國立清華大學

智慧化企業整合

Project2
二手車價格預測

指導教授:邱銘傳 教授

第八組

110034555 李勁緯
110034558 廖文辰
110034562 黃鉦程
110034586 熊夆峻

110/12/17

目錄

第一章、前言.....2
第二章、資料前處理.....2
2.1 資料來源.....2
2.2 資料欄位.....2
14. Price：該汽車的價格 2.1.3 資料整合.....2
2.3 資料整理.....3
2.4 特徵提取.....3
(1) 公里數.....3
下圖為公里數與價格，行駛公里數低於 8 萬公里的車價格會比較好，故新增一欄以行駛公里數 8 萬為界線的資料。.....4
.....4
.....4
(2)打檔方式.....4
下圖為自排手排平均價格，自排的價格高，故新增一欄位為是否為自排。.....4
.....5
.....5
座位數.....5
下圖為 X 軸為座位數 Y 軸為價格顯示出座位的平均售價及座位車種的總數，從這兩張圖可以發現兩人座位重要的少數。所以我們新增一欄是否為兩人座的欄位。.....5
.....6
.....6
12. is_kilometers_driven_less_80k：已行駛里程數低於 8 萬.....7
13. is_diesel：是否為柴油.....7
14. is_automatic：是否為自排.....7
15. is_2_seats：是否為兩人座.....7
2.6 資料整合.....7
第三章、模型介紹.....8
第四章、模型分析.....8
4.1 XG Boost.....8
第五章、結果與討論.....9
5.1 結論.....9
5.2 未來展望.....9
本次模型僅針對二手車較明顯之 14 項特性資料進行訓練，且其中幾項特性資料無法納入該模型之訓練，若未來能加入更多較容易被賣家隱藏之二手車特性資料，例如：是否為事故車、是否為泡水車等，此模型將能更準確的預測二手車之價格。本模型僅運用 XGboost 方法預測二手車之價格，未來若能運用 RNN 等機器學習方法於此模型中，能提升預測之準確性。最後一點未來須注意避免造假，以免影響預測準確度。.....9

第一章、前言

1.1 背景介紹

在新冠疫情影響下，多數人的財務狀況緊縮，然而新車的價格卻逐年攀升，對於首次購車或資金較不充裕的人，二手車是個較為合適的決定，根據報導指出，在印度二手車之購買族群年齡有 70%的比例為 40 歲以下的青壯年，且 2020 年印度二手車市場的銷售量高達 440 萬輛，超過同期新車市場規模，可見二手車市場相當龐大。若能使用機器學習模型有效預測二手車價格，必定能為許多人帶來更大的便利性。

1.2 5W1H

為提升買賣雙方交易之滿意度，本次報告欲以 Random Forest 及 XG Boost 兩種機器學習模型來預測二手車價格，以評估合理且公平之二手車價格。因此，本報告先以 5W1H 定義問題。

- Who 有意將手中舊車脫手之賣家及打算購入二手車之買家
- What 為了能夠評估合理且公平之二手車價格
- Why 買賣家交易二手車時，雙方皆能以合理且公平的價格交易，以提升雙方交易之滿意度
- When 二手車交易時
- Where 印度各城市為例
- How 資料分析、深度學習

第二章、資料前處理

2.1 資料來源

為了建立合適的模型，我們採用 Kaggle 平台上的數據作為本次 Project 的資料來源。Kaggle 是一個數據建模和數據分析競賽平台。企業和研究者可在此發布數據，提供統計學家和數據挖掘專家以此資料為基礎，進行競賽以產生最好的模型。

2.2 資料欄位

為了釐清各欄位資料所代表的資訊及意涵，以使後續資料處理與模型分析得以順利進行，本次資料欄位共有下列 14 項：

1. Unnamed：汽車編號
2. Name：汽車型號名稱
3. Location：汽車所在地點
4. Year：汽車出廠年份
5. Kilometers_Driven：已行駛里程數(公里)
6. Fuel_Type：燃油/燃料種類
7. Transmission：打檔方式
8. Owner_Type：汽車經過了幾手交易
9. Mileage：平均油耗
10. Engine：引擎規格
11. Power：馬力
12. Seats：汽車內的座位數
13. New_price：新車價格
14. Price：該汽車的價格 2.1.3 資料整合

