Human-Robot Detection for Online Auctions

Machine Learning Engineer Nanodegree

Capstone Project

Yu-Yu Lin
March 29th , 2017

I. Definition

Project Overview

Online auction has been adopted by major e-commerce platforms such as Ebay, uBid, and WeBidz, etc. By properly designed auction mechanisms, auctions have been proven to increase the profit of the sellers. On the other hand, buyers of products may adjust its purchasing strategy to maximize its probability of winning the auction with minimal costs.

A promising auction mechanism, however, can be seriously jeopardized by the use of intelligent robots. Unlike human beings, these robots are able to continuously monitor the status of the auction and submit bids in a timely and intensive manner. In this case, a bidder may win most of the auctions with the help of a well-designed robot agent. As a result, a majority of human bidders may seek for other platforms for their product purchasing, leading to decreased revenues for the platform provider. Therefore, it is essential for the website/platform to build up an effective algorithm to detect potential robot agents and block their accounts accordingly. Kaggle had raised a competition to serve this purpose. This project will be developed based on the data set provided by the Kaggle website.

Problem Statement

A data set containing the bidding history of a set of labeled agents (human or robot), each of which is distinguished by a unique bidder_id, has been released. The purpose of this project is to utilize this labeled data set (training set) to build up our

prediction model in order to detect whether an agent is a robot on another unlabeled data set (test set). The result is formatted a two-column matrix. The first column is the bidder_id of the test data. The second column is the probability that the agent is a robot, which is computed based on our model. The result is submitted through the Kaggle website and is compared with other competitors on the leaderboard.

To summarize, our proposed procedure to solve the problem includes the following:

1. Data wrangling: since agent label and bidding history are stored in different files (train.csv and bid.csv, respectively), we need to merge these two data sets for an easier processing.
2. Feature engineering: extracting important features from the bidding history.
3. Prediction: Building up learning models based on the training set and the extracted features.
4. Result evaluation and refinement: Submitting the model to the Kaggle website to see the ranking of the model. Repeat steps 2-4 to achieve higher ranking.

Metrics

As mentioned above, the probability that each bidder is a robot is submitted online for evaluation. Different probability thresholds are used to determine whether an agent is a robot or not. As a result, an ROC (Receiver Operating Characteristic) curve is obtained. This curve well captures the tradeoffs between the True Positive Rate (TPR) and the False Positive Rate (FPR), where

$$\text{TPR} = \frac{\#\text{robot bidders detected as robots}}{\#\text{robot bidders}}$$

$$\text{FPR} = \frac{\#\text{human bidders detected as robots}}{\#\text{human bidders}}$$

In this competition, we use the AUC (Area Under Curve) as the performance metric. AUC is the area under the ROC curve. The maximum value of AUC is equal to 1. For a random prediction, the AUC value is around 0.5.

II. Analysis

Data Exploration

There are two .csv files associate with the training information: **train.csv** and **bid.csv**. We summarize the entries of these two files in the following. Note that some of the information are encrypted for privacy, as highlighted in the description.

1. **train.csv**: Each row represents a unique bidder. Each bidder contains the following information:

   bidder_id: a unique id used to distinguish among different bidders

   payment_account: account number associated with the bidder (Encrypted)

   address: mailing address of the bidder (Encrypted**)**

   outcome: whether the bidder is a robot (outcome = 1.0) or a human (outcome = 0.0). There are a total of 1910 human bidders and 103 robot bidders.

2. **bid.csv**: bidding history collected. Each row represents a submitted bid with the following information:

   bid_id: a unique id used to distinguish among different bids submitted. A total of 7656334 bids are submitted.

   bidder_id: the bidder who submitted the bid

   auction: a unique id used to distinguish among different auctions. A total of 15051 auctions are included.

   merchandise: the category of the auction (such as “home goods”)

   device: phone model used to submit the bids. A total of 7351 different devices are used.

   time: time instants that the bid is submitted (Encrypted)

   ip: ip address of the bid (Encrypted). A total of 2303991 different ip addresses were used.

   country: country where the ip belongs to. Bids are submitted by bidders from 200 different countries.

url: url where the ip is referred from (Encrypted). A total of 1786351 urls are reported.

It is noted that although some of the data entries are encrypted, most of them can be treated as categorical data (e.g. ip and url). The model can still be built without knowing the true values of these entries.

