Rossmann 銷售預測

總結報告:康景皓(Austin)

1. 定義

1.1 項目概況

人工智能與數據分析隨著資訊量的暴增與資料量的迅速累積已經成為每一間企業都必須要去做的時間，如何有效利用過往數據的累積來分析跟預測是現今每一間企業都在做的事情，過往人們常常通過自己經驗來預測結果，但是每次預測的結果運好的時候預測跟實際狀況差不多但大多的時候通常都是預測失準。如今在人工智能研究中，可以通過數學的形式，對數據進行訓練，從而量化的方式得到預測結果。在機器學習領域裡，分為監督學習與非監督學習兩大類。監督學習是通過歷史的數據學習，該數據有特徵信息（輸入信息）和目標信息（預測信息），通過分析得出輸入與輸出之間的關係搭建模型，然後通過模型輸入的特性信息進行預測。非監督學習是對特徵信息進行聚類分析，但沒有目標結果。目前在金融產業中模型的使用已經非常頻繁了，在行銷的領域裡”量化行銷”是未來的趨勢，所以如何設計、運用模型將會是未來行銷人才價值所在。

由於自己對未來想要成為”行銷數據分析師”所以在這次的 Final Project 中我選擇的項目是 Rossmann 的銷售預測，本項目的數據集為 1115 個 Rossmann 門店的歷史銷售記錄和這些門店的相關信，由 Kaggle 提供 Rossmann Store Sales 所提供的 數據文件

主要分為以下三個文件： train.csv test.csv 以及 store.csv。其中， train.csv 文件中是包含銷售額數據，用來進行訓練模型； test.csv 文件中不包含銷售額數據，是用於預測的； store.csv 文件中記錄的是與商店有關的信息。

1.2 問題陳述

Rossmann 希望能對各個門店未來六週的銷售額進行預測，因為可靠的銷售額預測能讓經理提前進行合理的資源分配，從而提高更店面的資源分配。門店的銷售額會受很多因素的影響，該項目的目標就是要基於 Rossmann 各個門店的信息（比如促銷、競爭、節假日、季節性和位置等）以及過往的銷售情況來建模預測未來六週的銷售額。

實際上影響營收的狀況絕對不只有單單 Rossmann 所提供的資料，天氣、環境等因素都會影響到消費者的行為進而營收狀況有了變化但是此次仍先透過 Rossmann 提供的資訊進行預測，首先第一步就是數據的整理，找出缺失值進行填充，再來是日期的填充新增每月、每年甚至每季的情況可了解銷售狀況，再來接著進行將種類特徵（Categorical data）進行行行獨熱編碼（One-hot Encoding）數據的探索可以了解到各類型的店家實際的狀況是怎麼樣，之後主要使用 XGBoost 進行建模，並且嘗試優化參數以利後續達成目標，最後根據 Kaggle 的描述使用 RMSPE 來做為評價標準，需要對我建立的模型做一個評估。我將使用建立好的模型對於 test 數據進行預測然後將其上傳到 Kaggle 上驗證其 RMSPE 分數，分數越低表現越好

1.3 指標

在本項目中，我們使用 RMSPE 也就是 Root Mean Square Percentage Error 作為模型的評價指標。 RMSPE 的計算方式如下：

$$\mathrm{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

yi表示真實的銷售數據，$\hat{y}i$是預測數據，銷售為零的數據不進行統計。該函數可以解決線性回歸問題，預測結果是具體數值向量，所以計算預測值和實際值之間的差額平方並累加起來平均最後再開方得到的結果相比直接實際值和預測值做差額來說的大大減少了預測值與實際值之間過於詳細的數值匹配要求。

2. 分析

2.1 數據探索

Rossmann 提供的數據有 train、Test、Store 數據，數據內容如下

- Id - 一例特定日期和特定門店的樣本。
- Store - 各門店唯一的編號
- Sales - 銷售額（本項目的預測內容）。
- Customers - 日客流量。
- Open - 用來表徵商店開張或閉店的數據，0 表示閉店，1 表示開張。
- StateHoliday - 用來表徵法定假期。除了少數例外，通常所有門店都會在節假日關閉。值得注意的是，所有學校在法定假期以及週末都會關閉。數據 a 表示公共假期，b 表示復活節，c 表示聖誕節，0 則意味著不是假期。
- SchoolHoliday - 用來表徵當前樣本是否被學校的關閉所影響，也可以理解為學校放假。
- StoreType - 使用 a,b,c,d 四個值來表徵四種不同類型的商店
- Assortment - 表徵所售商品品類的等級，a 為基礎型，b 為大型，c 為特大型。
- CompetitionDistance - 距離最近競爭商家的距離（m）。
- CompetitionOpenSince[Month/Year] - 距離最近競爭商家的開業時間。
- Promo - 表徵某天是否有促銷活動。
- Promo2 - 表徵門店是否在持續推出促銷活動
- Promo2Since[Year/Week] - 以年和年中周數表徵該門店參與持續促銷的時間。
- PromoInterval - 週期性推出促銷活動的月份，例如 "Feb,May,Aug,Nov" 表示該門店在每年的 2 月 5 月 8 月和 11 月會周期性的推出促銷活動。

