巨量資料分析

410473074 王姿文、410473002 林易霆、410473014 康益豪、410478016 張維翰

2019/06/18

1. 資料說明

1.1 資料名稱

Soil erosion and organic matter for central Great Plains cropping systems under residue removal

---

1.2 資料來源：

https://catalog.data.gov/dataset/soil-erosion-and-organic-matter-for-central-great-plains-cropping-systems-under-residue-re?fbclid=IwAR3cLtZiyGGUOoCgmnmMnCkQkPiQe2yGKGW_1qXyfgDWGbI5kO3E6Ivpkyg (https://catalog.data.gov/dataset/soil-erosion-and-organic-matter-for-central-great-plains-cropping-systems-under-residue-re?fbclid=IwAR3cLtZiyGGUOoCgmnmMnCkQkPiQe2yGKGW_1qXyfgDWGbI5kO3E6Ivpkyg)

1.3 資料背景與原始目的

- 資料背景：

本資料集為美國農業部針對中部草原地帶之農地所做的土壤侵蝕(Soil Erosion)和土壤有機質(Soil Organic Carbon)的觀察型資料。

- 原始目的：

本資料蒐集之目的為依照土壤侵蝕的程度來規劃美國農業部執行土壤環境維護之資源應用及分配。此資料即包含雨蝕、風蝕程度以及各種農地的基本資料、使用紀錄。

1.4 資料變數

- 原始資料：

本報告的原始資料共有37個變數，其中soil erosion 此變數本為一個連續型變數，我們根據網路上土壤報告，將其數值大於5設為severe，小於5設為minor，作為預測二元變數。其餘為農地、土壤本身性質、農作收穫量等等的解釋變數。

- 變數說明：

原始資料扣除ID、名稱、重複及有明顯共線性(soil erosion = watereros + winderos)之變數以後，具有15個類別變數、16個連續變數，共有31個實質變數，以下為各變數的說明。

變數名稱	Type	說明	變數名稱.	Type.	說明.
CoFIPS	cat.	County區域	tillage	cat.	灌溉方式
musym	cat.	土壤土種	soil erosion	con.	土壤侵蝕程度 (T/acre/year)

變數名稱	Type	說明	變數名稱.	Type.	說明.
muacres	con.	農地大小 (英畝)	sci	con.	土壤狀況指標
comppct_r	con.	農地土壤占比	sciom	con.	土壤生物分解物質指標
tfact	cat.	土壤侵蝕容忍分級	scier	con.	土壤侵蝕物質狀況
nirrcapcl	cat.	未灌溉土壤主要分級	scifo	con.	土壤人為生產物質狀況
nirrcapscl	cat.	未灌溉土壤副分級	Slope	con.	農地坡度
irrcapcl	cat.	灌溉土壤主要分級	removal	cat.	移除農作殘餘
irrcapscl	cat.	灌溉土壤副分級	Rem Res - ann avg	con.	農作殘餘平均移除量
farmlndcl	cat.	農地分級	crop1	cat.	第一年農作作物
awc_r	con.	土壤最高容水量	Rem Res - crop1	con.	第一年農作作物殘餘量
texture	cat.	土壤土質	crop2	cat.	第二年農作作物
rotation	cat.	幾輪耕作	Rem Res - crop2	con.	第二年農作作物殘餘量
yield1	con.	第一年收穫量	crop3	cat.	第三年農作作物
yield2	con.	第二年收穫量	Rem Res - crop3	con.	第三年農作作物殘餘量
yield3	con.	第三年收穫量			

1.4 研究問題

Prediction:

希望藉由本資料去預測Soil Erosion(土壤侵蝕程度)變數。

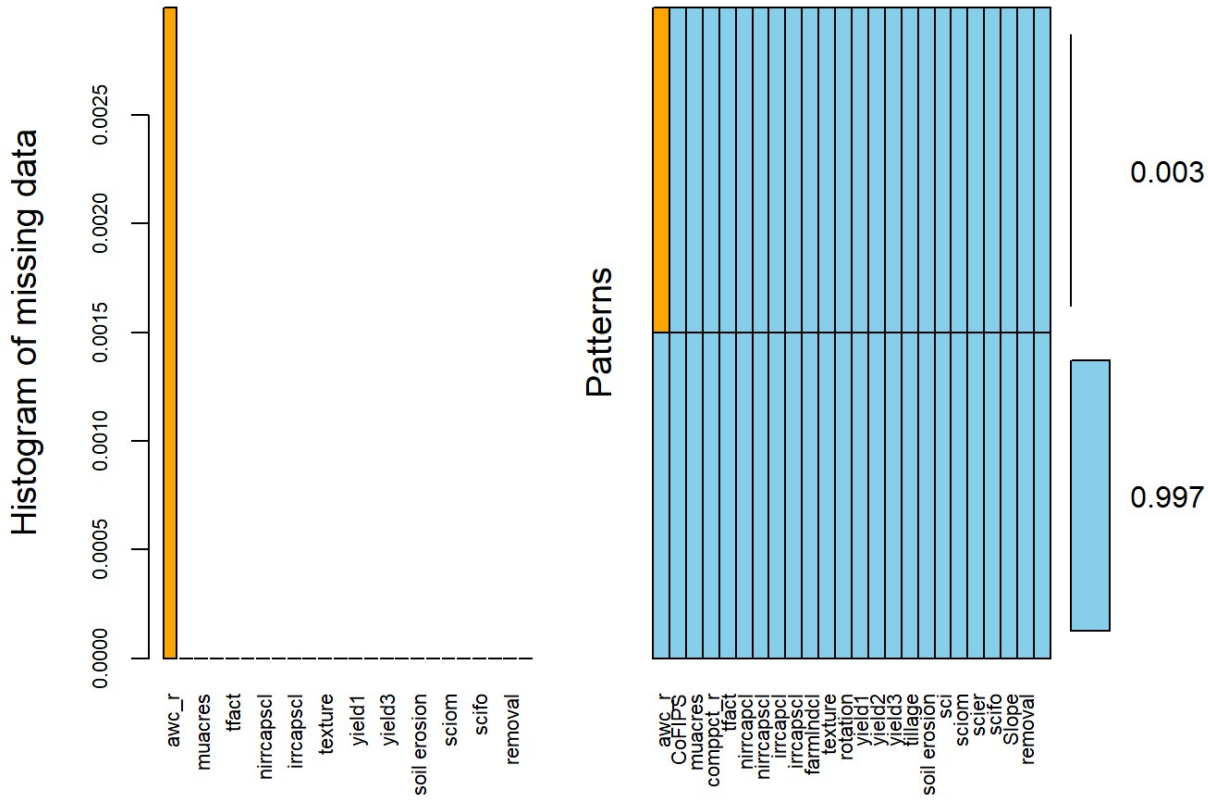
2. 資料清理

2.1 去除不必要的變數

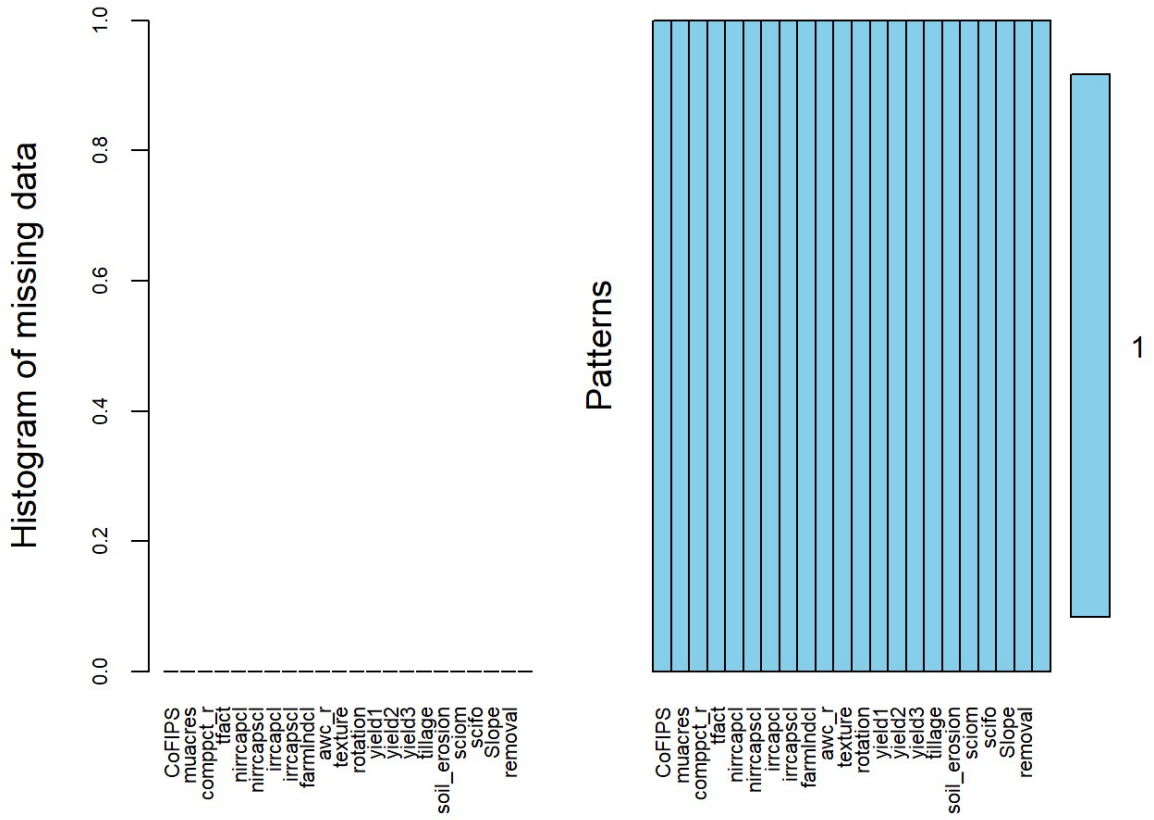
對於資料集而言沒有作用的變數以及存在嚴重關聯性問題的變數，都是我們不需要的。例如： `sci` 、 `scier` ，這些變數都與 `soil erosion` 有關聯。此外，例如： `musym` 、 `crop1` ...等對於資料集而言均沒有作用。我們便以上述來去除對於資料集而言不需要的變數，並且我們會在EDA部分再詳細探討關聯性問題。

