

**2024 Data Mining Final Project
Report
Facebook Recruiting IV: Human or
Robot?**

F74096255 鄭宇辰

F74091093 陳宥橋

F74109016 葉惟欣

1. 執行環境

Colab

2. 比賽介紹

甲、 競賽目標

藉由拍賣的資訊，判斷拍賣人是否為機器人。

乙、 資料集

i. 總論

1. 分為 bidder dataset 與 bid dataset，其中主辦方將 bidder dataset 分為 training set 與 test set 。（統一翻譯：bidder 競標者、auction 拍賣、bid 出價）
2. 以下資料中，ip, url, time, payment_account, address 都有加密。
3. 從兩個資料集結構可知，如果我們要判斷這個競標者是不是機器人，需要取出該競標者的所有交易資訊，然後以此作為判斷依據。

ii. 介紹

1. train, test：訓練和測試的目標，裡面包含
 - 甲、bidder_id：競標者的 id
(ex : 49bb5a3c944b8fc337981cc7a9ccae41u31d7)
 - 乙、payment_account: 競標者的付款帳號
(ex : a3d2de7675556553a5f08e4c88d2c228htx90)
 - 丙、address: 競標者的地址
(ex : 5d9fa1b71f992e7c7a106ce4b07a0a754le7c)
 - 丁、outcome：結果（只存在於 training dataset 中，0 表示機器人，1 表示人類）

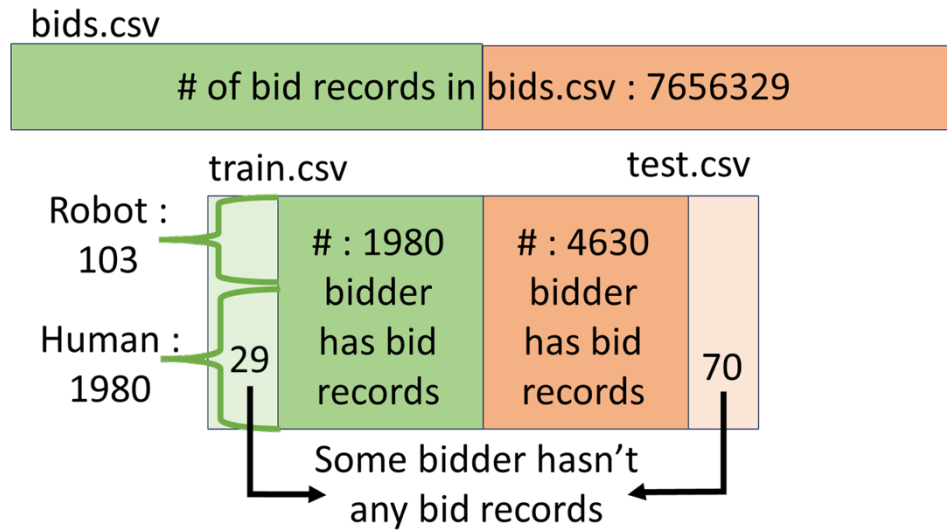
丙、 資料觀察

i. 思路 1

首先我們觀察 Kaggle 提供的資料集之間的關係，分別是：bids.csv, train.csv 與 test.csv。我們發現 bids.csv 裡面的 bid record 總數有 7656329，但其實只有 6614 個 bidders。也就是說一個 bidder 會有多個 bids。也因為上述提供的資料及觀察欄位，我們必須將 bids 的資訊融入至 train 裡面。藉此讓模型在 training dataset 中可以有更多資訊可以做判斷，而這些資訊就來自每一筆 bid。簡而言之，就是 bids.csv 是以 bid_id 作為主要 column，每個 bid 都有不同對應的屬性。而 train.csv 與 test.csv 則是以 bidder_id 作為主要 column。這也符合目的：分辨 bidder_id 是機器人還是人類。

