Nanodegree Engenheiro de Machine Learning

Projeto final

Juan Eduardo Cruz Maldonado

02 de Janeiro de 2018

I. Definição

Visão geral do projeto

Este projeto consiste em resolver um desafio [7] criado pelo facebook e disponibilizado pela plataforma Kaggle, plataforma voltada para Data Science. No desafio serão utilizadas técnicas de machine learning como aprendizagem supervisionada para classificação de classes. O Kaggle disponibiliza uma base de dados para trabalharmos em um problema que a própria tecnologia nos trouxe, identificar robôs em sites de leilão, a prática de fraudes em leilões já era comum[1], mas a IA trouxe novas "oportunidades", e essas tecnologias também estão sendo utilizadas em outros segmentos do mercado[3][4][9][10[11]. Este tipo de robô é ilegal, inclusive há empresas que foram acusadas de praticar este tipo de atividade e foram fechadas[8].Nos sites de leilões, de forma automática os robôs fazem lances[2] com mais precisão para ter sucesso de acordo com seu objetivo, evitando com que os humanos consigam ganhar o leilão ou pagar um preço justo. Os lances dos robôs são fraudes e para manter a confiabilidade no leilão, neste projeto irei criar um classificador binário para de identificar quais lances são feitos por robôs e quais são feitos por humanos, assim evidenciando as fraudes a serem removidas dos leilões.

Descrição do problema

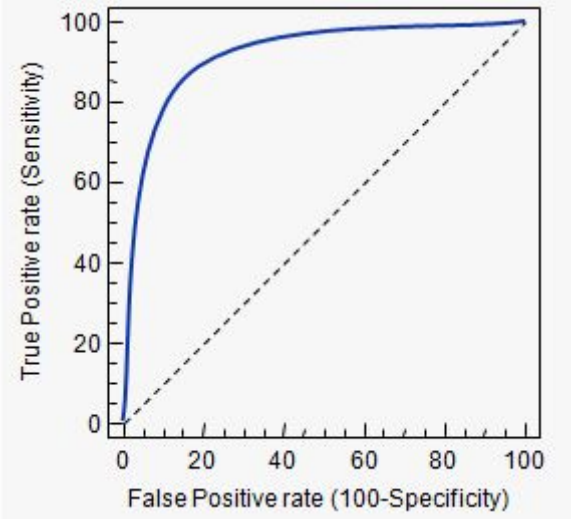
A proposta do desafio Kaggle, é identificar robôs em sites de leilão. Como os robôs trabalham de forma automatizada conseguem ter uma resposta mais rápida que os humanos aos leilões (ganhando os leilões, ou fazendo o aumento de preço do produto), fazendo assim com que os licitantes humanos se frustrem com os leilões que acabam desistindo de utilizar este tipo de serviço.Este projeto trata-se de aprendizagem supervisionada, onde irei construir um classificador

binário para identificar quais lances são provenientes de um robô, para assim excluí-los dos leilões e evitar fraudes nos lances.

O arquivo de dados cedido pelo kaggle [6] separa os lances de humanos e robôs como:
Humanos = 0
Robôs = 1

Métricas

A métrica para realizar este projeto foi estabelecida pelo próprio desafio, será utilizada a métrica Roc Curve, uma métrica usada para medir modelos classificadores. Nosso problema tem dados desbalanceados(em breve será mais aprofundado), a AUC ROC funciona muito bem com conjuntos com essa característica.



Roc Curve normalmente mostra as taxas positivas verdadeiras no eixo Y, e taxa de falsos positivos no eixo X. Com isso quanto mais linha do gráfico for para o canto esquerdo superior melhor é resultado da curva.

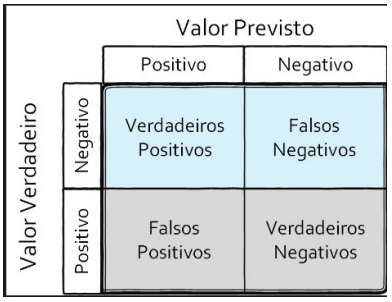
A "inclinação" das curvas ROC também é importante, já que é ideal para maximizar a taxa positiva verdadeira enquanto minimiza a taxa de falsos positivos.

Conceitos básicos :

verdadeiro positivo - uma instância positiva que é corretamente classificada como positiva;
falso positivo - uma instância negativa que é incorretamente classificada como positiva;

verdadeiro negativo - uma instância negativa que é corretamente classificada como negativa;
falso negativo - uma instância positiva que é incorretamente classificada como negativa;

Para ajudar na validação visual também usarei a matriz de confusão, que serve para verificar se o algoritmo está confundindo duas classes. Cada coluna da matriz representa instâncias de uma classe prevista, e as linhas representam os casos de uma classe real. Visualmente fica mais claro de identificar os erros dos algoritmo.



Conforme evolui com projeto encontrei outras duas métricas para avaliar o modelo.

