

Machine Learning Final Project

題目: Cyber Security Attack Defender

隊名: NTU_b03901057_XXXYYY 什麼鬼都可以吧

隊員:

b03901057 詹晉誠

b03901138 張晏祐

b03902035 黃兆緯

工作分配:

詹晉誠: Task1、Task3

張晏祐: Task1、Task3

黃兆緯: Task1、report

1. Preprocessing and Feature Engineering

1) Dataset

在做任何處理之前，首先檢查 dataset 的基本資料：

	Training set	Testing set
出處	DARPA'98 IDS evaluation program	
size	4408587	606779
# of features	41	
# of continuous features	34	
# of discrete features	7	

這個 task 跟 hw2 的 0/1 分類問題很相似，因此我們想用類似的方法來進行這個 task，這兩個問題的差別有：

- (a) hw2 資料裡的 feature 都是連續的，這次有部份離散 feature。
- (b) hw2 是 0/1 分類問題，這次是多分類問題（5 個分類、40 種攻擊）。
- (c) hw2 的資料集小很多，training set 只有 4000 筆，而這次有四百萬筆。

為了用類似 hw2 的方法來進行，在前處理時需要將離散 feature 轉為連續的（有數字大小意義的）。使用的 model 需要可以處理多分類問題（one-versus-rest 或 multinomial）。另外，由於資料集較大，model 的訓練時間是需要被考慮的因素。檢查完 dataset 的基本資料後，我們開始想辦法處理上述的三個問題。

2) Discrete features

Feature	Description	Type	Feature	Description	Type
1. duration	Duration of the connection.	Cont.	22. is guest login	1 if the login is a "guest" login; 0 otherwise	Disc.
2. protocol type	Connection protocol (e.g. tcp, udp)	Disc.	23. Count	number of connections to the same host as the current connection in the past two seconds	Cont.
3. service	Destination service (e.g. telnet, ftp)	Disc.	24. srv count	number of connections to the same service as the current connection in the past two seconds	Cont.
4. flag	Status flag of the connection	Disc.	25. error rate	% of connections that have "SYN" errors	Cont.
5. source bytes	Bytes sent from source to destination	Cont.	26. srv error rate	% of connections that have "SYN" errors	Cont.
6. destination bytes	Bytes sent from destination to source	Cont.	27. error rate	% of connections that have "REJ" errors	Cont.
7. land	1 if connection is from/to the same host/port; 0 otherwise	Disc.	28. srv error rate	% of connections that have "REJ" errors	Cont.
8. wrong fragment	number of wrong fragments	Cont.	29. same srv rate	% of connections to the same service	Cont.
9. urgent	number of urgent packets	Cont.	30. diff srv rate	% of connections to different services	Cont.
10. hot	number of "hot" indicators	Cont.	31. srv diff host rate	% of connections to different hosts	Cont.
11. failed logins	number of failed logins	Cont.	32. dst host count	count of connections having the same destination host	Cont.
12. logged in	1 if successfully logged in; 0 otherwise	Disc.	33. dst host srv count	count of connections having the same destination host and using the same service	Cont.
13. # compromised	number of "compromised" conditions	Cont.	34. dst host same srv rate	% of connections having the same destination host and using the same service	Cont.
14. root shell	1 if root shell is obtained; 0 otherwise	Disc.	35. dst host diff srv rate	% of different services on the current host	Cont.
15. su attempted	1 if "su root" command attempted; 0 otherwise	Cont.	36. dst host same src port rate	% of connections to the current host having the same src port	Cont.
16. # root	number of "root" accesses	Cont.	37. dst host srv diff host rate	% of connections to the same service coming from different hosts	Cont.
17. # file creations	number of file creation operations	Cont.	38. dst host error rate	% of connections to the current host that have an S0 error	Cont.
18. # shells	number of shell prompts	Cont.	39. dst host srv error rate	% of connections to the current host and specified service that have an S0 error	Cont.
19. # access files	number of operations on access control files	Cont.	40. dst host error rate	% of connections to the current host that have an RST error	Cont.
20. # outbound cmds	number of outbound commands in an ftp session	Cont.	41. dst host srv error rate	% of connections to the current host and specified service that have an RST error	Cont.
21. is hot login	1 if the login belongs to the "hot" list; 0 otherwise	Disc.			