2.2 遺失值處理

我們接下來計算各變數的遺失值比例，若遺失值比例大於百分之八十即予去除該變數。下圖中的左圖為 **Proportion Plot of Missing Data**，橫軸為變數名稱，縱軸為遺失值比例。若變數的遺失值比例超過百分之八十則予以刪除。由左圖可以看出只有一個變數 `awc_r` 存在遺失值比例(比例為0.003)，我們便對 `awc_r` 補值。



我們使用mice套件內的sample方法補值。下圖中的左圖為**Proportion Plot of Missing Data**，橫軸為變數名稱，縱軸為遺失值比例，可以看出最終無遺失值存在。刪減以及補值後共有 22 個變數，並以刪減完成的變數去做探索性資料分析。

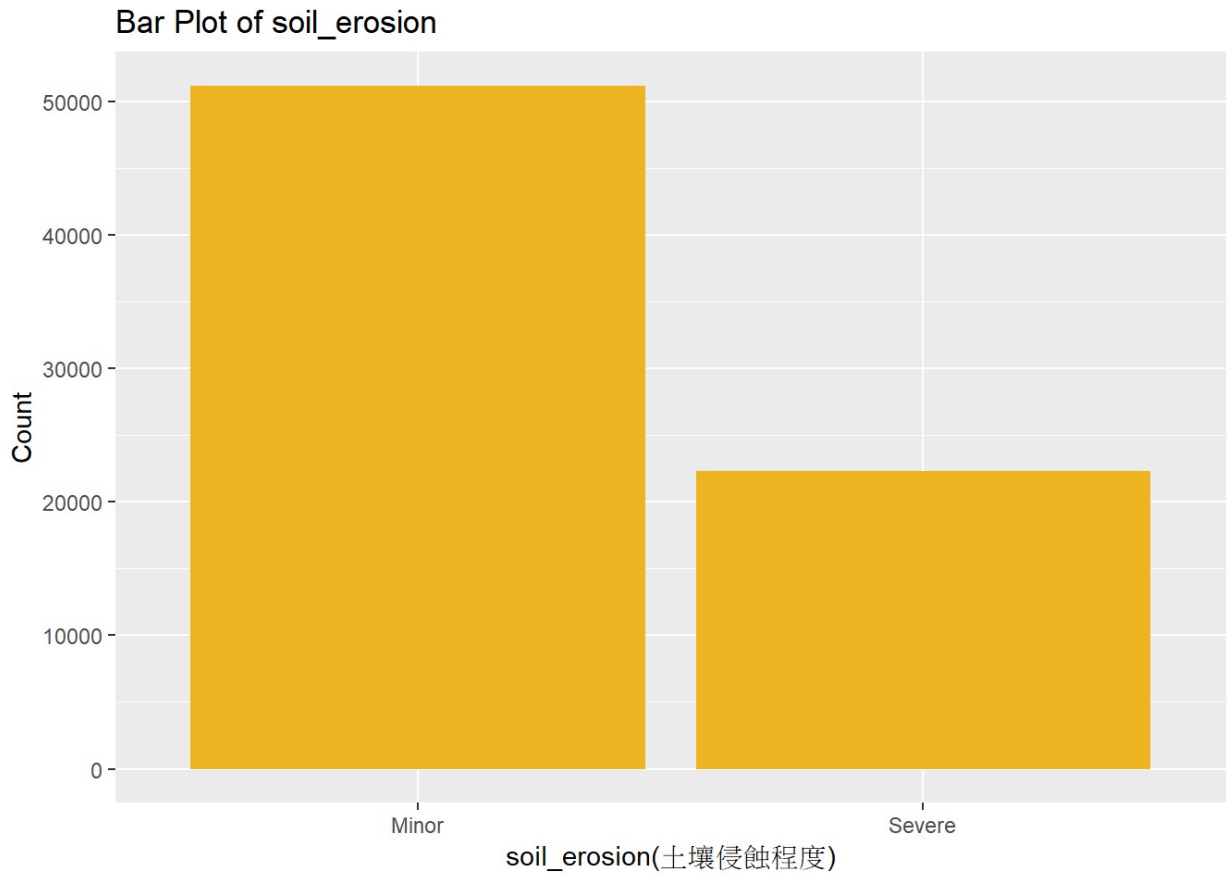


3. 探索性資料分析

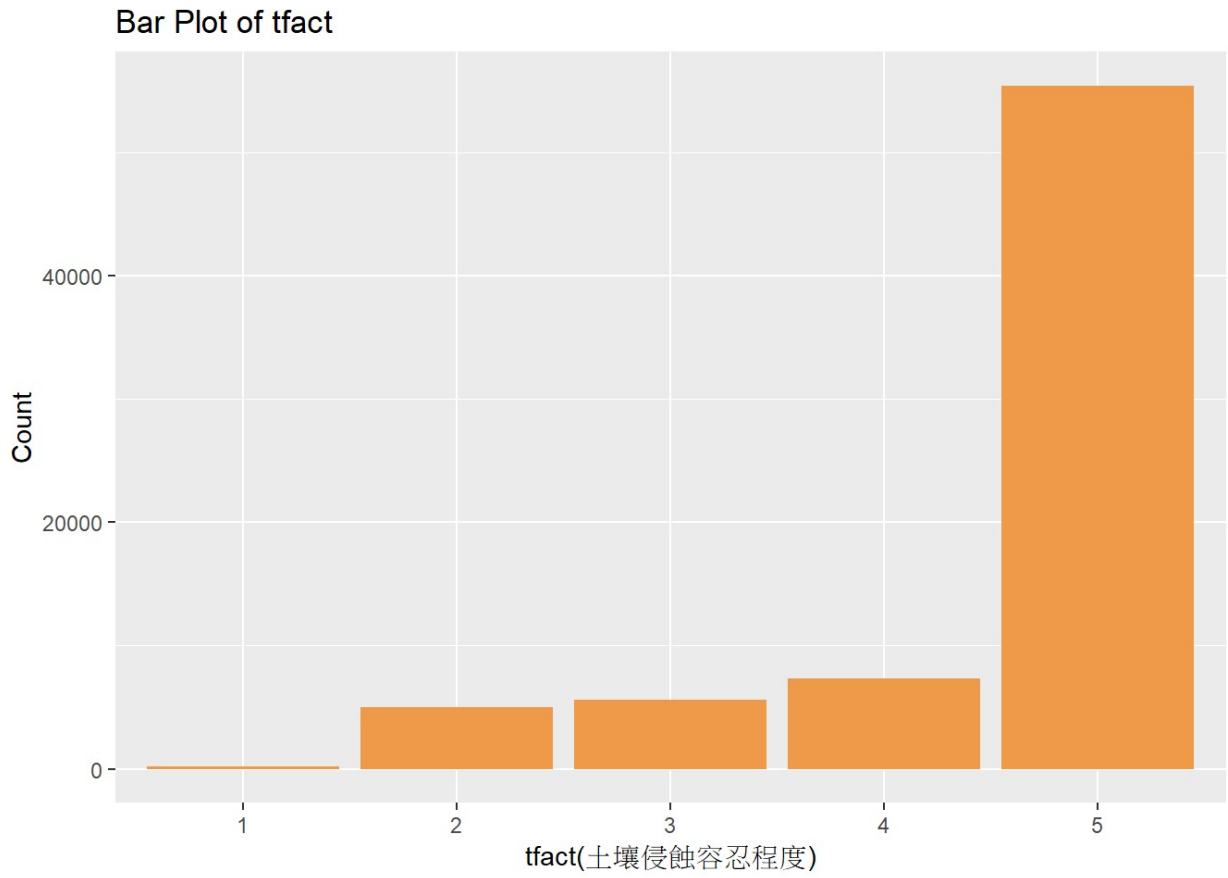
在執行EDA之前，我們先確認目前資料集的變數數量為 22 個變數。我們先將變數細分為類別型變數(categorical variables)以及連續型變數(continuous variables) ，其中有 11 個類別型變數以及 11 個連續型變數。

3.1 類別型變數

下圖為**Bar Plot of soil_erosion**，我們反應變數的長條圖，橫軸為 `soil_erosion` ，縱軸為 `Count` 。可以看出 `soil_erosion`(土壤侵蝕程度) 變數以 `Minor` 居多，代表土壤受到侵蝕程度大部分是輕微的。