However, we note that the **time** feature is encrypted to values between 9631916842105263 and 9772885210526315. We do want to extract the exact time information since the time series behaviors for human and robots may be of significant difference. Interestingly, we note that the differences between two encrypted time stamps are multiples of 52631580. It may make sense to consider this value as the length of 1 second. In addition, by shifting the minimum time stamp towards 0, we can thus convert the time stamps into numbers between 0 and 2678398. Surprisingly, the latter value corresponds to the number of seconds in a month.

Exploratory Visualization

In the following, we summarize the statistics of several counting statistics of human and robot bidders in Table I(a) and Tabile I(b), respectively.

	num_bids	num_auction	num_url	num_country	num_device	num_ip
count	1910.000000	1910.000000	1910.000000	1910.000000	1910.000000	1910.000000
mean	1392.046073	57.189005	330.097906	12.484293	72.824607	572.430890
std	14487.265477	142.021381	2714.979379	22.748897	183.376886	4109.721882
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	3.000000	2.000000	1.000000	1.000000	2.000000	2.000000
50%	14.000000	9.000000	4.000000	3.000000	7.000000	10.000000
75%	136.000000	40.000000	33.000000	10.000000	50.000000	86.750000
max	515033.000000	1623.000000	81376.000000	164.000000	2618.000000	109159.000000

Table I (a): Counting statistics of human bidders

	num_bids	num_auction	num_url	num_country	num_device	num_ip
count	103.000000	103.000000	103.000000	103.000000	103.000000	103.000000
mean	4004.038835	145.038835	544.582524	26.475728	163.611650	2387.796117
std	16370.097403	195.103186	1163.909786	31.158622	222.811854	11269.674137
min	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	288.000000	23.000000	4.500000	3.000000	4.500000	34.000000
50%	716.000000	74.000000	88.000000	13.000000	78.000000	290.000000
75%	2332.500000	170.500000	591.000000	41.500000	219.000000	1089.000000
max	161935.000000	1018.000000	8551.000000	179.000000	1144.000000	111918.000000

Table I (b): Counting statistics of robot bidders

The feature represented by each column is as follows:

1. **num_bids**: total number of bids submitted by each bidder
2. **num_auctions**: total number of auctions that each bidder participated in
3. **num_url**: total number of unique urls used by a bidder
4. **num_countries**: total number of countries where a bidder submitted its bids
5. **num_device**: total number of unique device models used by a bidder
6. **num_ip**: total number of unique ip addresses used by a bidder

It is observed that for the robot bidders, the 25, 50, and 75 percentiles of the six features are all higher than that of human bidders. One of the reasonable conjectures is that as robot could keep bidding without taking a rest, it may submitted more bids and participating in more auctions than human beings.

However, it is quite surprising that a human bidder submitted a total of 515033 bids, which is even larger than that of all the robot bidders. Therefore, it appears that although the six counting metrics are useful features, we need to explore additional features to distinguish between human and robot bidders.

We thus extract the human bidder and the robot bidder that made the largest number of bids. Based on the two samples, we try to explore the time series features to distinguish between the two highly aggressive bidders:

	bidder_id	outcome	num_bids	min_time_diff_same_auction	max_numbids_per_auction_20min
1823	f5b2bbad20d1d7ded3ed960393bec0f40u6hn	0.0	515033.0	18.0	8543.0

	bidder_id	outcome	num_bids	min_time_diff_same_auction	max_numbids_per_auction_20min
1512	9655ccc7c0c193f1549475f02c54dce45kjw7	1.0	161935.0	15.0	21104.0

Table II: Time series feature of sampled human (top) and robot (bottom) bidders with the maximum **num_bids** values

The two time series features identified here are the following:

min_time_diff_same_auction: the minimum time difference between two bids associated with the same auction

max_numbids_per_auction_20min: the maximum number of submitted bids associated with each auction within 20 minutes time slots.

