Machine Learning Final Project

Cyber Security Attack Defender

Team name: 我不是蘿莉控 只是我喜歡的女生 都剛好是蘿莉(ˋ・ω・ˊ)

Members and Work division

r05942040 沈文迪：preprocessing, building model, improve model (25%)

r05921075 鄭凱文：preprocessing, building model, improve model (25%)

r05921049 李建皜：preprocessing, building model, improve model (25%)

r05942072 吳昭霆：preprocessing, building model, improve model (25%)

Preprocessing/Feature Engineering

1. 我們觀察到 training data 的 4408587 筆中其實有很多是重複的，23 個 label 的統計重複資料如下表，去掉重複後剩下 979228 筆，我們將之輸出為 newtrain.csv，後面都是用這個 newtrain.csv 來 train。

Label	原資料筆數	去掉重複後的資料筆數
guess_passwd.	47	47
nmap.	2080	1396
loadmodule.	9	9
rootkit.	8	8
warezclient.	914	805
smurf.	2527107	2771
portsweep.	9328	3256
neptune.	964959	226289
normal.	875363	734536
spy.	1	1
ftp_write.	8	8
phf.	4	4
pod.	244	195
teardrop.	881	834
buffer_overflow.	25	25
land.	19	17
imap.	12	12
warezmaster.	18	18
perl.	1	1
multihop.	7	7
back.	1971	884
ipsweep.	11272	3469
satan.	14309	4636

2. 資料的第二三四個變數，分別是 protocol type, service, flag，是類別變項，也就是他們是文字，經過我們統計，protocol type 共 3 類，service 共 70 類，flag 共 11 類，所以我們的作法是將之分別轉換為 bag of word，以 protocol type 為例，其三個類別分別是 udp, icmp, tcp，轉換方式如下表。

udp	(1,0,0)
icmp	(0,1,0)
tcp	(0,0,1)

service, flag 也以此法轉換，故原本 training data 的長度為 41 維，現在減掉 3 維，加 3 加 70 加 11，變成 122 維。如此 data 中就都是數值變項，沒有類別變項了。

3. 訓練資料所需要用到的標籤, 在原始檔案裡面分別是 guess_passwd、 nmap.,、loadmodule.等 23 種標籤。所以我們一開始先把這 23 種標籤分別改成他對應到的 5 種輸出分類: dos、normal、probe、u2r、r2l。然後再根據這些分類，建立一個用來訓練模型的數列答案，和處理資料文字變數的方法一樣我們把它轉換成 bag of word，以分成這 5 類為例，轉換方式如下表。

Label	對應到的分類	轉換的數列
guess_passwd.	r2l	(1,0,0,0,0)
nmap.	probe	(0,1,0,0,0)
loadmodule.	u2r	(0,0,1,0,0)
rootkit.	u2r	(0,0,1,0,0)
warezclient.	r2l	(1,0,0,0,0)
smurf.	dos	(0,0,0,1,0)
portsweep.	probe	(0,1,0,0,0)
neptune.	dos	(0,0,0,1,0)
normal.	normal	(0,0,0,0,1)
spy.	r2l	(1,0,0,0,0)
ftp_write.	r2l	(1,0,0,0,0)
phf.	r2l	(1,0,0,0,0)
pod.	dos	(0,0,0,1,0)
teardrop.	dos	(0,0,0,1,0)
buffer_overflow.	u2r	(0,0,1,0,0)
land.	dos	(0,0,0,1,0)
imap.	r2l	(1,0,0,0,0)
warezmaster.	r2l	(1,0,0,0,0)
perl.	u2r	(0,0,1,0,0)

multihop.	r2l	(1,0,0,0,0)
back.	dos	(0,0,0,1,0)
ipsweep.	probe	(0,1,0,0,0)
satan.	probe	(0,1,0,0,0)

Model Description (At least two different models)

