Machine Learning Final

組別:MD709

D04922007 傅思維：DNN，Feature 抽取，Feature 分析。

R03921084 傅冠鈞：SVM，資料的前處理比較，Feature 抽取。

R03943133 陳重輝：Adaboost，Feature 抽取，Feature 分析。

一、資料的前處理

I. Data normalize的比較

首先我們嘗試以SVM比較Data有沒有經過正規化對performace的實驗。每一輪中我們隨機抽取10000筆資料，以9:1的比例把資料切成train跟validation data，再對3種正規化的方法進行比較，最後用validation data算error，這樣進行1000輪。

1. Minmax:

$$X_{normal} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

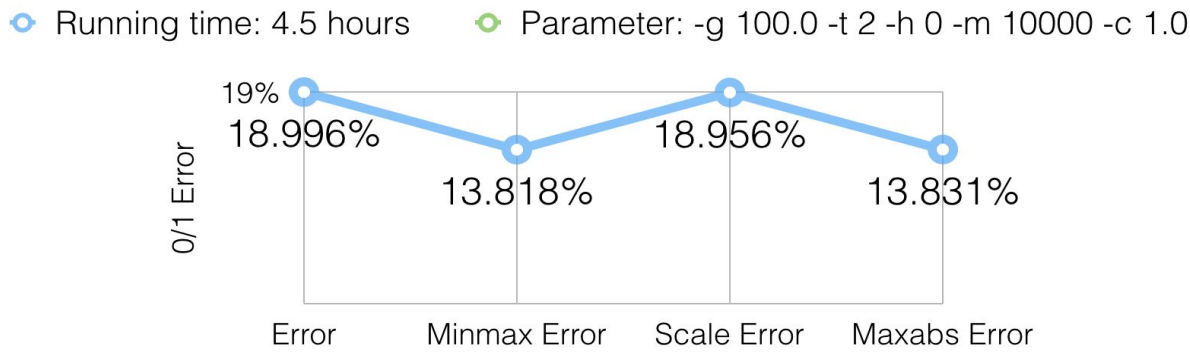
2. Scale:

$$X_{normal} = \frac{X - \mu}{\sigma}$$

$$\mu = mean \quad \sigma = standard\ deviation$$

3. Maxabs:

$$X_{normal} = \frac{X}{X_{max}}$$



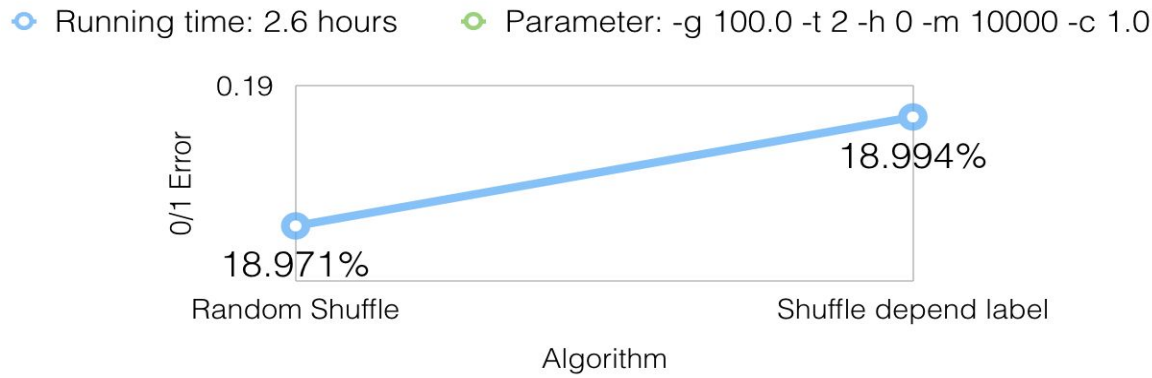
由實驗得知，對於SVM來說使用Minmax跟Maxabs這兩種正規化是有較好的performance，而不做正規化跟使用Scale兩者的performace其實差不多。而在Adaboost中尋找最好的decision stump時，對data做scaling並不影響結果，因此三種正規化方法皆不會影響adaboost的表現。而如果是用DNN，有沒有作正規化會對結果有很大的影響，如果是用Scale來做前處理由於有一些feature的值還是會很大，因而主宰著梯度的方向，使其收斂到較差的local minimun，所以使用Maxabs的表現會最好。