We see that the robot bidder submitted its bids more intensively in time compared with the human bidder. To validate our conjecture, we illustrate these two time series features in the following tables:

	min_time_diff_same_auction	max_numbids_per_auction_20min
count	1.910000e+03	1910.000000
mean	4.234820e+07	70.293717
std	6.320997e+07	446.081612
min	1.000000e+00	0.000000
25%	3.112000e+03	1.000000
50%	2.835100e+04	2.000000
75%	1.367542e+08	11.000000
max	1.367542e+08	8543.000000

Table III (a): Time series statistics of human bidders

	min_time_diff_same_auction	max_numbids_per_auction_20min
count	1.030000e+02	103.000000
mean	6.640188e+06	340.912621
std	2.953348e+07	2082.187157
min	8.000000e+00	1.000000
25%	1.000000e+02	24.000000
50%	6.090000e+02	55.000000
75%	1.405750e+03	137.000000
max	1.367542e+08	21104.000000

Table III (b): Time series statistics of robot bidders

It is observed that for a given auction, the minimum time difference between robot bidders at least 10 times shorter than that of the human bidders for the 25 and 50 percentiles. For the maximum number of bids of each auction within the 20 minute slots, the robot bidders were able to submit 10 times more bids than human bids in all 25, 50, and 75 percentiles. These two features are considered significant in distinguishing between human and robot bidders.

Surprisingly, from Table III, we noticed that the minimum value of the **min_time_diff_same_auction** of the human bidders is only 1 second; while that of the robot bidders is 8 second. Thus, we extract these two samples and see if we could explore other features to distinguish between the two:

	bidder_id	outcome	num_bids	min_time_diff_same_auction	max_numbids_per_auction_20min
1318	e76154d73bb7d7b86d9bd48d167d3c679z0va	0.0	4.0	1.0	2.0

	bidder_id	outcome	num_bids	min_time_diff_same_auction	max_numbids_per_auction_20min
1203	06ddad46112881df154e3bda0e90fe880r57n	1.0	177.0	8.0	35.0

Table IV: Time series feature of sampled human (top) and robot (bottom) bidders of those with the minimum value of min_time_diff_same_auction

It is noticed that the human bidder only submitted 4 bids while the robot bidder submitted 177 bids. The **max_numbids_per_auction_20mins** feature also successfully distinguishes the two samples: the human bidder has only a maximum of 2 bids while the robot bidder has 35 bids.

Another feature we would like to explore is the **entropy** feature. This set of features captures the variance of the categorical data. For example, if a bidder submits its bidder using only 1 device, its **device entropy** is low. If a bidder submits its bids within the same hours, its hour entropy is lower than bidders who submit their bids across different hours in a day. We suspect that as robot bidders are able to submit bids continuously without taking a rest, it should have higher hour entropy.

The formula that we use to calculate the entropy of a categorical data $\boldsymbol{a} = [a_1, \dots, a_N]$, where a_i is the number of sample belonging to category i, is given as follows:

$$Entropy(\boldsymbol{a}) = -\sum_{i=1}^{N} \frac{a_i}{\sum_{j=1}^{N} a_j} \log \frac{a_i}{\sum_{j=1}^{N} a_j}$$

We consider the following entropy measures:

1. **ip_entropy**: entropy of ip address of a bidder
2. **url_entropy**: entropy of url of a bidder
3. **device_entropy**: entropy of device of a bidder
4. **hour entropy**: entropy of hours when the bids are submitted of a bidder.

It is noticed that the **hour entropy** for each bidder is first calculated on the per-country basis and averaged out over all countries where the bidder submitted its bids. This calculation accommodates the fact that different countries locate in different time zones. The entropy statics between human and robot bidders are summarized in the following tables:

	ip_entropy	url_entropy	device_entropy	hour_entropy
count	1910.000000	1910.000000	1910.000000	1910.000000
mean	2.683290	1.560550	2.018459	0.700911
std	2.288652	1.624869	1.650189	0.628644
min	-0.000000	-0.000000	-0.000000	0.000000
25%	0.693147	-0.000000	0.636514	0.126346
50%	2.271869	1.070967	1.816563	0.654323
75%	4.246183	2.487472	3.433878	1.030862
max	10.106916	8.942253	5.924976	3.054211

Table V (a): Entropy statistics of human bidders

	ip_entropy	url_entropy	device_entropy	hour_entropy
count	103.000000	103.000000	103.000000	103.000000
mean	4.599000	2.638433	2.758170	1.075393
std	2.687466	2.347788	1.765078	0.601961
min	-0.000000	-0.000000	-0.000000	0.000000
25%	2.155149	0.197655	1.101417	0.679991
50%	5.137297	2.345790	3.270671	1.052194
75%	6.645241	5.033401	4.400002	1.412230
max	11.254165	6.658489	5.162447	3.078652

Table V (b): Entropy statistics of robot bidders

It is observed that robot bidders generally achieve a higher value in each of the four entropy statistics.