第一種方法　Generative method

分成 dos 和 normal (score=0.44287)：分數很低，模型沒辦法分類出來。

第二種方法　Logistic Regression

- 分成 dos 和 normal (score=0.88557)：因為 dos 和 normal 已經佔了大部分的資料數，所以光是只分兩類就已經有 0.88 的準確度。於是接下來嘗試分成 5 類。
- 分成 dos、normal、probe、u2r、r2l (score=0.88385)：Logistic 是使用 2 分法，所以我們按照下列順序：dos、probe、u2r、r2l 來分層分類出是否屬於這些資料，最後剩下沒被歸類在這 4 類的就將它標籤成 normal。分成 5 類之後，分數反而下降，我們直接觀察模型出來的分類狀況，發現訓練出來的模型輸入 test 資料後，u2r 和 r2l 完全沒有分類出來，統計數量一個都沒有。
- 先分出 dos、probe、和其他，再把所有資料和標籤是 u2r 和 r2l 的訓練資料直接做比較(score=0.88548)：由於上一個方法沒辦法分類出這兩筆資料，所以我們直接將 test 資料和 train 資料一對一進行比對，如果出現完全一樣的內容而且分類是 u2r 和 r2l 的話就把它標籤成 u2r 或是 r2l。結果使用此方法雖然增加一些正確率，整體的分數卻還是不如一開始只使用 Logistic 來分類 dos 和 normal 兩類，於是我們判斷 logistic 沒辦法處理分類出這種訓練資料某幾種資料樣本特別少的情況，於是改用別的方法。

第三種方法　Random Forest

- 分成 dos、normal、probe、u2r、r2l (score=0.95973)：分數明顯的增加。
- 分成 normal、probe、u2r、r2l 以及把 dos 細分成 6 類(score=0.96067)：由於 dos 佔了很大部份的資料比例，我們決定嘗試把 dos 分類細分一些，觀察能不能提高準確度，於是從訓練模型的時候，原本的分 5 類變成分 10 類，有觀察到分數有些增加。
- 先分出 normal 和其他，再把剩下不屬於 normal 的分成 u2r、r2l、probe 以及把 dos 細分成 6 類(score=0.96084)：由於把 dos 拆開分類有增加準確度，我們把第 2 位數量最多的 probe 也進行細分，把它拆開成 4 類，於是準確度有些微的改善。
- 先分出 normal 和其他，在把剩下不屬於 normal 的分成 u2r、r2l、以及 probe

細分成 4 類和 dos 細分成 6 類(score=0.96110)：由於 u2r 和 r2l 的數量太少，所以我們就不再把這兩個細分的更多。我們觀察到 normal 的數量似乎分類的太多，應該有一部份的 dos 被當成 normal，為了提高精準度，我們將 normal 和其他的分類先訓練出一個模型判斷是否為 normal，在把剩下不是 normal 的資料進行分類，做到這邊分數終於提升到高過 strong base line。

- 用答案數量來篩選正確標籤(score=0.96769)：透過上傳單一答案，我們分別找到各個分類的答案數量，再把我們分類出來的結果比較發現 normal 分類出來的數量太多，有很多不屬於 normal 的都沒有分類出來(約 2 萬筆)。於是我們改用模型來預測各個分類的機率，再按照順序：u2r、r2l、probe 來優先選取這幾筆資料。答案的 u2r 數量計算出來共 231 個所以我們就先取 u2r 機率最高的資料作為答案，以此類推接著做 r2l 和 probe，如果一筆資料同時被認為是 r2l 和 u2r，則以 u2r 為答案，因為 u2r 的數量最少。最後剩下的資料再直接用模型分類成 5 類加起來做為 test 資料的預測。用這個做法我們達到了目前 kaggle 上的最高分。各分類的數量如下表：