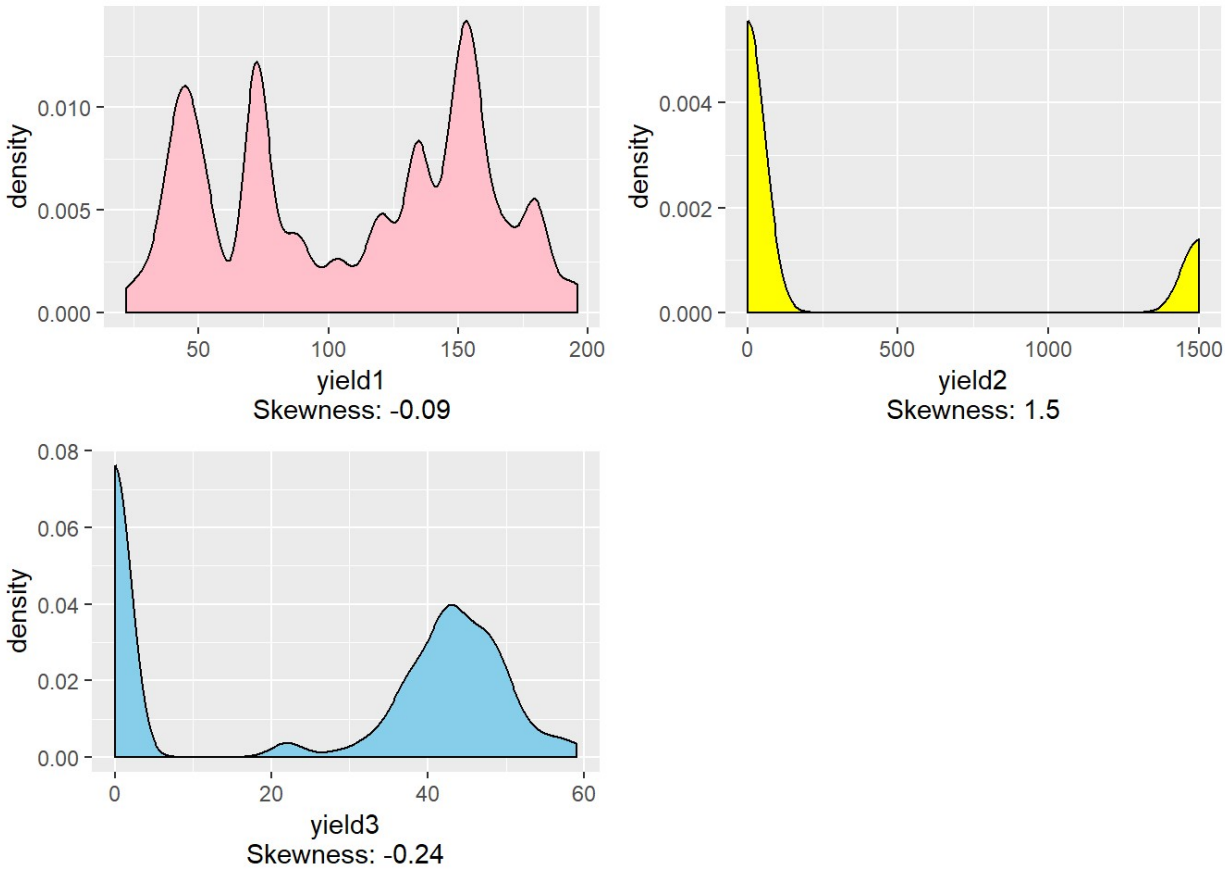


下圖為**Bar Plot of tfact**，橫軸為 `tfact` ，縱軸為 `Count` 。可以看出 `tfact(土壤侵蝕容忍程度)` 變數的值主要集中在5，代表該資料大部分的土壤，受侵蝕容忍程度是高的，也與我們的反應變數 `soil erosion` 相互對照。

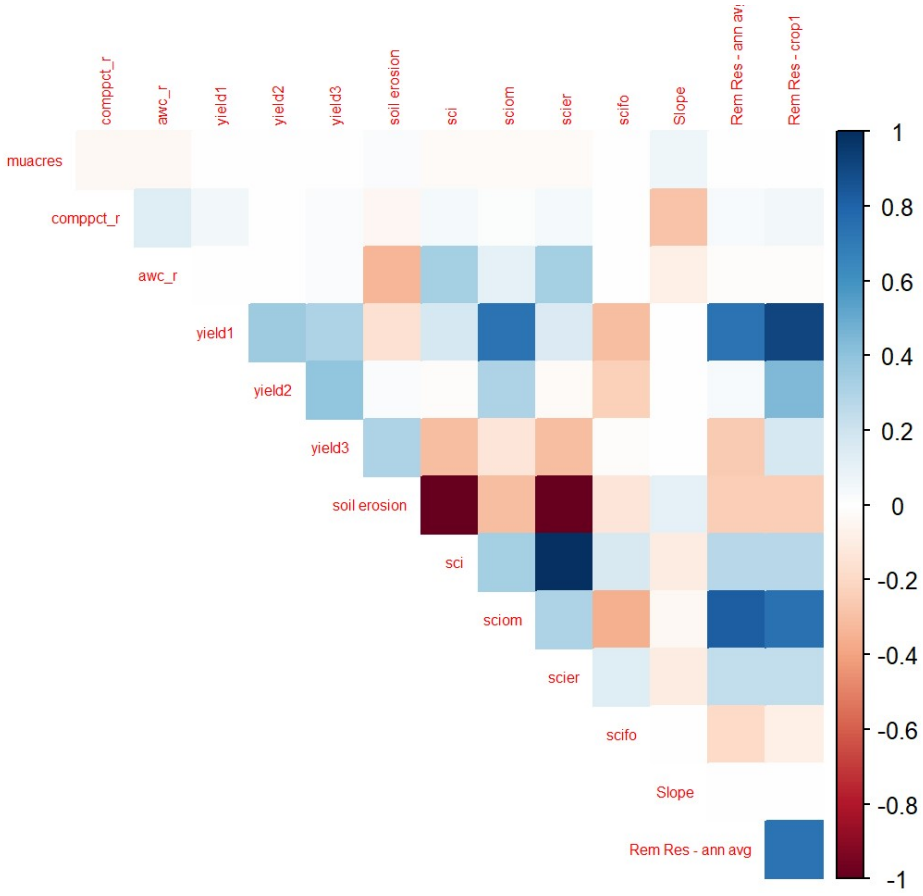


3.2 連續型變數

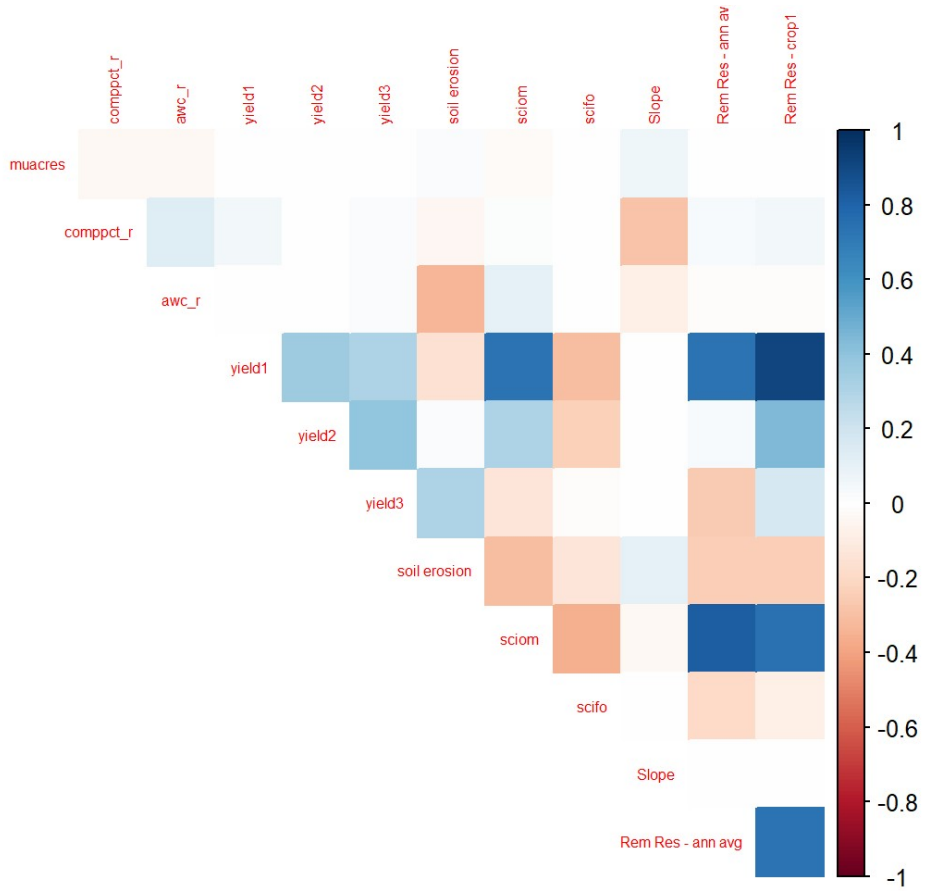
下圖為**Density Plot of yield3**，橫軸為 `yield1`，`yield2`，`yield3` ，縱軸為 `Density` 。此圖為第一年到第三年收穫量的Density plot，可以看出許多土地並無種植作物，因此此類土地汙染也較不嚴重。



下圖為**Correlation Plot 1**，橫軸為變數名稱，色條為相關係數，色條顏色越接近藍色則正關聯性越高，越接近紅色則負關聯性越高。由下圖可得知存在不同變數關聯性過高問題。我們便去除對於資料集而言不需要的變數。



下圖為**Correlation Plot 2**，是我們刪除一些較關聯性過高的變數後所繪製的。橫軸為變數名稱，色條為相關係數，色條顏色越接近藍色則正關聯性越高，越接近紅色則負關聯性越高，可看出已無關聯性過高的問題。