We are also curious in whether the robot bidders won auctions more often than human bidders. Although not explicitly given in the data set, we assume that the bidder who submits the last bid in an auction wins the auction. The winning probability is the number of winning auctions divided by the total number of auctions participated by a bidder. The following tables summarize the winning probability of human and robot bidders.

	win_prob
count	1910.000000
mean	0.011359
std	0.047504
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	0.916667

	win_prob
count	103.000000
mean	0.043726
std	0.113448
min	0.000000
25%	0.000000
50%	0.009901
75%	0.039779
max	1.000000

Table VI: Winning probability of human (left) and robot (right) bidders

It is observed that for the 50 and 75 percentile, the robot bidders have a higher winning probability than human bidders. Human bidders experienced a quite low winning probability (close to 0) even for the 75 percentile bidder.

Algorithms and Techniques

The problem is considered as a logistic regression problem since the values that we return is a real number between 0 and 1, indicating the probability that a bidder is a robot, rather than a binary value 0/1. We choose to use tree based algorithms such as Random Forest to build up our model.

To explain Random Forest model, we need to first understand what is a decision tree. A decision tree is a series of questions we design to predict whether a student will

pass or not (e.g. how old are you?, how long you study per week?). After asking these questions, a student is classified as "pass" or "not pass." However, a decision tree can prone to overfit. That is, even though we can classify all the students correctly by asking many questions, the performance can be poor when new data comes in.

Random forest is a method to solve this issue. We construct many decision trees to form a "forest." To construct each decision tree, we randomly pick K students. After the forest is formed, each decision tree gives a prediction on a new data point. Eventually, predictions made by different trees are combined (e.g. through weighted sum) to come up with the final prediction.

Decision tree based algorithms are flexible and can be used to construct complicated prediction models. The parameters such as tree depth and number of features used must be properly selected through cross validations to avoid overfitting. The major disadvantage of decision tree algorithms is that it does not support online learning. Every time a new data sample is obtained, the tree must be reconstructed. However, for this project, we are asked to train the model based on the bidding history in a prescribed period time. The learning is conducted offline and the training speed is not of major concern.

The difficulty in illustrating a random forest is anther disadvantage of the model. In this project, we thus focus on evaluating the final score of our model rather than explaining the detail of the derived model. On the other hand, we will check several sampled points to see if our prediction is consistent with our conjectures.

Benchmark

A trivial upper bound for AUC score is 1, when the prediction is perfect. On the other hand, 0.5 is a trivial lower bound when a predictor makes random guesses whether the bidder is a robot or not.

The Kaggle leaderboard, however, has provided us several benchmarks for us to compare our result with other competitors. There are actually two leader boards, public and private, are available. The public leaderboard only uses 30% of the test data to compute the AUC score and it is revealed to competitors during the competition. The private leaderboard uses 70% of the data set to compute the final AUC score after all competitors have submitted the final model at the end of the competition. It serves to determine the final ranking of the competitor.

However, since the competition is already closed when we worked on this project, both private and public leaderboard scores are available. We thus compare our result with those on the private leaderboard to see how the proposed model performs.

III. Methodology

Data Preprocessing

The purpose of data processing in this project is to construct a data frame with dimension *N* by (*M*+1), where *N* is the number of bidders and *M* is the number of features used in constructing the prediction model. We summarize all the features as presented in the Data Exploration section in the following:

(1) Counting statistics:

a. **num_bids**: total number of bids submitted by each bidder
b. **num_auctions**: total number of auctions that each bidder participated in
c. **num_url**: total number of unique urls used by a bidder
d. **num_countries**: total number of countries where a bidder submitted its bids
e. **num_device**: total number of unique device models used by a bidder
f. **num_ip**: total number of unique ip addresses used by a bidder

(2) Time series statistics:

a. **min_time_diff_same_auction**: the minimum time difference between two bids associated with the same auction.
b. **max_numbids_per_auction_20min**: the maximum number of bids associated with each auction within 20 minutes time slots.

