Pump it Up: Data Mining the Water Table

隊伍名稱：NTU_r05942101_queenfat

隊長：陳泓弦　學號：R05942101
組員：林建宏　學號：R05942052
黃梓維　學號：R05942058
潘維方　學號：R05942121

Work division

陳泓弦：feature 處理、報告撰寫
黃梓維：feature 處理、報告撰寫
林建宏：建立 DNN 模型、Random forest 模型、報告撰寫
潘維方：建立 XGBoost 模型、報告撰寫

Purpose

利用搜集的 40 個 feature，來對抽水站狀況做預測，在這個題目，我們用了三個模型，分別是 Xgboost、Random forest 和 DNN 來預測水井的狀態，資料的 Preprocessing 有各自的處理方式。

Preprocessing/Feature Engineering

(1) XGBoost

發現官網給的 feature 當中有一些資料是很奇怪的，例如 construction_year 有很多筆資料是零，這樣的資料肯定是有問題的，所以第一步我們必須修正這些資料，想透過一些插值的方法，來填補這些奇怪的資料，我們想到的做法是先把最小的 construction_year (1960)設定為零，1961 就是一，以此類推來標準化數值，然後找出 construction_year 的中位數，把缺失和奇怪的數據都用中位數來補上。其他比較特別的 feature 大概也是用上述的方法來填補，透過填上中位數的方法，來讓這些資料可以被使用。

再來探討的是 feature 的含意，對於我們的 training 到底有什麼影響，依據下列幾點來作為刪除不必要 feature 的標準，首先刪除一些意義不明，不太理解是什麼意思的 feature，例如 num_private，資料幾乎都是零，只有零星幾個非零的數字，這樣加入 model 可能會帶來一些不必要的影響，所以決定忽略它，再來是太多特例難以分類的 feature，例如 wpt_name，在整個 colume 當中，看了前十筆資料，大概就有十筆不一樣的種類，分類太多，我們覺得對於 model 也不會有太多的幫助，這樣看來就好像每個 data 都有各自的 wpt_name，所以最後就決定去掉了，接下來是一群 feature 有相似的意涵，我們覺得只要留下能夠代表的就好，例如有一組 feature，例如 extraction_type_class, extraction_type_group, extraction_type 這組，其實三個 feature 意思相近，所以最後只要留下一個就好。以上是我們這組抽取 feature 的基本精神，經過刪刪減減，從 40 個 feature 當中，留下了 26 個 feature 做 training，另外這些 feature 中如果是數字之外的分類我們並沒有做 multi-label 的分類方法，純粹對於不同分類用 1~X 的數字來做分類，雖然說這樣的分類方法以距離的觀點來看會比較有問題，但是因為在做 Random forest 時就有發現，如果做 multi-label 的話會容易造成參數過多的狀況，所以我們就想說先用數字代替就好，有需要再來做

multi-label 的分類法，而做出來的結果，比沒做 feature selection 的 model 好了許多。

(2) Random forest

我們認為水井的狀態和記錄的時間有相關性，因此我們將 data_recorded 的 year、month 分別抽取出來成為兩個新的 feature，year_recorded 和 month_recorded，有 Null 話填上平均值。permit 和 public_meeting 這兩個欄位我們推測和水井的建造有關，雖然不太理解他們確切代表的意思，我們還是丟進去當 feature，有 Null 的話填入 False。另外我們把 13 個我們認為對訓練沒有幫助的欄位丟掉，這些欄位有些是和預測水井狀態完全沒有關係；有些是 Null 或是 0 這些缺失奇怪的數據太多，沒辦法拿來當做有用的 feature；有些是類別太多，容易讓模型壞掉；有些 feature 彼此有相同的意義，所以我們只留下一個當做 feature，下面列出了 13 個欄位以及我們沒有拿來當做 feature 的原因：

id：和水井狀態無關，對預測沒有幫助
wpt_name：分成 37400 類，太多了
subvillage：分成 19287 類，太多了
amount_tsh：幾乎是 Null
num_private：幾乎是 0
region：和 region_code 一樣
quantity：和 quality_group 一樣
quality_group：和 quality 一樣
source_type：和 source 一樣
payment：和 payment_type 一樣
waterpoint_type_group：和 waterpoint_type 一樣
extraction_type_group：和 extraction_type 一樣
scheme_name：大概一半是 null