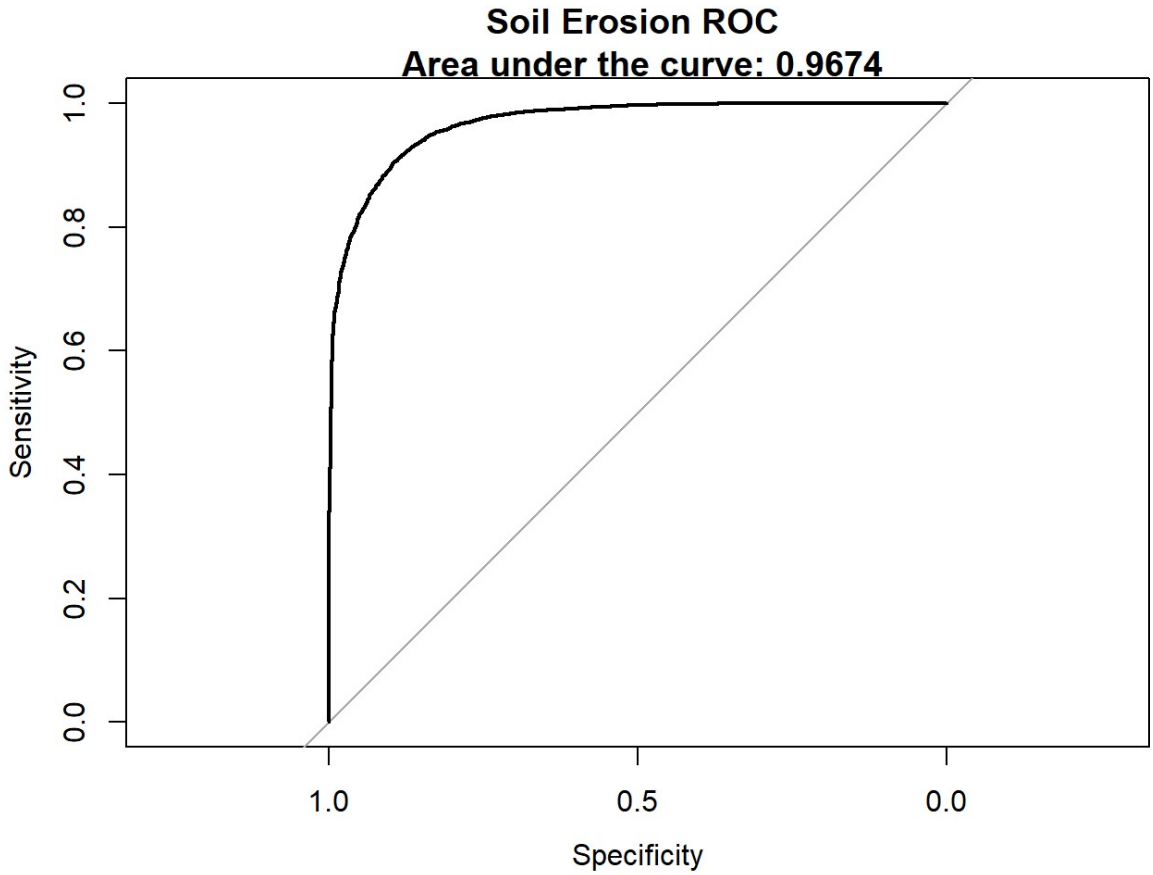


4. 預測分析

資料切割的方式，我們依照 80 % training 以及 20 % testing 的方式進行切割。其中，training data中的30 %會做為CV的validation資料，在執行模型CV的時候會進行設定。為了符合5次Bootstrap cross-testing的標準，我們再將testing data執行5次的 `createDataPartition()` ，隨機抽樣的機率設定為 p = 0.8。在切割完成以後再個別將結果設定為test_1 ~ test_5，以利建模後的cross-testing執行。

4.1 logistic regression without any variable/model selection

使用 `Caret` 套件 `train` 函數，並帶入10 fold的cross validation去建立模型。下圖為預測與實際值畫出的ROC圖。

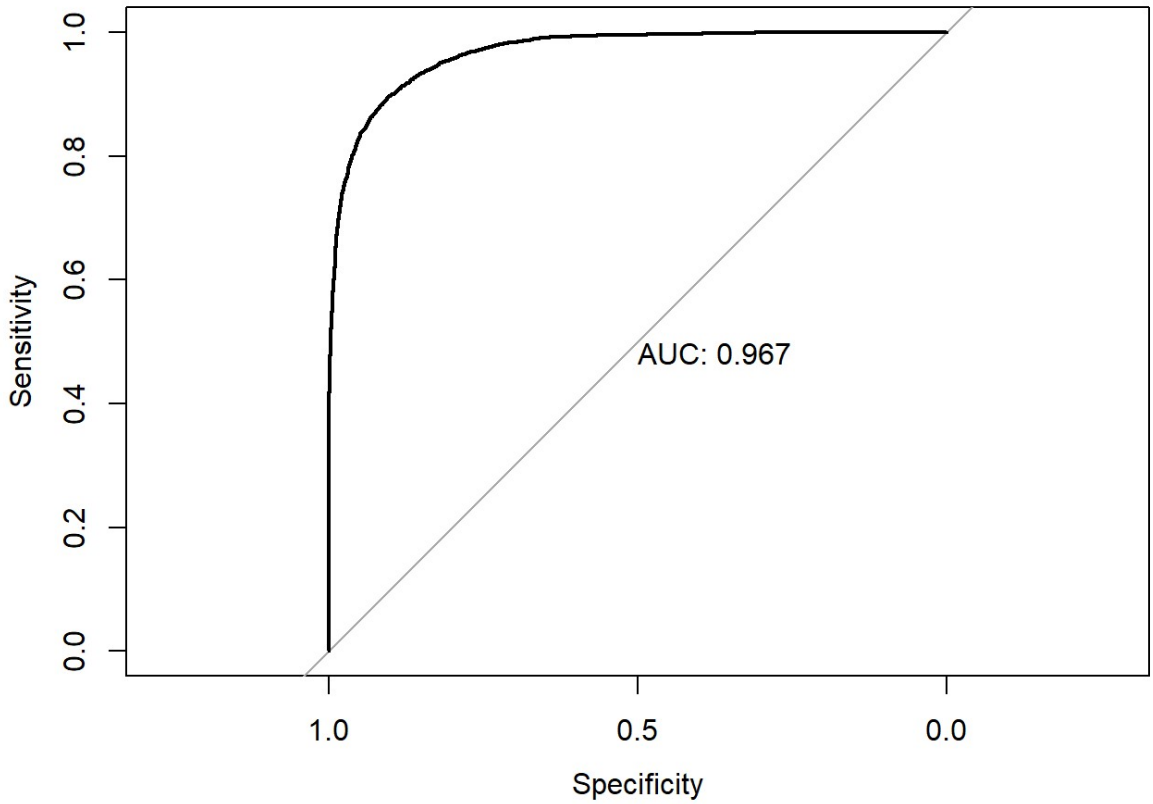


我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為91.4%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.913	0.914	0.914	0.913	0.914	**0.914**

4.2 logistic regression with forward selection

在執行向前逐步迴歸建模前，我們先分別建立包含全部X變數的 `Full model` 和只包含截距項的 `Null model` ，之後再帶入我們的向前逐步迴歸去建立模型。下圖為Test data預測值與實際值的ROC圖。

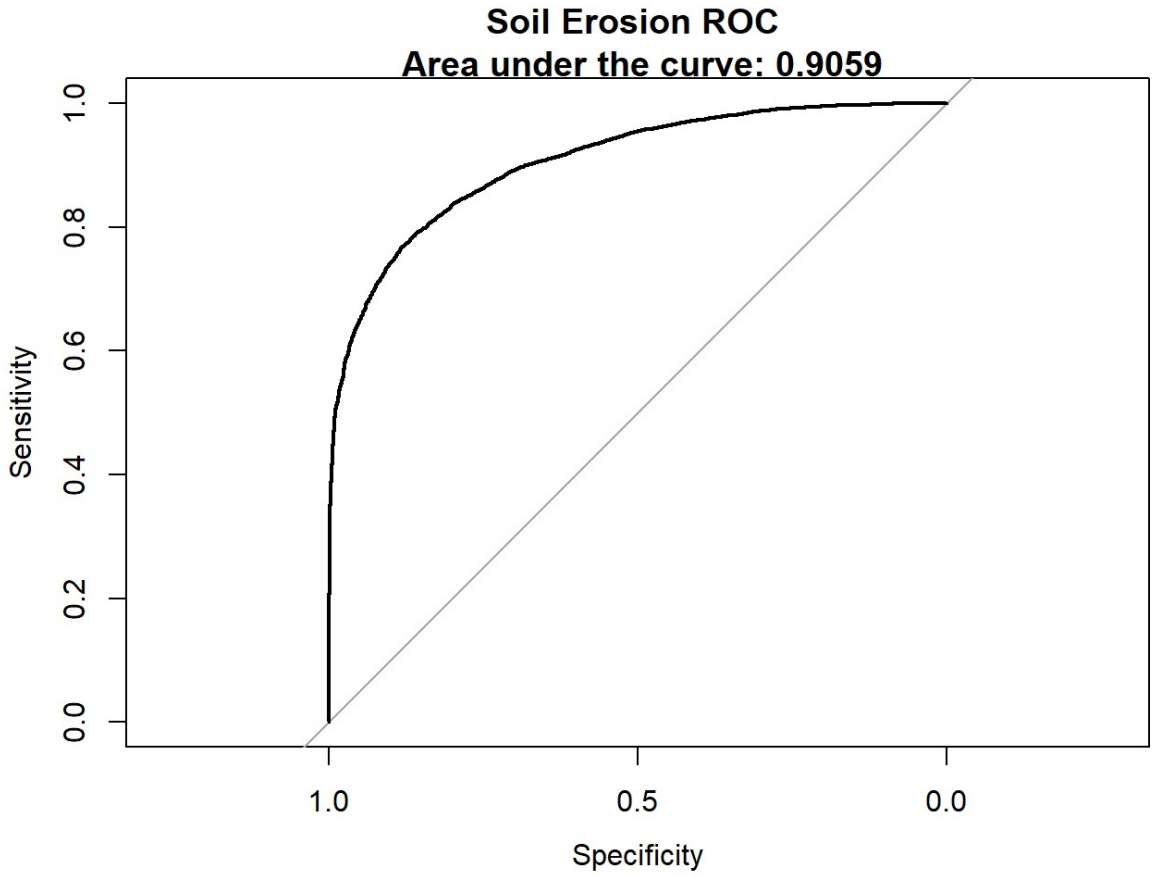


我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為91.2%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.911	0.911	0.912	0.912	0.911	**0.912**