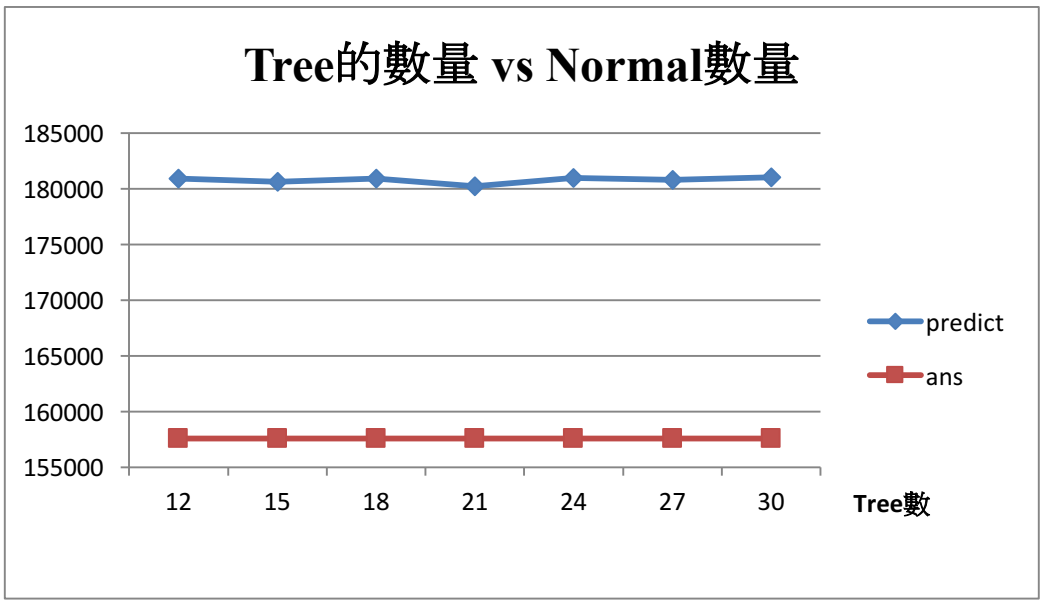
	Training data	Testing data
normal	734536 (75.0%)	157556 (26.0%)
dos	230990 (23.6%)	432209 (71.2%)
u2r	43 (0.00439%)	231 (0.038%)
r2l	902 (0.0921%)	8567 (1.41%)
probe	12757 (1.30%)	8216 (1.35%)

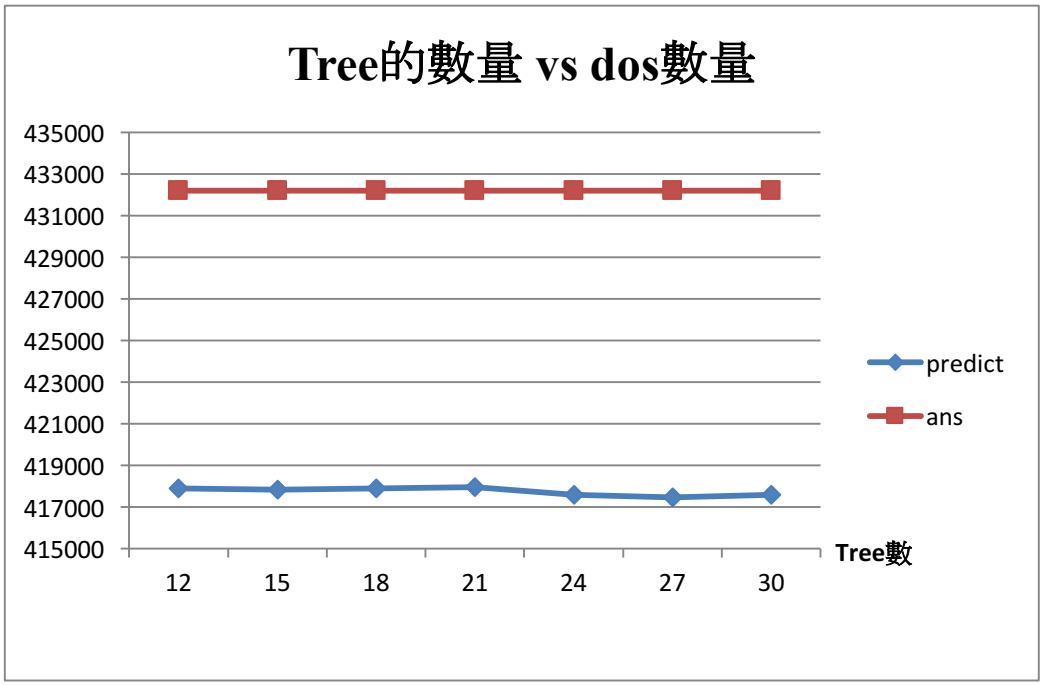
由於 training data 和 testing data 的分布比例很不一樣，因此單純使用 random forest 來做分類的方法有其上限。因此我們才會考慮依照模型預測的機率大小，由少量的類別優顯選取資料。

