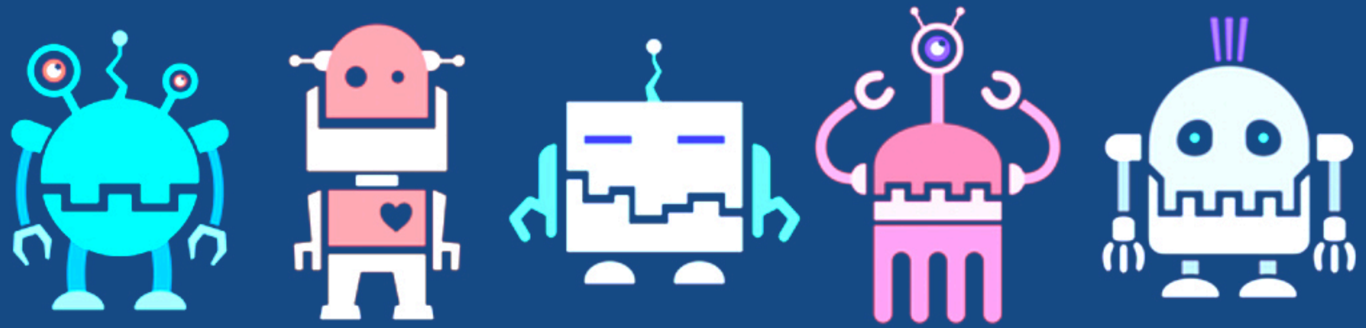
Facebook Recruiting: Human or Robot?

PREDICT IF AN ONLINE BID IS MADE BY A MACHINE OR A HUMAN

Overview

In this competition, you'll be chasing down robots for an online auction site. Human bidders on the site are becoming increasingly frustrated with their inability to win auctions vs. their software-controlled counterparts. As a result, usage from the site's core customer base is plummeting.



In order to rebuild customer happiness, the site owners need to eliminate computer generated bidding from their auctions. Their attempt at building a model to identify these bids using behavioral data, including bid frequency over short periods of time, has proven insufficient.

The goal of this competition is to identify online auction bids that are placed by "robots" helping the site owners easily flag these users for removal from their site to prevent unfair auction activity.

Preliminary plan

1. Get familiar with data
2. Create features
3. Select the most important features
4. Try different models & parameters
5. Compare, improve
6. Submit
7. Repeat

Platforms

PostgreSQL (Postgres + Postico)

- Data analysis

Python (Jupyter Notebook + PyCharm)

- Feature engineering (pandas, numpy)
- Analyzing & visualisation (^ + matplotlib, seaborn)
- Building models (sklearn.ensemble, xgboost)
- Comparing (sklearn.model_selection)
- Scoring (sklearn.metrics)

Understanding data: Bids

bids table (size: 7 656 334 = 412 416 for humans + 2 658 808 for robots + 4 585 110 for unknown)

bid_id	bidder_id	auction	merchandise	device	time	country	ip	url
607221	1096778aca48c41c8d0e288804d5e4e128pco	4tuo8	jewelry	phone6	9762786421052631	it	207.125.55.60	vasstdc27m7nks3
607222	5f7613fcbb6662c05241f6d31c8f8de6v9xpc	boegs	jewelry	phone2	9762786421052631	az	127.80.227.44	5286y6d9wljgzo9
607223	dd055846eb553ab8e953b084f2048d53bdqec	h1ko2	mobile	phone129	9762786421052631	tr	143.250.66.218	vasstdc27m7nks3
607224	7fe34a652dab73b90d39f3d965cdb09b5jnh9	012wh	sporting goods	phone446	9762786421052631	ru	17.138.193.10	vasstdc27m7nks3
607225	c4856fd5abe8f6d6dea36ca2fec444faauos8	sjcg0	jewelry	phone46	9762786473684210	id	81.2.190.0	gw4cowefrfwry6p
607226	7d0f7d3602118db86f8b79910ebb239bh16yb	jefix	sporting goods	phone3367	9762786473684210	cn	84.248.216.90	1l4036y9oxdadxm
607227	8dac2b259fd1c6d1120e519fb1ac14fbqvax8	ekodf	jewelry	phone684	9762786473684210	gt	134.163.232.162	jzt3dgewwxajbez
607228	c709b6c05ebd88124fe639a70103c8aerho8s	7mkw3	home goods	phone318	9762786473684210	tr	151.213.44.163	vasstdc27m7nks3
607229	1ba5b20f16d914e3d76652d5c9cdcbf5iazkd	w152e	mobile	phone222	9762786473684210	in	41.175.168.83	x907rwvp65b0d5b