II. Data shuffle比較

在觀察資料的時候發現，Label是0或是1的數量差別很大，因此我們比較兩種抽取train data跟validation data的方法，對這兩種方法做實驗。每輪抽10000筆data，進行1000輪。

(A)Random Shuffle：Data進來的時候，直接使用Random的方式以9:1的比例去切train data跟validation data。

(B)Shuffle Depend Label：Data進來的時候，會先依照Label是0還是1做分類，各別對Label是0的data以9:1的比例切成train data跟validation data，Label是1的data也用同樣方式分類。最後在組合起整個train data跟validation data。



由實驗得知，有沒有根據Label去切data對performace是沒有差別的，甚至Random的切Error還比較低。

二、各別的Model的介紹及參數使用

I. Adaboost

每次iteration在各維度中找出最佳的decision stump作為g_t，並更新資料點權重，由這些較弱的g_t aggregation得到較強的G。在track1 預測dropout機率時，輸出 $G(x) = (\sum_{t}^{T} \alpha_t g_t(x))/(\sum_{t}^{T} \alpha_t)$，在track2處理帶有權重的資料時，將各資料點的權重起始值調整為題目要求即可。

上傳測試結果 track 1 : 0.957 accuracy track 2 : 0.873 accuracy

時間複雜度: (D: 維度數目 N: training data數目 T: iteration次數)

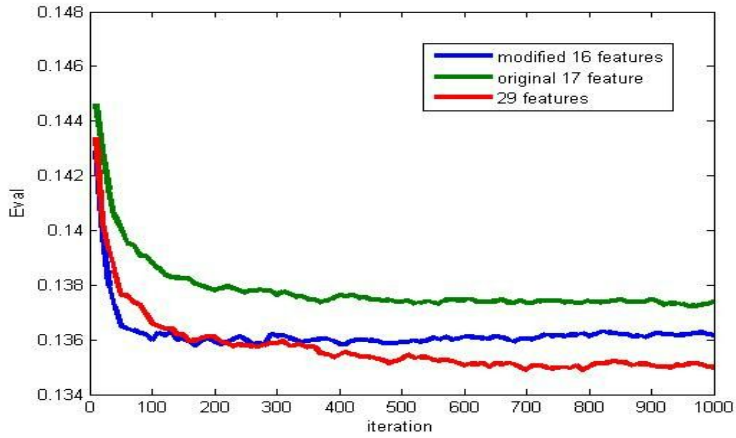
開始尋找各個g_t前，必須先對各維度做排序，所需時間為O(D*N*logN)，此步驟僅須執行一次。 每次iteration在一個維度中找到最好的decision stump所需時間為O(N)，有D維度以及T iterations，因此需要O(D*N*T) 。

結合以上結果，adaboost所需時間為O(D*N*(logN+T)) 。

Validation測試結果:

1. 使用助教一開始給予的17維度資料 (original 17 feature)
2. 加入組員們取出的feature (29 features)
3. 分析各個feature重要性後留下的16維度 (modified 16 features)

以上三者比較圖 (iteration次數:1000 執行5次平均結果)



II. SVM

直接使用林智仁老師的Libsvm library，因為在SVM中，如果超過幾萬的資料一起下去train時間會過長，因此在train之前會先tune參數，我們先抽取30000筆data以9:1的比例切成train data跟validation data根據不同的gamma,cost,degree(如果kernel是polynomial)，先以train data學習，以validation data驗證，如此去tune到最好的參數。最後以最好的參數把整筆data去train，這裡就不再切出validation data了，而直接去預測test data。

因為有Libsvm這個library，所以寫SVM時都滿好寫的，把資料處理好，參數調整好丟進去就可以了，只是train的時間真的太久了（往往都要好幾個小時到一天）。或許可能有些加速的方式，是以後在用SVM的時候可以去研究的。

Type	Kernel	Test Acc (#1 track1) (#2 track2)	Validation Acc	Parameter	Running Time Tune參數加上Train時間
c_svc	rbf	86.5%(#2)	86.3%	-g 10	---
c_svc	sigmoid	86.1%(#2)	85.1%	-g 0.037 -c 27.0	38 hours
nu_svr	rbf	92.3%(#1)	-------	-g 0.333 -b 1 -c 1.0	10 hours