F1 score :

Em estatística, o F Score (ou F-measure) é uma medida da acurácia do teste. Ele considera tanto a precision p e a recall r do teste para calcular a pontuação: p é o número de resultados positivos corretos dividido pelo número de todos os resultados positivos devolvidos pelo classificador, e r é o número de resultados positivos corretos dividido pelo o número de todas as amostras relevantes (todas as amostras que deveriam ter sido identificadas como positivas). O F Score pode ser interpretado como uma média ponderada de precision e da recall, em que o F score alcança seu melhor valor em 1 e o pior escore em 0. A fórmula para a pontuação de F é:

F1 = 2 * (precisão * recall) / (precisão + recall)

II. Análise

Exploração dos dados

Foram disponibilizados dois conjuntos de dados para resolver o desafio, um conjunto com dados do licitante do leilão, com dados como id, conta de pagamento e endereço.
O segundo conjunto de dados tem os dados dos lances, com 7,6 milhões de lances em diferentes leilões. A plataforma de leilão on-line tem um incremento fixo de valor em dólar para cada lance, portanto, não inclui um valor para cada lance.

Descrição dos arquivos

- train.csv - conjunto de treino de licitantes
- test.csv - conjunto de teste para licitantes
- sampleSubmission.csv - exemplo de submissão de arquivo
- bids.csv - conjunto de dados de lances

Colunas do conjunto de dados

train.csv(usuário, o atributo "outcome" define se é robô ou não.)

- **bidder_id** - identificador exclusivo de um licitante.
- **payment_account** - conta de pagamento associada a um licitante. Estes são criptografados para proteger a privacidade.
- **address** - Endereço para correspondência de um licitante. Estes são criptografados para proteger a privacidade.
- **outcome** - Indica se um licitante é ou não um robô. O valor 1.0 indica um robô, o valor 0.0 indica humano.

bids.csv(lances nos leilões)

- **bid_id** - id único para um lance
- **bidder_id** - Identificador exclusivo de um licitante(o mesmo usado em train.csv e test.csv)
- **auction** - identificador único de um leilão
- **merchandise** - A categoria da campanha do site de leilões, que significa que o licitante pode chegar a este site pesquisando "produtos domésticos", mas acabou fazendo lances para "artigos esportivos" - e isso faz com que esse campo seja "produtos domésticos". Esse campo categórico pode ser um termo de pesquisa ou um anúncio on-line.
- **Device** - dispositivo do licitante
- **time** - Horário em que o lance é feito (criptografados para proteger a privacidade).

- **Country** - o país ao qual o IP pertence
- **ip** - endereço IP de um licitante (criptografados para proteger a privacidade).
- **url** - url de onde o licitante foi encaminhado (criptografados para proteger a privacidade).

Amostra do conjunto de dados de lances (bids.csv)

	bid_id	bidder_id	auction	merchandise	device	time	country	ip	url
0	0	8dac2b259fd1c6d1120e519fb1ac14fbqvax8	ewmzr	jewelry	phone0	9759243157894736	us	69.166.231.58	vasstdc27m7nks3
1	1	668d393e858e8126275433046bbd35c6tywop	aeqok	furniture	phone1	9759243157894736	in	50.201.125.84	jmqlhflrzwuay9c
2	2	aa5f360084278b35d746fa6af3a7a1a5ra3xe	wa00e	home goods	phone2	9759243157894736	py	112.54.208.157	vasstdc27m7nks3
3	3	3939ac3ef7d472a59a9c5f893dd3e39fh9ofi	jefix	jewelry	phone4	9759243157894736	in	18.99.175.133	vasstdc27m7nks3
4	4	8393c48eaf4b8fa96886edc7cf27b372dsibi	jefix	jewelry	phone5	9759243157894736	in	145.138.5.37	vasstdc27m7nks3
5	5	e8291466de91b0eb4e1515143c7f74dexy2yr	3vi4t	mobile	phone7	9759243157894736	ru	91.107.221.27	vasstdc27m7nks3
6	6	eef4c687daf977f64fc1d08675c44444raj3s	kjlzx	mobile	phone2	9759243210526315	th	152.235.155.159	j9nl1xmo6fqhcc6

Após unificar os dois datasets verifiquei se há ruídos para tratar, como por exemplo dados nulos :

Um tratamento na coluna country teve que ser realizado.

	Valores Faltantes	% do total
bid_id	0	0.00
bidder_id	0	0.00
auction	0	0.00
merchandise	0	0.00
device	0	0.00
time	0	0.00
country	8859	0.12
ip	0	0.00
url	0	0.00

Valores totais distintos agrupados por atributos

```
Total da coluna bid_id 7,656,334
Total da coluna bidder_id 6,614
Total da coluna auction 15,051
Total da coluna merchandise 10
Total da coluna device 7,351
Total da coluna time 776,529
Total da coluna country 200
Total da coluna ip 2,303,991
Total da coluna url 1,786,351
```

Amostra do Conjunto de dados Licitantes(train.csv)