4.3 logistic lasso regression

將lambda值設為從0.0001到0.01去做建立模型。下圖為預測與實際值畫出的ROC圖。

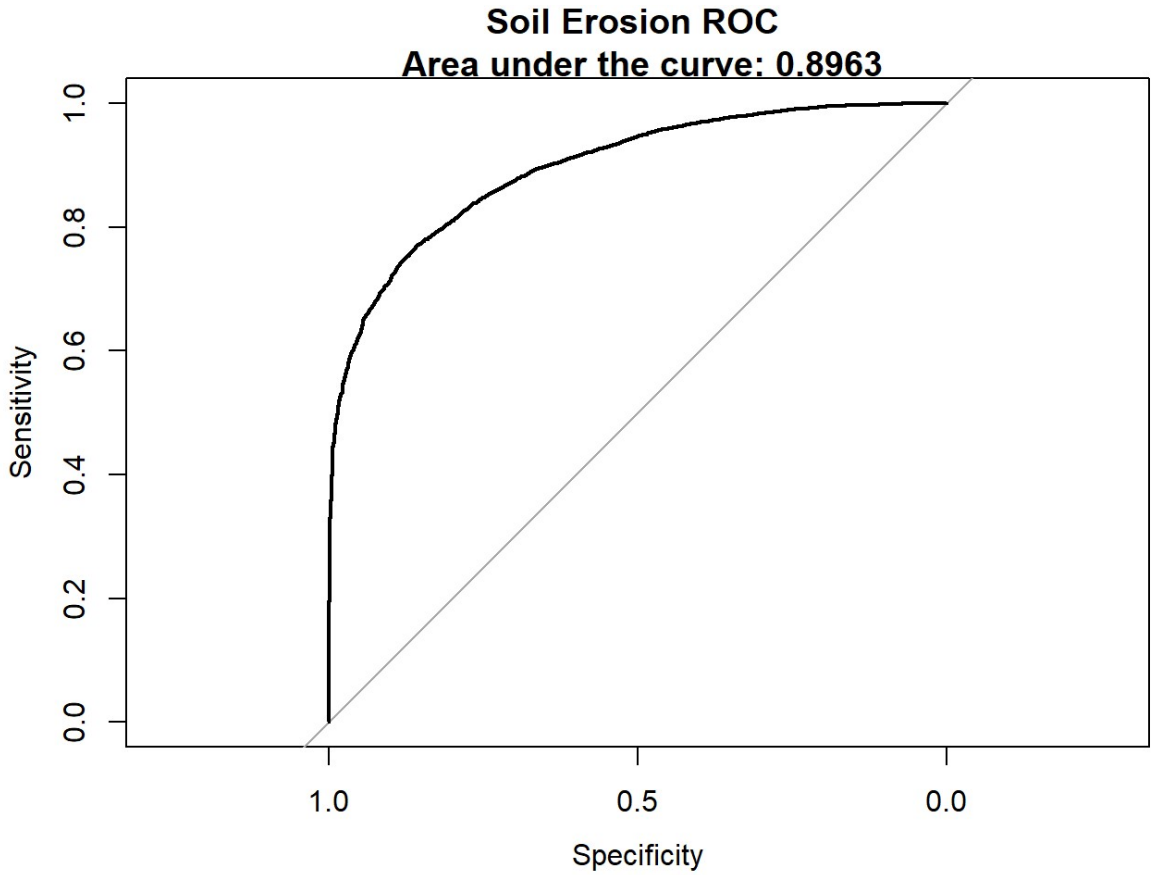


我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為82.9%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.827	0.832	0.829	0.831	0.828	**0.829**

4.4 logistic rigid regression

將lambda值設為從0.0001到0.01去做建立模型。下圖為預測與實際值畫出的ROC圖。



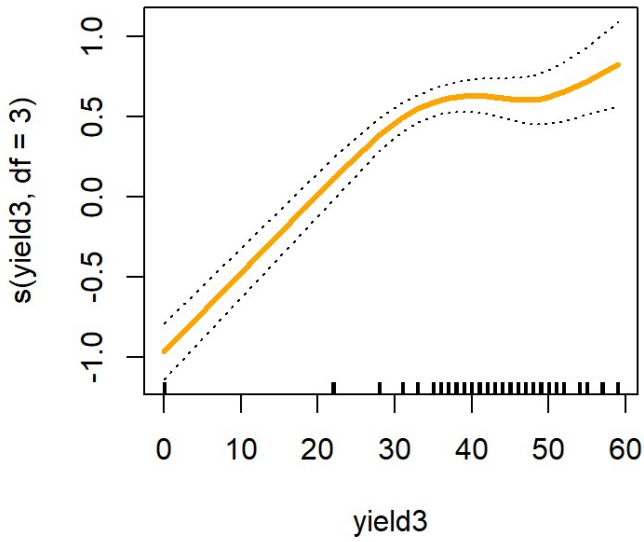
我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為82%。

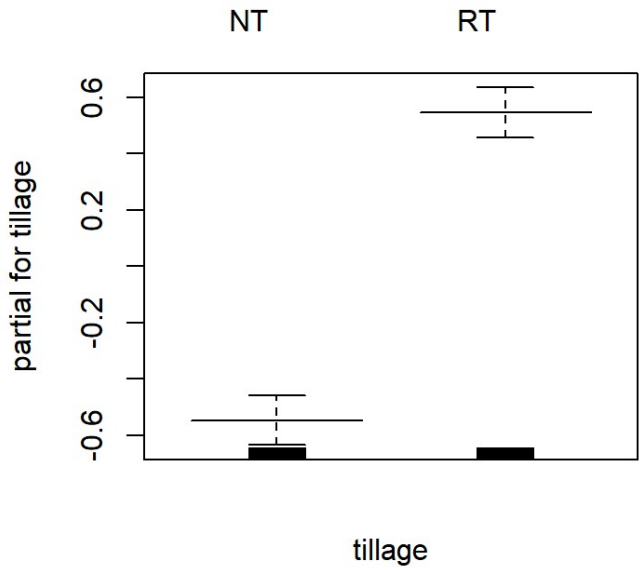
	test1	test2	test3	test4	test5	test_average
Accuracy	0.818	0.823	0.820	0.821	0.820	**0.820**

4.5 Logistic GAM

在進行 `Logistic GAM` 建模以前，我們必須先將training data 以及 testing data的Y變數的二元變數轉換為0跟1個Factor。轉換完成以後，我們再開始進行建模。因為Logistic GAM為Binary的Y變數，因此沒有辦法透過簡易繪圖的方式抓出應該轉為非線性關聯的X變數。因此我們先將沒有刪減變數也沒有設定spline的full model進行Logistic GAM，並利用 `summary(gam.fit)` 的變數顯著性檢定將不顯著的連續型變數進行Spline的設定。我們針對需要調整的兩個變數: `muacres` 、 `yield3` 進行多次的DF設定，並利用 `ANOVA(model1, model2,test = 'Chisq')` 指令執行模型的選擇。我們最後選出的模型是yield3分3段的spline, `muacres` 分10段的spline的Logistic GAM模型。

確立我們的Logistic GAM 模型以後，我們利用 `plot.GAM()` 針對我們有興趣的幾個變數進行簡易的繪圖，以利了解造成嚴重土壤侵蝕的潛藏原因。





左圖: 第3年收穫量 , 右圖: 灌溉系統差異

由第3年收穫量與Logistic GAM的圖可知，若當年沒有種植農作物則土壤侵蝕嚴重的機率將大幅上升。反之，若當年有種植作物則較不容易有嚴重的土壤侵蝕，但農作的多寡卻不太影響嚴重土壤侵蝕發生的機率，或許農作收穫量受到農地大小和農作種植面積的影響。

灌溉系統和土壤侵蝕嚴重的機率相關圖則顯示，有灌溉系統的農地較不容易發生嚴重的圖壤侵蝕，且此效果十分顯著。由此可知，水分應能降低土壤侵蝕的發生機率。

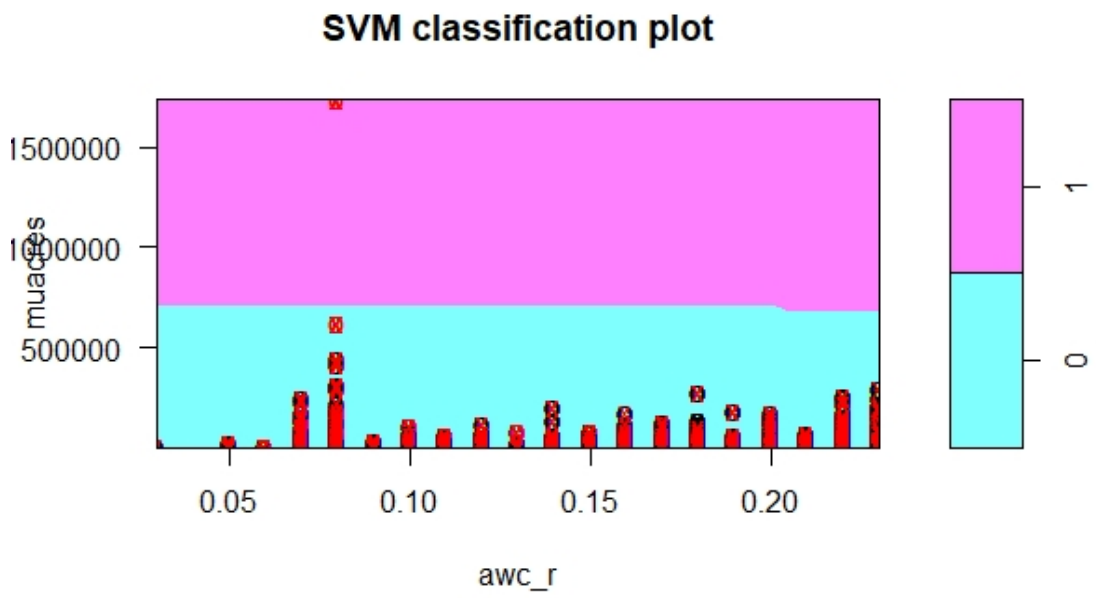
接著我們將5組Bootstrap得出的testing data分別進行預測，並個別計算出其預測的準確率。