Insights:

platform has a fixed increment of dollar amount for each bid

time is transformed to protect privacy, but the order is preserved

ip is obfuscated to protect privacy

merchandise is uninformative - it's not the item of auction bid

Understanding data: Bidders

bidders_train table (size: 2 013 = 1 910 for humans + 103 for robots)

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

bidders_test table (size: 4 700)

bidder_i	payment_account	address
49bb5a address (null) c337981cc7a9ccae41u31d7	a3d2de7675556553a5f08e4c88d2c228htx90	5d9fa1b71f992e7c7a106ce4b07a0a754le7c
a921612b85a1494456e74c09393ccb65ylp4y	a3d2de7675556553a5f08e4c88d2c228rs17i	a3d2de7675556553a5f08e4c88d2c228klidn
6b601e72a4d264dab9ace9d7b229b47479v6i	925381cce086b8cc9594eee1c77edf665zjpl	a3d2de7675556553a5f08e4c88d2c228aght0
eaf0ed0afc9689779417274b4791726cn5udi	a3d2de7675556553a5f08e4c88d2c228nclv5	b5714de1fd69d4a0d2e39d59e53fe9e15vwat
cdecd8d02ed8c6037e38042c7745f688mx5sf	a3d2de7675556553a5f08e4c88d2c228dtdkd	c3b363a3c3b838d58c85acf0fc9964cb4pnfa

Insights:

payment_account is uninformative

address is uninformative

many of the outcomes were hand labelled and some were stats based

(hence) dataset is imbalanced (95 : 5)

the data is noisy and messy

Understanding goals

Having bidders' information in the train and test table & their activities in the bids table we have to combine & process them to create feature vectors for every bidder; then use it as a train & test data for solving classification problem.

The submission file looks like this. Prediction stands for prediction of the probability that the bidder is a robot.

bidder_id	prediction
49bb5a3c944b8fc337981cc7a9ccae41u31d7	0.0004950495049504951
a921612b85a1494456e74c09393ccb65ylp4y	0.0026644258699104564
6b601e72a4d264dab9ace9d7b229b47479v6i	0.197347242547483
eaf0ed0afc9689779417274b4791726cn5udi	0.004008839827560129
cdecd8d02ed8c6037e38042c7745f688mx5sf	0.001525165928255305
d4aed439bdc854a56fc6cc3bdb986775w7hxw	0.1624730765667744
ed591299b162a19ff77f0479495831b31hl1q	0.0026644258699104564
eebdee08b0f67283126ef60307f49680sb9va	0.20516732443995317
6887f0abc4eb4c79eb0e23c48ceea186vjfih	0.01812241226434708