2.3 套件選擇

，在此次專案中使用之五個主要的套件：

1. Numpy：提供維度陣列與矩陣的運算。
2. Pandas：提供數據操作和分析的 data frame 結構。
3. Matplotlib：提供圖形繪製的工具。
4. Warnings：提示使用者一些錯誤或是過時的用法。
5. Seaborn：提供進階的圖表繪製，是以 matplotlib 為基礎的工具

2.3 資料整理

```
[ ]  print(train.isnull().sum())
```

```
Unnamed: 0           0
Name                 0
Location             0
Year                 0
Kilometers_Driven    0
Fuel_Type            0
Transmission         0
Owner_Type           0
Mileage              2
Engine              36
Power               36
Seats               42
Price                0
dtype: int64
```

圖 1 空缺值數量

首先，找出所有為空值之欄位以及空值之數量。

```
[29]  #######mileage
      train['Mileage'].value_counts(dropna=False)
```

```
17.0 kmpl     172
18.9 kmpl     172
18.6 kmpl     119
20.36 kmpl     87
21.1 kmpl      86
              ...
9.1 kmpl        1
15.11 kmpl      1
20.86 kmpl      1
17.16 kmpl      1
23.03 kmpl      1
Name: Mileage, Length: 443, dtype: int64
```

圖 2 油耗統計

對照車種及燃油種類，將「Mileage」「Engine」、「Power」、「Seats」四欄中的空值以手動的方式輸入

2.4 特徵提取

(1) 公里數

下圖為公里數與價格，行駛公里數低於 8 萬公里的車價格會比較好，故新增一欄以行駛公里數 8 萬為界線的資料。

```
plt.figure(figsize=(10, 7))
plt.scatter(train['Kilometers_Driven'], train['Price'])
```

```
<matplotlib.collections.PathCollection at 0x7efe2b224390>
```

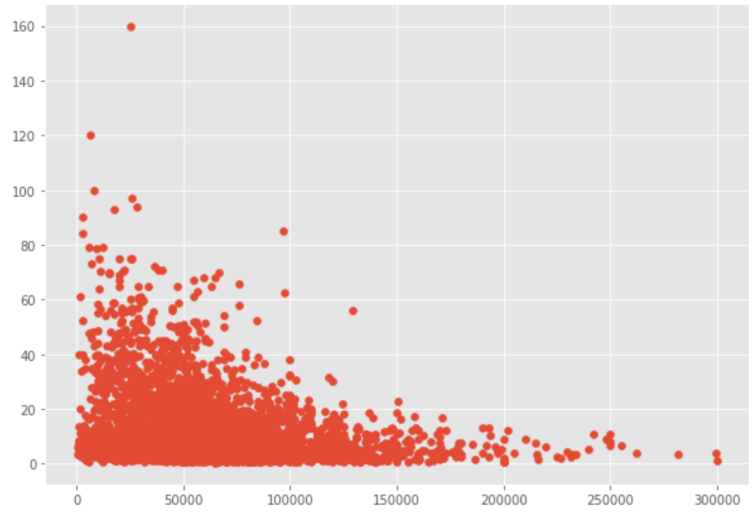

圖 3 價格與公里數散佈趨勢圖

```
less_80k = pd.DataFrame(columns=['is_kilometers_driven_less_80k'])
train = pd.concat([train, less_80k])


for i in train.index:
        value = train['Kilometers_Driven'][i]
        if value < 80000:
                train['is_kilometers_driven_less_80k'][i] = 1
        else:
                train['is_kilometers_driven_less_80k'][i] = 0
```

圖 4 公里數特徵提取

(2)打檔方式

下圖為自排手排平均價格，自排的價格高，故新增一欄位為是否為自排。

```
plt.figure(figsize=(10, 7))
sns.barplot('Transmission', 'Price', data=train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efe228fe650>
```