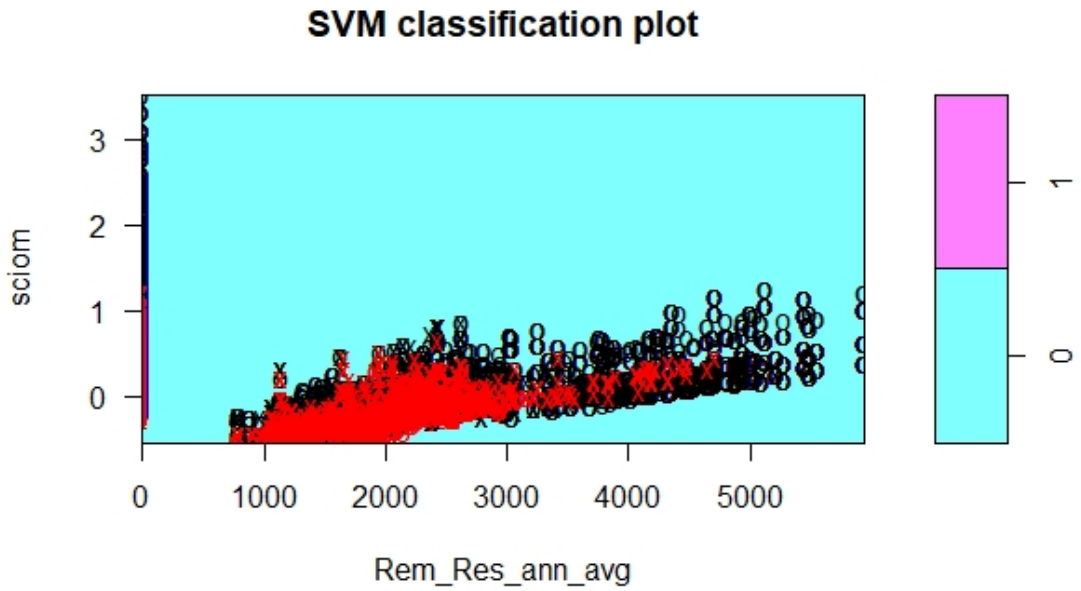
	test1	test2	test3	test4	test5	test_average
Accuracy	0.912	0.914	0.914	0.914	0.914	**0.914**

五次Testing Data預測的平均準確率即為91.35%，預測的準確率十分高。
我們推測準確率這麼高的原因在於Logistic GAM的模型建構缺少交叉驗證的步驟，因此在做testing data的預測時應該多做幾組bootstrap testing data的預測，再來計算平均的預測準確率。

4.6 SVM

SVM模型的前處理相對其他模型來講相對簡單，僅需把預測目標的Y變數轉換為0和1即可。但SVM在電腦運算以及模型參數設定的調整上卻需花費即大量的時間。在進行更為縝密的參數調整以前，我們先對Training Data進行沒有特殊設定的SVM建模。由配適出的結果可知，SVM的種類為 `C-Classification` 、Kernel的種類為 `radial` 、cost的值為1 (預設值)且gamma為 `0.0116` 。在沒有多餘設定的狀況之下，支援向量的數量也高達了12558個向量空間。從SVM model的繪圖結果可發現，原始的變數中僅有部分的變數有明顯的分群效果。我們推測此即為SVM支援向量高達12000多個的原因。





不同變數的分群圖

鑒於tuning的過程耗費太多時間，來不及跑完最後的模型結果。因此我們先以上述的SVM模型進行testing data的預測。其預測的結果如下表:

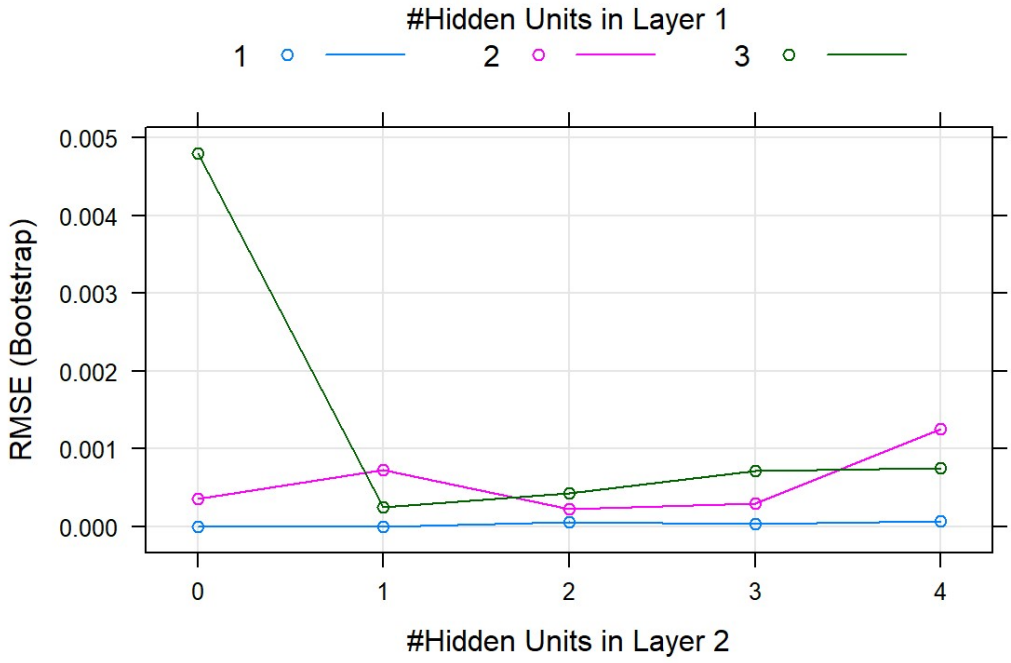
	test1	test2	test3	test4	test5	test_average
Accuracy	0.934	0.936	0.934	0.936	0.935	**0.935**

五次的預測準確率平均高達93.51%。但此SVM模型沒有經過tune model的過程，因此支援向量的數量非常多，也使得模型的複雜程度大幅提升。若完成tune model的步驟，應該會得到更為合理的模型的複雜程度以及預測的準確率。

4.7 Neural Network

在執行類神經網絡的建模以前，我們必須先對原本的training data做轉換才能做模型建模。首先，我們必須將我們的Y變數(soil_erosion) 拆解為兩個Dummy variable。接著，我們再將其他類別型的X變數也拆解為Dummy variables，並移除原始的類別型變數。完成上述的步驟以後，再寫出完整的模型預測方程式。

完成準備步驟以後，我們先用 `caret` 套件協助判定類神經網絡模型的參數設定。如下圖所示，第一隱藏層的取1個節點具有最低的RMSE，第二隱藏層的節點則較不明顯。我們以表格的方式招出最好的組合為: 第1隱藏層1個節點、第2隱藏層1個節點。

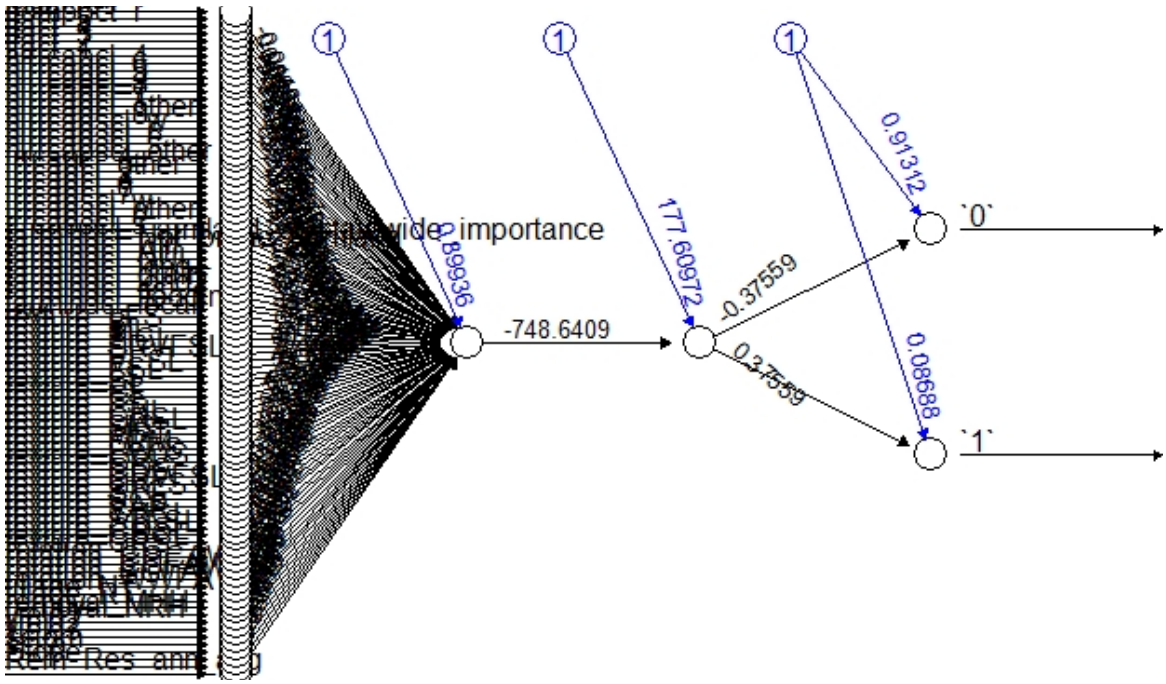


Caret Tune model 結果呈現

layer1	layer2	layer3	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
1	0	0	0.0000001	NaN	0.0000001	0.0000001	NA	0.0000001
1	**1**	**0**	**0.0000001**	**NaN**	**0.0000001**	**0.0000002**	**NA**	**0.0000002**
1	2	0	0.0000600	NaN	0.0000033	0.0001197	NA	0.0000100
1	3	0	0.0000341	NaN	0.0000034	0.0000641	NA	0.0000113
1	4	0	0.0000736	NaN	0.0000038	0.0001364	NA	0.0000088
2	0	0	0.0003613	NaN	0.0000047	0.0012325	NA	0.0000151

layer1	layer2	layer3	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
2	1	0	0.0007345	NaN	0.0000236	0.0017080	NA	0.0000590
2	2	0	0.0002237	NaN	0.0000093	0.0005962	NA	0.0000228
2	3	0	0.0003027	NaN	0.0000119	0.0005638	NA	0.0000212
2	4	0	0.0012557	NaN	0.0000436	0.0027930	NA	0.0000991
3	0	0	0.0048012	NaN	0.0001665	0.0103647	NA	0.0003637
3	1	0	0.0002476	NaN	0.0000091	0.0011183	NA	0.0000300
3	2	0	0.0004281	NaN	0.0000142	0.0006393	NA	0.0000229
3	3	0	0.0007177	NaN	0.0000232	0.0013740	NA	0.0000431
3	4	0	0.0007555	NaN	0.0000345	0.0008496	NA	0.0000353

決定好neural network有幾個隱藏層和幾個節點以後，我們將再對training data進行一次類神經網絡的模型建構，得到最終的neural network模型。最終的模型如下圖所示。之後我們再對5組testing data進行預測，並計算出平均的Accuracy。



