdays*** também tiveram seus outliers removidos, identificados por meio de gráficos de densidade.

Ainda, verificamos a existência de dados desbalanceados graficamente, para facilitar a decisão em relação à necessidade de ser feito um oversampling ou undersampling na etapa de otimização do método escolhido e, além disso, com base na análise de influência de uma variável categórica sobre o target, pudemos eliminar as features ***marital*** e ***contact***, por não possuírem uma relação aparente com o target.

Para finalizar a primeira etapa, realizamos o processo de encoding de variáveis categóricas, tratamento de valores ausentes e normalização de dados. Para o tratamento dos valores ausentes, identificamos os valores "unknown" nas colunas

education e poutcome, substituindo-os por ***NaN*** e também optamos por excluir a coluna ***pout come*** devido à alta proporção de valores ausentes. Para o encoding de variáveis categóricas, realizamos uma ordinarização manual na coluna ***education***, transformando as categorias em valores numéricos. Já para as colunas ***job*** e ***month***, aplicamos o One-Hot Encoding para transformar as categorias em variáveis binárias (0 ou 1). Então, realizamos uma normalização das colunas numéricas para garantir que os valores fiquem entre 0 e 1 e finalizamos essa parte de limpeza de dados usando o KNN Imputer para preencher valores ausentes nas features, baseado nos valores mais próximos .

3. Treinamento Inicial e Avaliação

Com nossos dados limpos e prontos para uso, separamos-os entre dados de treino e teste. Em seguida, utilizamos alguns modelos de Machine Learning para treiná-los e avaliá-los. A escolha dos métodos e métricas de avaliação foram feitas com base nas aulas assistidas do Turing Academy. Entre os métodos aprendidos, utilizamos o ***KNN, Decision Trees, Regressão Logística, Naive Bayes, Random Forest e Support Vector Machine (SVM)***. Já para as métricas de avaliação, optamos por utilizar, inicialmente, a ***acurácia, precisão, recall, AUC-ROC e F1***, obtendo a seguinte tabela.

Estimadores/Modelo (%)	Decision Tree	KNN	Regressão Logística	Nayve Bayes	SVM	Random Forest (Pré-Otimização)
Acurácia	86	89	89	86	89	90
Precisão	40	48	57	39	52	59
Recall	43	23	23	43	11	37
AUC	67	76	88	79	89	91
F1	42	31	32	41	18	45
Pontuação(3|1|2)	6,82	6,75	6,98	6,98	6,53	7,52
Pontuação(1|1|1)	6,50	6,53	6,97	6,87	6,53	7,53
Pontuação(2|3|1)	6,92	7,28	7,90	7,50	7,72	8,30
Pontuação(1|2|1)	6,55	6,80	7,43	7,13	7,13	7,93

Figura 1: Tabela com os valores das Métricas de Avaliação obtidas com os Modelos Iniciais.

Assim, criamos um esquema de pontuações atribuindo diferentes pesos para os principais estimadores, em nossa percepção, ***acurácia, AUC-ROC e F1***, respectivamente, seguindo a ordem dos pesos entre parênteses na tabela. Para os diferentes pesos atribuídos às métricas, notamos que o modelo Random Forest possui os maiores valores em todas as opções, de maneira que o escolhemos para passar pelo processo de otimização, entretanto, logo será explicado o porquê ele não foi a nossa escolha final.

O Random Forest é um modelo de Machine Learning do tipo Ensembling, combinando o resultado de múltiplos modelos para realizar a predição, mais especificamente do modelo Decision Trees. Seu funcionamento consiste em construir e treinar diversas árvores de decisão por meio de um subconjunto aleatório do nosso dataset e, então, fazer a predição final com base na moda dos resultados obtidos, para problemas de classificação, ou na média, para problemas de regressão.

4. Treinamento Otimizado

Com o modelo Random Forest escolhido, a princípio, passamos para o processo de otimização, por meio do ***resampling,*** através da técnica de SMOTE, ***cross-validation e otimização bayesiana de hiperparâmetros***. Os hiperparâmetros utilizados para o modelo foram:

- ***n_estimators***, ou seja, o número de árvores na floresta, que possui uma melhor performance conforme maior o número, entretanto um tempo de computação mior também;
- ***max_depth***, ou seja, a profundidade máxima de cada árvore, controlando o tamanho das árvores. Para árvores rasas, pode haver sub ajuste dos dados, já para árvores profundas, pode ocorrer o sobreajuste;
- ***min_samples_split***, ou seja, o número mínimo de amostras necessárias para dividir um nó, controlando a forma como a árvores cresce e sua complexidade;
- ***min_samples_leaf***, ou seja, o número mínimo de amostras que um nó folha deve ter, ajudando a evitar, assim, nós muito específicos;

- ***max_features***, ou seja, o número máximo de features a serem consideradas para encontrar a melhor divisão.

