資料採掘期中報告

法科所 二年級 105652006 王選仲

資科所 一年級 106753004 蘇裕勝

# Titanic

此題得分0.84688[[1]](#footnote-1)。以下分三個部分說明分析的過程與嘗試：Feature Selection、Model Selection、Cross-Validation

## Feature Selection

總共以下features： Pclass, Sex, Age, Sibsp, Parch, Ticket, Fare, Cabin, Embarked，以下分別說明如何處理。

### 處理Null

檢視了資料過後，發現Age, Cabin, Embarked有null。Age是連續變數，嘗試了平均數、中位數以及用其分佈創造出的隨機list塞入null值當中，發現用中位數塞入null值可以得到最好的預測效果。

另外，Cabin原本是字串，但處理後會變成類別變數，因此直接將null值分為一類即可。Embarked則參考大多數人的做法，把最多的一類，也就是S，塞入null值。

### Discretization

這部分要討論如何切分連續變數，而資料中有兩個欄位是連續變數：Fare、Age，這部分我嘗試了三種切分方式。第一種，直接用直覺去切分，但後來發現這種做法很不科學，沒有理論依據可循。第二種方式則直接將每一個群切成相同人數即可，但並沒有因此得到較好的效果。接著，第三中方式，我希望可以嘗試看看透過entropy的大小去切分族群，但是這種作法在sklearn中並沒有被實作，所以只好找了一些網路上別人寫好的程式拼拼湊湊來實作，這樣理論上來說應該會得到比較好的效果，但從結果面來看，也沒有得到比較好的效果。總體而言，我最後還是選擇了，用直覺切分並慢慢調整的做法，以達到最佳的預測效果。

### 其他變數的創造

#### Name => Name\_Length, Name\_HasSpeacialChar, Title

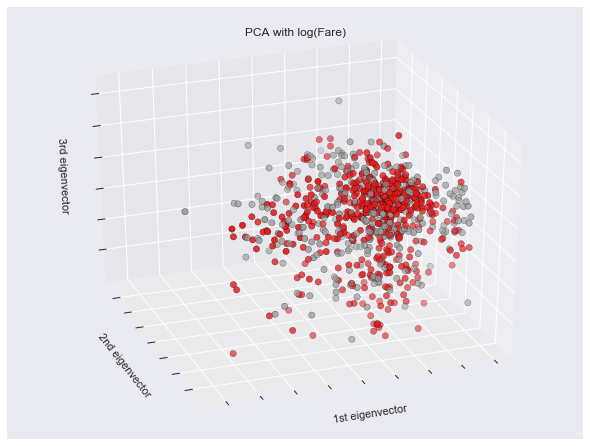
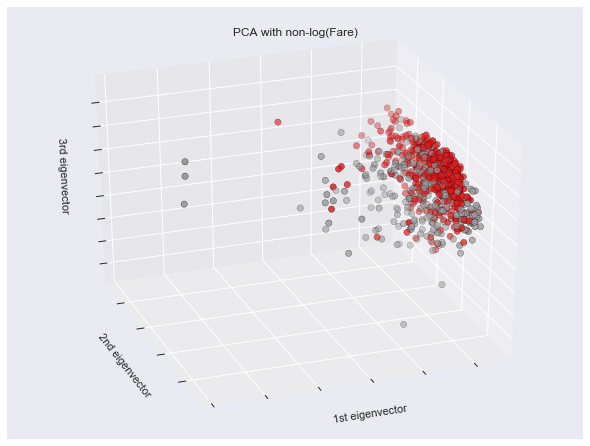
普遍上網路的做法，都直接取Name的「長度」以及「稱謂」，但除此之外，我還發現了一個重要的Feature，可以取標點符號的個數（另外從﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽同人入標點符號不包括逗號、與英文句號），雖然理論上很難理解為什麼這樣的feature可以增加預測效果，但實際上卻幫助我從80%的accuracy提升到81%。

#### SibSp, Parch => Faily\_Size, Is\_Alone

這兩個變數實質上指得是「兄弟姐妹及配偶人數」以及「父母及子女人數」，而透過這些數字則可以直觀想像出「家庭總人數」以及「是否孤身一人」兩個變數。

但經過測試發現「是否孤身一人」在整個預測的過程中，扮演的角色並不重要，另外從其對於是否會存活的影響上來看，「孤身一人的」存活率是55%，但是「非孤身一人的存活率」僅有34%，與預期有極大的落差。

