Nanodegree Engenheiro de Machine Learning

Relatório de Projeto Final

Rafael Novello
21 de março de 2018

Definição

Visão Geral do Projeto

Este documento compõe o relatório do projeto final do curso Nanodegree Engenheiro de Machine learning. O projeto foi desenvolvido usando de um desafio da plataforma Kaggle que consiste na classificação de lances realizados por pessoas e robôs em um leilão digital.

O leilão é uma forma de negociação, venda ou compra, de bens ou serviços muito utilizada. A dinâmica consiste, de forma muito resumida, na oferta de um item ou lote ao qual é estabelecido um preço mínimo inicial. Os interessados no item fazem suas ofertas (lances) por um determinado período e adquire o item aquele que fizer o maior lance até o fim do período do leilão.

Uma modalidade de leilão muito popular no Brasil é o Pregão Eletrônico que é realizado pelo governo em processos de licitação (aquisição de produtos ou serviços pelo governo). O pregão tem uma dinâmica inversa ao leilão. Neste, o governo anuncia publicamente a necessidade de compra ou contratação e os interessados competem entre si para alcançar o menor preço.

Segundo dados no Ministério do Planejamento, em 2013 o pregão eletrônico proporcionou uma economia de 9,1 bilhões de reais nas compra públicas do governo federal [1].

Descrição do Problema

Como descrito na seção anterior, o objetivo do leilão, assim como, e principalmente, do pregão eletrônico é permitir a competição igualitária entre os interessados. Porém, o uso de robôs (softwares que automatizam tarefas, neste caso o envio do lance ao leilão ou pregão) vem prejudicando sites de leilão e o pregão eletrônico do governo brasileiro, como podemos ver em várias matérias jornalísticas [2], [3], [4]

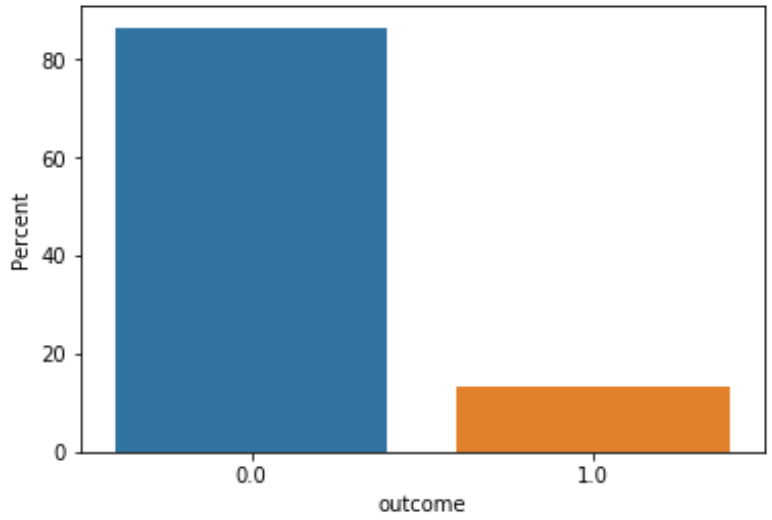
Identificar lances feitos por robôs é um grande desafio para plataformas de leilão, que precisam garantir a igualdade entre os “competidores”. Apenas assim as plataformas serão capazes de evitar que todos os ganhadores serão sistemas automatizados.

Para a identificação dos lances feitos por robôs e por operadores humanos vamos usar uma base de lances históricos feitos em uma plataforma de leilão online. Nesta base os operadores foram categorizados em robôs e humanos e desta forma podemos identificar todos os lances feitos por cada categoria de operador. Esta será nossa entrada.

O resultado esperado é a identificação dos lances feitos por cada uma das categorias de operadores através de uma variável binária, onde o valor 1 representa um lance feito por robô e o valor 0 representa um lance feito por humano. Esta será nossa saída.

Para obter o resultado esperado será usado um modelo de aprendizagem supervisionada, um classificador binário, que será melhor detalhado a frente.

Métricas



Como apresentado na imagem acima, a quantidade de amostras pertencentes a categoria 0 (humanos) é muito maior que a categoria 1 (robôs). A proporção das categorias no dataset é de 87% lances válidos e 13% de lances feitos por robôs.