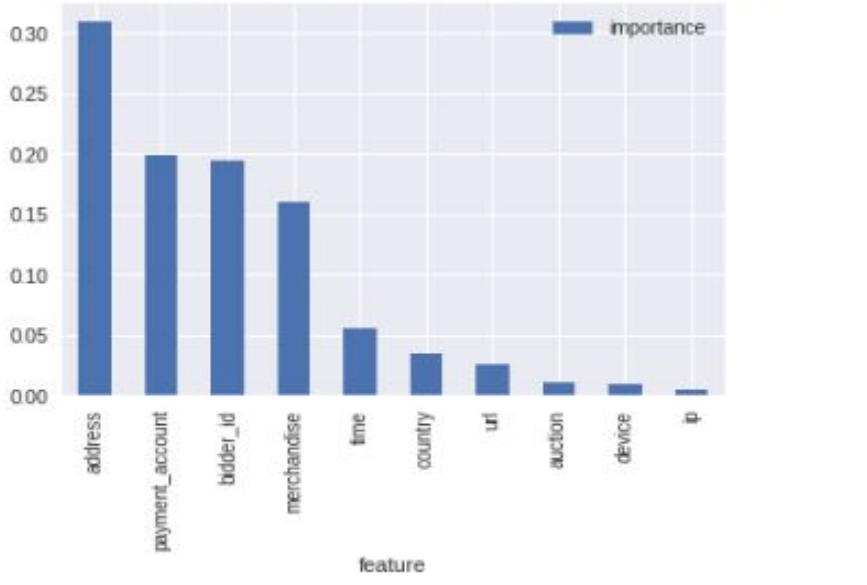
	bidder_id	payment_account	address	outcome
0	91a3c57b13234af24875c56fb7e2b2f4rb56a	a3d2de7675556553a5f08e4c88d2c228754av	a3d2de7675556553a5f08e4c88d2c228vt0u4	0.0
1	624f258b49e77713fc34034560f93fb3hu3jo	a3d2de7675556553a5f08e4c88d2c228v1sga	ae87054e5a97a8f840a3991d12611fdcrfbq3	0.0
2	1c5f4fc669099bfbfac515cd26997bd12ruaj	a3d2de7675556553a5f08e4c88d2c2280cybl	92520288b50f03907041887884ba49c0cl0pd	0.0
3	4bee9aba2abda51bf43d639013d6efe12iycd	51d80e233f7b6a7dfdee484a3c120f3b2ita8	4cb9717c8ad7e88a9a284989dd79b98dbevyi	0.0
4	4ab12bc61c82ddd9c2d65e60555808acqgos1	a3d2de7675556553a5f08e4c88d2c22857ddh	2a96c3ce94b3be921e0296097b88b56a7x1ji	0.0

Analisando o arquivo Train.csv verifiquei 1910 humanos e 103 robôs, sendo assim em proporção 94,88% do dataset está marcado como humano e o restante 5,12% está marcado como robô. São poucos dados marcados como Robô, característica de um desbalanceamento dos dados[5].

Visualização exploratória

Para realizar a primeira classificação com os dados, após realizar a limpeza no dataset, precisei identificar quais seriam os atributos mais relevantes para classificar as classes, para isso utilizei a técnica de “feature importance” da biblioteca sklearn.ensemble.ExtraTreesClassifier(no arquivo EDA.ipynb utilizei outras formas e cheguei no mesmo resultado nas features).

```
                importance
feature
address              0.309
payment_account      0.198
bidder_id            0.194
merchandise          0.159
time                 0.055
country              0.035
url                  0.026
auction              0.011
device               0.009
ip                   0.004
<matplotlib.axes._subplots.AxesSubplot at 0x7fecdb10fa10>
```



As features mais relevantes para o modelo são Address, payment_account, bidder_id e merchandise respectivamente. Poderia incluir o atributo “time”, mas o valor de diferença entre o atributo merchandise é grande, por tanto vou deixar de fora, acredito também que o time seria um dos atributos mais importantes, mas teria que haver um tratamento diferente para ele, pois o tempo está criptografado neste desafio.

Algoritmos e técnicas

Durante o projeto decidi utilizar algoritmos de classificação com características diferentes e analisar o resultado de como cada um conforme ajustava os parâmetros.

O algoritmos utilizados foram :

Naive Bayes (GaussianNB):

Naive Bayes é um tipo de classificador baseado no Teorema de Bayes. Ele prevê as probabilidades de associação para cada classe, como a probabilidade de que determinado registro ou ponto de dados pertença a uma determinada classe (humanos ou robôs). A classe com maior probabilidade é considerada como a classe mais provável.Porém ele não consegue aprender relação entre as classes. Ele estima sem relacionar os atributos do dataset.

No projeto o utilizado o Gaussian Naive Bayes, que é usado na classificação e assume uma distribuição normal.

Parâmetro(s) usado :

priors : Probabilidades anteriores das classes. Se especificado, os antecedentes não são ajustados de acordo com os dados.

Random Forest (RandomForestClassifier):

Random Forest é um algoritmo que pode ser utilizado para classificação ou regressão. é um tipo de ensemble learning, método que gera muitos classificadores e combina o seu resultado. A Random Forest cria vários decision trees, cada um com suas particularidades e combina o resultado da classificação de todos eles.

Parâmetros utilizados :

n_estimators: Número de estimadores (Decision Trees) que serão utilizados pelo Random Forest.
n_jobs: Número de execuções em paralelo que serão usadas pelo seu modelo, ao

passar -1, o valor será igual ao número de núcleos do computador executando. Quanto mais paralelizado for a execução, mais rápido será,mas é necessário um hardware que comporte a execução.
max_depth: A profundidade máxima da árvore. Se Nenhum, os nós serão expandidos até que todas as folhas/nós fiquem puras ou até que todas as folhas contenham menos de amostras de min_samples_split.
min_samples_split : é o número mínimo de amostras para cada divisão/nó.