Train 的描述性統計

	Store	DayOfWeek	Sales	Customers	Open	Promo	SchoolHoliday
count	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06
mean	5.584297e+02	3.998341e+00	5.773819e+03	6.331459e+02	8.301067e-01	3.815145e-01	1.786467e-01
std	3.219087e+02	1.997391e+00	3.849926e+03	4.644117e+02	3.755392e-01	4.857586e-01	3.830564e-01
min	1.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	2.800000e+02	2.000000e+00	3.727000e+03	4.050000e+02	1.000000e+00	0.000000e+00	0.000000e+00
50%	5.580000e+02	4.000000e+00	5.744000e+03	6.090000e+02	1.000000e+00	0.000000e+00	0.000000e+00
75%	8.380000e+02	6.000000e+00	7.856000e+03	8.370000e+02	1.000000e+00	1.000000e+00	0.000000e+00
max	1.115000e+03	7.000000e+00	4.155100e+04	7.388000e+03	1.000000e+00	1.000000e+00	1.000000e+00

Test 的描述性統計

	Id	Store	DayOfWeek	Open	Promo	SchoolHoliday
count	41088.000000	41088.000000	41088.000000	41077.000000	41088.000000	41088.000000
mean	20544.500000	555.899533	3.979167	0.854322	0.395833	0.443487
std	11861.228267	320.274496	2.015481	0.352787	0.489035	0.496802
min	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000
25%	10272.750000	279.750000	2.000000	1.000000	0.000000	0.000000
50%	20544.500000	553.500000	4.000000	1.000000	0.000000	0.000000
75%	30816.250000	832.250000	6.000000	1.000000	1.000000	1.000000
max	41088.000000	1115.000000	7.000000	1.000000	1.000000	1.000000

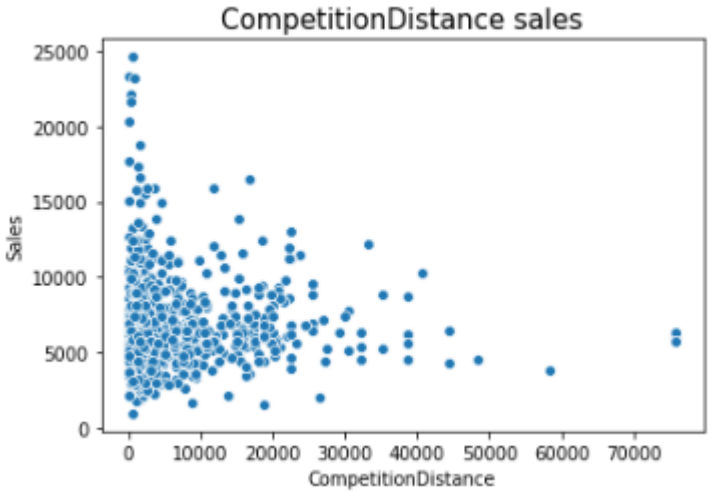
Store 的描述性統計

	Store	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear
count	1115.00000	1112.000000	761.000000	761.000000	1115.000000	571.000000	571.000000
mean	558.00000	5404.901079	7.224704	2008.668857	0.512108	23.595447	2011.763573
std	322.01708	7663.174720	3.212348	6.195983	0.500078	14.141984	1.674935
min	1.00000	20.000000	1.000000	1900.000000	0.000000	1.000000	2009.000000
25%	279.50000	717.500000	4.000000	2006.000000	0.000000	13.000000	2011.000000
50%	558.00000	2325.000000	8.000000	2010.000000	1.000000	22.000000	2012.000000
75%	836.50000	6882.500000	10.000000	2013.000000	1.000000	37.000000	2013.000000
max	1115.00000	75860.000000	12.000000	2015.000000	1.000000	50.000000	2015.000000

- Test 數據中發現到總共有 41088 條數據，其中缺失數據是在 Open 上缺失了 11 條數據
- Store 數據中有 6 種數據有了缺失值分別為 CompetitionDistance 缺了 3 條、CompetitionOpenSinceMonth 缺 354 條、CompetitionOpenSinceYear 缺 354 條、Promo2SinceWeek 缺 544 條、Promo2SinceYear 缺 544、PromoInterval 缺 544 條
- 觀察到有將近 354 家店家不知道競爭對手開始的時間，推斷可能為這些店家開設的時間比 Rossman 店面來的早所以無法推斷，又或者真的是資料缺失，此外也 500 多間的店面沒有折扣促銷的情況，代表可能整間公司比較屬於單打獨鬥類型的公司，並沒有統一化的策略，又或

