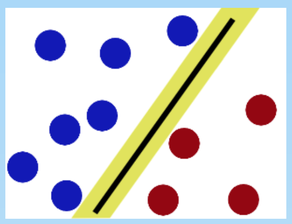
Privacy Preserving local SVM with Secret Sharing

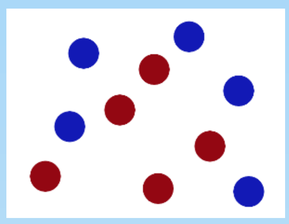
專題生: 江仕瑄　　元智大學資訊工程學系　110學年度專題製作成果　　指導老師:陳昱圻

一、摘要

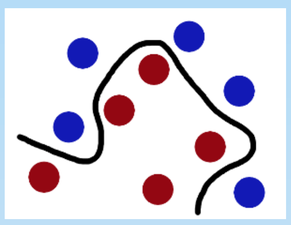
SVM是一種監督式的學習方法，用統計風險最小化的原則來估計一個分類的超平面(hyperplane)，其基礎的概念非常簡單，就是找到一個決策邊界(decision boundary)讓兩類之間的邊界(margins)最大化，使其可以完美區隔開來。
以下圖(一)而言，我們要透過SVM找到的就是那條黑色直線，使它能讓我們成功的分類成藍色區與紅色區。若是遇到圖(二)情況，我們可以用一曲線分割，而這條曲線在立體空間中，可視為用一平面分割，稱為超平面(hyperplane)。



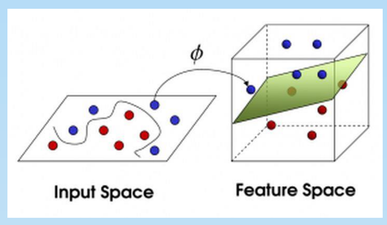
圖(一)



圖(二)



圖(三)



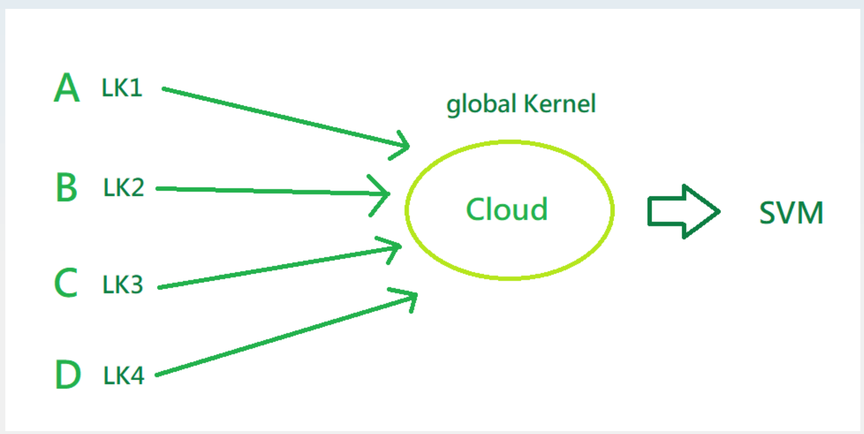
圖(四)

二、問題描述

為了使SVM達到一個更好的Privacy Preserving，假設在有一個不受信任的第三方下，如何安全的計算global SVM model，又不會將有關雙方的數據外漏是我們本次要解決的問題。

要達成這個目標，首先，我們先要有一個可以正常運作的local SVM，原本我們打算利用市面上已經做好的local SVM model去增加防護，但很遺憾的是我們並沒有找到一個合適的選項，所以我決定先自己刻一個local SVM model。以下是這個local SVM model的架構:每個參與者計算出各自的local kernel，傳到cloud後，計算出global Kernel，再透過SVM演算法計算出SVM model(圖五)。