SGDClassifier

SGD é um classificador, que implementa uma rotina de aprendizado de descida de gradiente estocástica simples que suporta diferentes funções de perda e penalidades para classificação. Algoritmo eficiente que consegue trabalhar com problemas de larga escala. É de fácil implementação e tem muitos parâmetros para otimizar o algoritmo.

penalty :A penalidade (termo de regularização) a ser usada. O padrão é 'l2', que é o regularizador padrão para modelos lineares de SVM. 'l1' e 'elasticnet' podem trazer esparsidade ao modelo (seleção de recursos) não alcançável com 'l2'.

max_iter :O número máximo de iterações sobre os dados de treinamento. Isso afeta apenas o comportamento no método fit e não o partial_fit .

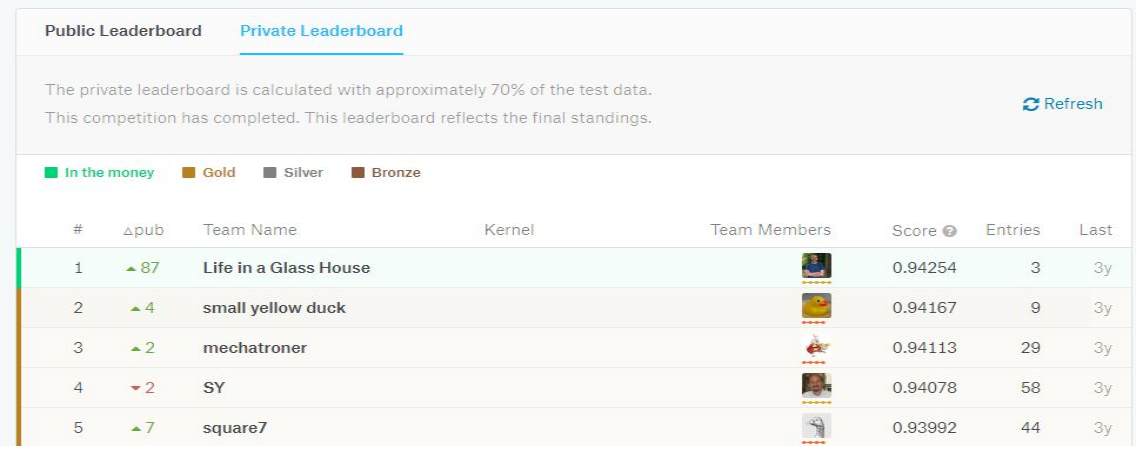
VotingClassifier
Combina classificadores conceitualmente diferentes, e utiliza um votação ou as probabilidades médias previstas para prever o tipo/Rótulo das classes. Este modelo é bom para ser utilizados com modelos com bom desempenho para equilibrar suas fraquezas individuais.

XGBClassifier
Extreme Gradient Boosting O XGBoost implementa um algoritmo de Gradient Boosting baseado em árvores de decisão.O Algoritmo vai criando árvores intituladas como fracas, e cada próxima vai corrigindo o que a anterior não foi capaz de prever até encontrar o número de estimadores(árvores) para o modelo.

Benchmark

O desafio já tem uma solução, no site do kaggle podemos ver o leaderboard com os resultados. Meu benchmark será o vencedor do desafio com o score de

0.94254



Metodologia

Pré-processamento de dados

Fiz a unificação dos datasets de bids e train e realizei o data clean, removendo dados nulos e ruídos. Criei uma árvore de decisão para descobrir os atributos mais importantes para o modelo classificar o target corretamente. Identifiquei dados nulos e realizei os tratamentos necessários com a biblioteca scikit learn. As colunas do dataset contém dados do tipo string, para utilizar o dataset nos algoritmos de classificação deve ser feito a conversão para dados categóricos onde usei o LabelEncoder (biblioteca do scikit learn para pré processamento). A divisão dos dados foi de 70% para treino e 30% para teste. Para a avaliação do classificador irei realizar validação cruzada com 5 k-folds, com a biblioteca de model_selection-stratifiedKFold.

Implementação

A implementação dos algoritmos foi realizada após uma análise exploratória nos datasets presente no arquivo “EDA - ROBOT vs HUMAN.ipynb” , que me levou as primeiras hipóteses para a classificação. Na exploração de dados fiz uma breve classificação e já constatei que algumas alterações deveriam ser realizadas nas features do projeto.

Como citado anteriormente na metodologia apliquei pré processamentos nos dados para remover dados nulos, e fiz um join entre os datasets bids e train, depois apliquei um algoritmo de “feature importance” para encontrar quais features poderias ser mais importantes para classificar corretamente os lances.