A few thoughts…

Auction example

bid_id	time	bidder_id	device	country	ip
2351229	9631917000000000	f5b2bbad20d1d7ded3ed960393bec0f40u6hn	phone49	in	1.158.230.21
2351423	9631917789473684	ac2d643e0c0d3bfe8e54e5a961c6cfdb5xn9v	phone4	in	127.197.177.224
2351460	9631917947368421	b222d659b11bccdaf7221a74b2632d04zieq3	phone159	pk	145.46.168.151
2351672	9631918842105263	53ca80b8f2d7ba5d7458bdb9e2aecf3aewlh6	phone45	in	152.54.151.219
2351764	9631919210526315	e3ccc9bd16fa1f4a92fe87b4a54e5e72uficw	phone57	in	116.208.30.59
…	…	…	…	…	…
7655643	9709219000000000	b222d659b11bccdaf7221a74b2632d04zieq3	phone188	lk	176.56.137.21
7655850	9709219947368421	626159dd6f2228ede002d9f9340f75b7puk8d	phone2353	cn	155.212.2.15
7655964	9709220473684210	626159dd6f2228ede002d9f9340f75b7puk8d	phone2353	cn	155.212.2.15
7656220	9709221578947368	9aca61f505f40a3b4f7865c2d00e2418s0e1f	phone45	in	141.152.173.54
7656248	9709221684210526	626159dd6f2228ede002d9f9340f75b7puk8d	phone168	qa	7.130.90.114

Feature engineering

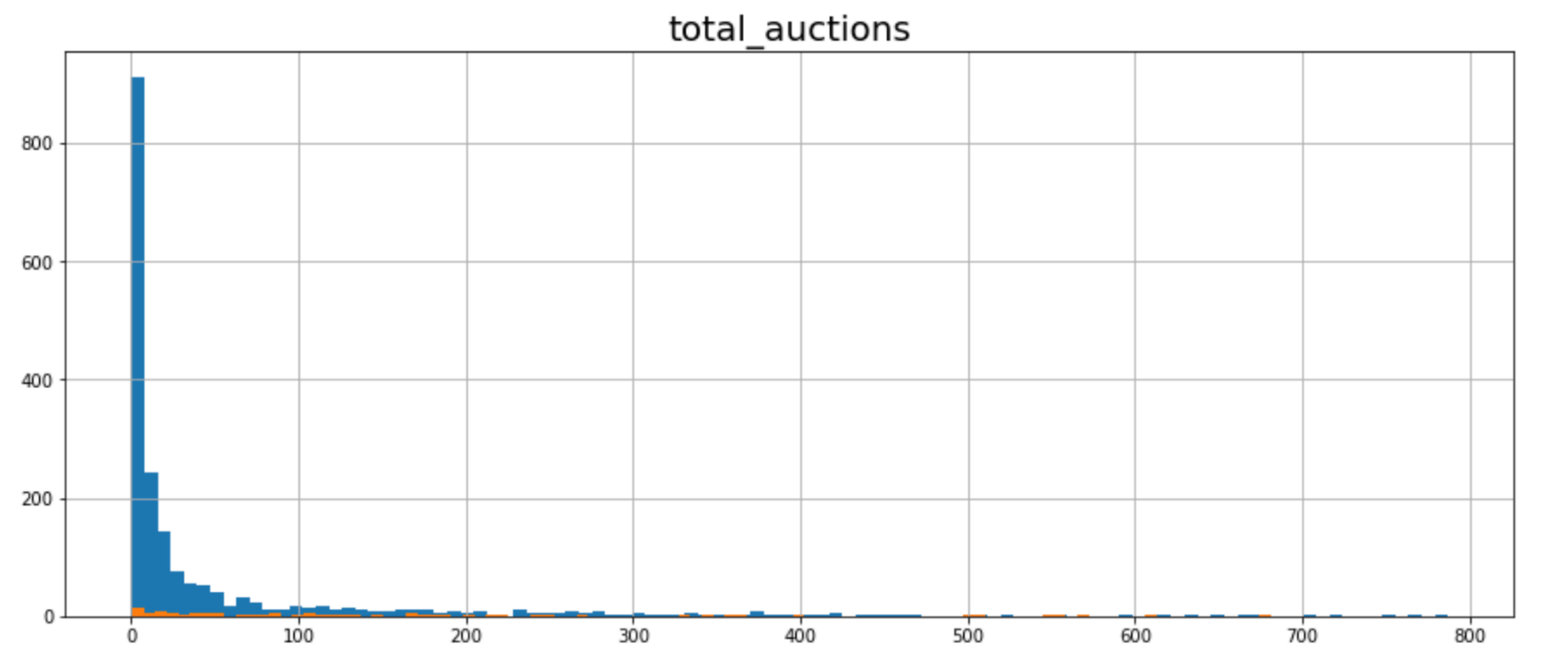
One of the main problems is which features should be generated for a good representation of bidder's behavior.