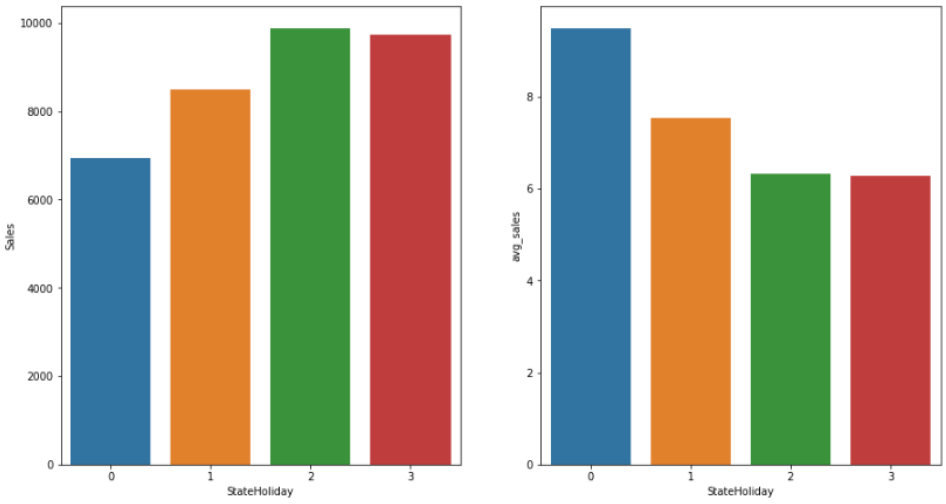
者真的是資料缺失，有三間店面附近競爭者的距離是缺失的，有可能這地方只有 rossman 一間店面，並沒有其他類型的店面，

2.2 探索性可視化



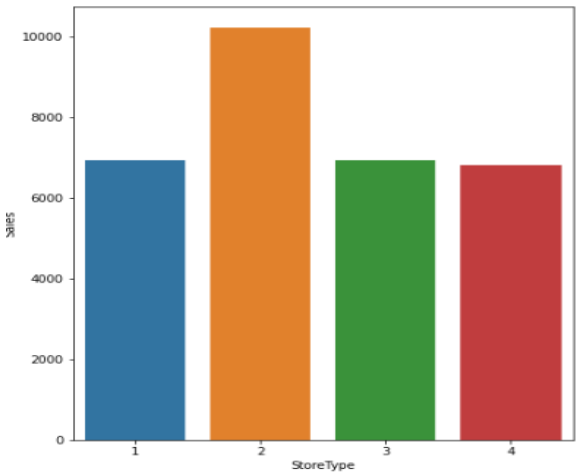
- 上圖可以看到與競爭對手距離躍進銷售額就會受到影響

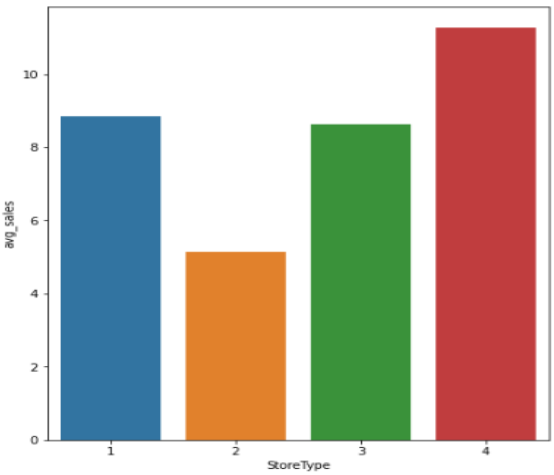
1 表示是公共假日，2 表示復活節，3 表示聖誕節，4 表示不是法定節日



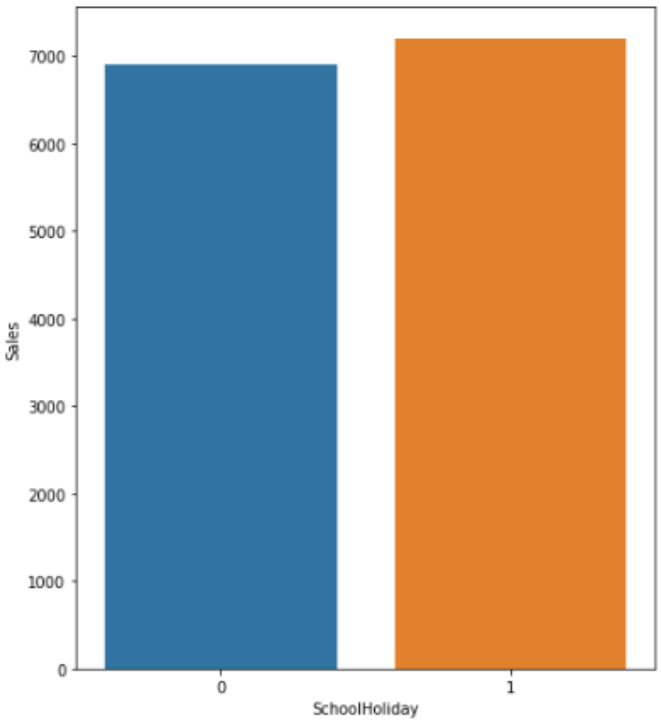
- 上圖得知不是假期的時候整體消費金額較少，但平均消費金額較高，可以推斷平日可能要上班或上學，買的人不多但是會買的人就會一次買較大量的金額，另外在重大節慶的時候消額金額會大增，顯示大家都在這些節慶買的熱度很高，後續節慶的促銷活動要把握，發想更多活動帶入更高營收