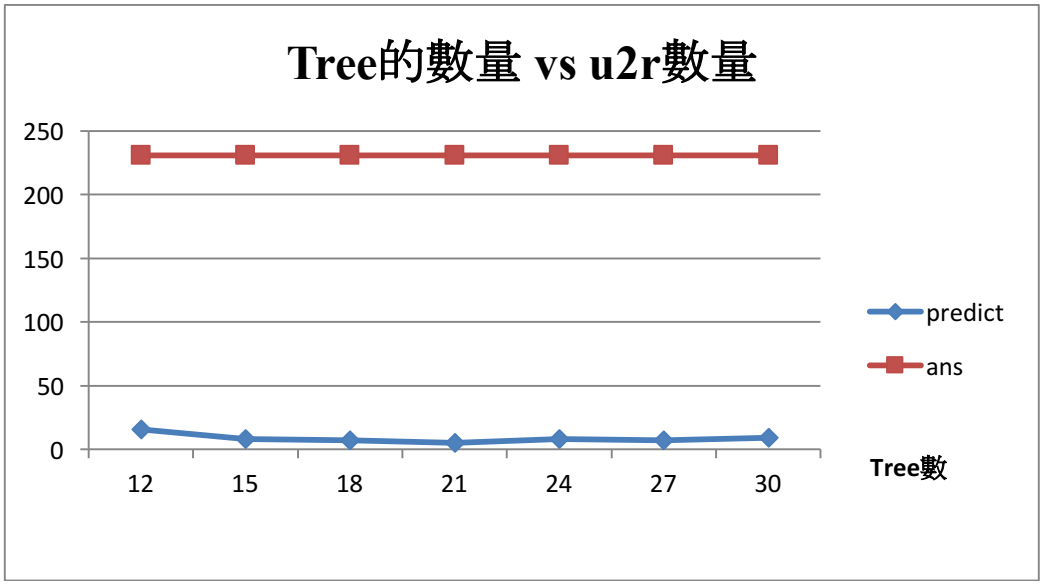
- 訓練模型的時候不使用 normal 來訓練，用答案數量按照順序篩選最後剩下的都歸類成 normal(score=0.93008)：上一個方法雖然用篩選的方式來做，可是 normal 的數量還是比正確答案差了 1 萬多筆，直接觀察各機率發現這些資料被判斷成 normal 的機率都是 1。為了達到正確的 normal 數量，我們重新做了一個新的模型，把 training 資料中屬於 normal 的不要丟進去訓練模型，因此這個模型只會分類出 4 種分類 dos、u2r、r2l、probe。按照上一個篩選答案數量的方法我們取了各個答案的數量的資料後，剩下的再分類給 normal。結果這個方法反而正確率大大的下降了，推測是因為 normal 的資料可能也具有某些特徵，用這個做法無法考量到那些特徵。