#### Fare

雖然已經經過分類得到Fare的類別變數，但是Fare本身還是Feature。經過PCA發現，Fare本身的range差距過大，在預測時會遇到瓶頸，因此將其取以2為底的log使其平滑，這讓Accuracy從81%提升到82%。

#### Age, Parch, Title => Mother

謠傳說母性堅強，因此將Age>18, Parch>0, Title==’Miss’的歸一類為母親。但是謠傳終究只是謠傳，經實際測試，這對於預測的結果只有負向效果。

### 每一個Feature的重要性Macintosh HD:Users:jeremy:Desktop:螢幕快照 2017-11-14 下午8.48.05.png

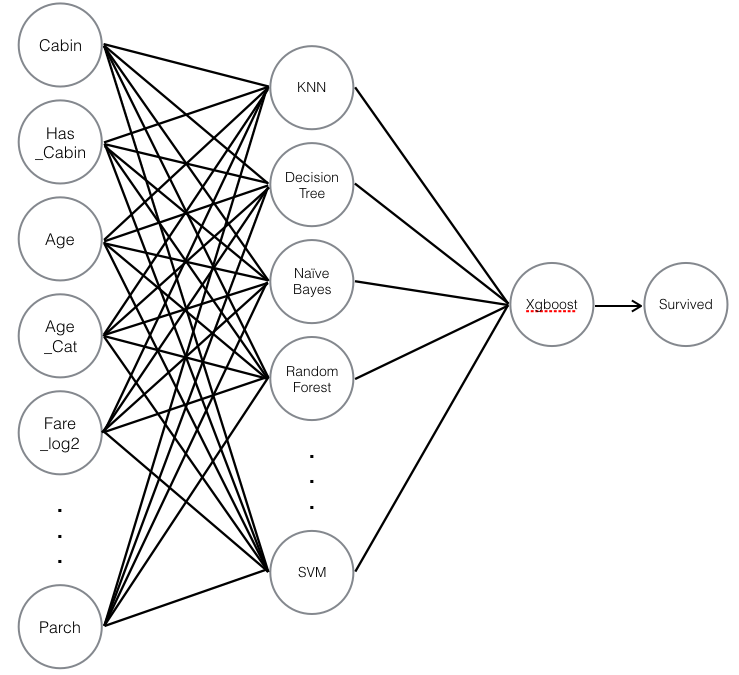
### 最終Feature的選擇

由於Feature很多，選擇上對最終預測結果影響也很難預測，加上每次重新跑一次程式，需耗費一定的時間，因此寫了一段自動化程式，根據上圖去選擇性的丟掉不重要的參數進行測試。最終，我謹將以下參數放入：Cabin 、Has\_Cabin、Age、Age\_Cat、Fare\_log2、Fare\_Cat、Ticket、Embarked、Sex、Name\_Length、Name\_With\_Special\_Char、Title、Family\_Size、SibSp、Parch。

## Model Selection

### 架構

第一層首先使用各個基本的Ｍodel進行嘗試，如KNN、Decision Tree、Naïve Bayes、SVM、Random Forest、AdaBoost、Extra Trees、Gradient Boosting，接著再將這些Model的產出放到Xgboost作為input，做最終的預測。