選好模型的 feature 後，接著要處理 missing data 的問題，我們發現除了有些資料是 Null 外，還有很多為 0 的資料，而這些為 0 的值其實是不合理的，所以 Null 和 0 都是我們要修正的資料。修正的方法大致上都是填入這個 feature 所有資料的平均值，比較特別的是，像 longitude, latitude, gps_height, population 這四個 feature，其實是有地域關聯性，因此我們先將資料根據 discrete_code 和 basin 分群後計算每一群的平均值，用這個值來填入 Null，會比對全部取平均值還要更精確。另外將 population 的值取 log。

因為有些 feature 是分類別，我們必須將這些類別轉換成向量，最簡單的做法是做 one-hot encoding，但這樣做會讓 feature space 的維度太高，在這麼大的維度下 training 是很沒有效率的。因此我們對 gps_height、latitude 和 longitude

這三個 feature 做 Linear discriminant analysis (LDA)降維，選這三個 feature 的原因除了是連續的數據外，我們想某地區的水井可能都是抽同一條河流，當某個水井是壞的，那麼很有可能他附近的水井也是壞的，那麼這些 data 應該可以分成同一類。做 LDA 而不是 PCA 的原因是我們希望降維後能將 data 分群，雖然 PCA 降維能讓誤差最小，但没有將 data 分開的特性。在做 LDA 之前，我們會先對 feature 做 Normalization，將數值減去平均值後除以標準差。

另外我們會看每個 feature，將出現次數低於 cut-off 的類別都歸類到"other"這個新類別，雖然這個做法理論上會表現比較差，但這差異我們測試後認為是無傷大雅，可以忽略的，而且將這些出現次數低的類別歸在同一類還有一個好處，那就是可以減輕 overfitting 的問題。

Model Description

(1) Model1 - XGBosst

基於 Gradient Boosting Machine，基本思想是把很多分類準確率較低的樹，組合成一個模型，成為一個準確率較高的模型，而這個模型會不斷得迭代，每次迭代就生成一顆新的樹，XGBosst 就是改良 GBM 而來，其特點是能夠自動利用 CPU 的多線程進行並行的運算，同時在目標函數中加上了正規化並且使用二階導數，不僅提高效率同時也增加準確率。

我們的模型是使用 R 語言為基礎的 XGBosst 版本，首先使用 xgb.cv 來調整將要使用在 xgboost 當中的參數，使用 softmax 作為分類依據，設定 early_stop 並且最多迭代 500 次，然後把原本的 data 隨機切成四等分，樹的最大深度設為 14，當找到這樣一組參數，把它餵進 xgboost，以這樣的參數設定來當作一次 predict，總共做了 11 組，並且設定 seed 以用來 reproduce，最後把這 11 組預測值，對每筆資料都去看出現最多次預測的是哪一個分類，就把那筆資料分到該類別去，作為我們最後的預測答案。

(2) Model2 – Random forest

Random forest，顧名思義，是用隨機的方式建立一個森林，森林是由很多的決策樹(decision tree)組成，因為單一棵決策樹會有 overfitting 的問題，而 random forest 其實就是生很多棵樹然後做 bagging，隨機森林的每一棵決策樹之間是沒有關聯的。隨機森林在分類問題中，當有一個新的輸入樣本進入的時候，就讓森林中的每一棵決策樹分別進行一下判斷，看看這個樣本應該屬於哪一類，然後看看哪一類被選擇最多，就預測這個樣本為那一類。

因為這次的題目是分類問題，我們使用 scikit-learn 的 RandomForestClassifier，首先先簡單介紹一些會使用到的參數，n_estimators 是