Entretanto, nessa etapa, optamos por explorar mais alguns outros modelos para otimização dos nossos estimadores, como forma de curiosidade e maior aprendizado. Dessa forma, treinamos modelos como ***Gradient Boosting Decision Trees (GBDT) e Rede Neurais***, obtendo a seguinte tabela.

Estimadores/Modelo(%)	Random Forest (Pré-Otimização)	Random Forest (Pós-Otimização)	Redes Neurais (Pré-Otimização)	Redes Neurais (Pós-Otimização)	GDBT (Pré-Otimização)	GDBT (Pós-Otimização)
Acurácia	90	86	84	90	90	87
Precisão	59	44	39	48	59	46
Recall	37	74	74	44	37	75
AUC	91	91	87	87	67	91
F1	45	55	51	46	45	57
Pontuação(3|1|2)	7,52	7,65	7,35	7,48	7,12	7,77
Pontuação(1|1|1)	7,53	7,73	7,40	7,43	6,73	7,83
Pontuação(2|3|1)	8,30	8,33	8,00	8,12	7,10	8,40
Pontuação(1|2|1)	7,93	8,08	7,73	7,75	6,73	8,15

Figura 2: Tabela com os valores das Métricas de Avaliação obtidas com os Modelos Otimizados.

Com esses novos valores obtidos através do esquema de pontuação criado por nós, decidimos por escolher o método GBDT, que, mesmo tendo valores próximos do Random Forest, eles são levemente melhores.

O Gradient Boosting Decision Trees é um modelo de Machine Learning que utiliza um algoritmo de Ensemble - como no Random Forest, treina diversas árvores e combina seus resultados para predição -, juntamente com a técnica de boosting - a junção sequencial de árvores "fracas", de forma que cada árvore nova corrige erros da árvore passada -, tornando-o mais robusto. Seu funcionamento consiste de uma etapa de inicialização, no qual é definido um modelo inicial fraco, simples, podendo representar a média/moda dos valores. Em seguida, para cada ponto do conjunto de treino, calcula-se o resíduo associado entre as previsões do modelo e os valores reais, representando o que o modelo inicial não conseguiu prever corretamente. Então, um novo modelo é treinado para poder prever os resíduos do modelo anterior e esse novo modelo tenta corrigir os erros feitos pelo modelo anterior. O processo de

treinamento continua iterativamente até que atinja um ponto para minimização da função de custo, ou seja, os resíduos se tornem suficientemente pequenos.

A otimização do GBDT foi feita da mesma forma que explicado anteriormente, ou seja, pelo ***resampling,*** através da técnica de SMOTE, ***cross-validation e otimização bayesiana de hiperparâmetros***. Os hiperparâmetros utilizados para o modelo foram:

- ***n_estimators***, ou seja, o número de árvores na sequência de boosting, que leva a um melhor ajuste conforme a maior quantidade de árvores, mas deve-se atentar ao sobreajuste;
- ***learing_rate***, ou seja, controla o quanto cada árvore contribui para a previsão final. Valores menores podem resultar a um melhor desempenho, mesmo que torne o treinamento mais lento;
- ***max_depth***, ou seja, a profundidade máxima de cada árvore de decisão, controlando o tamanho das árvores. Para árvores rasas, pode haver subajuste dos dados, já para árvores profundas, pode ocorrer o sobreajuste;
- ***min_samples_split***, ou seja, o número mínimo de amostras necessárias para dividir um nó, controlando a forma como a árvores cresce e sua complexidade;
- ***min_samples_leaf***, ou seja, o número mínimo de amostras que um nó folha deve ter, ajudando a evitar, assim, nós muito específicos;
- ***subsample***, ou seja, a fração das amostras usadas para treinar cada árvore.

5. Conclusão

Após a análise, limpeza de dados e experimentação com diversos modelos, concluímos que o modelo Gradient Boosting Decision Trees (GBDT) apresentou os melhores resultados de predição. Mesmo que o modelo Random Forest tenha mostrado bom desempenho, o GBDT superou ligeiramente em todas as métricas avaliadas.

O processo envolveu uma preparação cuidadosa dos dados, considerando o tratamento de valores ausentes, remoção de outliers e normalização das variáveis. A

escolha do modelo final foi fundamentada em um esquema de pontuação que ponderou as métricas de avaliação mais relevantes para nós.

A implementação de técnicas avançadas como o resampling com SMOTE e a otimização bayesiana de hiperparâmetros foi importante para melhorar o desempenho dos modelos. Esse processo todo nos mostra que o modelo é capaz de realizar boas predições sobre a contratação de serviços bancários em campanhas de marketing. Esse modelo pode ser utilizado para otimizar campanhas, direcionando recursos de maneira mais eficaz e potencialmente aumentando as taxas de sucesso.