Devido a natureza desbalanceada dos dados e a importância de não descartar erroneamente um lance feito por um usuário real da plataforma, as métricas de avaliação usadas foram Precision, Recall e, para auxiliar foi usando ROC (Receiver Operating Characteristic ou Características operacionais do receptor em tradução livre).

Precision e Recall foram usados por lidarem bem com conjuntos desbalanceados e por permitirem comparar objetivamente os resultados de cada modelo testado. O índice ROC foi usado para avaliar a performance do modelo criado em relação ao desafio na plataforma Kaggle.

Análise

Exploração dos dados

Os dados disponibilizados na plataforma Kaggle para este desafio [5] estão organizados da seguinte forma:

Para o conjunto de dados do licitante (participante do leilão)

- bidder_id - Identificador exclusivo de um licitante.
- payment_account - conta de pagamento associado a um lance. Estes são obscuros para proteger a privacidade.

- endereço - endereço de correspondência de um licitante. Estes são obscuros para proteger a privacidade.
- Resultado - Etiqueta de um licitante que indica se é ou não um robô. O valor 1.0 indica um robô, onde o valor 0.0 indica humano.

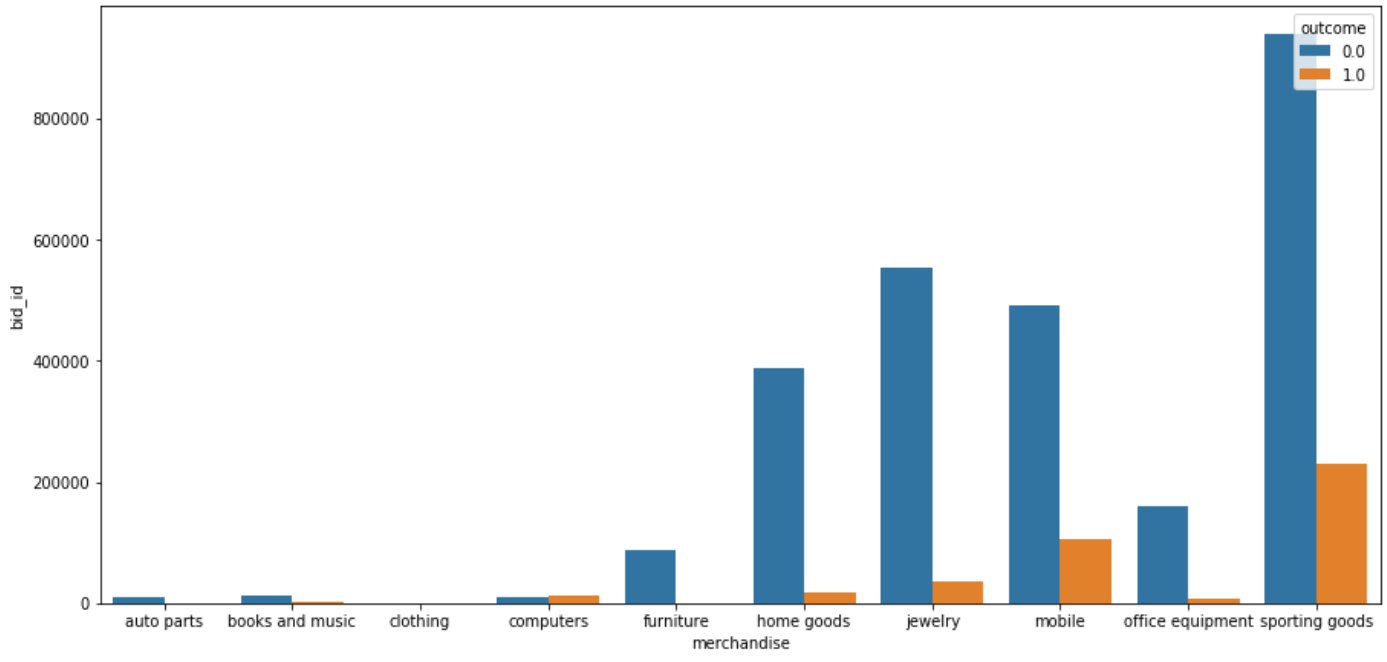
Para o conjunto de dados de licitação

- bid_id - ID exclusivo para este lance