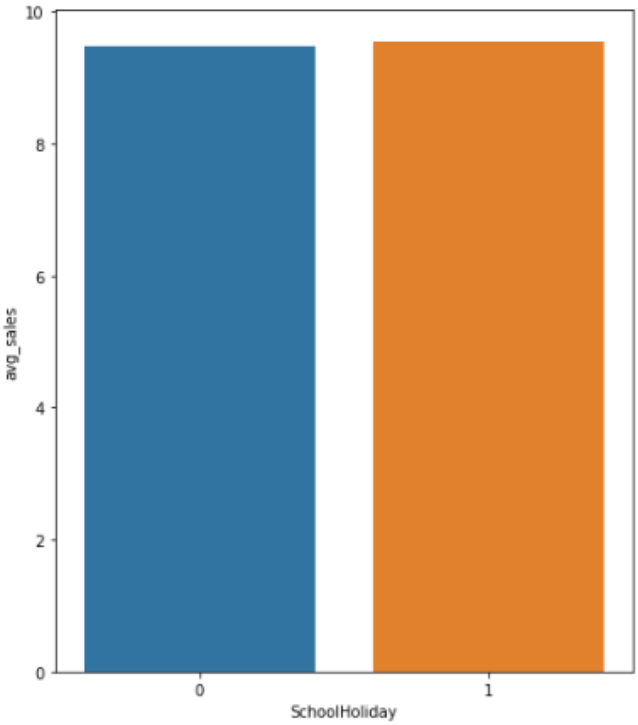
以下：1 代表 A 類店家;2 代表 B 類店家; 3 代表 C 類店家;4 代表 D 類店家



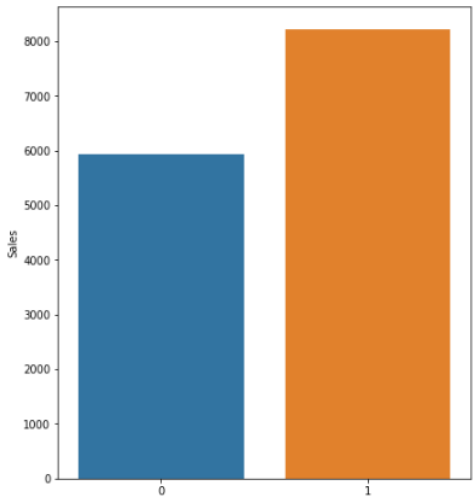


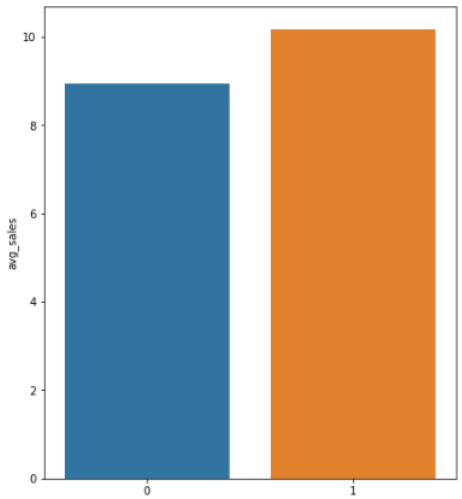
上圖了解到 B 類型店銷售額最高但是平均消費金額較低，推斷可能是在城市開的店面，其餘三者均無明顯差距