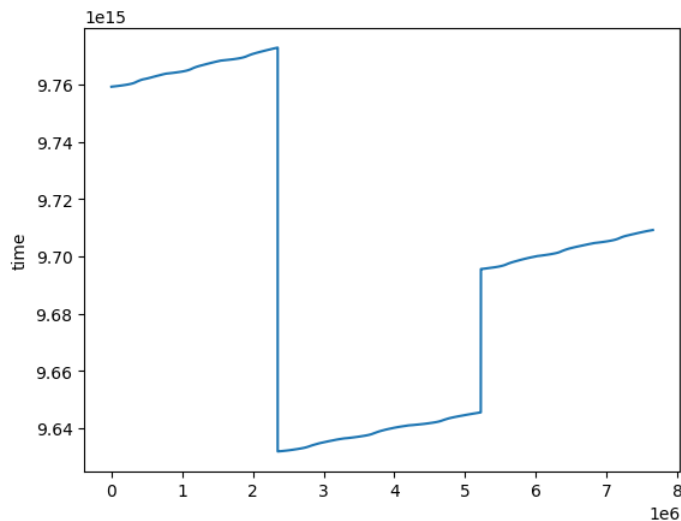
ii. 思路 2

為了要將 bids.csv 的資訊融入至 train.csv，我們對三個 dataset 進行分析。發現其中的集合關係，對於有些缺失的內容 (Some bidder hasn't any bid records) 我們將有數值缺失的資料丟棄(詳見參考資料 2-參考 Kaggle 比賽的討論區)。而 Robot 的數量與 Human 的數量非常不平衡，這是影響我們選擇模型的原因之一。同時由於 test.csv 裡面的 bidder 並沒有 output，我們將其先設為-1，再將資料拿去 fit model。因為我們在原先找資料集的集合關係過程中，發現將所有 output 設為-1 跑出來的答案，相當於全部都設定為 0，所以我們可以知道，如果今天輸入的值小於 0 或大於 1，會被自動設定成 0 或 1。而當我們再多測試幾次之後，我們發現，測試的資料集的 Robot 和 Human 的比例為 1:1，而這和我們訓練資料集的比例差距極大，而這正是我們其中一個要克服的問題。



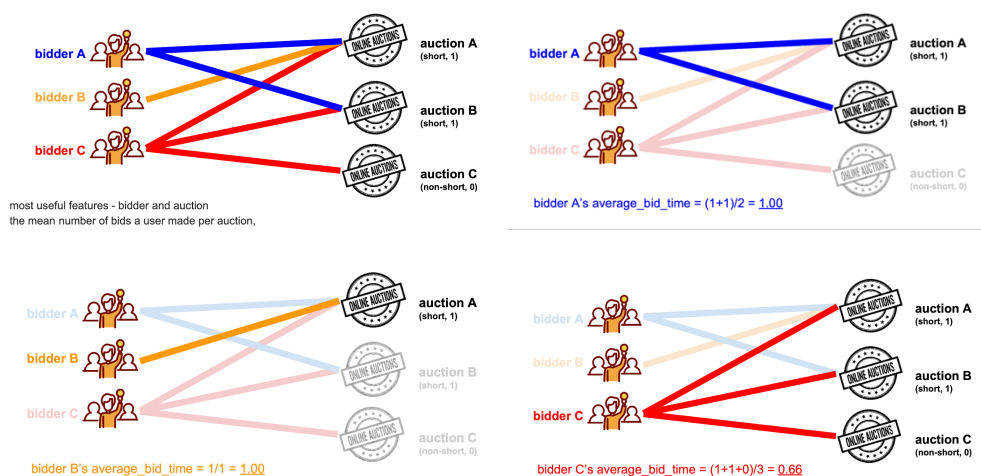
iii. 思路 3

- iv. 一個競標者有多次出價，原作者希望將多筆資料簡化為一個資訊。將出價時間直接轉換為 Unix time 會發現時間分佈呈現明顯的三個區間。原本經過加密、看似彼此之間關聯性低的資料，現在出現了明確的先後關係。



- v. 如此一來經過加密的時間就是一個特別的資訊。原作者結合時間欄位，與一個競標者(bidder)的出價(bid)紀錄，從而設計出一個簡化的函數。這個函數設計方式如下：一個 bidder 會參與多個 auction，以至於有很多的出價(bid)紀錄，同時在每個 auction 中也會進行多次出

價(bid)，如下圖。因此為了將這些出價(bid)紀錄都變成單一屬性 merge 到 training set。我們會將一個競標者(bidder)在一個 auction 的出價次數其中的時間為 78 個小時這時由上表看出來的，所有 bidder 頻繁出價的持續時間會為 3 天，在這 3 天內出價的次數分為「多次出價 / 少量出價」，分完後再將多次出價變成 1，少量出價為 0，然後投標者(bidder)參與多個 auction 再去做平均。下圖為計算方式的示意圖。