Features that was created:

1. total_bids - total count of user's bids
2. total_auctions - total count of auctions in which a bidder participated
3. bids_per_auction - mean number of user's bids in auction
4. mean_time_diff - mean time between a user's bid and that user's previous bid
5. mean_response - mean time between a user's bid in auction and previous bid in the same auction
6. min_response - minimum time between a user's bid in auction and previous bid in the same auction
7. ip_entropy - the entropy for how many ips a bidder used
8. url_entropy - the entropy for how many urls a bidder was reffered from

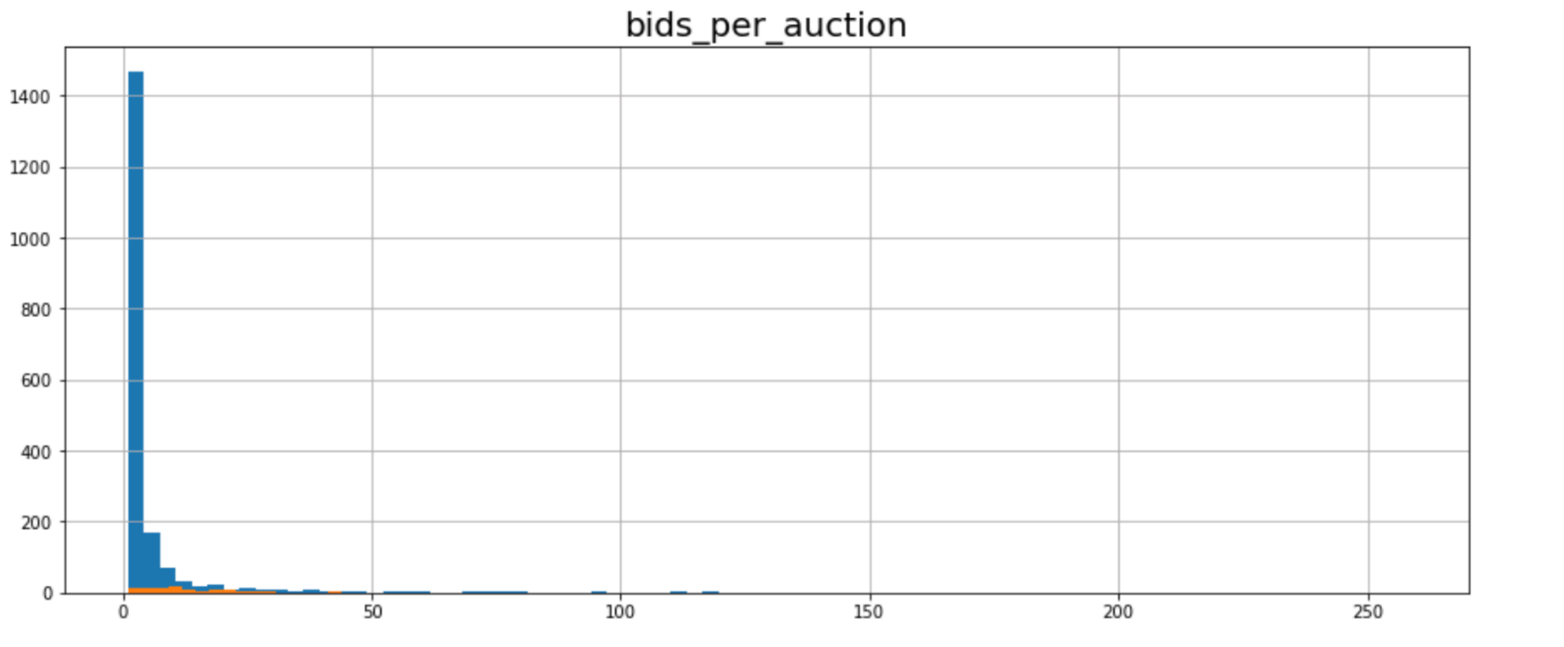
	importance	|corr_coef|
bids_per_auction	0.286	0.117
mean_time_diff	0.213	0.096
total_bids	0.192	0.037
ip_entropy	0.122	0.041
url_entropy	0.103	0.003
total_auctions	0.065	0.124
min_response	0.013	0.026
mean_response	0.007	0.019