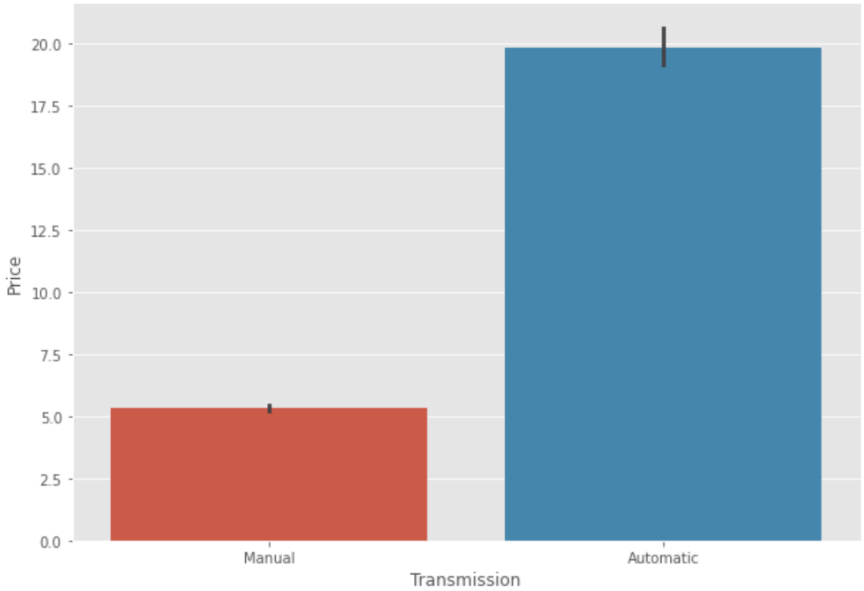

圖 5 打檔方式與價格之直方圖

```
is_automatic = pd.DataFrame(columns=['is_automatic'])
train = pd.concat([train, is_automatic], axis=1)


for i in train.index:
        value = train['Transmission'][i]
        if value == 'Automatic':
                train['is_automatic'][i] = 1
        else:
                train['is_automatic'][i] = 0
```

圖 6 打檔方式特徵提取

座位數

下圖為 X 軸為座位數 Y 軸為價格顯示出座位的平均售價及座位車種的總數，從這兩張圖可以發現兩人座位重要的少數。所以我們新增一欄是否為兩人座的欄位。

```
plt.figure(figsize=(8,  6))
sns.barplot('Seats',  'Price',  data=train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efe2b60a810>
```

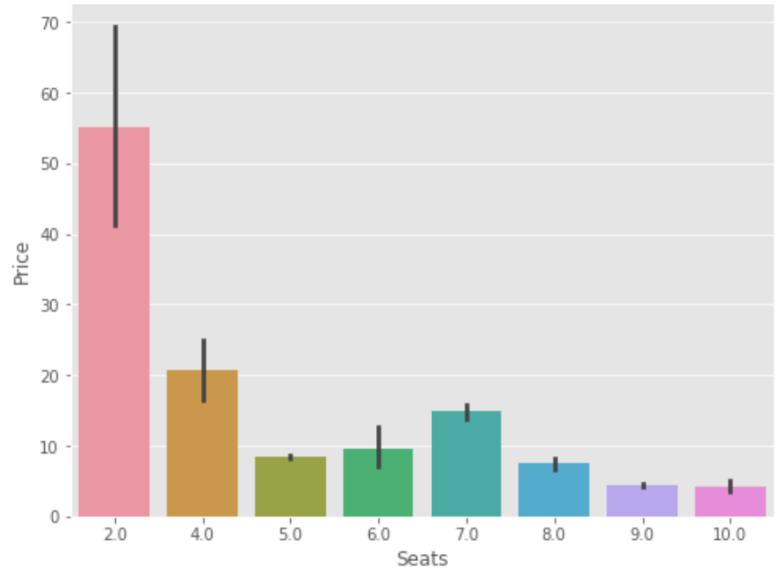

圖 7 座位與價格之直方圖

```
plt.figure(figsize=(8,  6))
sns.countplot('Seats',  data=train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efe22624350>
```