- bidder_id - Identificador exclusivo de um licitante (o mesmo que o bidder_id usado no train.csv e test.csv)
- Leilão - Identificador exclusivo de um leilão
- mercadoria - A categoria da campanha do site de leilões, o que significa que o licitante pode chegar a este site por meio da busca de "bens domésticos", mas acabou por licitar "bens esportivos" - e isso leva a que esse campo seja "bens domésticos". Este campo categórico pode ser um termo de pesquisa ou propaganda online.
- dispositivo - modelo de telefone de um visitante
- Hora - Tempo em que a oferta é feita (transformada para proteger a privacidade).
- país - O país ao qual o IP pertence
- IP - endereço IP de um licitante (ofuscado para proteger a privacidade).
- url - url do qual o licitante foi encaminhado (ofuscado para proteger a privacidade).

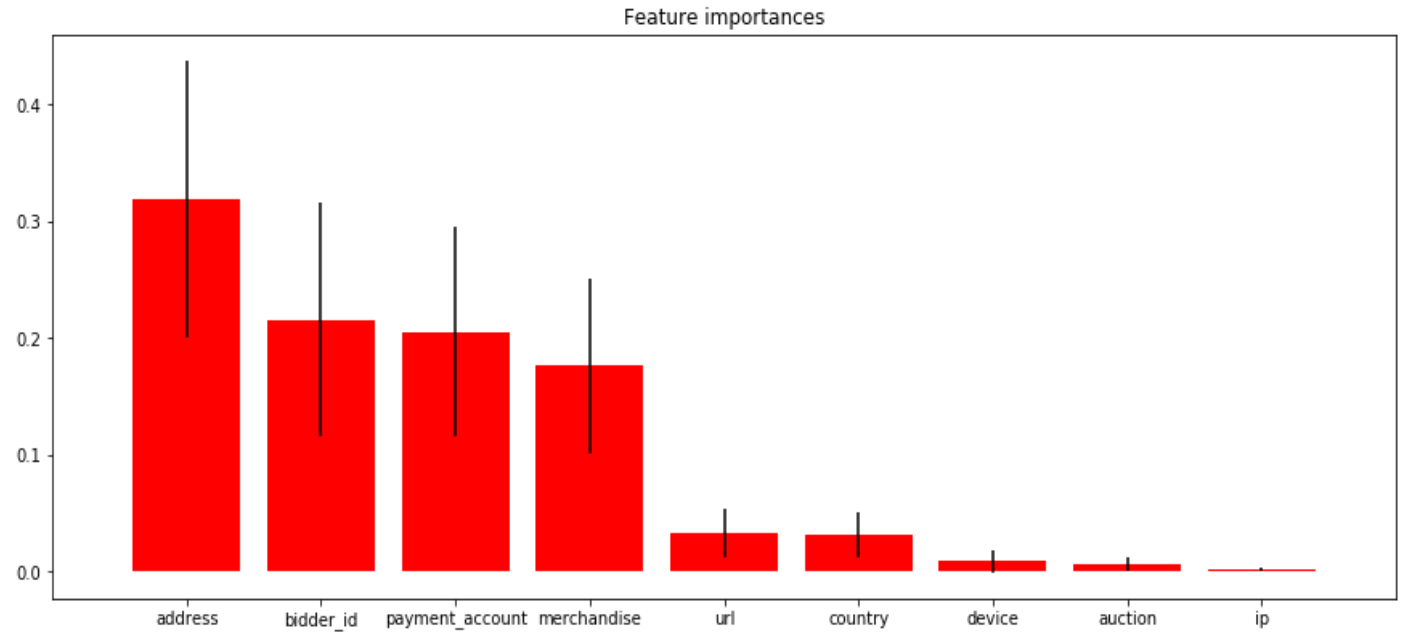
A base de lances contém 7.656.334 registros que são divididos em 3.071.224 no conjunto de treinamento (que deve ser dividido em treinamento e validação) e 4.585.110 no conjunto de testes. O conjunto de testes não contém a variável alvo, que deve ser inserida pelo resultado do modelo treinado para submissão na plataforma Kaggle e avaliação.

Na base de treinamento a proporção de lances feitos por humanos é de 87% e 13% feitos por robôs, o que mostra que o dataset está desbalanceado e por isso deve-se tomar algumas precauções.