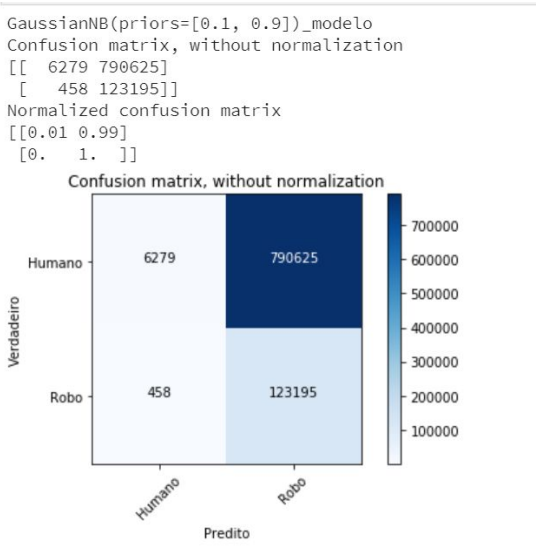
Após a exclusão dos atributos desnecessários apliquei o algoritmo da

curva de Roc em 3 classificadores que escolhi para este projeto (GaussianNB,RandomForestClassifier,SGDClassifier). Coloquei alguns parâmetros somente para verificar como os algoritmos se comportam após o pré-processamento.

Tabela de Scores ROC AUC.

Model	Score
GaussianNB	0.7902411219111749
RandomForestClassifier	0.9997274997250853
SGDClassifier	0.7024554102563458

Apliquei uma matriz de confusão para cada modelo, para entender se haviam erros na classificação(matrizes se encontram no arquivo EDA).



Nesta imagem , referente ao modelo gaussiano,

O modelo classificou 6279 instâncias como humanos e que realmente eram humanos.
O modelo classificou 790625 instâncias como Robôs que na verdade eram Humanos.
O modelo classificou 458 instâncias como Humanos que na verdade eram Robôs.
O modelo classificou 123195 instâncias como Robô e que realmente eram Robôs.
Ou seja nossa taxa de acertos, relativos aos verdadeiros/falsos ainda não está ideal, ajustes devem ser feitos nos modelos.

Arquivo enviados para o Kaggle :

Submission and Description	Private Score	Public Score
3_sub.csv just now by Juan Eduardo Maldonado SGDClassifier(alpha=0.0001, average=False, class_weight=None, epsilon=0.1, eta0=0.0, fit_intercept=True, l1_ratio=0.15, learning_rate='optimal', loss='log', max_iter=1000, n_iter=None, n_jobs=1, penalty='l2', power_t=0.5, random_state=None, shuffle=True, tol=None, verbose=0, warm_start=False)	0.49100	0.50353
2_sub.csv a minute ago by Juan Eduardo Maldonado RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini', max_depth=8, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=4, min_weight_fraction_leaf=0.0, n_estimators=300, n_jobs=1, oob_score=False, random_state=66, verbose=0, warm_start=False)	0.49388	0.48080
1_sub.csv 2 minutes ago by Juan Eduardo Maldonado GaussianNB(priors=[0.1, 0.9])	0.50505	0.50415

Nesta etapa já sabia que o score não era o ideal consegui mais de 50% com o modelo , e pelo menos gostaria de passar dos 51%, deveria testar outros algoritmos ou mudar os parâmetros. Nas pesquisas vi que o F1 Score e o Precision podem ajudar a interpretar melhor a curva de Roc.

Refinamento

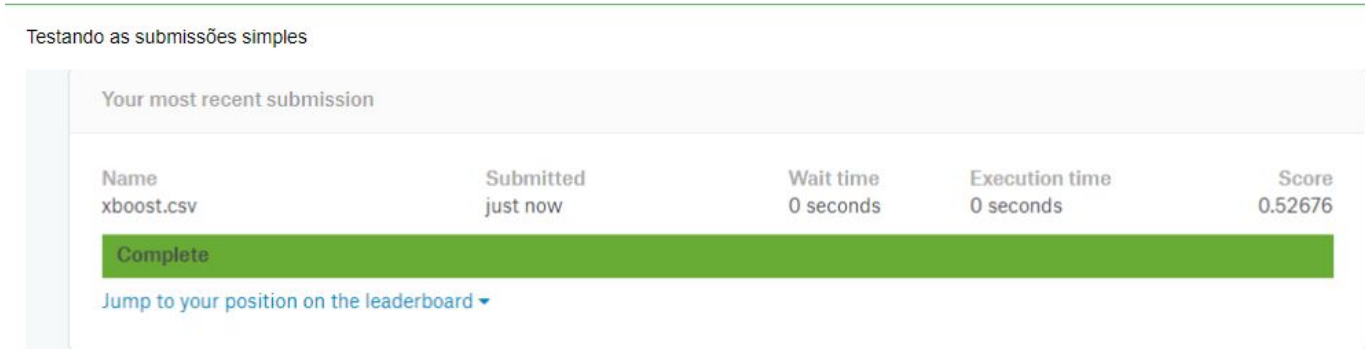
Realizei alguns testes com os parâmetros dos algoritmos citados anteriormente, sem muito sucesso optei por fazer a implementação de outros algoritmos de classificação.

O primeiro a ser testado foi o voting classifier, que também não obteve resultados muito diferentes dos demais classificadores.