Neural Network 結構

在做預測之前，我們必須先將testing data轉成和training data相同的格式。除此之外，neural network的建模過程會將判斷為沒有效果的變數自動刪除，且neural network的預測指令亦不接受testing data保留這些被刪除的變數。因此，我們必須再將testing data的變數做調整。經過繁雜的預測過程以後，我們得到5組的Accuracy 以及平均的Accuracy如下表:

	test1	test2	test3	test4	test5	test_average
Accuracy	0.697	0.697	0.697	0.697	0.697	**0.697**

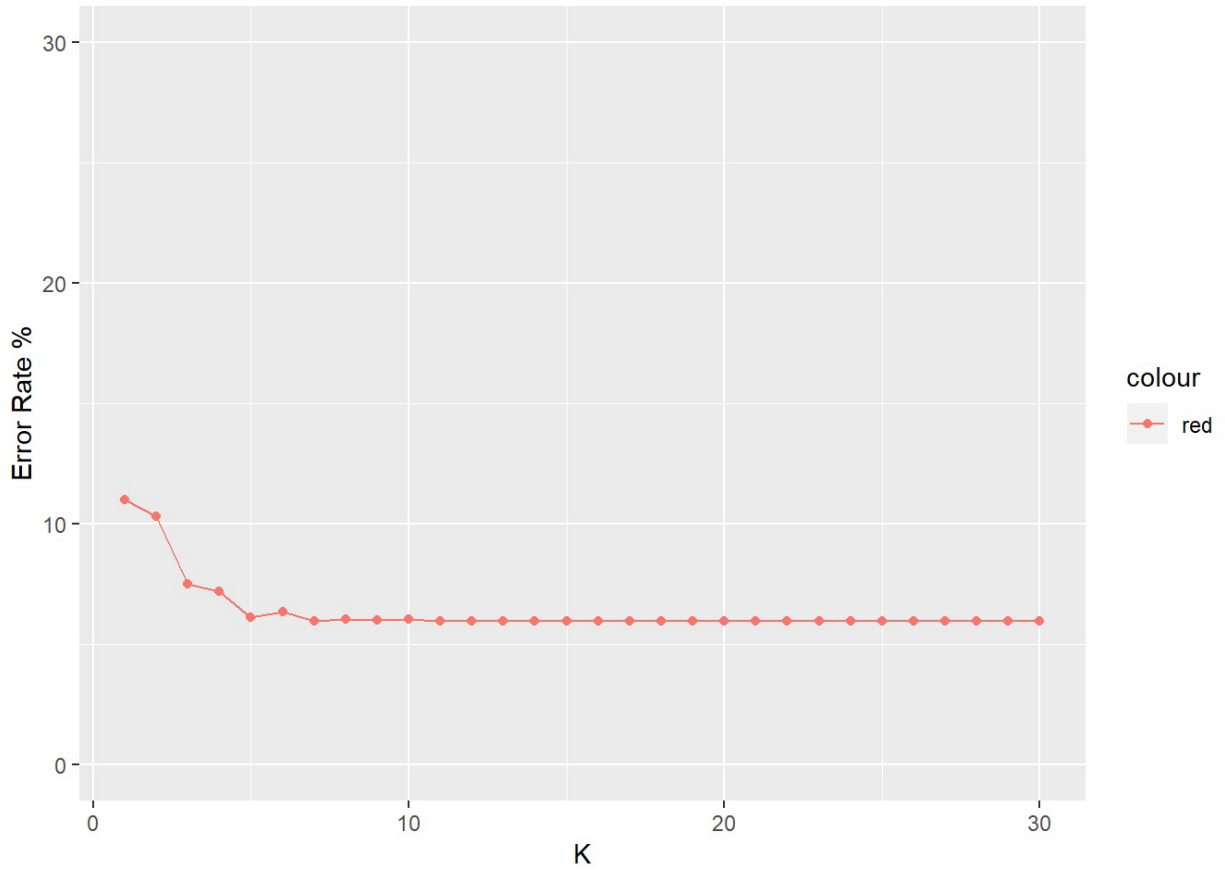
上表5組testing data的 Accuracy皆為0.6965，不太符合理想的狀況。我們推測是因為稍早tune出的模型參數過於簡單，此neural network 僅有2個隱藏層，且各層只有1個節點。實際觀察confusion matrix更可發現預測的結果是將所有觀察值歸類為 `Minor` 。因此5組testing data的預測準確率也完全相同。再深入檢視neural network回傳的預測分群機率可發現，實際為“Minor”的資料分為“Minor”的機率為0.91左右，但實際為“Severe”卻分為“Severe”的機率只有0.46而已。預測機率超過0.5的會被分類為此結果，因此所有的資料筆數最後都被分為“Minor”。也就造成5組testing data預測結果都為“Minor”，Accuracy也都一樣的詭異狀況。

	Minor	Severe
3	0.9131238	0.0868762
11	0.9131238	0.0868762
15	0.9131238	0.0868762
16	**0.5375380**	**0.4624620**
23	0.9131238	0.0868762
37	0.9131238	0.0868762

4.8 KNN

這筆資料集的解釋變數包含 `10` 個類別變數，而其中有不少Nominal variables，然而這些變數無法計算歐幾里得距離矩陣，因此我們先移除這些變數來建立KNN模型。

下圖為**Error Rate of Different K**，在Testing data中，使用不同的 `k` 的所跑出來的Error rate會不同，我們將挑選Error rate最低的k作為參數來建模。

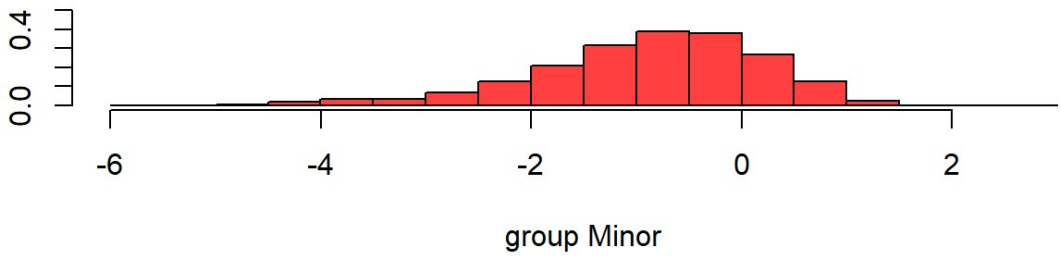


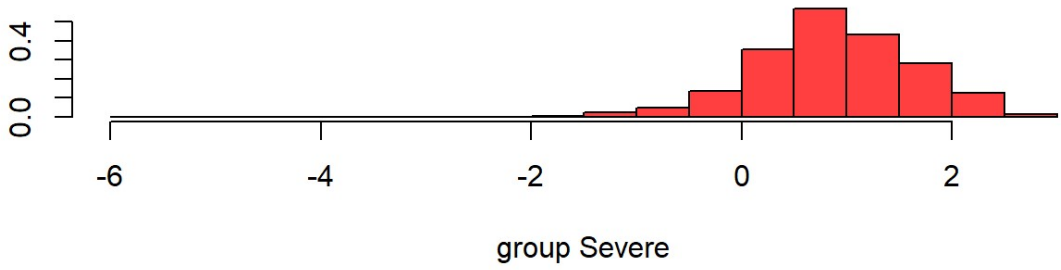
而我們將test1到test5所使用的不同 k 參數，以及代入模型後得出的不同Accuracy於下表呈現，最終得知平均Accuracy為76.6%。

	test1	test2	test3	test4	test5	test_average
k	9	9	13	9	7	
Accuracy	0.765	0.766	0.764	0.767	0.767	**0.766**

4.9 LDA

LDA的假設希望變數為常態，然而這筆資料集的變數並不是常態分佈，因此可能較不適合用LDA來建模。下圖為**Plot of LDA Model**，可以看出分類狀況。



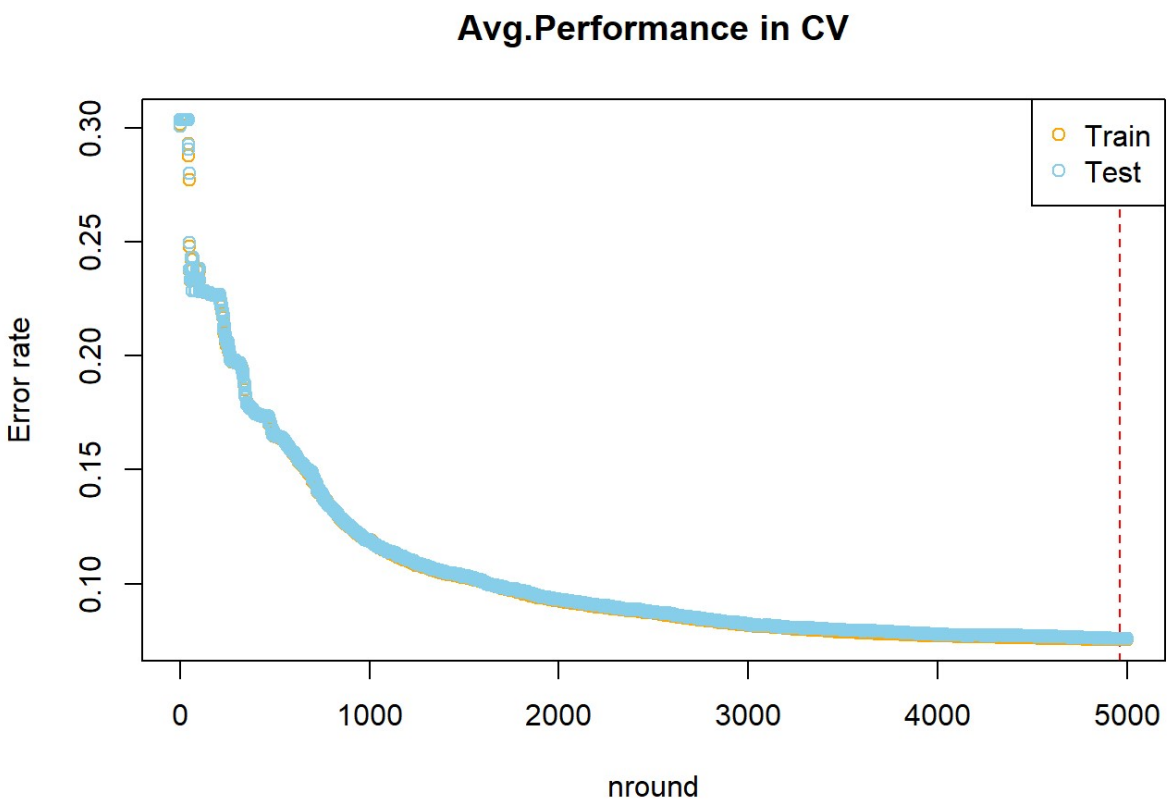


我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為86.1%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.86	0.862	0.862	0.862	0.861	**0.861**