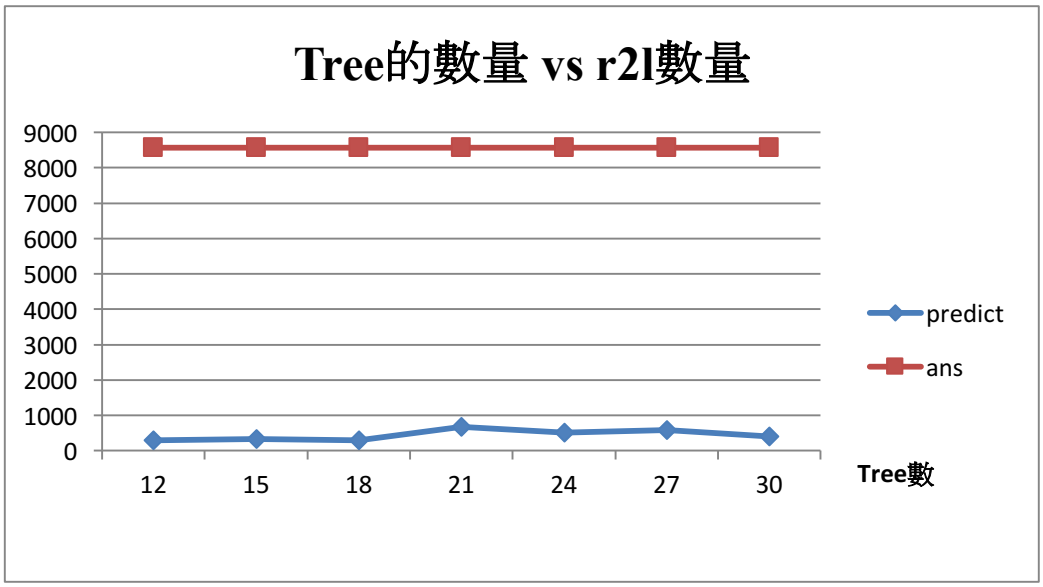
Experiments and Discussion

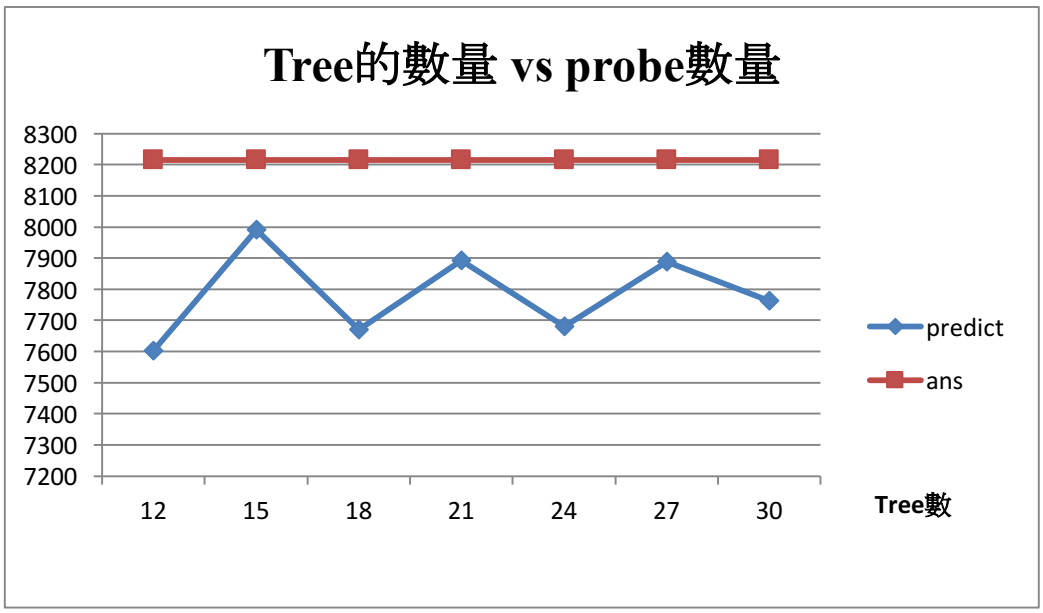
- Random forest Tree 數量變化





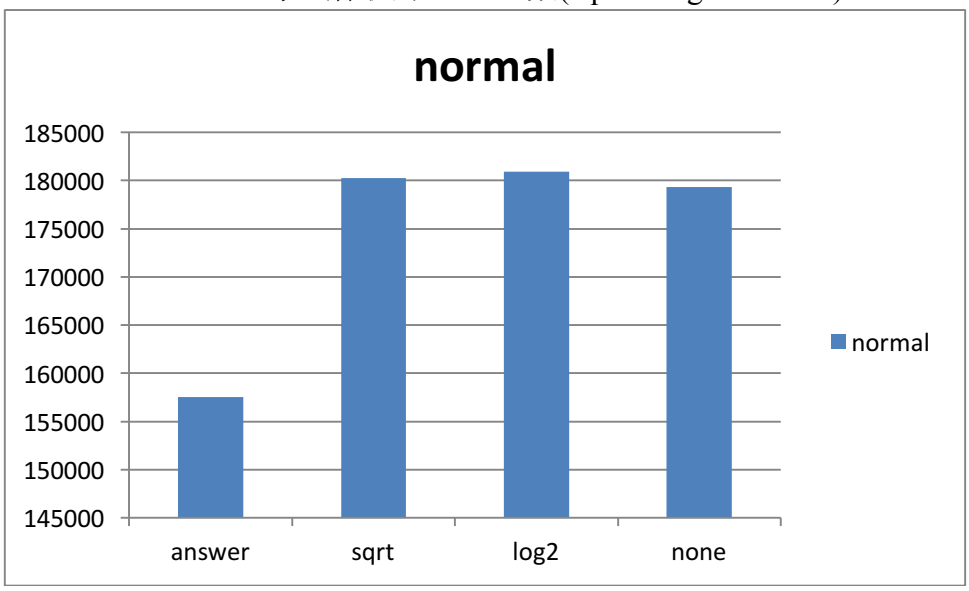

Tree的數量 vs u2r數量
250
200