VotingModel.csv a month ago by Juan Eduardo Maldonado add submission details	0.46863	0.50825	☐

Outra implementação foi do conhecido algoritmo xgboost muito utilizado nos desafios Kaggle.
Já no primeiro teste já obtive melhoras no modelo, e junto utilizei outras métricas de score para auxiliar no entendimento do algoritmo.

```
0.9999974885013277
F1 score: 0.5820327657842683
Precision: 0.7015000285176525
```



Como obtive um melhor resultado com este modelo, trabalhei mais nos parâmetros dele para ver o quanto poderia deixar mais preciso.

Parâmetros	F1 Score	Precision	Roc Score	Kaggle
XGBClassifier(n_estimators=300, learning_rate=0.05, max_depth=8)	0.5820327657842683	0.7015000285176525	0.9999974885013277	%52.676

Utilizando o GridSearchCV, consegui uma melhora :

GridSearchCV	F1 Score	Precision	Kaggle
GridSearchCV(model, parametros_grid, scoring="accuracy", n_jobs=-1, cv=kfold)	F1 score: 0.5582381268139966	0.8511759733479191	% 52.762

Realmente o Xgboost foi mais eficiente, e a utilização do gridSearch facilitou para encontrar os melhores parâmetros para o modelo com o método best_params(o código aplicado está todo no arquivo capstone.py).

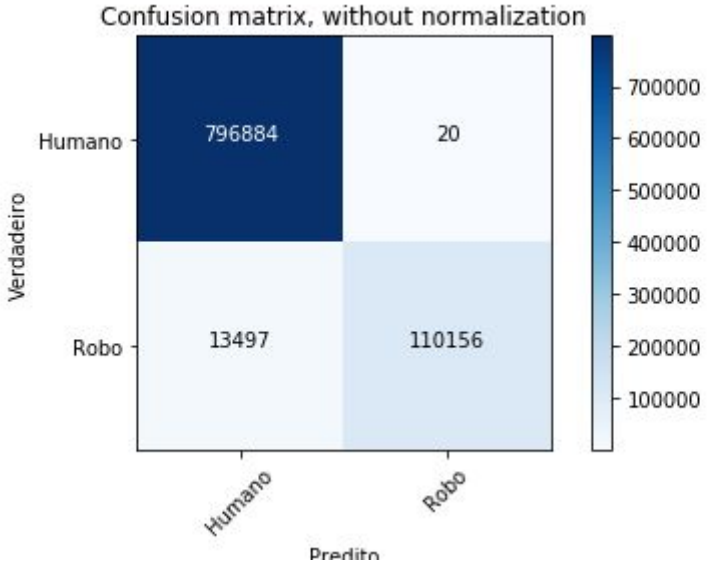
Resultados

O modelo final escolhido foi a implementação do XGboost com o Gridsearch, onde alcancei o maior o score nas classificações.

Comparando os modelos :

Modelo	Roc Auc	Kaggle
GaussianNB	0.7902411219111749	50,4
RandomForestClassifier	0.9997274997250853	48
SGDClassifier	0.7024554102563458	50,4
VotingClassifier	0.9755544149282063	50,8
XGBClassifier	0.9999974885013277	52.62
XGBClassifier + GRidSerach	0.9999944133193914	52.76

A curva de ROC não quer dizer muito sozinha, incluir os scores de precision e F1, ajudaram a visualizar melhor, conforme arquivo “capstone”. A Matriz de confusão também melhorou a visualização da taxa de erros, a melhor matriz foi da Árvore de decisão o que me levou a utilizar o modelo do Xbg que trabalha como uma árvore melhorada conforme citado anteriormente.



Os melhores parâmetros para o modelo foram

{'n_estimators': 350, 'learning_rate': 0.1, 'max_depth': 10, 'min_child_weight': 1}
E o melhor score :

0.9999944133193914

Comparado aos outros modelos como mostrado anteriormente ele teve o maior ROC Auc Score e “Kaggle Score”.

Justificativa

O modelo vencedor da competição teve um score muito alto, de 94%, os algoritmos que testei não conseguiram chegar perto do Benchmark.

Modelo	Kaggle
Benchmark	**94%**
GaussianNB	50,4%
RandomForestClassifier	48%
SGDClassifier	50,4%
VotingClassifier	50,8%
XGBClassifier	52.62%
XGBClassifier + GRidSerach	**52.76%**

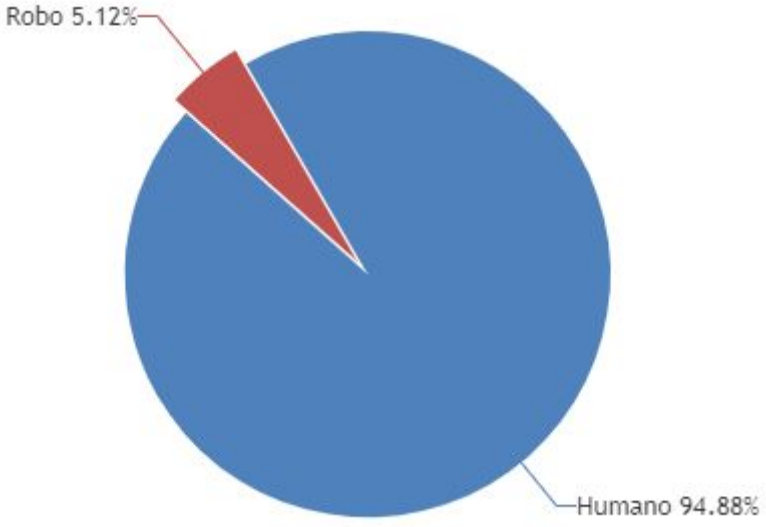
O modelo conseguiu um score no Kaggle de 52,76%, no desenvolvimento do projeto vi as diferenças entre os algoritmos classificadores, suas nuances e como é possível otimizar a aplicação deles com os parâmetros. O computador que roda o algoritmo também faz diferença, quanto mais estimators, ou mais profundidade nos parâmetros o processamento vai ficando mais lentos, demorando horas para executar. Com relação ao primeiro modelo e ao último(xgboost) tive uma diferença de 1h para 4h.