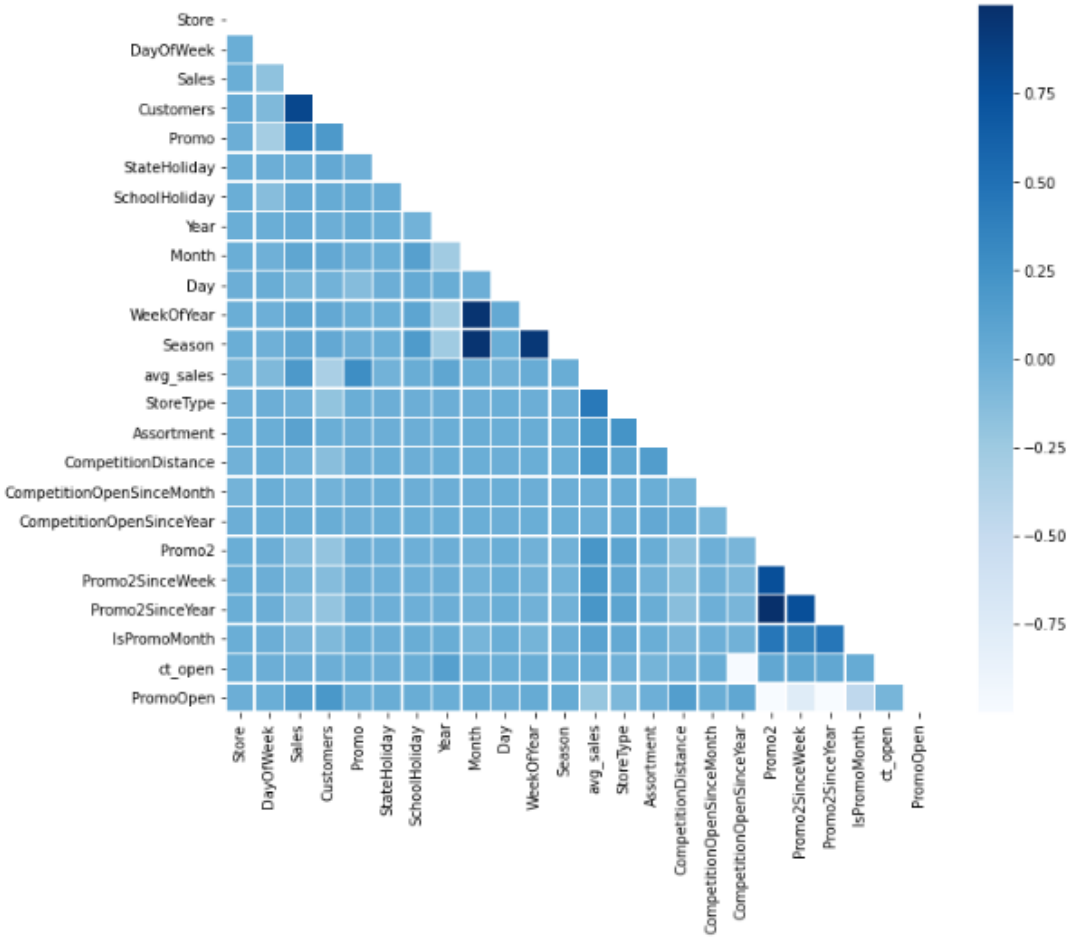
透過上圖了解到學校有無放假對於營收並無明顯差距



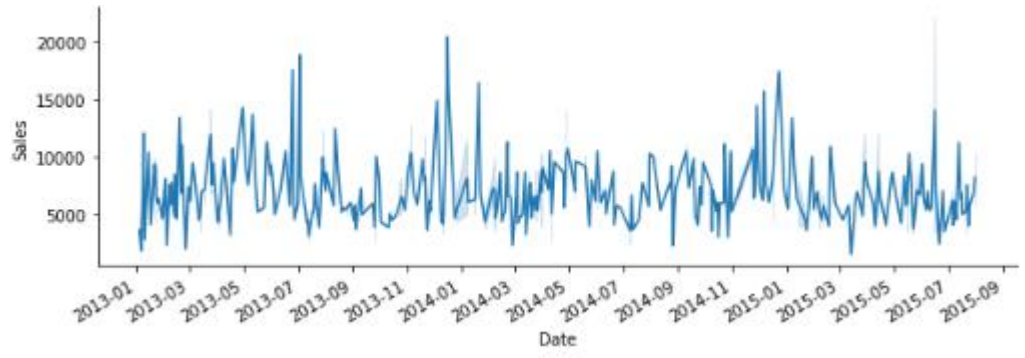


上圖觀察到，有促銷活

動能夠成功帶入較高的營收，顯示 Rossmann 的消費者是價格敏感者，後續可將促銷活動的利潤一起觀看，若利潤維持現狀或更好的話，未來可多嘗試打促銷活動



從相關係數的熱力圖來看，Customer 跟 Promo 對於營收來說有最蠻大的影響力



整體營收觀察到每月營收起伏較大，如同上述所說或許是有活動時候消費狀況才比較優異

2.3 算法與技術

綜合前述分析可以得知，此次我們的問題顯然是一個回歸的問題，模型的使用上我選擇的是隨機森林作為一開始的基準模型與 XGboost 作為主要使用的算法，來預測數據，一開始會先將數據整理好包括(某部分特徵如：StateHolidy、Storetype、Assortment)這些數據都會做進行處理，後續日期也會分成年、月、日、季度等維度以利後續分析，最後將 Store 的資料分別跟 Train 與 Test 進行結合，之後將不要的特徵移除後開始進行模型的建立與預測

2.3.1 隨機森林

在機器學習中，隨機森林是一個包含多個決策樹的分類器，並且其輸出的類別是由個別樹輸出的類別的眾數而定。

隨機森林顧名思義，是用隨機的方式建立一個森林，森林裡面有很多的決策樹組成，隨機森林的每一棵決策樹之間是沒有關聯的。在得到森林之後，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（對於分類算法），然後看看哪一類被選擇最多，就預測這個樣本為那一類。隨機森林可以既可以處理屬性為離散值的量，比如 ID3 算法，也可以處理屬性為連續值的量，比如 C4.5 算法。另外，隨機森林還可以用來進行無監督學習聚類和異常點檢測。

下面是隨機森林的構造過程：

1. 假如有 N 個樣本，則有放回的隨機選擇 N 個樣本(每次隨機選擇一個樣本，然後返回繼續選擇)。這選擇好了的 N 個樣本用來訓練一個決策樹，作為決策樹根節點處的樣本。

2. 當每個樣本有 M 個屬性時，在決策樹的每個節點需要分裂時，隨機從這 M 個屬性中選取出 m 個屬性，滿足條件 m << M。然後從這 m 個屬性中採用某種策略（比如說信息增益）來選擇 1 個屬性作為該節點的分裂屬性。

3. 決策樹形成過程中每個節點都要按照步驟 2 來分裂（很容易理解，如果下一次該節點選出來的那一個屬性是剛剛其父節點分裂時用過的屬性，則該節點已經達到了葉子節點，無須繼續分裂了）。一直到不能夠再分裂為止。注意整個決策樹形成過程中沒有進行剪枝。