### 第一層Model的選擇

首先，我先把所有我學過的以及網路上看到大家用的分類模型拿進去測試，並得到以下結果。

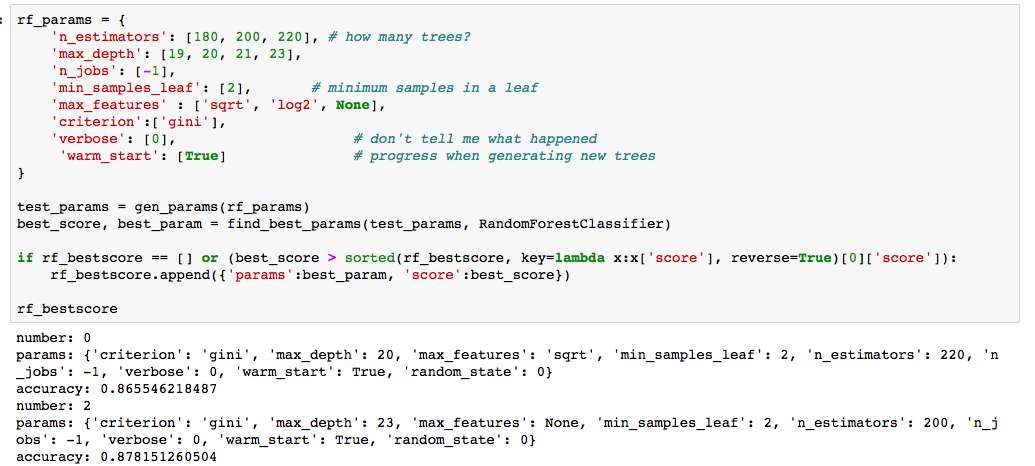
|  |  |
| --- | --- |
| **Model** | **Accuracy[[2]](#footnote-2)** |
| Random Forest | 0.860830527497 |
| Extra Trees | 0.82379349046 |
| AdaBoost | 0.814814814815 |
| Gradient Boosting | 0.814814814815 |
| SVM | 0.809203142536 |
| Decision Tree | 0.760942760943 |
| Naïve Bayes | 0.773288439955 |
| K Nearest Neighbor | 0.741863075196 |

從上面的數據可以得知，比較傳統的模型效果相對比較差一些，如簡單暴力的KNN、naïve Bayes、或是未經改良的Decision Tree。相對的，許多改良Decision Tree的方式都可以得到一些比較好的結果，如AdaBoost及Gradient Boosting透過對於分類結果權重的改變去增進Tree的準確度，又或者Extra Trees及Random Forest，透過Feature的選擇去剪裁出最好的Tree。

而實際上，從下一層XGboost產出的結果來看，若將這些比較不準確的Model結果放入，將產生非常嚴重的負面影響，因此最終我僅將Random Forest、Extra Trees、AdaBoost以及Gradient Boosting，放入Xgboost作為Input Features。

### Parameters Tuning

由於參數測試非常耗時，因此，透過了自動化的方式進行參數測試，如下圖。但是，參數測試的過程中，有一件事非常不解，從提交上Kaggle的資料看起來，如果參數測試的過程中，透過cross validation去評價accuracy，會導致嚴重的over-fitting。因此，最後僅用一組train set 跟test set進行驗證，而沒有將整個資料集分成K-Folds，輪流作為train set及test set進行驗證。

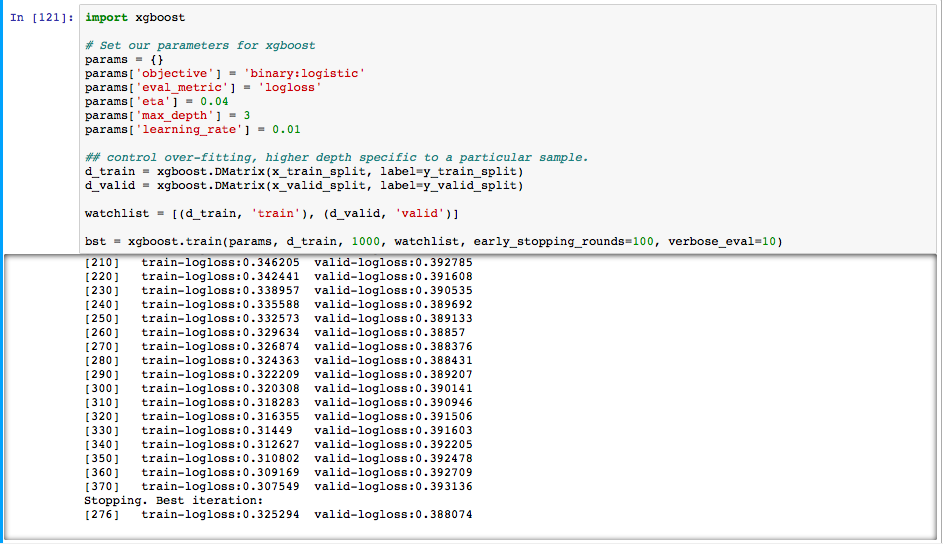


## Cross-Validation

### 第一層的Model們

在這一層中，我使用5-Folds的方式進行Cross Validation，把切成五等分，並輪流將這些folds作為train set及test set，如此一來，可以用相對較大量的test set確保不會over-fitting。