Feature distributions



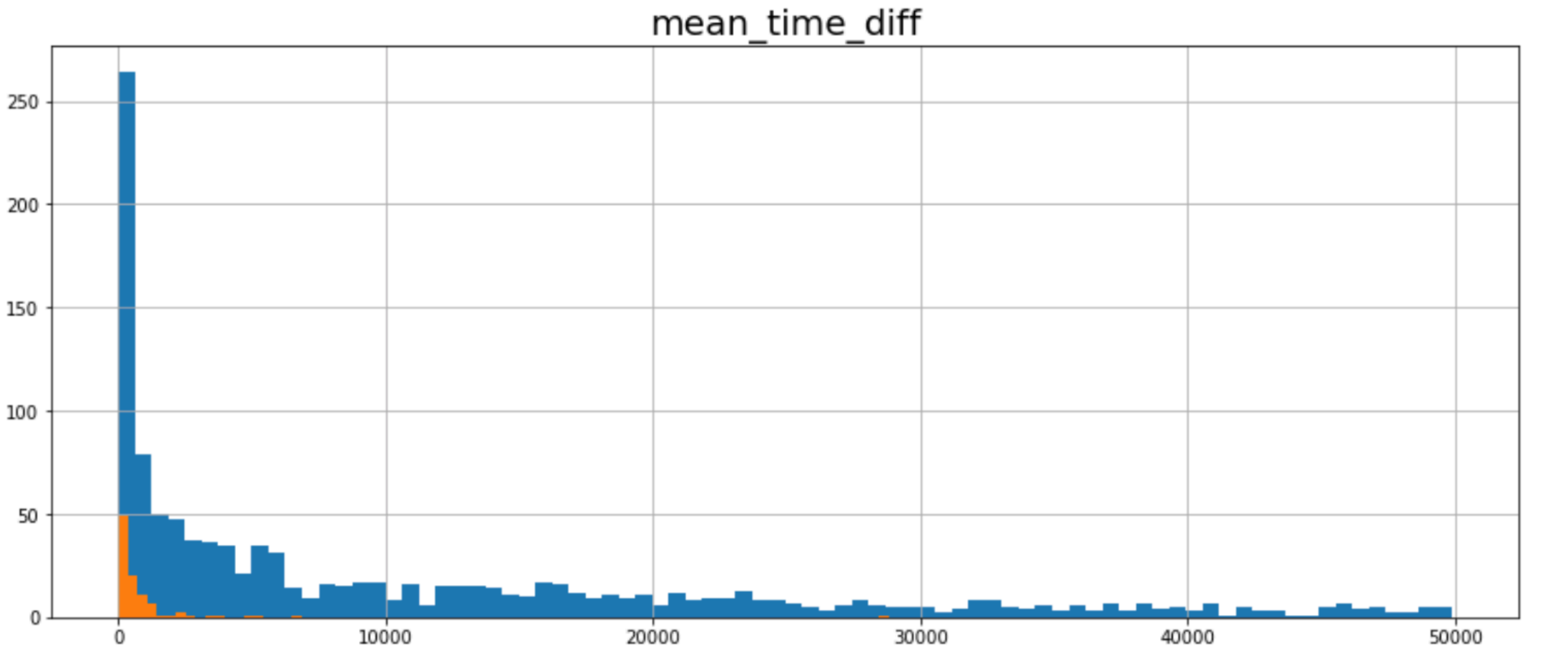
	total_auctions_x	total_auctions_y
count	1910.000000	103.000000
mean	57.189005	145.038835
std	142.021381	195.103186
min	0.000000	1.000000
50%	9.000000	74.000000
max	1623.000000	1018.000000