150
100
50
0
12
15
18
21
24
27
30
predict
ans
Tree數

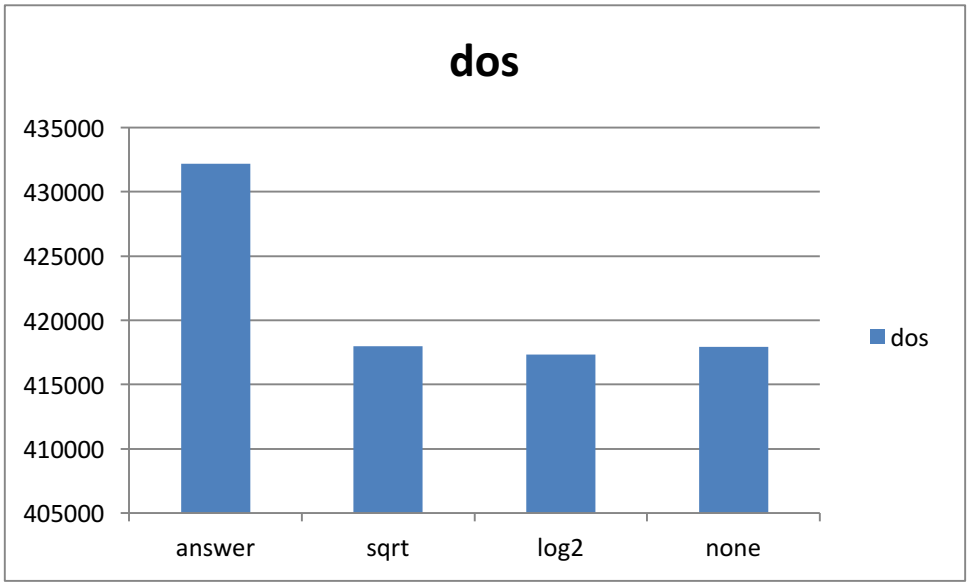

Tree的數量 vs r2l數量
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
12
15
18
21
24
27
30
predict
ans
Tree數