(3) Entropy statistics:

5. **ip_entropy**: entropy of ip address of a bidder
6. **url_entropy**: entropy of url of a bidder
7. **device_entropy**: entropy of device of a bidder
8. **hour entropy**: entropy of hours when the bids are submitted of a bidder.

(4) Other statistics:

a. **win_prob**: The probability that a bidder wins an auction. A bidder is considered as a winner if the bidder is the last one who submits a bid for the auction.
b. **avg_num_url_per_device**: average number of urls used by a device.

In order to compute Time series statistics and the hour entropy, we need to map the encrypted time stamps into seconds. Following the reasoning in the Data Exploration section, the time stamps are shifted towards 0 and are divided by 52631580 to convert to the units of seconds. For each bid, a **day** and an **hour** columns are created to map the corresponding day and hour indices. The **hour** column is then used to compute the hour entropy.

It is noted that some of the human bidders in the training sets submit 0 or 1 bid. If the agent submits less than one bid, the **min_time_diff_same_auction** feature is not defined. As a result, we set the value to 100 x (max value of the time difference between bids for an auction). For the other features, we simply set the values to 0.

No feature scaling is required for the decision tree based algorithm as the features are treated independently when determining the decision boundaries.

Implementation

Functions related to the data preprocessing includes the following:

1. ConvertTime(history):

   Input: data frame containing bidding history

   Output: None. But the column **time** is converted to into unit of seconds. In addition, columns **day**, **hour**, **minute**, **second** are computed and added to the input history.

2. entropy (arr):

   Input: a list containing the counts of a set of categorical data

   Output: integer, which is the calculated entropy.

3. GenerateFeature(bidders, history, training):

   Input:

   - Bidders: training set containing bidder ids and the type of each bidder as stored in the column **outcome**.
   - history: all the bid submitted
   - training: Boolean variable. True if the data set is a training set.

   Output: A data frame with rows containing all the bidders. Columns are the features identified in the Data Preprocessing section. If the training variable is true, the **outcome** column of each bidder is appended.

After the execution of the above three functions we apply the random forest model to obtain the prediction model. To properly select the parameters, we apply GridSearchCV. The number of folds is equal to 5. A total of 20 trees (as controlled by the parameter n_estimators) are generated. The GridSearchCV search over the two parameters: (1) **max_feature**, ranging from 2 – # features (2) max_depth, ranging from 2 to 9.

The function is implemented in the function PredictRandomForest(X_all, seed) with input/output specified as follows:

Input:

- X_all: training set with rows representing bidders and columns representing selected features
- seed: seed of the random number generator used to initialize the trees

Output:

- clf: trained regression model

Refinement

To better alleviate the overfitting issue, we take another layer of ensemble technique over the random forest. We generate a total of 9 random forests, each of which is constructed by a different initialization seed. As a result, the final predictor is constructed by averaging the result of the 9 random forests.

IV. Results

Model Evaluation and Validation

We first look at the prediction result of our random forest model on the test data:

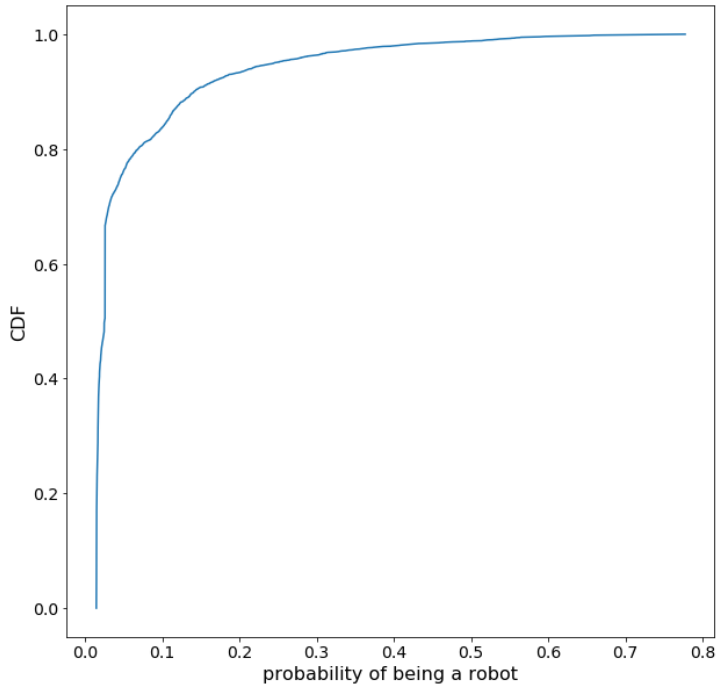

Fig. 1 CDF of the probability that an agent is a robot of the test data

It is noted that the maximum value of the prediction probability is less than 0.8. By observing Fig. 1, we see that over 90% of the agents are unlikely to be a robot (with a probability less than 20% of being a robot).