Para o melhor entendimento dos dados, as features foram apresentadas separando por lances feitos por humanos e por robôs o que gerou a visualização a seguir:



Esta visualização mostra a diferença de concentração dos lances feitos por robôs por cada feature, o que levou a uma avaliação da importância de cada feature para a distinção entre as categorias.



Esta visualização mostra que apenas 4 das 9 features realmente contribuem para a distinção entre as categorias, o que indica a necessidade de aplicação de técnicas de feature engineering para agregar mais informação ao dataset. Tais técnicas de feature engineering podem ser muito avançadas e fogem do escopo do projeto mas são uma sugestão de próximos passos.

Algoritmos e técnicas

Por se tratar de um problema de classificação entre duas categorias onde existe uma base para treinamento com a variável alvo preenchida, foi usada uma abordagem de aprendizado supervisionado, onde o(s) algoritmo(s) são previamente treinados com esta base classificada e depois avaliado(s) com uma amostra sem a variável alvo.

Em relação aos algoritmos usados, foram testados com sucesso os seguintes algoritmos:

- Nearest Neighbors
- Random Forest
- Neural Net
- AdaBoost
- Naive Bayes
- Quadratic Discriminant Analysis

O objetivo em treinar uma variedade de algoritmos distintos era observar aquele que apresentaria a melhor performance na classificação dos dados de treinamento, para então otimizar os parâmetros daquele que apresentasse os melhores resultados.

Benchmark

Por se tratar de um desafio na plataforma Kaggle, é provável que a melhor referência para comparação seja o ranking associado ao desafio. Neste ranking [6] o melhor modelo teve uma taxa de

acertos de 94,3% e as melhores 50 submissões ficaram acima de 92,3% de acerto. A plataforma Kaggle usa a métrica ROC (Receiver Operating Characteristic) como avaliação dos resultados apresentados pelos competidores.

Para apoiar durante o processo de treinamento e escolha do melhor modelo, foi utilizar o algoritmo Dummy Classifier da biblioteca SciKit-Learn [7] como baseline de comparação.

Metodologia

Pré-processamento de dados

Dada a existência de variáveis categóricas no dataset de treinamento, foi necessário aplicar um processo de encoding nas mesmas. Este processo foi feito com a ajuda da classe LabelEncoder da biblioteca SciKit-Learn [8].

Implementação

O projeto iniciou com a análise dos dados de treinamento, verificando se existe ausência de dados para alguma feature e fazendo algumas visualizações como para a distribuição entre os lances feitos por robôs e por humanos. O objetivo era saber qual a proporção do desbalanceamento do dataset entre os lances feitos por humanos e por robôs.

Em seguida foi verificada a correlação entre as features do modelo e a variável alvo. Neste processo foi utilizado um algoritmo de árvore de decisão para avaliar as features mais relevantes para a classificação.

Na sequência foi aplicado o pré-processamento descrito na seção anterior, transformando features categóricas com um processo de encoding. Este processo é necessário para a maioria dos modelos de machine learning.

Após o pré-processamento foi possível testar os modelos descritos na seção “Algoritmos e técnicas” com um subset dos dados de treinamento. Para esta fase foram utilizados os parâmetros padrão de cada modelo, pois o objetivo é descobrir apenas qual deles apresenta melhor taxa de acertos nas classificações.

Ao descobrir o melhor modelo, a última etapa foi a otimização do mesmo. Neste processo foi usada a classe GridSearch da biblioteca Scikit-Learn para encontrar o melhor conjunto de parâmetros para o modelo.

Resultados

Modelo de avaliação e validação

Os modelos citados na seção “Algoritmos e técnicas” foram treinados usando os parâmetros padrão e foi obtido dos mesmos as métricas ROC score, Precison e Recall e uma média entre as três métricas

foi calculada para cada modelo. Como é possível ver na tabela abaixo, os modelos usando os algoritmos AdaBoost e Random Forest apresentaram os melhores resultados médios usando os parâmetros padrão.