根據投影片上的 feature 介紹，總共 41 個 feature 中，有 7 個 discrete feature，分別是：protocol type、service、flag、land、logged in、is hot login、is guest login。應用一些網路攻擊的知識可以知道，這些 discrete feature 對於判斷攻擊類型有很大的作用。例如 pod(Ping of Death)攻擊是使用 ping，因此 protocol type 是 ICMP。

為了讓這些 discrete feature 變為具有數值意義的 feature，我們使用的是 one-hot encoding，將每個種類變為一個維度，該維度僅有 0/1 兩種可能，0 代表原始資料並不是這個種類，相反地 1 代表原始資料是這個種類。例如 protocol type 有 icmp、tcp、udp 三種種類，則 encoding 舉例如下：

原始資料	Encoded
icmp	(1, 0, 0)
udp	(0, 0, 1)
abc	(0, 0, 0)

使用 one-hot encoding 後，discrete features 都被轉為具有數值意義的 feature（但這些新 feature 其實比較適合 decision-based 的模型，因為 0/1 的意義並不明確）。最後將原本的 41 維資料轉為 122 維資料，維度變為三倍。因此訓練時間變得更久，之後選擇模型要更加考慮訓練時間。

3) 多分類問題

從 0/1 分類問題變為 5 分類甚至 40 分類的多分類問題，其實並不是很大的問題，很多套件提供的模型都支援多分類問題（例如我們使用的 scikit-learn），即使不支援多分類問題，也可以針對每個分類訓練一個 0/1 分類模型，再對這幾個模型的輸出進行組合即可。

4) Dataset Size

這個 task 的 training set 有 440 萬筆，是 hw2 的 1100 倍，這造成某些 model 的訓練時間太久，會不方便進行測試。

首先我們發現有一些 data 是完全一模一樣的，排除掉相同的 data 可以減少資料量，但這會導致 training set 的分布改變（完全相同的 data 也許是增加權重資訊），因此我們並沒有排除相同的 data。

5) Feature Extraction

利用 one-hot encoding 後，變成 122 維的 data。首先我們考慮每個 feature 間的交互關係，可以加上兩兩相乘的 feature，但這樣一來每筆 data 就會超過 10000 維，遠遠超過我們能處理的大小，因此並不考慮這個方法。

利用 chi-square test 和相關係數來檢查 feature 和 label 的相關性。檢查的結果為除了特定幾個 feature 和 label 相關性特別高，其他 feature 和 label 的相關性都差不多，因此進行了下面兩個實驗：

(a) 只取相關性最高的幾個 feature

(b) 取所有 feature

最後實驗的結果為 (b) 較優，因此我們都取全部的 feature 來訓練。

2. Model Description

進行完資料前處理和 feature 選擇後，我們選定了幾種模型來做接下來的實驗。首先仿照 hw2 的作法，使用 Logistic Regression 和 DNN，另外，由於我們對 discrete features 做了 one-hot encoding，這樣的 feature 會比較適合 decision-based 的模型，因此選用 Random Forest 和 Gradient Boosting Decision Tree。

1) Simple models

包含 regression-based 的 Logistic Regression 和 DNN，以及 decision-based 的 Random Forest 和 Gradient Boosting Decision Tree。這些模型在 scikit-learn 中都有實作，我們都是使用 scikit-learn 的模型。

(a) Logistic Regression

課堂上有介紹過的模型，輸出 $y = \text{sign}(\text{sigmoid}(w^T x + b))$ ，可以用 gradient descent 來接近最佳解。

(b) DNN

多層類神經網路，最基本的 DNN 就是 Multi-layer perceptron，也可以使用 gradient descent 來接近最佳解，但隨著初始化不同會有不同的走向。最後一層使用 softmax 當作激活函數就可以解多分類問題。

(c) Random Forest

使用 decision-based 的 Decision Tree 當作基本的小模型，利用 bootstrap（取後放回的抽樣）的技巧來給予每棵樹不同的 training data，再將這些分類能力比較弱的樹作 bagging ensemble，得到最後的結果。

(d) Gradient Boosting Decision Tree

Boosting 是一種「改變錯誤」的技巧，每次訓練完一個小模型後，將這個模型分類錯誤的 data 的權重調高，再進行下一次訓練，如此一來可以將錯誤調整回來。而 Gradient Boosting 則是使用 Boosting 概念的一種優化方法，和 Linear Regression 的 gradient descent 很像，但 Gradient Boosting 是計算先前模型的 loss function 的 gradient，往 gradient 的反方向走。

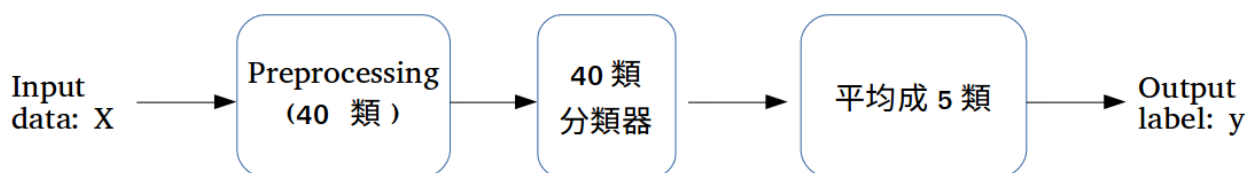
Gradient Boosting Decision Tree 就是使用 Decision Tree 作為 base model 的模型。