Validating the model is difficult since Kaggle does not provide actual labels of the test data. Hence, we take samples from the test data, look into their feature values, and determine if the predicted probability matches our intuition.

We sample four test bidders from the test set and summarize their feature values as follows. The four bidders correspond to 100 (id 2782), 75 (id 160), 50 (id 1609), and 25 (id 742) percentile in the predicted probability, respectively. The probability that a bidder is a robot is presented in the **outcome** column.

	outcome	num_bids	num_auction	num_url	num_country	num_device	num_ip
2782	0.751855	808.0	56.0	552.0	3.0	4.0	679.0
160	0.036948	196.0	45.0	15.0	28.0	102.0	187.0
1609	0.011144	159.0	102.0	82.0	22.0	76.0	128.0
742	0.001500	9.0	8.0	1.0	1.0	4.0	5.0

	outcome	min_time_diff_same_auction	max_numbids_per_auction_20min
2782	0.751855	5.400000e+01	56.0
160	0.036948	3.410000e+03	51.0
1609	0.011144	2.902300e+04	4.0
742	0.001500	1.195637e+06	1.0

	outcome	ip_entropy	url_entropy	device_entropy	hour_entropy
2782	0.751855	5.916960	5.373006	1.039706	0.495807
160	0.036948	5.211789	1.421829	4.343018	0.785046
1609	0.011144	4.750146	3.480569	3.889496	0.674203
742	0.001500	1.522955	-0.000000	1.214890	1.889159

	outcome	win_prob	avg_num_url_per_device
2782	0.751855	0.017857	138.000000
160	0.036948	0.000000	0.147059
1609	0.011144	0.000000	1.078947
742	0.001500	0.000000	0.250000

Table VII: Predicted probability of being a robot from samples of test data

It is observed that as predicted, for the following features, a larger feature value indicates a higher probability of being a robot: **num_bids**, **num_ip**, **max_numbids_per_auction_20mins**, **ip_entropy**, and **win_prob.** A smaller value of **min_time_diff_same_auction** is another indication of being a robot. This result actually matches with our original conjecture. It is reasonable and matches our expectation

On the other hand, there are features such as **num_country** and **num_device**, which do not quite match our initial guess. The bidder who is determined to be the robot with probability 0.75 submitted its bids in only 3 countries with only 4 devices.

Therefore, we may conclude that some features may be more important than others; we will justify this point in the conclusion.

Justification

We submitted our result to the Kaggle website. As the competition is already closed, we simply report our private board score 0.92255, and the ranking is 200 out of 985 (top 20%). Interestingly, we note that one of our previous submissions which mistakenly put zero values to **min_time_diff_same_auction** to bidders submitting no bids, achieves an even higher score: 0.92439, and the ranking is 173 out of 985 (top 18%). The maximum score achieved on the private leader board in this competition is 0.94254. The high ranking of our model may be an indication of the successful categorization of most of the bidders. We may conclude that our model is robust enough to solve the problem

We also summarized in the following tables how the performance of the model is improved by introducing new features:

	Public score	Private score
(1) = counting statistics + win_prob + avg_num_url_per_device	0.87951	0.89326
(2) = (1) + time series statistics	0.88733	0.90148
(3) = (2) + entropy statistics	0.90154	0.91073
(4) = (3) + initializing with different seeds	0.90179	0.92255

It is observed that the use of new statistics improves the original private score in (1) by 0.01747. The use of multiple seeds and the averaged random forest as the final model effectively improve the private score of (3) by another 0.01182.

V. Conclusion

Free-Form Visualization

We would like to look into the feature importance score provided by the random forest model to illustrate which feature is most relevant in differentiating between

human and robot bidders. Since we have taken an average over multiple random forests, each of which is initialized using a different seed, the importance score is also averaged over the 9 random forests. The result is shown in the following figure.