vi. 透過這樣的方式將兩種行為屬性

(1) 一個投標者在一個 auction 的投標次數

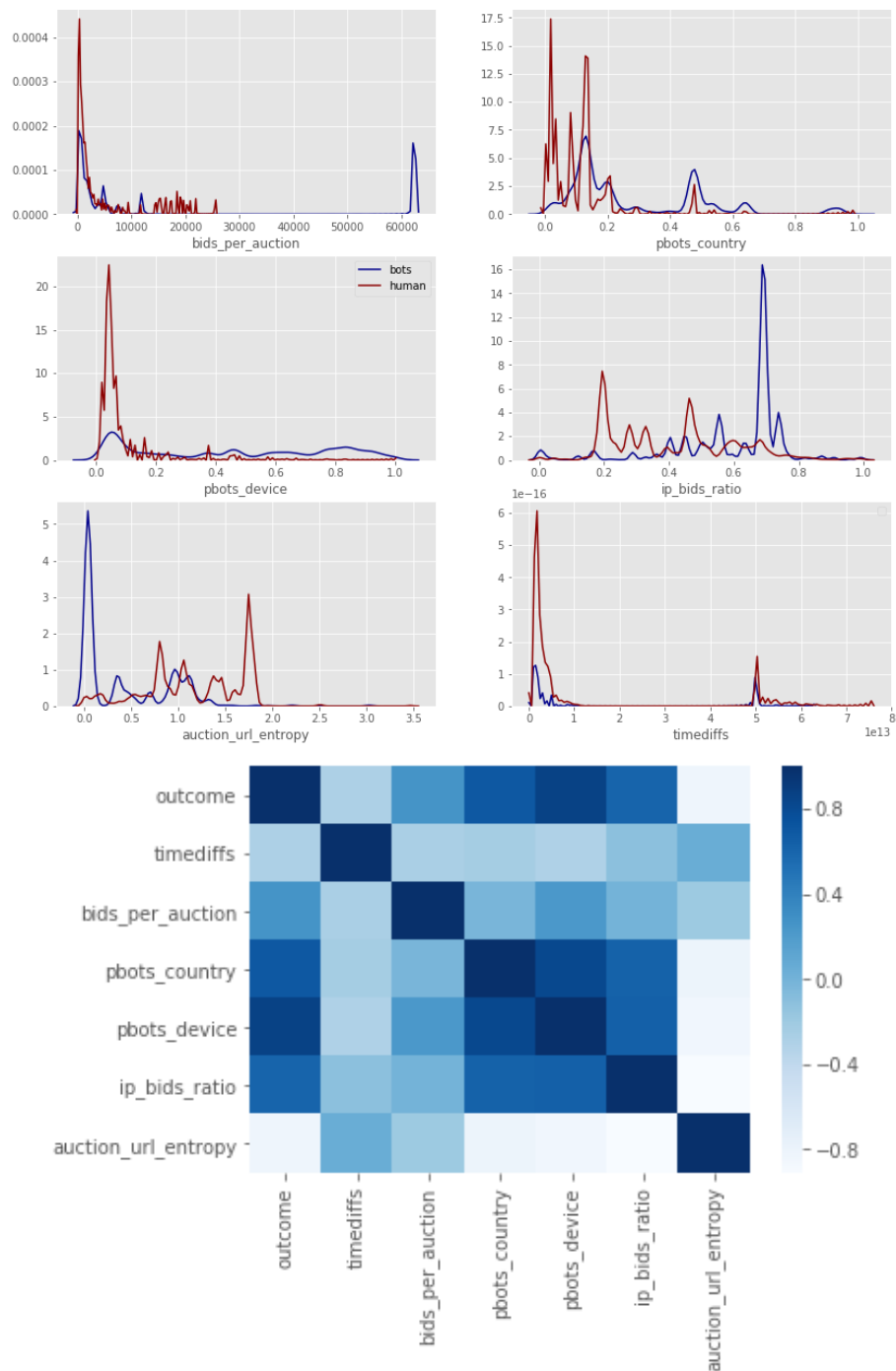
(2) 一個投標者參加多個 auction

以一種形式表達。我們自己思考其數學式背後的原理，robot 被設計用來贏得 auction，那麼他就會在一個 auction 中多次出價。然而為了不希望，這樣 robot 擁有一個 auction 中多次出價的資訊遺失，比如我們只是計算一個競標者出價的次數。那麼 robot 在一個拍賣會出價 5 次，等於 human 在五個拍賣會中各投出價 1 次，因為加起來都是 5 次。顯而易見這樣的方式會 information loss。所以才設計出這樣的方式來將 bids.csv 的資訊融入到 training set 中。

3. 資料處理

甲、資料觀察

i. 首先我們先檢查了那些 features 可以供我們使用。



- ii. 由於我們觀察到觀察單一屬性實際上找不到特別的特徵。所以我們開始進行到處 survey 前者的工作。看到別人會對於兩兩屬性進行相關度的比較。讓我們產生一種想法就是相關度如果很高，那代表這兩個屬性(舉例：A 屬性與 B 屬性)可能是有用的，舉例來說兩個 Bidder 擁有同樣的 A 屬性，而他們的 output 也是相同的(都為機器人或人類)，這件事情並不能說明這個 A 屬性特別有用，但是如果這兩個 Bidder 又都擁有同樣的 B 屬性，那麼就像一個輔助證明 A 屬性是具有資訊性的，因為其他的屬性也有相似的表現。所以從上述的 correlation matrix heatmap，我們可以發現一件事情，像是每個 device 的機器人比例，每個 country 的 robot 比例，以及唯一 IP 數量與競標者出價的次數有真正相關性。這件事情在我們 survey 很多個參賽者前輩也有提到，他們對此的解釋是，robot 的代理人(agent)，也就是會出一堆 robots 去設法贏得出價的人，也因此代理人可能只有幾個，代每個代理人都有多個 robot，那這樣 robot 出現在同個國家的比例就會提升，簡言之，robots 可能來自某些組織。所以這是一個有用的資訊可以去分析與探討的。