A solução final com o Xgboost mesmo tendo o melhor Score entre os outros modelos que testei ainda não é o suficiente para alcançar o Benchmark, consegui apenas 52% enquanto o benchmark conseguiu 94% de acurácia. Tive dificuldades para criar outras features, que acredito que poderia aumentar o score do modelo (features serão debatidas no tópico Melhorias.)

V. Conclusão

Forma livre de visualização

Um dos primeiros passos foi identificar as proporções das classes e descobri que o dataset era desbalanceado, com apenas 5% de dados de um classe (Robôs) o que poderia ser um desafio a ser vencido.



Após realizar um merge entre os datasets, consegui visualizar alguns problemas que teria quando incluísse os dados nos classificadores.Dados nulos, dados criptografados e tipo string teriam que ser tratados.

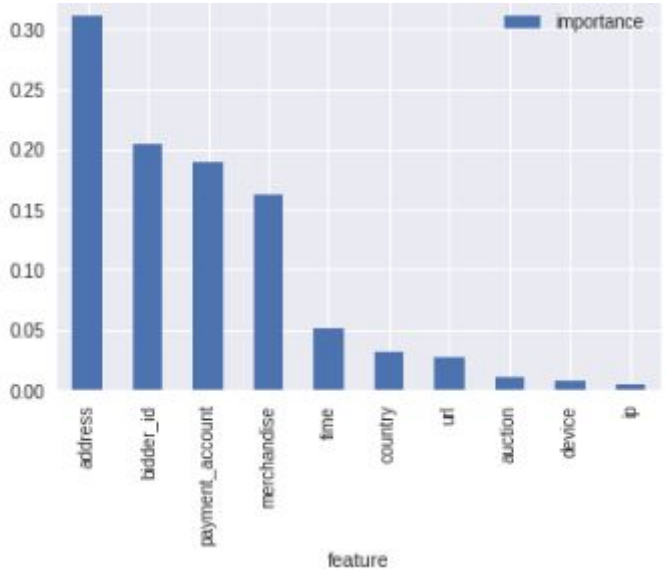
bid_id	bidder_id	auction	merchandise	device	time
0	8dac2b259fd1c6d1120e519fb1ac14fbqvax8	ewmzr	jewelry	phone0	9759243157894736
1	668d393e858e8126275433046bbd35c6tywop	aeqok	furniture	phone1	9759243157894736

country	ip	url	payment_account	address	outcome
us	69.166.231.58	vasstdc27m7nks3	NaN	NaN	NaN
in	50.201.125.84	jmqlhflrzwuay9c	a3d2de7675556553a5f08e4c88d2c228ucoac	42a3b61a1fe69d66ad60f3e347aa09b1erfe2	0.0

Uma limpeza nos dados foi feita. Para fazer a implementação dos classificadores, também foi realizada a transformação dos dados para tipos categóricos.

bidder_id	auction	merchandise	device	time	country	ip	url	payment_account	address
796	3645	4	1	498335	84	813216	361821	1497	438
1803	1317	7	5212	498335	154	992702	576774	153	522
1295	1354	9	3089	498336	194	114979	576774	649	1288
1295	12435	9	2	498336	132	497796	191118	649	1288
985	8288	7	331	498337	84	202110	261143	1811	1979

E o que que foi essencial para o projeto, foi identificar quais destas features são mais importantes para prever a classe do licitante.



Utilizando uma árvore de decisão para verificar a feature importance do dataset, consegui implementar os algoritmos de classificação no projeto. Não consegui realizar tratamento para utilizar a feature tempo, e não inclui no modelo final pois a diferença entre as importâncias era muito grande.

Os processos de tratamento do machine learning são extremamente importantes para os processos do projeto, os dados que são nossa fonte informação e devem sempre ser analisados e ajustados da melhor forma possível para atender as necessidades do projeto. A aplicação de técnicas de Pré-processamento, feature engineering e data cleaning sempre devem ser

estudadas, são muito abrangentes e tem muitas soluções para os problemas de machine learning, com certeza fezem toda a diferença para o resultado final do projeto.