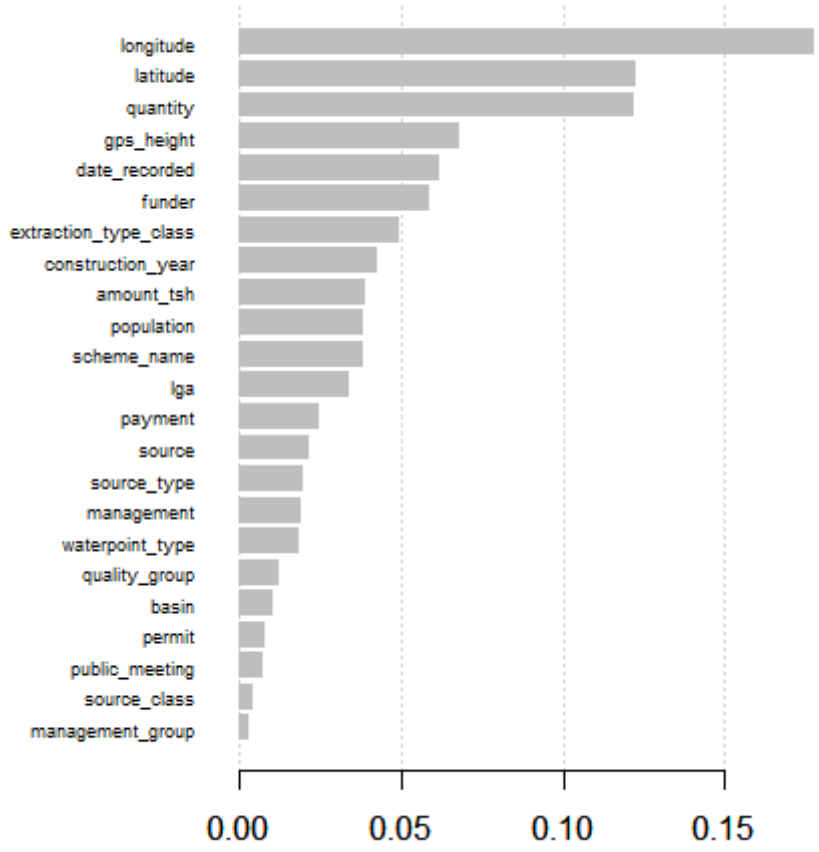
決策樹的個數，太小容易 underfitting，太大又容易 overfitting，我們最後設 500。oob_score = True，使用 out-of-bag 樣本去評估模型的好壞，若使用 out-of-bag 不需要切 training set 和 validation set，準確度就能蠻準確的反應在 testing set 上的結果。max_features = 'auto'表示在每個分子樹時最多考慮$\sqrt{N}$個 feature，N 是總 feature 數。

(3) DNN

Deep neural network，也就是課堂上教的最基本模型，利用 keras 中 model.add(Dense(….)))來架構出 DNN 的模型，藉由其中 Dropout、regularizer、activation、layer 層數跟大小等來建立 DNN 的架構，目的是架構出瘦長型的 DNN，並且有高的準確率和訓練速度。

Experiments and Discussion

(1) XGBosst



一開始我們是在 python 上實作，但發現效果不錯卻還是差一點過 strong

baseline，後來發現有同學指出在 R 語言上使用同樣的模型訓練效果會提升，於是最終我們這組使用 R 訓練的 model 再丟回給 python 下去左 predict。上圖是用 xgboost 使用 R 跑出來的特徵相關性，可以看到影響最大的是經緯度的特徵，也就是地理位置對水井的影響最大，其中的關聯會在下一章節做詳細的討論。而在訓練過程中我主要調整的參數是 nrounds，去決定疊代需要的次數，經過隨機的組合竟然意外的效果就不差了，因為有過 strong baseline 所以後來就沒有針對每項參數去做實驗觀察對結果的好壞優劣了。

(2) Random forest

Random forest 在訓練時，有兩個重要的參數要設：min_samples_split 和 n_estimators，這兩個參數設的好不好對訓練的結果影響重大。但每個案例有不同適用的值，那我們只能一個一個試，上傳後根據分數來選最好的模型。我們試了 min_samples_split = 4、6、8，n_estimators = 500、700、1000，最後的結果，min_samples_split 設 6，n_estimators 設 1000 的表現最好，可以達到 0.81 左右。

此外我們還有嘗試一些選 feature 的方法，但結果並沒有變好。對每個 feature，我們在歸類 other 類別時，我們還有做根據 functional, non-functional, or in-need of repair 來做分類，但效果沒有比較好，我們也嘗試了不同的 cut-off value，但也沒有進步，幾乎都是維持在 0.821 左右。

Random forest 參數調整比 XGBoost 簡單很多，我們只需要選樹的數目和每個葉子的最少樣本數，random forest 給人的感覺像是一個黑盒子，我們沒辦法去控制模型內部的運行，只能在不同的少量參數做調整，然後上傳看結果如何，這可能是 Random forest 的弱點，太多內部 random 產生的結果，無法讓人簡單的就去做一些限制或是修改，讓模型效果較好；不過這也就是他的好處，讓人簡單可以製作出模型，且速度不慢，而且結果也不差。