乙、 前處理

i. 總論

在最先開始的時候，我們原先的想法是想要在 training 時，讓每個 bidder 都可以保有其所有的 bids 資料，但經過一陣子的常識以後我們發現這是不可行的。取而代之的做法是，我們拿每個 bidder 的 bids 資料去作整合，擷取出多個 features 然後 append 回原本的 bidder 後面，以此達成提供額外的資訊。

ii. 個別介紹

1. $ip(X, bids)$

甲、此函式針對原始資料集中的 “bidder”, “auction”, “ip” 三個欄位討論。

乙、透過以下三個問答創造新的欄位加入原始資料集：

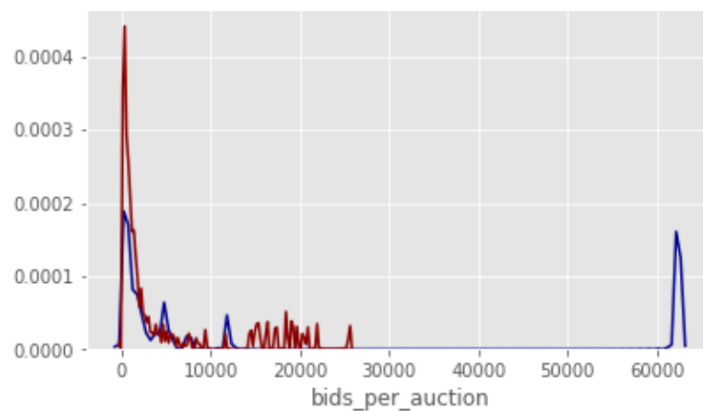
- i. bidder 在同一個 auction 中使用過多少不同的 ip? (計算 entropy)
- ii. 一個 ip 在同一個 auction 中曾經被多少個不同 bidder 使用? (計算 entropy)
- iii. ip 曾經被多少 bidder 使用過? (將數值資料轉換為二分。only_one_usr / many)

2. $bid_order(X, bids)$

甲、此函式針對原始資料集中的 “auction”, “bidder_id” 兩個欄位討論。

乙、統計同一個拍賣(auction)中不同 bidder_id 曾經做過的首次出價(first bid)，並將此出價順序數值 normalized。

丙、# of bids a user made per auction



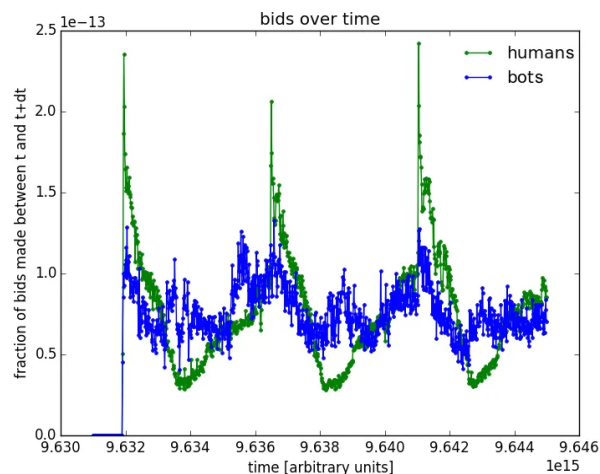
3. $dt(X, bids)$

甲、此函式針對原始資料集中的 “time” 欄位討論。

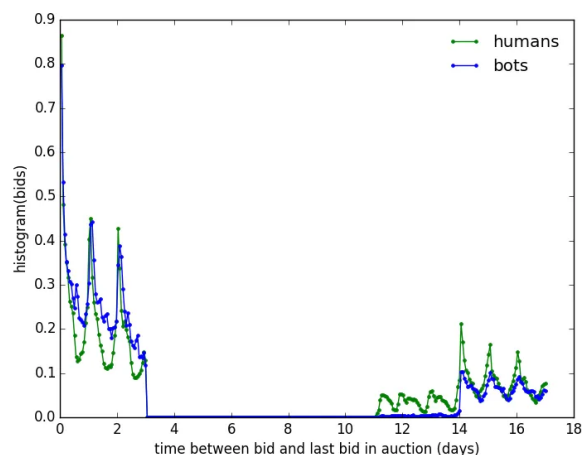
乙、由於 time 欄位是加密過的數值，直接轉換為 Unix time 之後，可以發現出價時間有一定的規律，一個週期大約是三天，而整體資料的時間尺度大約是一個月。藉此得到每個 auction 的 startt(start time)以及 one_day (一天) 的長度。

丙、原作者對於 time 欄位十分看重，但我們測試後發現不進行 dt 前處理的效果會比較好，推測是因為機器人和人類的行為基於時間的比較對於分辨兩者沒有特別行險的效果。

丁、bids-time



戊、bids-day (day 是基於 dt 產生的新欄位，會在後續介紹)



4. *day(X, bids)*

甲、基於在 *dt* 獲得的 “startt” 以及 “one_day” 數值計算每一次出價在拍賣的第幾天發生。

乙、將 “day” 資料轉為 “n_days”
(n=0,1,2,14,15,16,28,29,30)

