Relatório Competição 1

Tharic de Freitas Araujo
Alexandre Alves da Silva Filho

19 de Dezembro de 2022

1 Análise exploratória dos dados

O dataset de treino fornecido possui 227.122 dados e 32 variáveis, das quais uma é o label de "aprovado ou negado" e outra apenas representa o ID do usuário. Das outras 30 variáveis fornecidas, apenas 15, ou 50%, não apresentam campos nulos. As features que representam o tempo de doença e a sua unidade(dias, semanas, meses), assim como a que representa o tipo de doença, estavam com 99% dos campos em branco, a do tipo de consulta possui 95% dos campos inutilizados. O Tipo de saída também estava completamente nulo. O campo da data de nascimento possui apenas 10 campos nulos. Há outros campos, como o caso do CD_ITEM, que tinha um valor diferente para cada uma das entradas, não sendo tão benéfico à intenção final do sistema. O campo de texto livre da indicação clínica também possui diversos tipos de mensagens não padronizadas, cerca de 20 mil dos campos continham o termo "anexo", outras 15 mil entradas repetiam o termo "CID", já presente em outros campos.

2 Pré-processamentos realizados

No pré-processamento dos dados, alguns campos nulos foram preenchidos, outras features foram descartadas e feature engineering também foi necessário. Os campos com acima de 95% dos campos nulos foram excluídos, sendo eles:

- QT_TEMPO_DOENCA;
- DS_UNIDADE_TEMPO_DOENCA
- DS_TIPO_DOENCA;
- DS_TIPO_SAIDA;
- CD_GUIA_REFERENCIA.

Também foi excluído o campo "DS_INDICACAO_CLINICA por não seguir um padrão bem definido, repetir dados constantes em outras variáveis e ter muita coisa que não diz o necessário sobre o campo, nos casos de explicações em anexo

ou campos nulos. A feature de "CD_ITEM" também não possui informações valiosas, já que toda entrada possui um valor distinto, não levando a uma generalização do caso.
Referente ao "Feature Engineering", houve alterações nas variáveis do CID, referente ao código internacional de doenças. O CID foi manipulado para reduzir a quantidade de opções, tirando especificidades e melhor separando o campo em categorias. Reduzindo para os dois primeiros caracteres, foi possível reduzir de cerca de 1600 diferentes variáveis para apenas 200 no set de treino. Para melhorar representar as entradas, foi criado o campo "IDADE" para simplificar visualização e reduzir dados distintos, sobretudo no campo de "DT_NASCIMENTO", ou Data de nascimento. Através da função "jd_to_date", disponível nesse repositório, o dia juliano, valor inicial das variáveis de DT_NASCIMENTO e DT_REQUISICAO, foi convertido para data, tendo então dia, mês e ano, depois criado a variável "IDADE, recebendo a subtração da data de requisição da data de nascimento.
Para melhor preencher os dados de quantidade de dias solicitados, tendo uma quantia considerável como nula, dentro da variável "QT_DIA_SOLICITADO", foi criada uma função para não preencher apenas com a média dos dados atuais, mas também com o desvio padrão dos mesmos. Dessa forma, os dados seriam preenchidos dentro da faixa da média menos o desvio padrão e a média mais o desvio padrão, convertidos a números inteiros. Por fim, a função drop_duplicates, do Pandas, foi utilizada para desconsiderar variáveis repetidas, distorcendo os dados e melhorando a acurácia do algoritmo.
Por fim, as features úteis restantes foram separadas em numéricas e categóricas, da seguinte maneira:

- Numéricas:

  IDADE
  NR_SEQ_REQUISICAO
  QT_DIA_SOLICITADO
  QT_SOLICITADA

- Categóricas:

  DS_TIPO_GUIA
  DS_TIPO_PREST_SOLICITANTE
  DS_CBO
  DS_TIPO_ITEM
  DS_TIPO_CONSULTA
  DS_INDICACAO_ACIDENTE
  DS_TIPO_INTERNACAO
  DS_REGIME_INTERNACAO
  DS_CARATER_ATENDIMENTO
  DS_TIPO_ACOMODACAO

CD_CID

DS_CLASSE

DS_GRUPO

DS_SUBGRUPO

Tal separação serviu para que a codificação do OneHotEncoder e scaling do StandardScaler fossem mais simples, práticas e melhores para ler e editar posterioremente, caso necessário.

3 Configuração experimental e algoritmos utilizados

O projeto foi desenvolvido em Python versão 3.10. Já, para a importação da base de dados, análise e manipulações necessárias, o Pandas foi a biblioteca escolhida. A biblioteca do Sweetviz também foi utilizada para auxiliar na leitura dos dados, gerando um único arquvio com dados relevantes de todas as features em questão. As bibliotecas fornecidas pelo SKLearn foram de imensa serventia, possibilitando a implantação das funções de train test split para dividir a base em dados de treino e teste. Nessa função foram utilizados os parâmetros te test_size = 0.3, tendo 30% para teste e 70% para treino, e shuffle = True, buscando melhorar a distribuição de aleatoriedade dos dados. O Pré processamento utilizou as funções StandardScaler, LabelEncoder, e OneHotEncoder, da biblioteca "preprocessing" do Sklearn, separando os dados numéricos no StandardScaler, categóricos no OneHotEncoder e target (aprovado ou negado) dentro do LabelEncoder.
Random Forest Classifier, também foi o algoritmo implementado para o objetivo final devido à sua baixa taxa de overfitting e facilidade de utilização. Os parâmetros utilizados foram class_weight = 'balanced', n_estimators=50, e random_state=0, O primeiro buscando balançar as classes "aprovadas" e "negadas". Por fim também foram importadas as funções "classification_report, confusion_matrix, accuracy_score e f1_score, da biblioteca sklearn.metrics, para avaliar a performance do algoritmo.

4 Resultados

Os resultados em teste foram bastante satisfatórios, tendo acurácia de 83%, f1 score de 72% nos autorizados e 87% nos negados. A matriz de confusão é apresentada na tabela 1. A classificação de acurácia pode ser vista detalhadamente na tabela 2. O treino obteve uma acurácia final de 0.8257, ou 82,57%

		Resposta prevista	
		Autorizado	**Negado**
Resposta Verdadeira	**Autorizado**	15241	6735
	Negado	5141	41020

Tabela 1 - Matriz de Confusão.

	Precision	Recall	F1-Score	Support
Autorizado	0.75	0.69	0.72	21976
Negado	0.86	0.89	0.87	56121
Accuracy			0.83	68137
Macro avg	0.80	0.79	0.80	68137
Weighted avg	0.82	0.83	0.82	68137

Tabela 2 - Classification Report

No entanto, as taxas de acurácia decente ficaram apenas no teste. Ao submeter as respostas finais na plataforma Kaggle, a acurácia ficou reduziu para 0,67785, ou 67,78%. Cabe agora a reavaliação das features utilizadas, manipulações realizadas, até o modelo de classificação utilizado, para melhorar a performance do código e obter respostas mais satisfatórias. Também será necessário um olhar mais crítico e estatístico aos dados fornecidos para melhorar tais reavaliações, revendo a importância de cada uma e as respectivas relevância à decisão final.