Feature distributions



	bids_per_auction_x	bids_per_auction_y
count	1881.000000	103.000000
mean	6.441526	23.154676
std	29.986961	42.999723
min	1.000000	1.000000
50%	1.610400	11.863000
max	1023.500000	325.000000

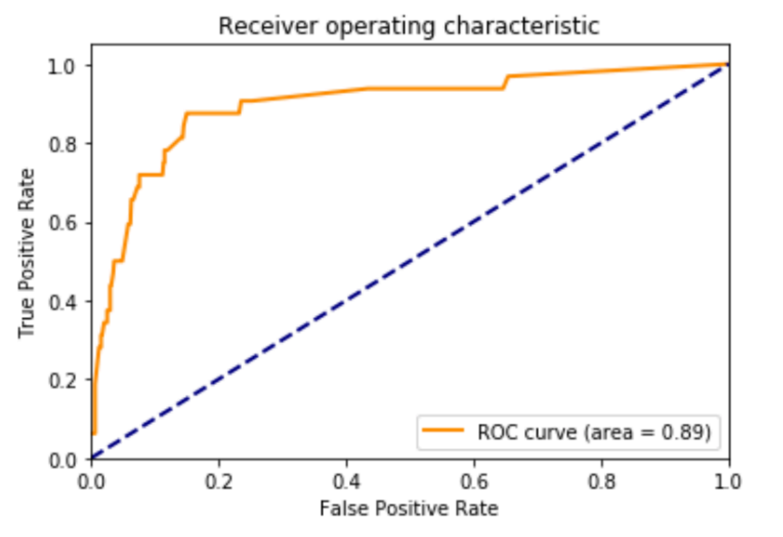
Feature distributions



	mean_time_diff_x	mean_time_diff_y
count	1.584000e+03	98.000000
mean	6.450692e+04	1013.142857
std	1.591896e+05	3052.776883
min	1.350000e+00	1.890000
50%	1.347196e+04	356.235000
max	1.445956e+06	28804.000000

Building models: first try

XGBoost with default parameters on the first features:




```
features = ['total_bids', 'total_auctions',
            'mean_time_diff']
```

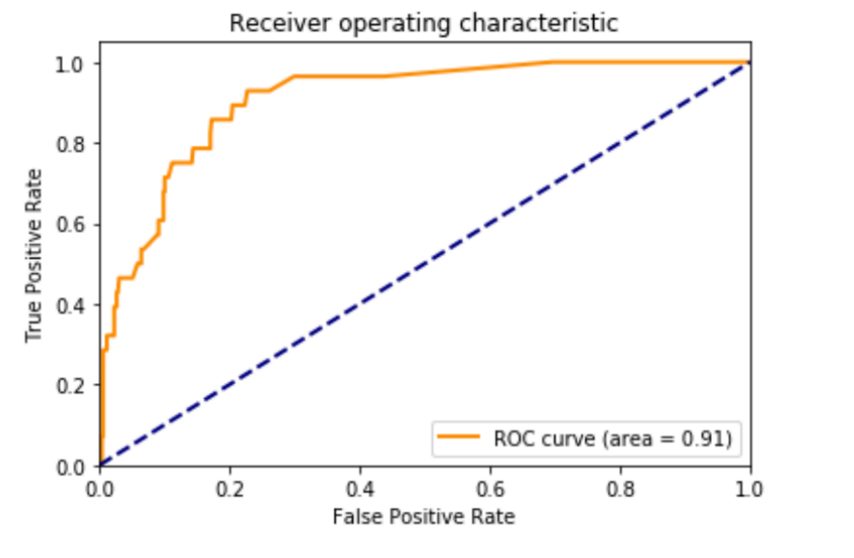
```
num_round = 10
bst = xgb.train(parameters, dtrain, num_round, evallist)
```

```
[0]     eval-auc:0.871322       train-auc:0.865817
[1]     eval-auc:0.879813       train-auc:0.886111
[2]     eval-auc:0.89534        train-auc:0.900631
[3]     eval-auc:0.903016       train-auc:0.92157
[4]     eval-auc:0.910249       train-auc:0.926373
[5]     eval-auc:0.904028       train-auc:0.93062
[6]     eval-auc:0.904103       train-auc:0.93368
[7]     eval-auc:0.900301       train-auc:0.940894
[8]     eval-auc:0.900844       train-auc:0.941451
[9]     eval-auc:0.894402       train-auc:0.945272
```