丙、新增 “balance”，此欄位表示 bidder 在 ['monday', 'tuesday', 'wednesday'] 欄位間數值的變異程度

丁、新增 “balance2”，此欄位表示 bidder 在 ['0_day', '1_day', '2_day', '14_day', '15_day', '16_day', '28_day', '29_day', '30_day'] 欄位間數值的變異程度

5. *n_bids(X, bids)*

甲、此函式針對原始資料集中的 “auction”，“bidder_id”，“bid” 欄位討論。計算 bidder 在各個 auction 中的出價次數。

乙、統計中位數、平均數以及總出價次數製造新欄位。

6. *urls(X, bids)*

甲、此函式針對原始資料集中的 “url” 欄位討論，計算重複出現的 urls 及其出現次數。

乙、挑選其中出現超過 18 次的 url，並且以這些 urls 作為 column，計算 “bidder_id” 以及 “repeat_url_18” 對應的 entropy 數值作為新欄位。

7. *user_countries_per_auction(X, bids)*

甲、此函式針對原始資料集中的 “bidder_id”，“auction” 以及 “country” 欄位討論。

乙、計算 bidder, auction 與 country 的多樣程度，取棋 median, mean 以及 max value 製作新欄位。

8. *merch(X, bids)*

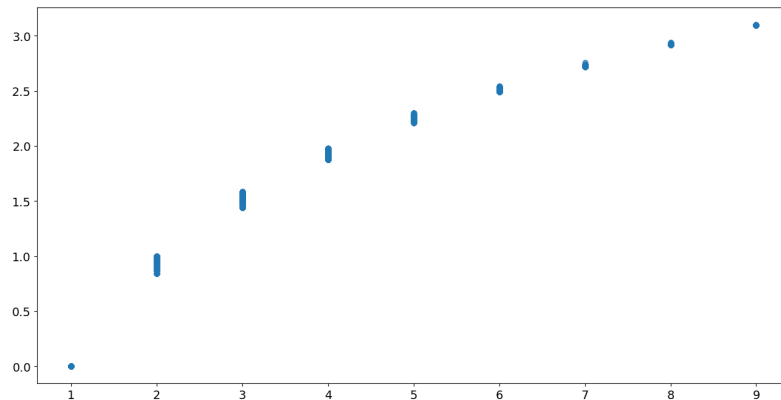
甲、全名為 merchandise，此函式針對原始資料集中的

“merchandise” 欄位討論。

乙、計算 auction 與 merchandise 欄位之間的 entropy 數值。

丙、將其轉為 one-hot encoding，我們曾經嘗試移除 one-hot encoding 的操作，但在保留 entropy、但沒有轉換為 one-hot encoding 的狀況下，訓練成果沒有進步，所以保留了此前處理。

丁、# of merchandise - entropy graph



丙、捨棄資料

Some data doesn't have its own information of bits. After searching in kaggle, we found that the competition hoster declared that those data would not be counted.

丁、異常值

There are five data are outliers in bid's info. These five data pretend themselves as robots by having only one bid info.

戊、資料數量級落差

這次比賽的一大特點為，train data 和 test data 的資料差距。train data 只有 2013 筆；但 test data 則有 4700 筆。但是如果藉由上傳假資料到 kaggle 可以發現，這次的比賽資料兩者的比例是 1:1，故這次比賽的其中

一個重點就是在於，如何利用比例懸殊(1910:103)的 train data 結果去學出正確的判斷模式就是這次比賽的重點。

4. 演算法與模型選擇

甲、嘗試過的演算法

- i. GradientBoostClassifier
- ii. Random Forest
- iii. SGD
- iv. SVC
- v. AdaBoost
- vi. ExtraTrees
- vii. Multilayer Perceptron

乙、最後的選擇

- i. Ensemble Random forest

丙、小結

- i. Multilayer Perceptron

在嘗試的過程中，我們嘗試自己建立 mlp，我們借由 one hot encoding，讓我們每個 bidder 有充足的 feature 可以去作訓練，但是因為 training dataset 整體資料還是太少，導致即使我們只使用三成的架構，分數仍然最高只能到 0.8 多。

- ii. Ensemble Random Forest

1. 為什麼我們不單單只使用 random forest 的原因不單單只是因為我們測出來的成績比較好，我認為 ensemble random forest 還有這些好處：