Simple model 的整體架構非常簡單，模型示意圖如下：



2) 40 Labels classifier

由於原本的 training data 的 label 是 40 種，然後再對應到 normal 和四種攻擊，因此我們嘗試不直接把 training data 對應成 5 種分類，而是直接訓練一個 40 分類的分類器，再將四種攻擊對應到的分類的機率平均起來，最後輸出成 5 種分類，模型示意圖更改如下：

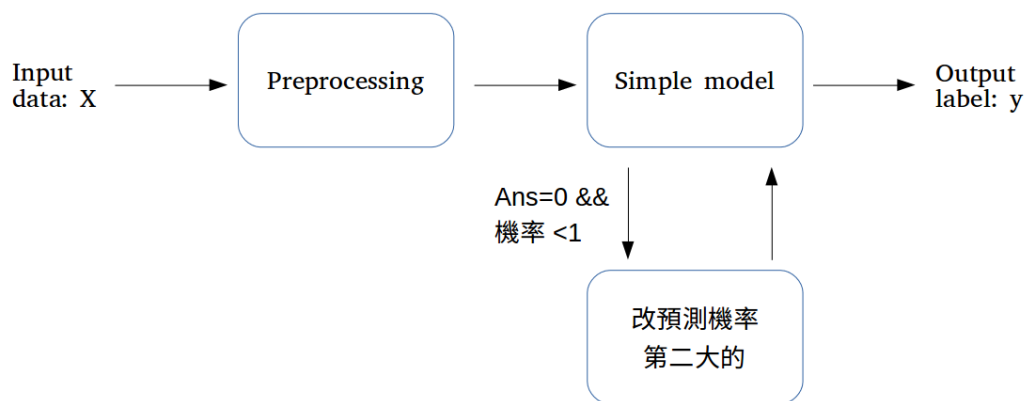


3) Altered models

由於這個 task 的 training set 和 testing set 分布相當不同，因此適當的矯正模型的輸出分布可以使預測 testing set 的效果變好（我們可以透過 kaggle 知道 testing set 的分布）。Altered model 就是在 simple model 的基礎上，對模型輸出的分布做調整，而我們做的調整有兩種：

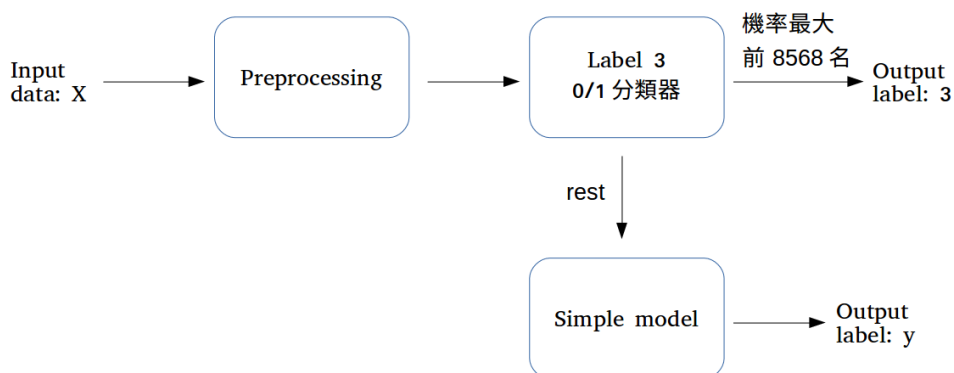
(a) 將預測為 label 0 重新檢查

由於預測 label 0 的比例太高了，因此將 label 0 預測機率小於 1 的所有 data，改為預測機率第二大的 label，模型示意圖更改如下：



(b) 優先填滿分布差異較大的分類

在比較 training set 和 testing set 的分布差異時，我們發現 label 3 的分布差異最大，因此優先考慮 label 3，訓練一個 label 3 的 0/1 分類模型，先將這個模型預測機率最大的 data 標記為 label 3，剩下的 data 再給 simple model 預測得到剩下的 label，模型示意圖如下：



3. Experiments and Discussions

1) Model 比較

		CV/OOB score	Public score	Training time
Simple model	Logistic Regression			>12 hours
	DNN	0.9997	0.95702	10 mins
	Random Forest n=100	0.99999	0.95752	3 mins / 8 thread
	Random Forest n=1000	0.99999	0.96128	30 mins / 8 thread
	GBDT, n=100	0.99999	0.96265	6 hours / 1 thread
40 labels	Random Forest n=1000	0.99999	0.96038	1 hour / 8 thread
Altered model	2.(a)		0.96415	
	2.(b)		0.96895	