選擇的調參如下幾個參數：

1.n_estimators：隨機森林中的決策樹數目，默認為 10。

2.max_features：最大特徵數，可以用整數形式考慮確定數目的特徵，用浮點數形式來考慮確定比例的特徵，也可以使用 log2，sqrt 來指定一種計算考慮特徵數目的方法。

3. max_depth：決策樹最大深度，可以指定一個整數來限制決策樹建立過程中子數的深

2.3.2 XGboost

XGBoost 是在每輪迭代中生成一棵新的回歸樹，並綜合所有回歸樹的結果，使預測值越來越逼近真實值。該算法有很強的泛化能力，其核心是每棵樹通過學習之前所有樹的殘差。與隨機森林不同，隨機森林采取的多數投票的結果，而 XGBoost 採取是所有結果的累加。如圖 2.5 所示具體進步在目標函數中細化了正則化項也就是可以更好的泛化避免過擬合，正則化項受葉子的數量和每個葉子的值來決定。

- booster：gbtree
- objective ：reg:linear →線性回歸問題
- gamma→用於控制是否後剪枝的參數越大越保守，一般 0.1、0.2 這樣子。
- max_depth→構建樹的深度，越大越容易過擬合
- subsample→隨機採樣訓練樣本
- colsample_bytre→生成樹時進行的列採樣
- min_child_weight
- silent→設置成 1 則沒有運行信息輸出，最好是設置為 0.
- eta→如同學習率

2.3.3 基準模型

此次要做的預測回歸的問題，所以基準模型部分會使用回歸經常使用到的隨機森林的跑出來的結果在 Kaggle 得出來的 RSMPE 的分數作為基準點，也因為是個基礎模型所以不會特別優化參數

3. 方法

3.1 數據預處理

(1) 先處理日期，將 train 與 Test 得資料都進行切割，切分成年、月、日、weekforyear 等值以利後續分析運用

(2) 因為後續要將 Store 的資料合併到 Train 與 test 中所以相處裡 Store 的缺失值，因認為競爭者的相關資訊會影響整體數據很大故不用 0 作為填充會使用”平均數”做填充(包含：CompetitionDistance、CompetitionOpenSinceMonth、CompetitionOpenSinceYear)，其他的(Promo2SinceWeek、Promo2SinceYear、PromoInterval)將用 0 填充資料

(3) 對定性特徵進行編碼分別將對 train 和 test 數據 中的 StateHoliday StoreType 以及 Assortment 分類特徵進行編碼。具體來說將 StateHoliday 特徵 的 取值為 0、 a、 b、 c 重新編碼成 0、 1、 2、 3。將 StoreType 特徵取值為 a、 b、 c、 d 重新編碼成 1、 2、 3、 4。Assortment 特徵 取值為 a、 b、 c 重新編碼成 1、 2、 3

(4) Store 的缺失值，StateHoliday 和 SchoolHolida 來判斷推測 Open 均為 1 並填充

(5)後續將不要的數值進行刪除，讓 train 與 test 的維度相符合

3.2 執行過程

(1)先將資料拆分成訓練集與測試集

```
train_x = train_df.drop(['Sales'],axis=1)
train_y = np.log(train_df['Sales'])

X_train, X_test, y_train, y_test = train_test_split(train_x,
                                                    train_y,
                                                    test_size = 0.3,
                                                    random_state = 42)
```

(2)開始使用隨機森林

默認參數開始測試，測試結果以此當作基準

```
randomforest_test = RandomForestRegressor()
randomforest_test.fit(X_train1,np.log1p(y_train1))
```

(4)使用 XGboost

初始 XGBoost 使用，每個參數都先盲選來測試看看，總共訓練了 1000 次，其實觀察到最終的結果為 0.11850 比 Top10%的要求 0.11773 多了一些後續透過 GridSearchCV 來進行參數的調整

```
params = {
    'booster': 'gbtree',
    'objective': 'reg:linear',  # 多分类的问题
    'gamma': 0.1,                 # 用于控制是否后剪枝的参数,越大越保守，一般0.1、0.2这样子。
    'max_depth': 8,               # 构建树的深度，越大越容易过拟合
    'subsample': 0.5,             # 随机采样训练样本
    'colsample_bytree': 0.3,      # 生成树时进行的列采样
    'min_child_weight': 3,
    'silent': 1,                  # 设置成1则没有运行信息输出，最好是设置为0.
    'random_state':5,
    'eta': 0.1                    # 如同学习率
     }
```

(4)使用 XGboost 並且用 GridSearchCV 找尋最優化參數

```
params_grid = {
    'learning_rate': [0.1,0.2,0.3,0.01,0.05],
    'max_depth': [6,10,15,50,100,150],
    'gamma': [0.7,0.8,0.9,0.1,1],
    'min_child_weight': [3,5,10,20,30,50,100],
    'subsample': [0.8,1,0.5,0.3],
    'random_state':[5,10,100,200,300],
    'colsample_bytree': [0.7,0.5,0.3,0.1]
    }


search_xgb = RandomizedSearchCV(skxgb_model, params_grid, cv = 3) # 3 fold cross validation
search_xgb.fit(X_train, y_train)

# best parameters
print(search_xgb.best_params_); print(search_xgb.best_score_)
```

```
{'subsample': 0.8, 'reg_alpha': 25, 'random_state': 10, 'min_child_weight': 3, 'max_depth': 100, 'learning_rate': 0.1, 'gamma': 0.7, 'colsample_bytree': 0.7}
0.9012337984579629
```