甲、更高的準確度

乙、更不容易 overfitting

因為我們的 training data 沒有很多，所以如果只使用單棵的 random forest 較容易 overfitting。

2. 使用的參數設定

甲、number of random forest = 5

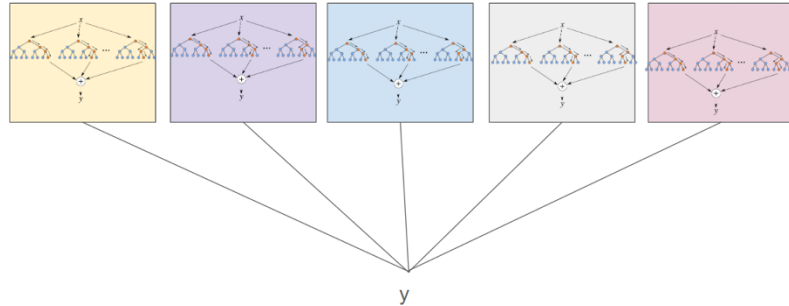
乙、n_estimators = 800

丙、max_depth = None

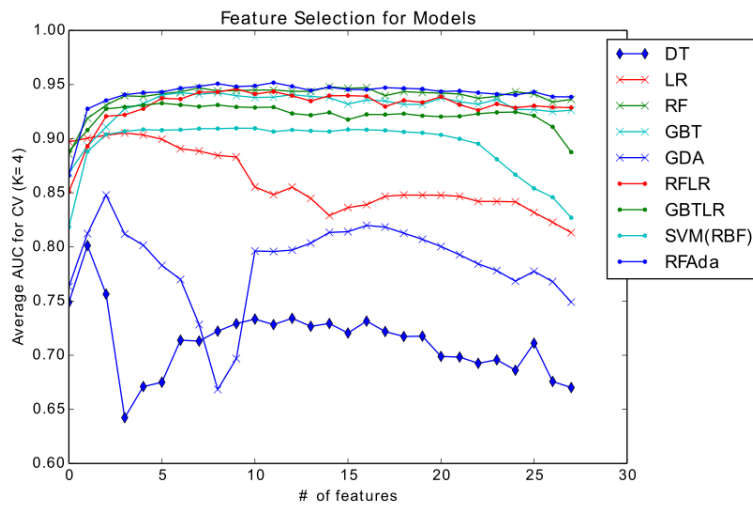
丁、min_samples_leaf = 1

戊、random_State = 1~5

己、criterion = entropy

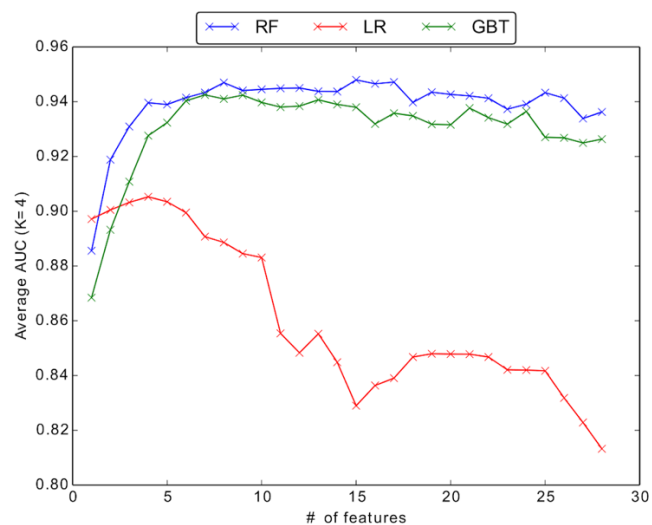


iii. Tree-based model



觀察上圖可以發現，SVM 的 AUC 較差，因為 data 不是可以透過 linear 去區分的，同時我們也可以發現除了 Decision tree model 以外，Tree-based model 在交叉驗證的準確性會表現好，這意味著複雜的 tree model 可以是區分 bidder 為 human or robot 很好的方式，同時因為 tree model 的方式也比較符合我們設計 dataset 的方式，因為我們都會將 country 還有 merchandise 的 label 作為 one-hot encoding。而像剛剛雷神巧克力(詳見 4.創新性 b 的討論)的例子，這樣背後的

行為模式在 tree based model 比較好被利用達到效果。像 tree based model 隨著# of features 增加，依然保持其高 AUC 的穩定性。



iv. Optuna

Optuna 是一個可以幫我們測試以及找出最適合的超參數的工具，但在我們尋找的過程中，叫難以找到比我們現行分數高的參數。我認為主要原因為訓練時間太長。如果要找到適合的超參數，則需要把所有可能的組合都是過一遍，且我們跑的過程中，還先跑 Cross Validation，然後再把總和相加當作這次跑的分數。故光跑一次的時間就非常高，如果再加上要跑所有的參數，我們認為投入的時間與收益不成正比。

5. 訓練方式

甲、data preprocess

乙、cross validation

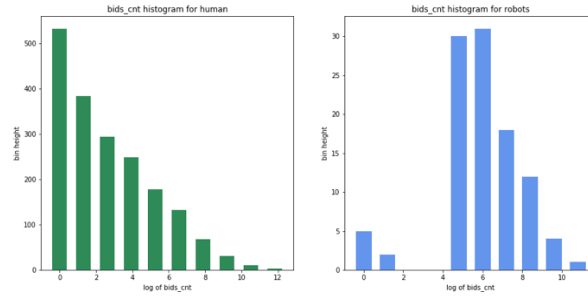
丙、Generate output

6. 創新性

甲、分析第一名的做法

第一名的做法是 robot 如果投標者(bidder)為 robot 的投標(bid)次數為 0，就會將其 5 個刪掉，因為他認為這個不符合常理，因為 robot 本身就是設計用來投標(bid)的，如果投標(bid)的次數為 0，那是違反其原先的目的，所以將其視為異常值將其丟掉。

```
outlier list = ['7fab82fa5eaea6a44eb743bc4bf356b3tarle',  
                'f35082c6d72f1f1be3dd23f949db1f577t6wd',  
                'bd0071b98d9479130e5c053a244fe6f1muj8h',  
                '91c749114e26abdb9a4536169f9b4580huern',  
                '74a35c4376559c911fdb5e9cfb78c5e4btqew' ]
```



