Kaggle: House Prices: Advanced Regression Techniques

- 比賽敘述:利用 1460 筆房屋交易資料作為訓練資料，預測另外 1459 筆房價
- 資料敘述:所有房屋皆位在 Ames, Iowa(人口 6.6 萬，63 平方公里)，每筆資料有 79 個屬性欄位，例如 YearBuilt, OverallQual, HouseStyle, 1stFlrSF, 2ndFlrSF 等，但有部分資料有空白的欄位。

使用到 Python(Pandas, scikit-learn, seaborn)，會在整理資料後用 scikit-learn 中的 Gradient Boosting Regressor 模型來進行學習和預測。

流程:

1. 先讀取 trainingData 和 testingData:

```
trainingData = pd.read_csv('C:/Users/.../train.csv')
testingData = pd.read_csv('C:/Users/.../test.csv')
```

2. 檢視 trainingData 中欄位缺失的情形:

	Total	Percent
PoolQC	1453	0.995205
MiscFeature	1406	0.963014
Alley	1369	0.937671
Fence	1179	0.807534
FireplaceQu	690	0.472603
LotFrontage	259	0.177397
GarageCond	81	0.055479
GarageType	81	0.055479
GarageYrBlt	81	0.055479
GarageFinish	81	0.055479
GarageQual	81	0.055479
BsmtExposure	38	0.026027
BsmtFinType2	38	0.026027
BsmtFinType1	37	0.025342
BsmtCond	37	0.025342
BsmtQual	37	0.025342
MasVnrArea	8	0.005479
MasVnrType	8	0.005479
Electrical	1	0.000685
Utilities	0	0.000000

直接捨棄缺失最嚴重的六個欄位(PoolQC, MiscFeature, Alley, Fence, FireplaceQu, LotFrontage)

再檢視 GarageArea 欄位(沒有缺失)後可知道 Garage 相關欄位的缺失大多是因為該房屋沒有停車場造成的，可補上應該填入的值。而 Bsmt 和 MasVnr 相關欄位也做對應的處裡。Electrical 欄位則是填入 trainingData 中該欄位的眾數。

而 testingData 中 Garage, Bsmt 和 MasVnr 欄位的缺失也是由檢察該房屋是否沒有停車場、地下室或磚砌表面補上缺失資料，其他欄位的缺失則是填入 trainingData 中該類別的眾數做為補值依據。

3. 補上 TotalSF 和 TotalBath 兩欄位，其中

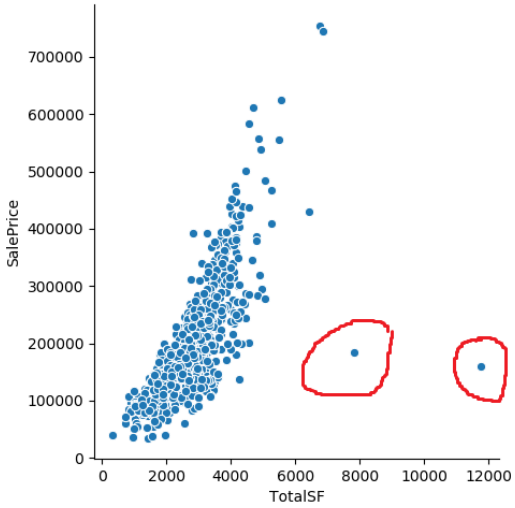
TotalSF=TotalBsmtSF+1stFlrSF+2ndFlrSF

TotalBath=BsmtFullBath+BsmtHalfBath+FullBath+HalfBath

4. 將部分有數值意義的類別欄位做轉型:

```
#change some attributes to numerical
trainingData["ExterQual"] = trainingData["ExterQual"].map({"Ex":5, "Gd":4, "TA":3, "Fa":2, "Po":1})
trainingData["ExterCond"] = trainingData["ExterCond"].map({"Ex":5, "Gd":4, "TA":3, "Fa":2, "Po":1})
trainingData["HeatingQC"] = trainingData["HeatingQC"].map({"Ex":5, "Gd":4, "TA":3, "Fa":2, "Po":1})
trainingData["KitchenQual"] = trainingData["KitchenQual"].map({"Ex":5, "Gd":4, "TA":3, "Fa":2, "Po":1})
```

5. 找出並刪除 outlier:



6. 將 categorical feature 轉換:

```
#prosess categorical features
len_train_x=train_X.shape[0]
tmp_all=pd.concat(objs=[train_X, test_X], axis=0, sort=False).reset_index(drop=True)
tmp_all=pd.get_dummies(tmp_all)
tmp_all= tmp_all.astype(float)

new_train_X=tmp_all[:len_train_x]
new_test_X=tmp_all[len_train_x:]
```

7. 用 Cross Validation 的方式調整 GradientBoostingRegressor 的參數:

```
#Set model
GBR = ensemble.GradientBoostingRegressor(n_estimators=3000, learning_rate=0.05, max_depth=3, max_features='sqrt',
                                         min_samples_leaf=10, loss='huber')
```

8. 最後再訓練出 30 個 GBR 模型，用 uniform blending 的方式組合出最後的預測模型。在預測 testing data 上得到的分數是 0.11809，PR 值大約 81。