SubmissionXGBoost.csv	**0.88570**	**0.87131**
3 features: mean_time_diff, total_bids_count, total_auctions_count		

Building models: best try (1)

RandomForestClassifier with max depth 3



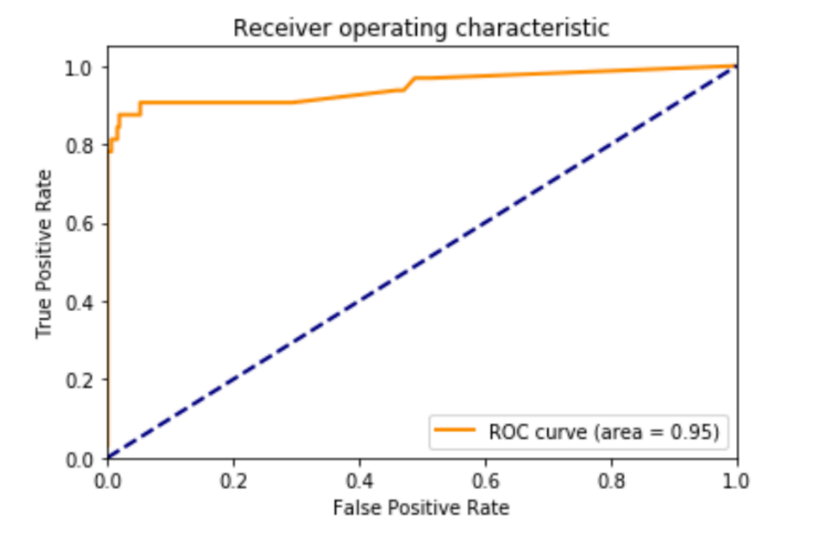
on 6 features:

```
features = ['total_bids', 'total_auctions',
            'bids_per_auction',
            'mean_time_diff',
            'ip_entropy', 'url_entropy']
```

SubmissionRF3_6.csv 'total_bids', 'total_auctions', 'bids_per_auction', 'mean_time_diff', 'ip_entropy', 'url_entropy'	0.90431	0.89186

Building models: best try (2)

XGBoost with tuned parameters:



{'learning_rate': 0.4,
'max_depth': 20,
'n_estimators': 10,
'reg_alpha': 1.4,
'reg_lambda': 1.8}

on 6 features:

```
features = ['total_bids', 'total_auctions',
            'bids_per_auction',
            'mean_time_diff',
            'ip_entropy', 'url_entropy']
```

SubmissionXGBoostGrid_6.csv	**0.90837**	**0.89006**
'total_bids', 'total_auctions', 'bids_per_auction', 'mean_time_diff', 'ip_entropy', 'url_entropy'; {'learning_rate': 0.4, 'max_depth': 20, 'n_estimators': 10, 'reg_alpha': 1.4, 'reg_lambda': 1.8}		

What next?

Try another algorithms (AdaBoostClassifier, ExtraTreesClassifier, Neural Networks?)

Try to make clusters based on bidders' activities

Create feature related to devices used by bidder

Create feature related to amount of bid