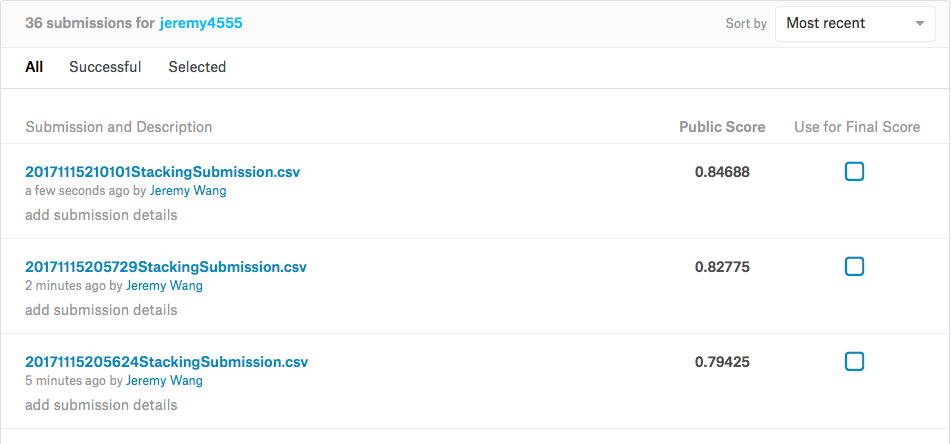
### Xgboost

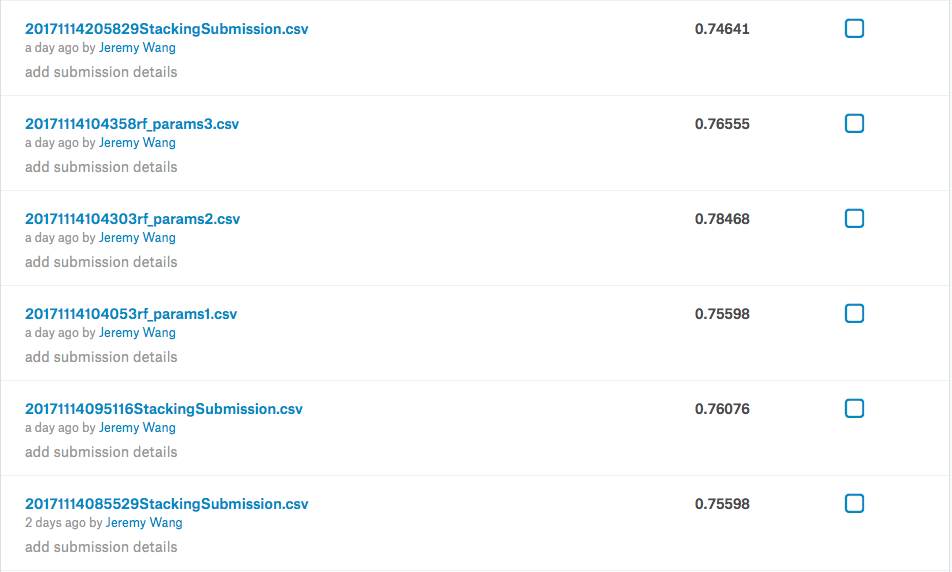
因為Xgboost有其cross-validation跟Boost的方式。首先，先把整個資料集以六四的比例切分成train set及test set，然後將其放入Xgboost進行訓練。