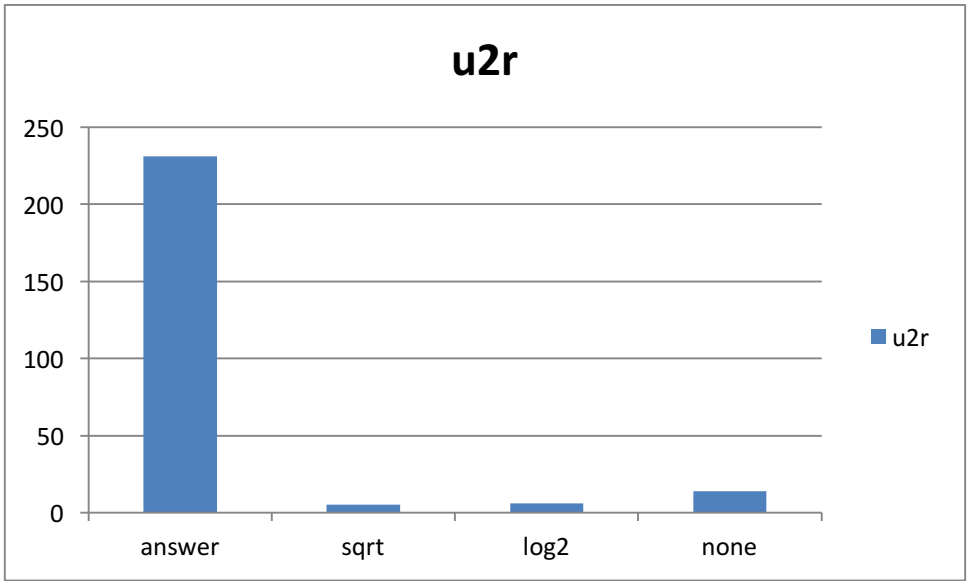

Tree的數量 vs probe數量
8300
8200
8100
8000
7900
7800
7700
7600
7500
7400
7300
7200
12
15
18
21
24
27
30
predict
ans
Tree數

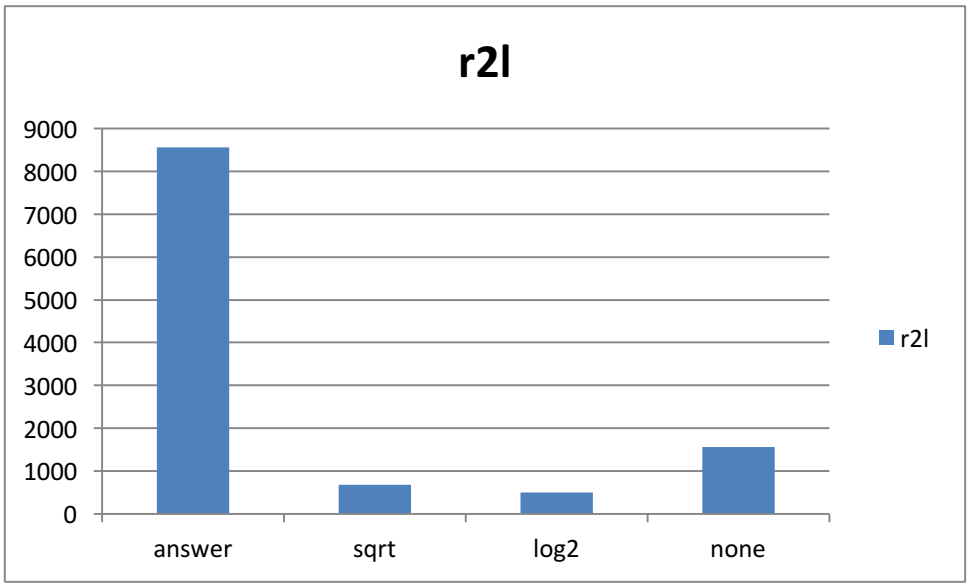
Normal, dos, r2l 的圖可以發現在 Tree 數量等於 21 時，predict 的結果都是最接近 answer 數量的，所以可得知較適合的 Tree 數量大約在 21 棵左右。