(5)持續使用網格搜索找尋最佳參數

使用參數後，原先想根據參數內容進行最後的執行但是發現到完全跑不動，所以後續某部分參數並未使用，且因為 max_depth 使用到 100 的關係造成速度非常之慢，最終訓練次數僅 300 次

(6)最終將調整後參數，輸出後上傳至 Kaggle 進行評分，最終結果 RSMPE 為 0.11156 結果非常的好回去對應到 Kaggle 最終的得分排名應該是 41 名表現蠻不錯的

```
params_final1 = {
    'booster': 'gbtree',
    'objective': 'reg:linear',  # 多分类的问题
    'gamma': 0.1,                  # 用于控制是否后剪枝的参数,越大越保守，一般0.1、0.2这样子。
    'max_depth': 100,               # 构建树的深度，越大越容易过拟合
    'subsample': 0.8,              # 随机采样训练样本
    'colsample_bytree': 0.7,       # 生成树时进行的列采样
    'min_child_weight': 3,
    'silent': 1,                   # 设置成1则没有运行信息输出，最好是设置为0.
    'random_state':5,
    'eta': 0.1                    # 如同学习率
     }
```

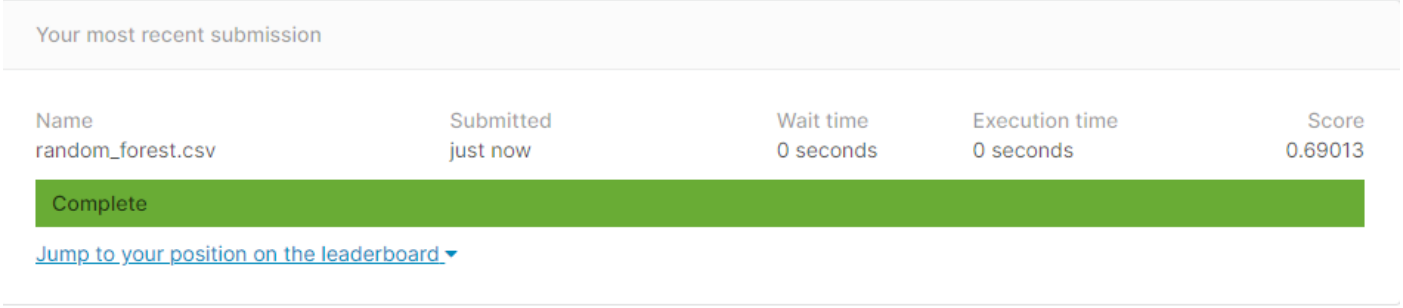
3.3 完善

首先使用隨機森林模型去執行，但因並沒有使用任何參數所以實際結果效果不好，後續使用 XGBoost 後續效果非常優異，雖然網格搜索可以幫助我們找到最優化參數但是若您測試的參數較高，電腦不好的話其實根本無法用