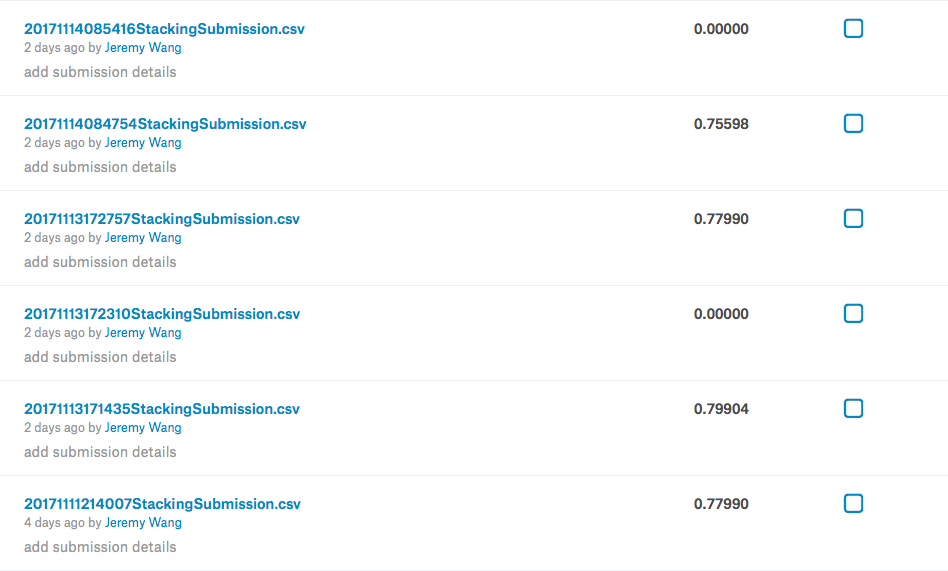
# Amazon.com-Employee Access Challenge

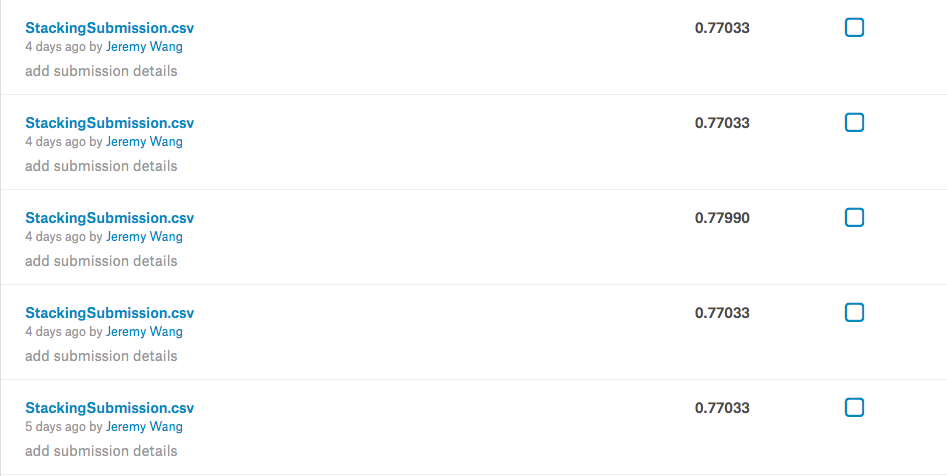
# 附註

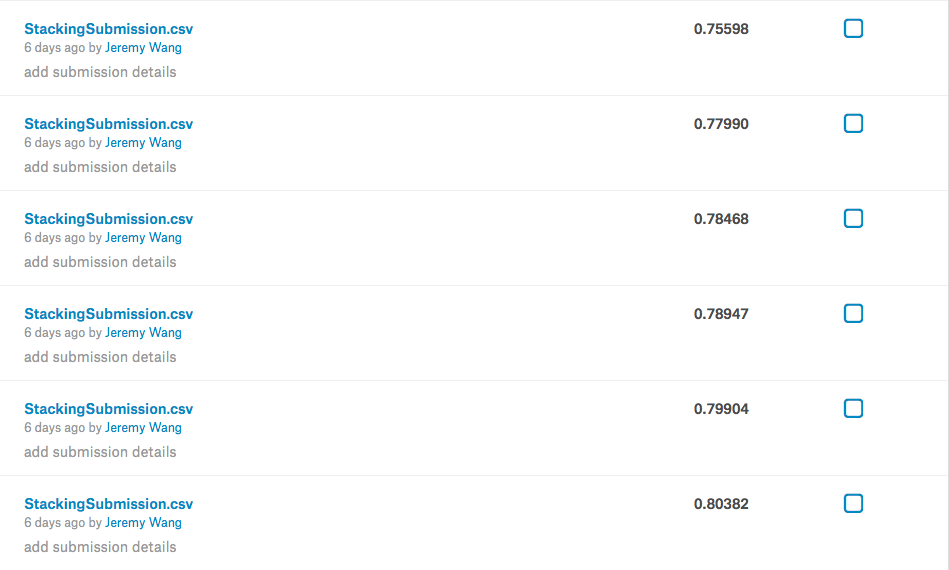