但因為我們一開始就將投標者(bidder)如果沒有投標(bid)紀錄的刪掉了，所以已將這個做法考慮進去了。而至於為什麼我們沒有比第一名高分的原因，我們認為是因為我們沒有將資料處理的非常好，像是 attribute merchandise 與 attribute country，因為他是類別，也因此原先是采用 one-hot encoding 的方式，但這樣就會有老師上課提及的維度詛咒的問題，也就是當維度很高時，會需要用更多筆資料才能去完整描述所有維度。不然會有高機率有那種 case 是這個維度所有筆的 data 都相當分散，而如果想當分散，那那個維度的意義完全不會對模型學習帶來甚麼資訊量，反正會使模型陷入 overfitting 的問題。

乙、實驗

也因此我們有嘗試將這個 one-hot encoding 的資訊變成 entropy 的計算方式，而我計算 entropy 的方式是一個 auction 的參與者們會透過搜尋多個 merchandise 找到當前的拍賣。而我將一個 auction 是由多少個 merchandise 搜尋進來的來替代 one-hot encoding。這麼做的想法是如下：因為機器人可能都是透過查詢一個 merchandise 而進到該拍賣 auction，所以可能機器人參與的 auction 的 entropy 就會比較低。因為相比於人類一系列神奇的思維與跳躍性的想法而點了某商品引導到該拍賣會 auction，機器人比較沒有這麼複雜的思維。然而，我們卻發現我們做出來的實驗結果卻不如我們的預期。one-hot encoding 仍然為模型帶來最大的資訊量。附圖是我們的結果。

Methods	Area under ROC curve
one-hot encoding	0.9388 (best)
entropy	0.93644 (worst)
without	0.93849

對此結果我們思考了其中的原因，為什麼我們就算用其他方式而避免用 one-hot encoding 至少解決了維度詛咒的問題，然而分數卻比甚麼都不做還來的低，反而是最糟糕的。分析如下：用一個例子句例，因為 merchandise 就相當於查詢 keyword，有一年大家(human)都會因為查詢雷神而得到巧克力拍賣(auction)的資訊。然而，這件事情可能不會被 robots 學習到。robots 可能還是透過他們既寫好的程式去廣大的搜尋 keywords 找到各大拍賣會，再進行拍賣。因此假如雷神是一個 merchandise 的 label，那如果我們將 merchandise 的 label 變成 entropy 去做分析，就將失去這個 bidder 他是從雷神這個 merchandise 找到這個拍賣會的資訊。當失去這個有利的資訊，模型便無法透過發現人類都會透過搜尋雷神而找到巧克力拍賣(auction)的行為模式。這樣可能會導致資訊量下獎，甚至比甚麼都不做直接拿 label 去辨識還來的糟糕。








丙、 消融實驗

因為有些屬性就如同我們在 4.a，我們並不明白每一件事情背後真正發現甚麼唯有，做了實驗再透過該屬性的變化對於最後 AUC 值的影響來猜測這樣的做法是否符合人類社會常見的現象，還是只是猜想。因為我們找的最原始的程式碼對於資料進行非常多特別的處理，這樣的處理可能有很多 background domain knowledge 的人才知道要怎樣

處理，也因此為了確認是否每個資訊是否都有效，我們做了 ablation study，發現拿掉了分析時間的函數卻比原先所有的嘗試來的高(詳見分析與討論 ablation study)。對此我們提出了猜想，因為可以看到原先的 EDA，機器人可能會有固定投標(bid)的行為模式，而人類也有固定投標(bid)的行為模式。然而看到時間軸的分布這人類跟機器人幾乎在那些時間段都會出現。只是出現的曲線長的不一樣(如下圖)，但其實頻率相同。這件事情可能有一定程度的影響，但需要真正的讓模型學會這樣的行為模式我認為是更有高難度的。而像採訪中第一名說獲勝的方法只是將 robot 如果 bid 次數為 0 的 data 拿掉(也就是在 a 出現的五筆 bid id)就可以。而我們最後的結果也是進行 ablation study 才獲得最高分的。