(3) DNN

DNN 部份就用前兩種 preprocessing 的方法拿來直接訓練，測試過程中有一直修改 layer 層數跟大小跟其他變數、加入 regularizer 等，但是發現不論怎麼改都無法像 Random forest 或是 XGBosst 那樣準確率那麼高，甚至在訓練過程就已經發現即使在準確率只有 0.6 左右就會有準確率會有無法衝高的狀況，大概最高只能達到 0.76 左右，而且速度方面明顯會比其他兩種方法慢了許多，每次訓練完全動輒 3 小時以上，所以在多次試驗過後決定放棄 DNN 的方法，專心於修改其他兩種方法就好。

Result

模型	分數
XGBoost	0.82
Random forest	0.81
DNN	0.75

使用 **XGBoost(最好的成績)**

BEST SCORE	CURRENT RANK	# COMPETITORS	SUBS. TODAY
0.8250	39	3467	0 / 3

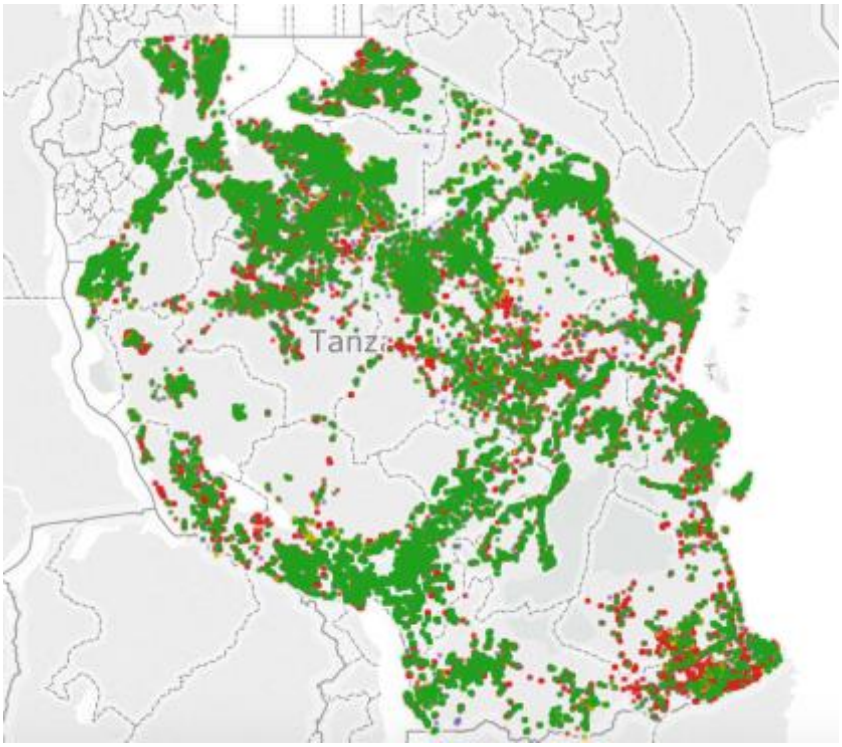
使用 **Random forest**

0.8145	jimmytkps	June 11, 2017, 9:40 a.m.

使用 **DNN**

0.7521	jimmytkps	May 22, 2017, 3:56 p.m.

我們最後結果如上，最後我們是利用 XGBoost 的方法來達到這個分數，雖然 XGBoost 每次結果會有部分不同，但是做了多次取眾數之後，最後可以大概達到 0.825 左右。並且其中可以發現在權重部分 longitude 和 latitude 部份是相對其他 feature 明顯大了許多，跟我們一開始預期來說差了許多，但是經過同學的簡報跟我們去找分佈圖確實發現這是有地域性的。



綠點：functional

紅點：non functional

黃點：functional needs repairs

所以可以確實推測，經緯度跟地域性會影響很大，以圖中右下和中間散佈比較分散的地方來說就有比較多是 non functional，在其他周圍部分就比較高比例是 functional，推論畢竟因為抽水機是藉由地下水源來抽水，所當地下水源沒水時，即使抽水機本身是好的，也會因為無法供給出水而被被認定為 non functional，所以在這樣的考量下就可以明顯感受到 longitude 和 latitude 是非常重要的 feature，也十分符合我們先前使用 R 看到的特徵影響程度。