## 附註一：Titnic截圖

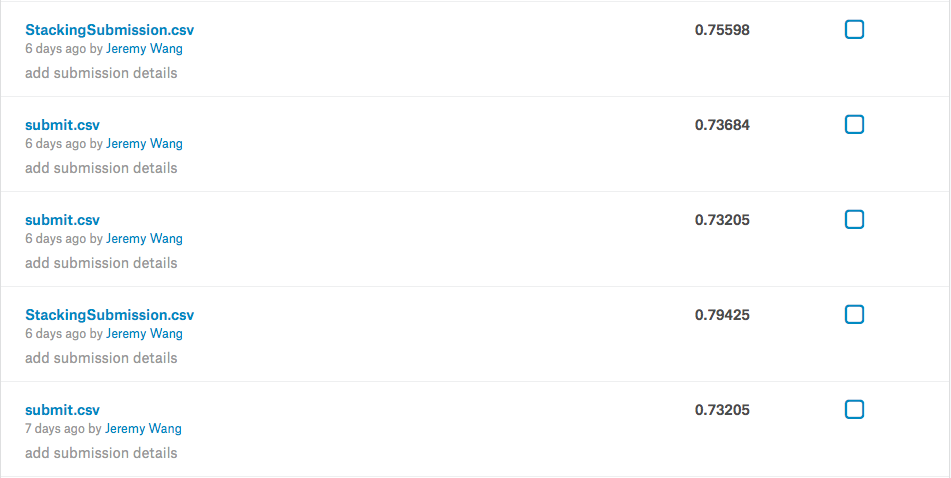


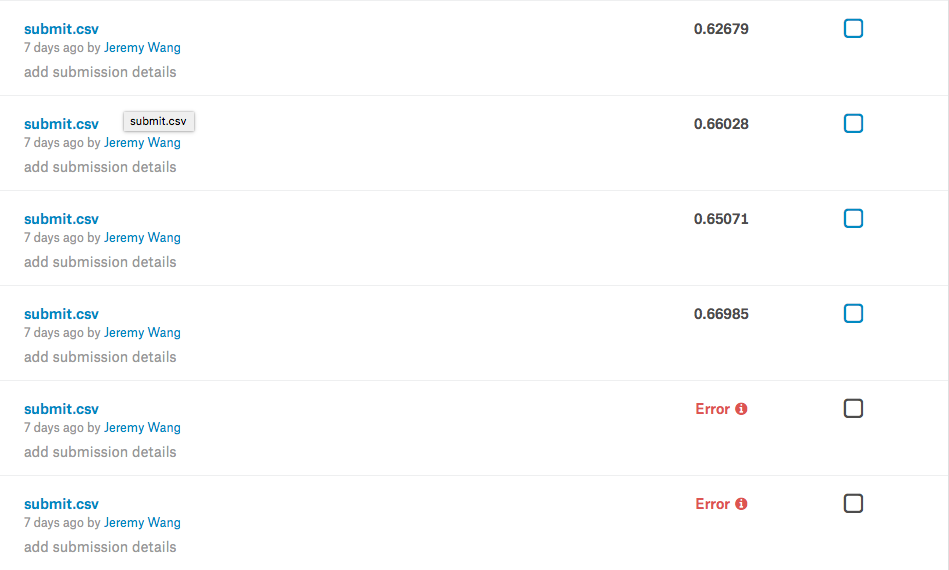












1. 見附註一 [↑](#footnote-ref-1)
2. 透過5個Folds分別作為testing data時，與Survived的比較，而得出的Accuracy。 [↑](#footnote-ref-2)