Reflexão

Ao iniciar o curso não tinha conhecimento na área, escolhi o projeto Kaggle por ser muito desafiador, aprender as técnicas com um processo seletivo ao meu ver é um ótimo começo. O dataset logo no começo já apresentou desafios, mostrando-se desbalanceado para fazer as classificações. com poucos dados evidenciando licitantes robôs cerca de 5%, acredito que por isso foi utilizada a métrica de ROC AUC (métrica que funciona bem para este tipo de problema) e dados criptografados difíceis para definir informações, nesta etapa ainda estava trabalhando na exploração analitica dos dados (EDA), evidiencia as hipoteses. Outro detalhe importante foi que enquanto eu realizava o EDA eu fiz uma predição simples, só para teste, porém com a divisão entre treino e teste de 60-40 e isso me deu um resultado maior de acurácia do que utilizando 70-30, acredito que devido ao desbalanceamento.

Ao trabalhar no dataset para conseguir implementar os classificadores tive outras dificuldades, pois havia dados criptografados e muitos atributos desnecessários no dataset, para reduzir os atributos utilizei técnicas para analisar quais eram os atributos mais relevantes para a classificação(uma árvore de decisão). Eu também tentei implementar outros atributos, pois as hipóteses sobre o problema apontavam algumas particularidades como, um licitante que cobria lances em menos tempo poderia ter mais chances de ser um robô, eu acabei não conseguindo criar uma feature para essa hipótese, mas acredito que trabalhar com tempo e com agrupamento de resultados (mais leilões ganhos, mais ips, países e devices diferentes) neste desafio poderia aumentar o Score final na classificação.

Para alcançar o resultado final na classificação, tive que realizar outros processos além dos citados acima, como divisão do dataset em treino e teste, transformação dos dados com formato string para dados categóricos, pois nem todos os algoritmos aceitam dados tipo string para fazer a classificação. Após testar e comprar alguns classificadores, utilizei os hiperparâmetros para dar uma ajustada no modelo final.

As atividades realizadas durante o projeto, são atividades comuns em projetos de machine learning, acredito que com algumas outras features poderia alcançar um score mais preciso.

O machine learning tem suas nuances, e para o problema de classificação as etapas descritas atendem para criar o modelo, claro que ainda podem haver refinamentos no modelo incorporando outras técnicas e outros modelos mais

precisos. No meu ponto de vista o resultado foi bom, deu uma base excelente para expandir os conhecimentos na área, que é o meu objeto para este projeto.

Melhorias

Conforme o desenvolvimento do modelo foi evoluindo, várias hipóteses sobre o que evidenciaria as ações de um robô foram surgindo. Para conseguir evidenciar essas hipóteses devem ser criadas novas features no dataset, que para este projeto acredito que essa seria uma melhoria importante. Poderiam ser criadas features com estatísticas como número de ip e url por leilão, tempo de lances em diferentes leilões, ou tempo de resposta em leilões. Seguindo a lógica, um robô poderia efetuar lances por mais devices, países e ips diferentes, enquanto um humano estaria de casa ou em um dispositivo único fazendo os lances. O robô também pode ser o primeiro a dar lance em um leilão, ou ser o mais rápido a cobrir lances, isso devemos levar em consideração, o tempo. Outra observação é que o robô além de ganhar os leilões pode somente gerar um valor maior, ou seja provavelmente o robô terminará alguns leilões em 2 lugar. Essas features com certeza ajudariam a classificar com mais precisão os lances.

Referência:

[1]https://www.bbc.com/news/10510086

[2]https://community.ebay.com/t5/Archive-Bidding-Buying/Removal-of-Bots-On-Ebay/td-p/19721112

[3]https://moz.com/blog/online-advertising-fraud

[4]https://en.wikipedia.org/wiki/Click_fraud

[5]http://www.dcc.fc.up.pt/~ines/enia07_html/pdf/28076.pdf

[6] https://www.kaggle.com/c/facebook-recruiting-iv-human-or-bot/data

[7]https://www.kaggle.com/c/facebook-recruiting-iv-human-or-bot

[8]https://smith.queensu.ca/insight/articles/robot_auction_bidders_are_such_buzzkills

[9]https://www.nbcnews.com/businessmain/bidbots-sometimes-used-rig-internet-penny-auction-sites-1C8673443

[10]https://g1.globo.com/musica/noticia/sucesso-fake-musicos-fraudam-numeros-de-streaming-usando-robos-e-jaba-20.ghtml

[11]http://tiinside.com.br/tiinside/28/08/2017/ministerio-do-trabalho-utiliza-analytics-para-deteccao-de-fraudes-no-programa-de-seguro-desemprego/

Modelos :

https://en.wikipedia.org/wiki/F1_score
https://pt.wikipedia.org/wiki/Precis%C3%A3o_e_revoca%C3%A7%C3%A3o
http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html
https://en.wikipedia.org/wiki/Naive_Bayes_classifier
http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html
http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html
http://scikit-learn.org/stable/auto_examples/ensemble/plot_voting_probas.html
http://pandas-ml.readthedocs.io/en/latest/xgboost.html
https://xgboost.readthedocs.io/en/latest/python/python_api.html
http://danielhnyk.cz/how-to-use-xgboost-in-python/
http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html
http://conteudo.icmc.usp.br/pessoas/gbatista/files/ieee_la2008.pdfhttp://minerandodados.com.br/index.php/2018/01/16/matriz-de-confusao/
https://www.kaggle.com/c/facebook-recruiting-iv-human-or-bot/data
http://minerandodados.com.br/index.php/2018/02/04/one-hot-encoding-como-funciona-python/