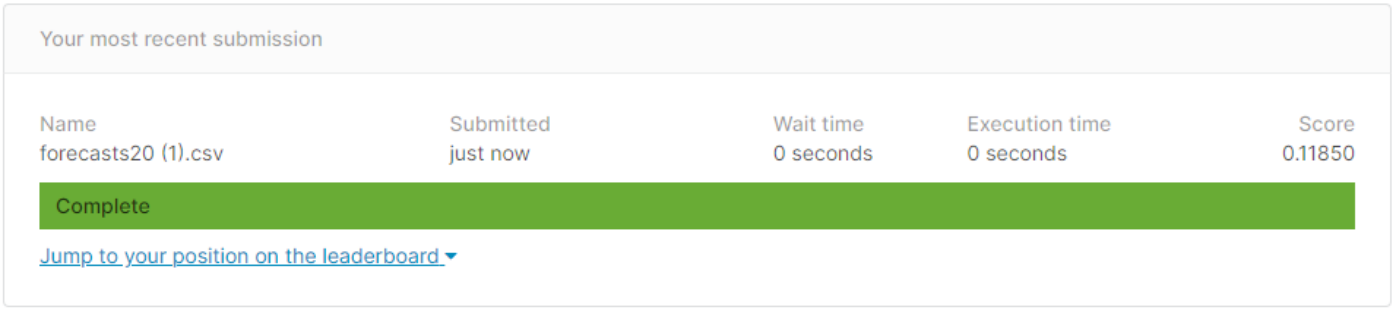
4 結果

4.1 模型的评价与验证

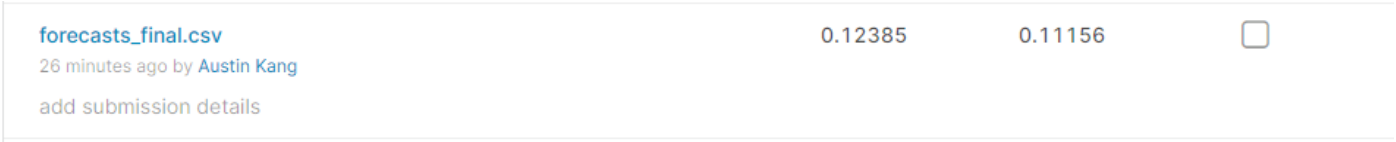
隨機森林分數



初始 XGB 分數



網格搜索後 XGB 分數



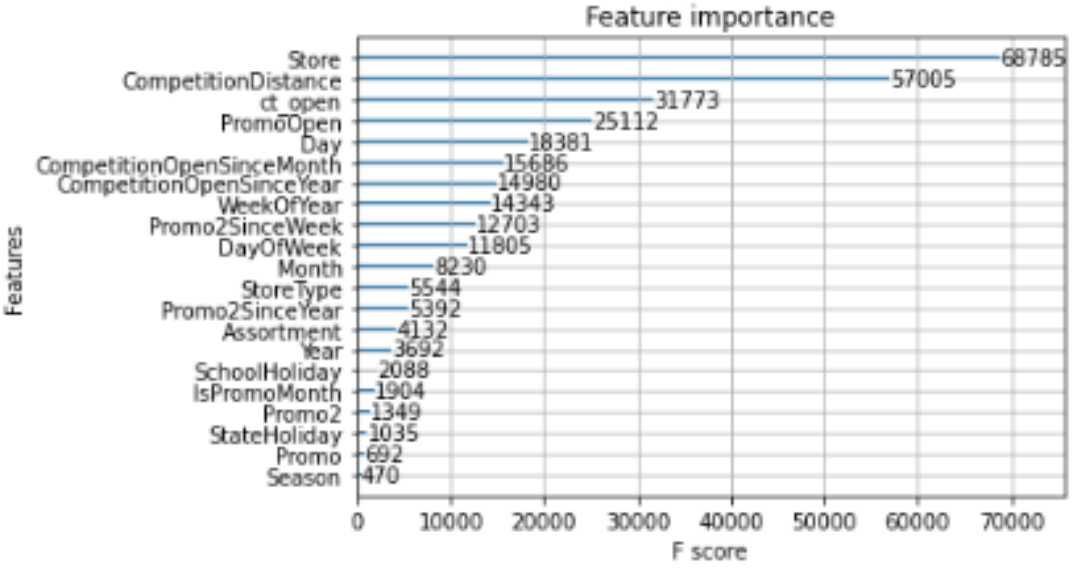
4.2 合理性分析

一開始隨機森林因非常單純未添加任何參數，最終跑出來的分數其實非常可怕，顯示這模型在真實生活動並無法使用的那麼理想，XGBoost 訓練時間比較長，特別是深度增加，樹的數量增加以後，但是模型的預測表現很好，但是加深深度後一般電腦其實負荷不了，後續網格搜索的值未來需要再多考慮

5. 項目結論

5.1 可視化分析

XGB 特徵重要性排序



從上述可觀察到店面整體來看與銷售額關係最大，並不難理解一間店面開立在一間市區的店面，自然而然對於銷售額來看就會比較高，或者是開業很久的店面也會累積較忠實的客群，另外競爭者的距離對於銷售額影響蠻大的也很正常若一個地區只有這一間店面自然而然地會成為獨大的店面銷售額也會比較好，此外如同之前在做資料探索的時候發現到，是否有優惠活動對於營業額也會有部分影響，顯示 Rossmann 的消費者是價格的敏感者，適當的促銷活動對於業績也持續有正面的影響

5.2 對項目的思考

這算是第一次完整的透過資料的整理，數據的探索、建立模型的評估來進行撰寫，其實對我來說最難的部分算是資料的整理，因為時常聽有經驗的前輩說資料整理得好可以帶你上天堂，資料整的不好你做什麼厲害的模型都是沒有用的，所以缺失值怎麼補比較正確，特徵值應該怎麼轉換這些都是花了最多的時間在處理，後續自然就是模型的選擇與調整參數，因為人工智能火熱已經好幾年了所以模型選哪個自然而然都會知道那些模型比較好準確度比較高，剩下的就是調整參數而已，對我來說重要的其實不是這些基本調整參數之類的，最重要的其實是我把資料最好分析後最終的結果該如何跟內部同仁講解並且能夠實際的把結果運行到實際的生活中，這可能是我之後要思考的點

5.3 需要作出的改善

特徵工程的部分可以考慮使用 PCA 降維的方式進行，此次並沒有使用此種方式，另外影響消費者購買意願的實際狀況肯定比實際多例如：天氣、景氣等因素對於消費者都是蠻大的影響，另外未來可嘗試使用一些深度學習的方式進行預測。

6. 參考

https://www.kaggle.com/holoong9291/udacity-final-project-rossmann

https://www.kaggle.com/c/rossmann-store-sales/discussion/18024

https://blog.csdn.net/onepiecehuiyu/article/details/51628931

https://zhuanlan.zhihu.com/p/31182879