Out[4]:

	ROC score	precision	recal	mean
name				
Nearest Neighbors	0.812510	0.695071	0.535943	0.681175
Random Forest	0.977921	0.998034	0.610606	0.862187
Neural Net	0.568696	0.170755	0.459884	0.399778
AdaBoost	0.975941	0.933923	0.664541	0.858135
Naive Bayes	0.764188	0.972258	0.183873	0.640106
QDA	0.828702	0.781649	0.411405	0.673919

O modelo escolhido foi o AdaBoost pois o mesmo apresentou o melhor Recall entre os modelos testados, mesmo ficando um pouco atrás na média entre as métricas calculadas se comparado com Random Forest. Esta escolha foi tomada pois tanto o modelo AdaBoost quanto o Random Forest tiveram ótimos resultados nas métricas ROC e Precision mas o modelo escolhido apresentou melhor Recall.

Justificativa

O modelo obtido foi comparado com um modelo de baseline criado com a ajuda do algoritmo Dummy Classifier da biblioteca SciKit-Learn e com o scores apresentados no desafio Kaggle. Abaixo seguem os resultados das comparações:

Métrica / Modelo	Dummy Classifier	AdaBoost Classifier
Precision	13,4%	96,2%
Recall	13,3%	81,8%
ROC Score	49,9%	99,3%

Quando comparado com os modelos submetidos no desafio Kaggle, o modelo AdaBoost performou bem já que as melhores 50 submissões ficaram acima de 92,3% de ROC Score.

Conclusão

Reflexão

Ao escolher este desafio Kaggle a impressão inicial é que este seria um projeto relativamente simples pois, apesar do grande volume de dados, trata-se de uma classificação binária entre lances feitos por pessoas reais e por robôs autômatos em um leilão online. Porém, ao iniciar a manipulação dos dados

surgiu o primeiro desafio, o dataset desbalanceado. Este foi um aspecto interessante e desafiador do projeto que vale a pena ser melhor abordado como sugestões de melhorias futuras.

Outro ponto bastante interessante deste projeto foi a necessidade de engenharia de features. Em consulta no fórum da plataforma Kaggle e na primeira avaliação do mentor da Udacity ficou claro a necessidade de se aprofundar em técnicas de engenharia de features que fogem ao escopo do projeto de capstone mas que podem ser aplicadas futuramente.

Por final, apesar das dificuldades, os resultados obtidos foram satisfatórios em vista das técnicas aplicadas e o projeto permitiu a consolidação e aprofundamento dos conhecimentos adquiridos neste Nanodegree.

Melhorias

Como apontado na seção anterior, os principais pontos de melhoria observados estão relacionados ao desbalanceamento do dataset e a aplicação de técnicas de engenharia de features e estes pontos serão discutidos mais detalhadamente abaixo:

- Sobre o desbalanceamento do dataset: Trabalhar com datasets desbalanceados representa um grande desafio e durante os estudos para a solução deste projeto foram observadas duas principais técnicas para lidar com este cenário que são o Oversampling e undersampling [9]. No primeiro, o objetivo é aumentar a quantidade de registros da categoria com menos amostras enquanto no segundo o objetivo é o oposto e aplicado à categoria com mais amostras. Estas técnicas podem ser aplicadas a este projeto para balancear o dataset e assim obter melhores resultados.
- Sobre a aplicação de engenharia de features: O objetivo das técnicas de engenharia de features é, em resumo, aumentar a quantidade de features utilizando técnicas de combinação e transformação das features existentes. Estas técnicas podem ser aplicadas a este projeto com o objetivo de melhorar a capacidade do modelo em distinguir as classes de lances.

Referências

1. http://agenciabrasil.ebc.com.br/economia/noticia/2014-02/com-pregao-eletronico-governo-economizou-r-91-bilhoes-em-2013
2. https://istoe.com.br/139247_GOLPE+NO+PREGAO+ELETRONICO/
3. https://idag.jusbrasil.com.br/noticias/2611942/pregao-eletronico-robos-ganham-licitacoes
4. http://g1.globo.com/tecnologia/blog/seguranca-digital/post/robos-participam-de-pregoes-compram-ingressos-e-atuam-na-bolsa.html
5. https://www.kaggle.com/c/facebook-recruiting-iv-human-or-bot/data
6. https://www.kaggle.com/c/facebook-recruiting-iv-human-or-bot/leaderboard
7. http://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyClassifier.html
8. http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html
9. https://en.wikipedia.org/wiki/Oversampling_and_undersampling_in_data_analysis