4.10 Boosting

在Boosting建模前，我們首先將類別變數分別都轉成Dummy Variables，並設定 `eta = 0.01`，`colsample_bytree = 0.95`，`subsample = 0.55`，`max_depth = 1`，`alpha = 0.1` ，然後找出最佳的 `nrounds` 參數。下圖為**Avg.Performance in CV**，我們希望能找出Train Error與Test Error最小差距時的 `nrounds` ，並得出 `nrounds = 4959`



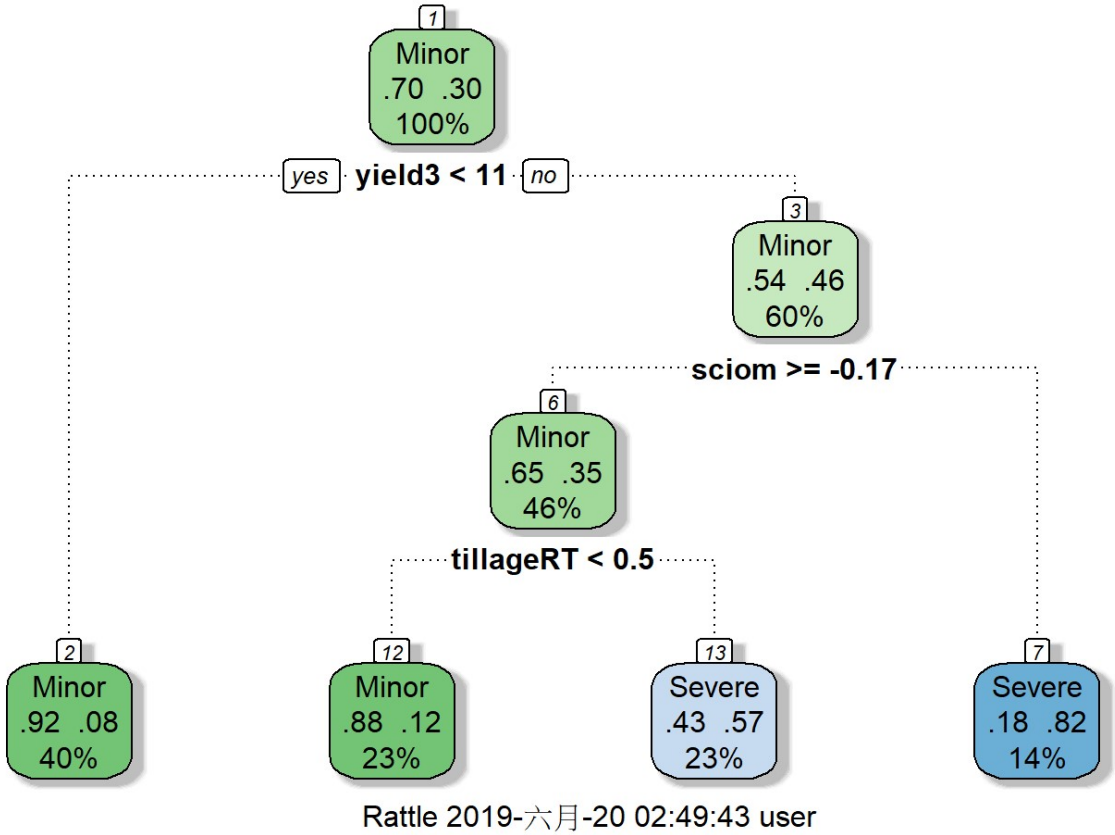
我們接著將test1到test5代入入模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為86.1%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.919	0.923	0.921	0.921	0.921	**0.921**

4.11 Decision Tree

決策樹是將資料進行一層一層的分割，而分割的原則是要得到最大的資訊增益。資訊量通常是以 `熵(Entropy)` 以及 `Gini不純度(Gini Impurity)` 為衡量標準。

下圖為**決策樹模型**，所使用到的變數有 `yield3(第三年收穫量)` `sciom(生物分解指標)` `tillage(灌溉方式)` ，可以看出Minor分類的情形比Severe好。



我們接著將test1到test5代入進模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為81.7%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.815	0.818	0.818	0.816	0.818	**0.817**

4.12 Bagging

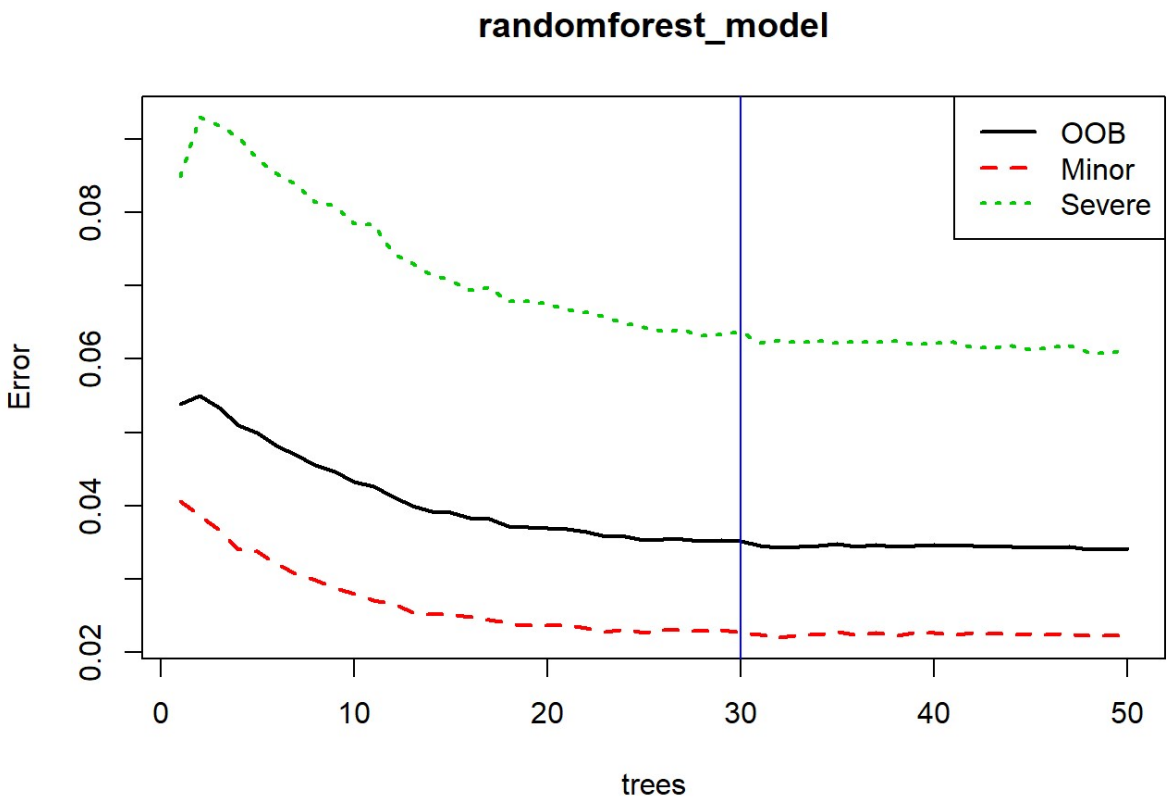
`Bagging` 是 `Bootstrap Aggregating` 的簡稱，透過統計學的 `Bootstrap sampling` 得到不同的訓練資料，然後根據這些訓練資料得到一系列的預測結果，並加以整合與平均，可以有效降低單一模型的變異程度。另外，一般在做 train 時會把手上的 label data 切成 training set 跟 validation set，但使用 bagging 的時候，不用如此，同樣可以擁有 validation 的效果，叫做 `Out-of-bag validation`。

最後以 `bootstrap replications = 27` ，所得到的 `Out-of-bag estimate of misclassification error` 最小，因此將模型參數 `nbagg` 設為27，再將test1到test5代入進模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為90.8%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.908	0.908	0.909	0.910	0.907	**0.908**

4.13 Random Forest

隨機森林是結合多棵決策樹，並加入隨機分配的訓練資料，以大幅增進最終的預測結果，然而決策樹的數量是透過下圖 最小錯誤率 來進行選擇，並可以由 `Minor` `Servere` `OOB` 三條線，看出在決策樹數量皆為30時，所得到的錯誤率是最低的，因此最後選擇 `tree=30` 。



於是，將模型參數 `ntree` 設為30，接著將test1到test5代入進模型中，分別得出不同的Accuracy，於下表呈現，最終得知平均Accuracy為96.5%。

	test1	test2	test3	test4	test5	test_average
Accuracy	0.966	0.965	0.966	0.966	0.966	**0.965**

---

5. 結論

下表為所有模型的Accuracy比較表格。可以看出Random Forest Model預測效果最好，而Neural Network表現最差。可以得知我們本次資料適合Random Forest Model。然而不同資料可能會適合不同的方法，不能以此作為分析方法的優劣評斷標標準。

	test_average	Ranking
Logistic_full	0.914	4
Logistic_forward	0.912	6
Logistic_lasso	0.829	9

	test_average	Ranking
Logistic_rigid	0.820	10
Logistic_GAM	0.914	5
SVM	0.935	2
Neural_Network	**0.697**	**13**
KNN	0.766	12
LDA	0.861	8
Boosting	0.921	3
Decision_Tree	0.817	11
Bagging	0.908	7
Random_Forest	**0.965**	**1**