III. DNN

這裡所使用的DNN為input-128-120-70-20-2的網路架構，每一層的activation function都是用leakyRelu直到最後一層用softmax，並使用5%的dropout rate來避免overfitting。由於是分類問題，cost function因而選用cross-entropy。最佳化的方式選用mini-batch gradient descent(batch size=128)並配合momentum、learning rate decay等。而此模型所使用的package是建構於theano和tensorflow之上的keras。

由於training data中dropout(“1”)的比例佔了快8成，所以是極度biased data，因此在算cross entropy時，我們嘗試針對不同類別的犯錯有不同的權重處罰(下表中的class_weight)。而此外因為track2的評分方式為weighted accuracy，所以我們也特別針對此提出一個處罰權重會隨sample(number of courses the student takes)改變的

cost function(下表中的sample_weight)，而其各種結果如下表所示。雖然我們試著改變處罰的權重，但對分類的正確率似乎沒有太大的影響。

DNN(17-dim)	minimun cross entropy	Validation (weighted) accuracy (%)	Test accuracy (Track2) (%)
no_weight	0.3360	87.60%	87.44%
class_weight	0.3356	87.56%	87.37%
sample_weight	0.2213	87.60%	87.43%
class&sample_ weight	0.3466	87.68%	87.41%

三、三個Model的比較

首先我們用助教給的17維特徵來做初步的嘗試，並且比較3種模型的正確率及執行時間(雖然執行的平台及使用的硬體皆不相同，但也可當作參考)，如下表所示:

17-dims	Validation accuracy(%)	Test accuracy(%)	Execution time (min)
Adaboost	86.26%	87.01%	26min
SVM	85.1%	86.1%	2271 mins
DNN	87.68%	87.41%	63min

從上表可知從模型的準確度來看，DNN的表現最好。由於DNN在每個batch中可以做平行化運算並透過GPU加速，所以可以大幅縮短執行的時間。而Adaboost需要透過前一輪的錯誤率來更新這次每個點的懲罰權重，儘管可能較不容易做平行化處理(雖然可以在算每個點的錯誤時平行化來算，但如果只是簡單的透過decision stump來預測的話，可能速度和sequential執行架構差不多)，不過由於演算法比較簡單，所以所需要的執行時間最短。而SVM如果是用QP來解，似乎也可以在kernel的計算上平行運算，或是用Gradient descent求解時也許也可以如DNN一樣在batch中加速，不過GPU的支援似乎比較不像DNN那麼普及。

四、新抽取的Feature及其對performace的比較

除了助教提供的17維特徵外，我們另外想出了12維可能對於判斷是否dropout的特徵列舉如下:

18. 這個enrollment最長多久沒有log紀錄（以秒為單位）
19. 這個學生最長多久沒有log紀錄（以秒為單位）
20. 這個學生對於這個enrollment的重視率
21. course dropout rate
22. 學生花多少時間(分鐘)在這個enrollment上

23. 學生有多少天有登入這個enrollment上

24. 登入時間的週期性衡量

25: 該enrollment所屬課程上傳discussion次數

26: 該enrollment所屬課程上傳problem次數

27: 該enrollment所屬課程上傳video次數

28: log時間點的平均值(只取時分秒 單位為秒)

29: log時間點的標準差(只取時分秒 單位為秒)

並使用這29維特徵，來輸入我們的模型裡，結果如下表所示:

29-dims	Minimun cross entropy	Validation accuracy(%)	Test accuracy(%)	Execution time (min)
Adaboost	ND*	86.5%	87.37%	35min
SVM	ND*	83.30%	82.40%	12min**
DNN	0.3398	87.77%	87.72%	61min

* ND means not defined.

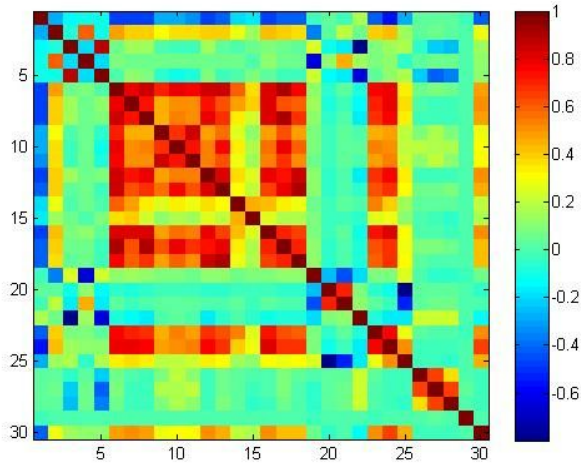
**在SVM的Execution time是拿之前的model所tune過的參數直接train29維的data，因此執行時間沒有那麼長。