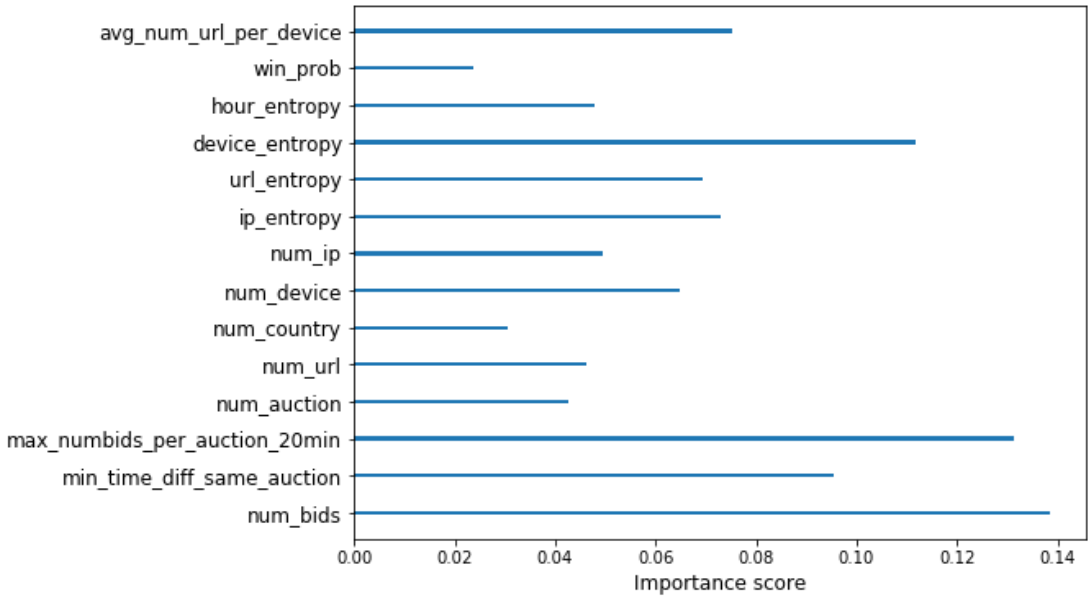

Fig. 2 Average importance scores

The top 5 most important features used to build up the prediction model are: **num_bids**, **max_numbids_per_auction_20mins**, **device_entropy,** **min_time_diff_same_auction**, and **avg_num_url_per_device**. It is noticed that the two time series statistics are of significant importance in differentiating between human and robot bidders. Features such as **num_country**, **num_auction**, and **num_url** have lower importance score. However, this is not an implication that these features are not relevant to the model. One possibility is that these features may be correlated with others. Therefore, incorporating all the features in our prediction model can decrease the importance feature for each of them.

Reflection

In this project, we have proposed the following solution to detect whether an online bidder could be a robot or not. We extract counting statistics, time series statistics, entropy values, and winning probability from the bidder's one-month bid history. We then build a random forest model by using GridSearchCV over the parameters **max_feature** and **max_depth** for parameter optimization. In the final step, we repeat the GridSearchCV with 9 different seeds and construct the final model by taking the averaged prediction value generated by the 9 random forests.

The most difficult part of the project is the feature engineering. Features given in this project such as time stamps are encrypted. We must use the common sense and make certain assumptions about the data. For example, by observing the time stamps are multiples of 52631580, we assume that this number corresponds to one second. When calculating the winning probability, we also assume that the bidder who submitted the last bid is the winner. The accuracy of these assumptions is difficult to justify. We are not convinced that these assumptions are useful until the prediction results in high score of AUC.

Dealing with outliers is another tricky part in this project. Originally we thought that humans cannot submit more bids than robots and tried to drop the human bidder with the maximum number of bids submitted. However, the AUC score decreased after we dropped the bidder. After a second thought, we realize that the data set may refer to multiple human bidders associated with one bidder_id. They may cooperate with each other to submit the bids. In turn, instead trying to drop the outlier, we are motivated to explore other time series and entropy features to distinguish humans submitting a large number of bids from robots. Eventually, this approach results in a higher score comparing with the approach that outliers are dropped.

Improvement

Techniques such as deep neural networks or XGBoost can also be implemented to solve the problem. These models can also be used in combination of the random forest model constructed in this report to avoid overfitting.