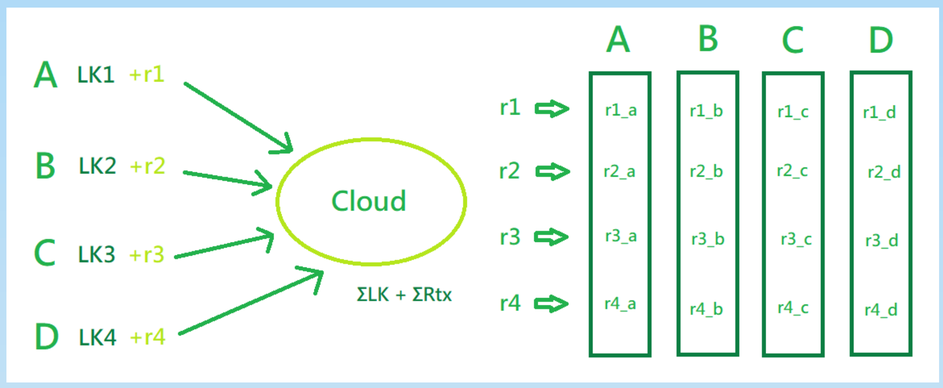
有了這個利用local Kernel實作的Linear SVM後，我們將會在合併global Kernel時使用Shamir's Secret Sharing Algorithm來達到更好的privacy preserving.



圖(五)

三、方法與步驟

整體的流程為:
1. 資料分割: 將dataset分成數筆local data，代表不同的參與者所擁有的資料。
2. SVM + SSS: (圖六)
(1) 分別計算local data的local kernel。
(2) 每個local kernel生成一個隨機的random matrix(ex: 參與者A 生成r1)。
(3) 每個local kernel將各自的random matrix透過Shamir's Secret Sharing Algorithm傳給每個參與者(ex: A 擁有r1_a, B 擁有r1_b, C 擁有r1_c, D 擁有r1_d)。
(4) 此時，每個參與者都擁有自己及其他參與者生成的random matrix的一部分(ex: A有r1_a +r2_a +r3_a +r4_a)
(5) 參與者將擁有的random matrix相加，連同local kernel傳給cloud。
(6) 此時，cloud擁有local matrix的總和及random matrix的總和，只要減掉random matrix的總和就可以得到 global kernel。



圖(六)

四、實驗結果

表(一)、表(二)為使用Dataset - Breast cancer diagnostic的結果，SVM accuracy: 0.934523。表(一)為在3個參與者、每個參與者有10 feature的情況；表(二)為在10個參與者、每個參與者有3 feature的情況。

以下記錄著三個階段的時間:
1.各個local 產生亂數 + 分割shares 傳給各個local
2.各個local把data轉成kernel matrix + 把拿到的shares加總傳給cloud的時間
3.Cloud 把 2.收到的東西還原secret最後得到global kernel的時間

表(一)、3 local kernels, 10 features in the local kernels

Prime(in secret sharing):	$2^{17}-1$	$2^{23}-1$	$2^{31}-1$
Each LK generate random matrix + sharing (on average)	5.370585(s)	4.891469(s)	5.059458(s)
Data to LK + Sum of LK & rMtx on cloud	0.008981(s)	0.007962(s)	0.007946(s)
Reconstruct secret + Get global kernel	3.235362(s)	2.771904(s)	3.811612(s)

表(二)、10 local kernels, 3 features in the local kernels

Prime(in secret sharing):	$2^{17}-1$	$2^{23}-1$	$2^{31}-1$
Each LK generate random matrix + sharing (on average)	13.161426(s)	12.712590(s)	13.333990(s)
Data to LK + Sum of LK & rMtx on cloud	0.024929(s)	0.024909(s)	0.025939(s)
Reconstruct secret + Get global kernel	32.686150(s)	27.258705(s)	32.175280(s)

最終我們成功用SSS演算法時做出一個local SVM，但由表可知，參與者越多，所花費時間成本越長，未來我們會著重在如何分配參與者、亂數矩陣生成，才不會使cost消耗太多。