可以發現和原始17維度的特徵相比，除了SVM外正確率大概都提升0.3%左右。

五、Feature的分析

I. 以相關係數分析

我們首先針對助教提供的17dim特徵及我們另外想出來的12dim特徵(共29dim)，分析其重要性。首先我們將training date中的label和29dim特徵串起來形成一個96434x30的矩陣，並計算彼此之間的相關係數(Correlation coefficients)如下圖所示。



第一行(列)代表各個特徵對label的相關係數，如果越趨近於紅色，代表正相關越大，即此特徵越大時label 越可能為1(dropout)，反之越趨近於深藍色時代表此特徵越大時

越不可能dropout。而這只是簡單初步分析哪些特徵可能比較重要，如果是相關係數小的特徵，並不一定代表沒用，也許和label之間只是存在著較複雜的非線性關係。

而如果我們將相關係數的絕對值大於0.4的特徵抽出來看，分別為第5、 6、7、11、12、15、16、17、22、23、29個特徵。

對應到的實體意義為:

學生在這個enrollment上登入幾次

學生在這個enrollment上登入幾次with different event sources

學生花多少時間(分鐘)在這個enrollment上

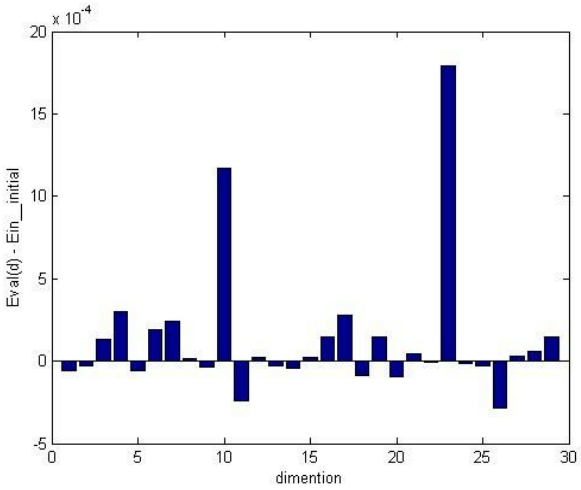
學生有多少天有登入這個enrollment上

log時間點的標準差

其中後面三個特徵是我們新增加的，而這些都是相關係數小於-0.4的，代表登入次數越多，花的時間越長，登入時間的標準差越大可能越不容易dropout。

II. 以加入雜訊(隨機排序)分析

接著我們對於特徵的重要性做了第二種分析。首先計算出未加入雜訊的Eval_initial，之後分別測試各維度加入雜訊(對特定維度d做隨機排序)之後Eval發生的變化。計算出Eval(d)，並與Eval_initial比較。所有維度都分析完畢後，留下Eval(d) > Eval_initial的維度，因為這些維度再加入noise後對整體表現造成負面的影響。至於Eval(d) < Eval_initial的維度，在加入noise之後反而得到更好的表現，因此我們認為這些維度的參考性較 。Training以及testing的時候或許可以忽略這些維度以節省時間與記憶體，同時不會對結果造成太大的影響。左圖顯示各維度的 Eval(d)-Eval_initial。由此推測 browser_problem 次數 (第10維)登入天數 (第23維) 可能是較為重要的feature。



六、推薦使用的Model與結論

除了第三小節討論的原因外，由於將DNN 最後一層的softmax結果直接拿出來用時就有機率的解釋，很適合拿來用於track1的應用。而track2的weighted accuracy也可以很輕易的整合到cost function裡因此我們推薦用DNN來實作，而且DNN還有很多另外的延伸，也是最近最熱門的機器學習模型之一。

這次的期末專題，不只要執行並比較不同的模型及演算法，還有一個重點是特徵的抽取。由於特徵工程也是機器學習裡重要的一環，因此我們也嘗試另外找出不少特徵，但對於預測的結果卻只有少量的幫助，要怎麼找出更有用的特徵，及有系統地把不重要的特徵刪除掉，可能是這次期末報告影響正確率的關鍵。