7. 分數

最高分 public score : 0.93344

Facebook Recruiting IV: Human or Robot?			
Overview Data Code Models Discussion Leaderboard Rules Team Submissions			
All Successful Selected Errors			
Submission and Description	Private Score	Public Score	
 best_score_remove_ip.csv Complete (after deadline) - 4d ago	0.93799	0.92871	
 best_score_remove_urls.csv Complete (after deadline) - 4d ago	0.93613	0.92504	
 best_score_remove_merchandise.csv Complete (after deadline) - 4d ago	0.93636	0.93344	<input checked="" type="checkbox"/>
 best_score_remove_ip.csv Complete (after deadline) - 4d ago	0.93799	0.92871	<input type="checkbox"/>
 best_score_remove_bid_order (1).csv Complete (after deadline) - 4d ago	0.93837	0.9308	<input type="checkbox"/>
 best_score_remove_bid_order.csv Complete (after deadline) - 5d ago	0.93837	0.9308	<input type="checkbox"/>
 best_score_remove_dt.csv Complete (after deadline) - 5d ago	0.93947	0.93294	<input type="checkbox"/>

8. 分析與結論

甲、 實驗

進行消融實驗，可以發現像如果移掉 ip 與 countries per auction，綠色的那兩個資訊後，AUC 分數下降到 0.92 多，可以參考 5 資料處理 (Feature Engineer) 的 a 資料觀察。因為多個 robot 其實都來自同一個代理人(agent)，因此如何發現他們來自同個代理人可以透過觀察他們的屬性 ip 以及 在該 country 的 bidder 數量，因為一個代理人只會在一個 country，而由 ip 這個屬性本身就含有 country 的部分資訊。如果移掉這兩個資訊，就失去了這些分辨背後是否有代理人的資訊，因為人類不會由代理人去操控行為。

Ablation study		Private Score	Public Score
	Previous best private score	0.93918	0.93256
	Previous best public score	0.93887	0.93293
	remove ip	0.93799	0.92871
	remove bid_order	0.93837	0.9308
	remove dt	0.93947	0.93294
	remove day	0.9333	0.92368
	remove n_bids	0.93879	0.93193
	remove urls	0.93613	0.92504
	remove merchandise	0.93636	0.93344
	remove user_countries_per_auction	0.93503	0.92748

乙、 特徵擷取

從這次的比賽當中，我們可以知道 feature engineering 的重要性。由其是要怎麼從大量的數據去擷取出我們要的資訊，更是一大難題。這個任務除了要對程式碼本身熟悉以外，還需要有這個領域的 domain knowledge，才能判斷出重要的數據並整理和擷取出來。

9. 心得

在這個期末 project 中，我們這組因為看了很多人的研究與 code 所以做了很多的嘗試，在每次看到別人的作法時，都會覺得很驚訝他們是怎麼想到的，像是第一名的作法是會去觀察 dataset 哪裡有 outlier 異常值，而第二名的做法是會去一直觀察 attribute，並對每個 attribute 進行非常多前處理。我覺得這場比賽非常具有 Data Mining 的精神。一堆看似沒有相關的 attribute，我們要做的事情是從中發掘出他們的意義，轉化成更適合模型去分辨是 human 還是 robot 的 attribute，亦或是增加 attribute。最後再思考哪樣的模型更適合我們設計的屬性。

這個期末 project 是 data based 的比賽。真的很符合 data mining 的感覺。並不是要訓練我們去設計模型長怎樣。而是從問題本身出發，去探討這個問題給予的資訊量要如何使用轉化，使他們成為符合這個問題本身更有意義的資料供後面去做很好的分析。所以我們這組花了很多心力在專研別人的作法，研究他們的 code，思考他們這樣的做法背後符合甚麼社會現象，因為身為學生沒有問題的 domain knowledge，再觀察別人的作法後學習到很多我從未擁有的思考方式。這也是老師上課常常提到的不是看到甚麼東西都直接餵給模型，要先去思考問題本身，自己擁有甚麼樣的 data，才是找到解決之道最好的方法。

10. 程式碼

Notebook，這份沒有做消融實驗，所以分數會有小落差

(https://drive.google.com/file/d/1rNGSpUA1Q5OQZt6aFiVx0hNaIpr1g4Li/view?usp=share_link)

11. 外部資源與參考文獻

甲、 Kaggle 比賽的討論區

(<https://www.analyticsvidhya.com/blog/2015/07/top-10-kaggle-fb-recruiting-competition/>)

乙、 知道主辦方給的 dataset 有問題

(<https://www.kaggle.com/competitions/facebook-recruiting-iv-human-or-bot/discussion/14628>)

丙、 Reference code (<https://github.com/rinze/kaggle-public>)

丁、 Reference code – 2 (<https://github.com/crystalxs/Human-or-Robot>)

戊、 第一名的紀錄訪談 (<https://medium.com/kaggle-blog/facebook-iv-winners-interview-1st-place-peter-best-aka-fakeplastictrees-ea6090528db4>)

己、 第二名的紀錄訪談 (<https://medium.com/kaggle-blog/facebook-iv-winners-interview-2nd-place-kiri-nichol-aka-small-yellow-duck-7cc26c3cbac1>)

庚、 報告紀錄 (https://cs229.stanford.edu/proj2017/final-reports/5161146.pdf?fbclid=IwAR35LtWWY-hfNq0UaQuX0JFBpobfCou8afkXE_XxgKK4bxoKm_7LlxyVZYQ)