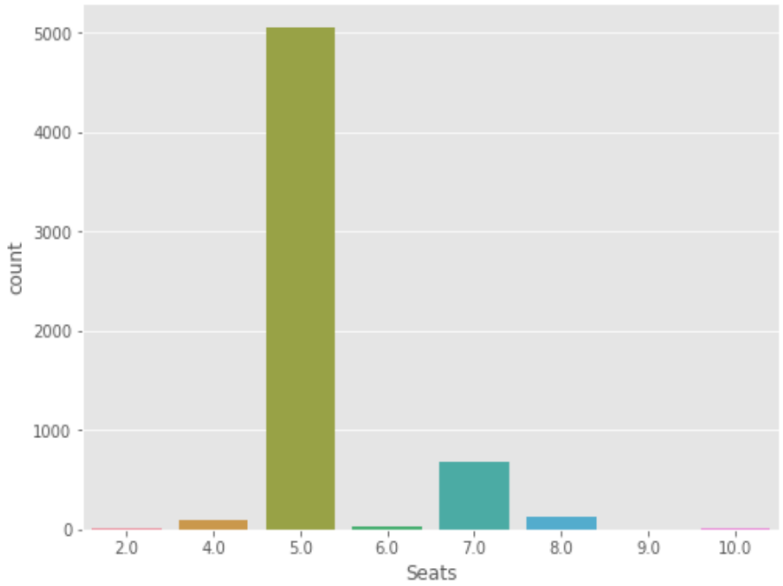

圖 8 座位數統計

2.5 處理後之資料欄位

將資料處理完，，經過重複資料刪除以及特徵提取新增後，總共有 15 個欄位。處理後之資料欄位共有下列 14 項：

1. Location：汽車所在地點
2. Year：汽車出廠年份
3. Kilometers_Driven：已行駛里程數(公里)
4. Fuel_Type：燃油/燃料種類
5. Transmission：打檔方式
6. Owner_Type：汽車經過了幾手交易
7. Seats：汽車內的座位數
8. Mileage_value：平均油耗
9. Mileage_measure：平均油耗單位
10. Engine_value：引擎數據
11. Power_value：馬力數據
12. is_kilometers_driven_less_80k：已行駛里程數低於 8 萬
13. is_diesel：是否為柴油
14. is_automatic：是否為自排
15. is_2_seats：是否為兩人座

2.6 資料整合

```
for i in ['is_kilometers_driven_less_80k',
                'is_diesel', 'is_automatic', 'is_2_seats',
                'Mileage_value', 'Engine_value','Power_value']:
        print(i)
        train[i] = train[i].astype('float32')
        test[i] = test[i].astype('float32')
```

圖 9 型態轉換

將新增的欄位以及原本的「Mileage_value」、「Engine_value」及「Power_value」轉為浮點數

```
print(train.shape)
cat = train.select_dtypes(include='object')

for i in cat.columns:
        le = LabelEncoder()
        train[i] = le.fit_transform(train[i])
```

圖 10 資料正規化

使用 LabelEncoder 將特徵為「Object」的資料正規化，再將資料擬和

```
X_train, X_test, y_train, y_test = train_test_split(train, train_target, test_size=0.15, random_state=42)
```

將資料劃分為「X_train」、「X_test」、「y_train」、「y_test」

第三章、模型介紹

3.1 XG Boost

XGboost 每一棵樹是互相關聯的，目標是希望後面生成的樹能夠修正前面一棵樹犯錯的地方。此外 XGboost 是採用特徵隨機採樣的技巧，和隨機森林一樣在生成每一棵樹的時候隨機抽取特徵，因此在每棵樹的生成中並不會每一次都拿全部的特徵參與決策。此外為了讓模型過於複雜，XGboost 在目標函數添加了標準化。因為模型在訓練時為了擬合訓練資料，會產生很多高次項的函數，但反而容易被雜訊干擾導致過度擬合。因此 L1/L2 Regularization 目的是讓損失函數更佳平滑，且抗雜訊干擾能力更大。