可以看到幾乎所有 model 的 CV/OOB score 都是 0.999 以上，這代表對於 training set 的分布來說，這些 model 都可以分類的很好。然而到了預測 testing set 時，分數就降到了 0.96 左右，被分布差異影響很大。

訓練時間的部份，Logistic Regression 訓練花費太久（超過 12 小時），因此之後都不使用；DNN 的訓練時間雖短，但性能太不穩定，因此也不使用；Random Forest 和 GBDT 的性能差不多，但訓練時間短很多，所以之後都經常使用 Random Forest 來做實驗。

Altered models 的部份，2(a)更改 label 0 答案的模型，對於 public score 有微小的進步，但是不能確定這個進步是運氣好或是實際有效；2(b)強調 label 3 的模型，public score 進步的幅度就比較明顯。

2) 分布差異

從先前的實驗中可以發現，training set 和 testing set 的分布有著很大的差異，而 testing set (public set) 的分布可以透過 kaggle 的分數來知道，因此我們用了 4 次 submission 來看 testing set 每個分類各佔了多少比例，得出來的數字如下：

	0	1	2	3	4
Training set	19.856	79.281	0.00098	0.023	0.839
Testing set	25.966	71.230	0.038	1.412	1.354
預測結果	29.858	68.840	0.00066	0.066	1.235

在 training set 和 testing set 的分布中，差最多筆的雖然是 label 0 和 label 1，但以比例來說差異最大的是 label 3。從預測結果來看也可以發現，label 3 幾乎沒辦法正確地預測出來，而 label 0 則是預測錯誤太多，錯誤的 4% 中幾乎全部都是誤判為 label 0。

在這個分布差異下，我們想到了兩種模型來解決這個問題，也就是上面所提到的 Altered models。首先 2(a) 是想要將預測結果的 label 0 數量減少，而 2(b) 是想要將預測結果的 label 3 增加，Altered model 都是基於調整分布的想法建構的。

3) Altered models

使用 Altered model 來調整分布後，public score 都有上升，因此我們認為 Altered model 是有用的，以下簡單分析兩個模型

(a) 更改 label 0 答案 [2.(a)]

這個模型的 public score 是 0.96415，而原本的 simple model 是使用 Random Forest (n=1000)，所以原本的 public score 為 0.96128，大約有 0.3% 的進步。進步幅度雖然小，但仔細檢查後發現，更改答案後 label 0 大約佔 26%，其中正確的有 24.9%，也就是說誤判為 label 0 的錯誤從原本有 4%，更改後剩下 1%，這是很大的進步，也證明這個模型是可行的，可惜即使判斷出哪些是誤判，還是很難重新判斷是哪種攻擊，因此這個模型的進步幅度並不大。

(b) 強調 label 3 [2.(b)]

這個模型的 public score 是 0.96895，而原本的 simple model 是使用 Random Forest (n=1000)，所以原本的 public score 為 0.96128，大約有 0.75% 的進步，這個進

步幅度比起上一個模型來的明顯多了，原因我們推測是由於 label 3 的分布差異太大了，原本的 simple model 幾乎沒有能力分類出 label 3，要是能完美分類出 label 3，那麼就能有 1.4%的進步，因此現在 0.75%的進步是可以想像的。

4) Normalization

由於 training set 和 testing set 的連續性 feature 的大小不一定意義相同（時間不同可能會有差異），而且我們使用的是 decision-based 的模型，對於這種 scale 可能不同的因素很敏感，因此我們想要使用 normalization 來減少這種差別。

我們對 training set 和 testing set 分別做 feature-wise 的 normalization，接著一樣使用 simple model 來進行訓練，最後的結果如下：

		CV/OOB score	Public score
Simple model	Random Forest n=1000	0.99999	0.95028
	GBDT, n=100	0.99999	0.95296

這個結果並沒有達到我們的預期，因此實驗都沒有使用 normalization

5) 40 label classifier

由於原本的 training data 的 label 是 40 種，然後再對應到 normal 和四種攻擊，因此我們直接訓練一個 40 分類的分類器，再將預測出來的機率作平均得到 5 分類的結果。這樣的 model 的 public score 為 0.96038，略遜於原本的 simple model。我們推測原因為這 40 種 label 其實也是非常不平均，大部分集中在其中幾個 label，因此其他小 label 一樣難以預測，所以效果和原本的 5 分類問題非常接近，甚至稍微遜於原本的模型。

5. Reference

1) MIT Lincoln Lab. DARPA Intrusion Detection Evaluation
<https://www.ll.mit.edu/ideval/docs/attackDB.html#pod>

2) A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection, Anna L. Buczak and Erhan Guven, IEEE