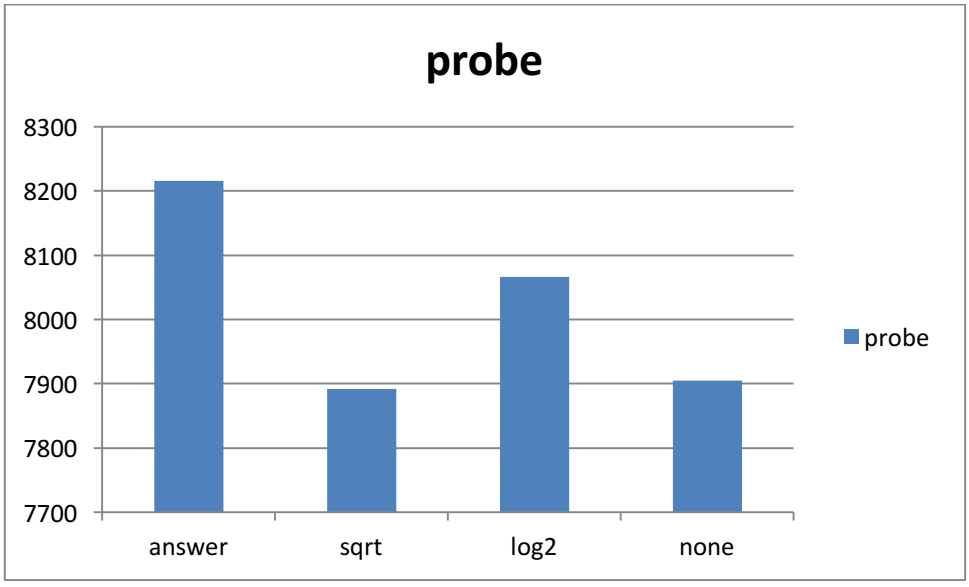
- Random forest 每一層取的 feature 數(sqrt vs log2 vs None)






u2r
250
200
150
100
50
0
answer
sqrt
log2
none
u2r


r2l
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
answer
sqrt
log2
none
r2l


probe
8300
8200
8100
8000
7900
7800
7700
answer
sqrt
log2
none
probe

Normal, u2r, r2l 可以顯著看到每個分支 train 時考慮所有 feature 時，會比較接近 answer 的數量，只有在 probe 會表現稍差一點，所以整體而言可以知道每個分支 train 時考慮所有的 feature 時會比較優，但缺點就是 training 時間顯著的增長。

我們 feature 的總數為 122，也就是 none 考慮的 feature 數量；sqrt 考慮的 feature 數量為 122 開根號，是 11 個；log2 考慮的 feature 數量是 122 取以 2 為底的對數值是 7。由於 feature 數量明顯增加了超過 10 倍，所以 model training 時間才會顯著上升，因此我們最後還是採用 default 的 sqrt。

- Logistic 複製樣本數過少的資料(balance)

有些分類在 training 資料裡面的數量太少(u2r, r2l)，訓練出來的模型也幾乎無法分類出這兩種分類，因此我們將 u2r 和 r2l 的資料複製數量到跟 dos 差不多的數量級。觀察訓練出來的模型，結果發現一樣分辨不出這兩個類別，沒有什麼作用，這個方法對於 logistic 沒有用處。