第四章、模型分析

4.1 XG Boost

以下列了六種參數，並分別介紹每個參數，將透過 Grid Search 的方式來找出最佳參數組合。

- booster：gbtree 樹模型/ dart(將深度神經網絡的 dropout 技術添加到 boosted 樹)/xgblinear。
  Boosting 是一種基於集成原理的順序技術。它結合了一組弱學習器並提高了預測精度。在任何時刻 t，模型結果都基於前一個時刻 t-1 的結果進行加權。正確預測的結果被賦予較低的權重，而錯誤分類的結果權重較高。
- Eta: 為學習率，範圍通常在 0.01-0.2 之間。
- gamma: 懲罰項係數，指定節點分裂所需的最小損失函數下降值。這個參數的值越大，算法越保守。這個參數的值和損失函數有關。
- Max_depth：為樹的最大深度，範圍通常在 3-10 之間。
- lambda: 權重的 L2 正則化項。這個參數是用來控制 XGBoost 的正則化部分的。
- alpha: 權重的 L1 正則化項。可以應用在很高維度的情況下，使得算法的速度更快。

首先三個 booster，我們先測試是否每個 booster 的效果都不錯，跑完的結果如下，gblinear 的 MSE 非常高，所以就不繼續考慮使用此一 booster。

```
# XGBoost
xgbmodel = xgb.XGBRegressor(booster = 'gbtree' )
xgbmodel.fit(X_train, y_train)

# Predict training data
y_pred_xg_train = xgbmodel.predict(X_train)

# Evaluation training data
print("MSE on Training Set (XGBoost):", mean_squared_error(y_pred_xg_train, y_train))

# Predict test data
y_pred_xg_dev = xgbmodel.predict(X_test)

# Evaluation test data
print("MSE on Test Set (XGBoost):", mean_squared_error(y_pred_xg_dev, y_test))
```

圖 11 XGBoost 資料分析

booster	gbtree	Dart	gblinear
Test MSE	11.04	11.04	33.5

接著使用 XGBoost Grid Search 來尋找最佳解。Booster 有兩種參數、eta 三種、gamma 三種、max_depth 四種、lambda 三種、alpha 三種，並且 cross validation 是五組、總共有 2*3*3*4*3*3*5 組為 3240 種組合。

```
xgbb = xgb.XGBRegressor()

param_grid = {
        "booster": ['gbtree', 'dart'],
        "eta": [0.01, 0.05, 0.1],
        "gamma": [0, 0.5 ,1],
        "max_depth": [3, 5, 7, 9 ],
        "lambda": [0.1, 1, 10],
        "alpha": [0.1, 1, 10]
}

xgb_model = GridSearchCV(estimator=xgbb, cv=5, param_grid=param_grid, verbose=2)
xgb_model.fit(train, train_target)

print(xgb_model.best_score_)
print(xgb_model.best_estimator_.get_params())
```

圖 12 XGBoost Grid Search

以下為最佳參數組合，MSE 值為 1.21，Accuracy 達到 0.87536。

最佳參數組合	
Booster	Gbtree
Eta	0.01
Gamma	0.5
Max_depth	7
Lambda	0.1
alpha	0.1
Test MSE	1.21
Accuracy	0.87536

第五章、結果與討論

5.1 結論

本次報告首先對 XG Boost 尋找最佳參數。XGBoost 準確度高，但我們發現尋找參數時間較長。因此，若實際利運用此套件，使用者應於準確度及時間兩者之間做考量。再者座位數、燃油種類、里程數為影響二手車售價的重要因素。

5.2 未來展望

本次模型僅針對二手車較明顯之 14 項特性資料進行訓練，且其中幾項特性資料無法納入該模型之訓練，若未來能加入更多較容易被賣家隱藏之二手車特性資料，例如：是否為事故車、是否為泡水車等，此模型將能更準確的預測二手車之價格。本模型僅運用 XGboost 方法預測二手車之價格，未來若能運用 RNN 等機器學習方法於此模型中，能提升預測之準確性。最後一點未來須注意避免造假，以免影響預測準確度。