資料探勘期末報告

主題: 利用租借腳踏車狀況的資料集來達到預測租借人數

1.

- 資料: **bike seoul sharing**
- 來源: kaggle open datasets
  - 網址: https://www.kaggle.com/datasets/willianoliveiragibin/bike-seoul-sharing/
- 欄位：
  - **Date**(日期)
  - **Rented Bike Count**(租用腳踏車數量)
  - **Hour**(時辰 0~23 點)
  - **Temperature**(溫度)
  - **Humidity** (%)(濕度)
  - **Wind speed** (m/s)(風速)
  - **Visibility** (10m)(可見度)
  - **Dew point temperature**(露點溫度)
  - **Solar Radiation** (MJ/m2)(太陽輻射)
  - **Rainfall** (mm)(雨量)
  - **Snowfall** (cm)(降雪量)

- **Seasons** (季節)
- **Holiday** (節日)
- **Functioning Day** (運作日)

	Date	Rented Bike Count	Hour	Temperature(C)	Humidity(%)	Wind speed (m/s)	Visibility (10m)	Dew point temperature(C)	Solar Radiation (MJ/m2)	Rainfall(mm)	Snowfall (cm)	Seasons	Holiday	Functioning Day
0	01/12/2017	254	0	-5.2	37	2.2	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
1	01/12/2017	204	1	-5.5	38	0.8	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
2	01/12/2017	173	2	-6.0	39	1.0	2000	-17.7	0.0	0.0	0.0	Winter	No Holiday	Yes
3	01/12/2017	107	3	-6.2	40	0.9	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
4	01/12/2017	78	4	-6.0	36	2.3	2000	-18.6	0.0	0.0	0.0	Winter	No Holiday	Yes

2. 為了達到這次的主題目的，首先要先了解資料的狀態，觀察並確定哪些資料是我們所需要的，然後對其資料做預處裡。

3. 首先觀察資料內容、屬性，並把需要改動的資料進行修改。

```
Date                       object
Rented Bike Count           int64
Hour                        int64
Temperature(C)            float64
Humidity(%)                 int64
Wind speed (m/s)          float64
Visibility (10m)            int64
Dew point temperature(C)  float64
Solar Radiation (MJ/m2)   float64
Rainfall(mm)              float64
Snowfall (cm)             float64
Seasons                    object
Holiday                    object
Functioning Day            object
dtype: object
```

可以發現有些資料屬性為 object，為了後面能方便使用，先將其屬性改為我們所需要的狀態。

	Month	RentedBikeCount	Hour	Temperature	Humidity	WindSpeed	Visibility	DewPointTemperature	SolarRadiation	Rainfall	Snowfall	Seasons	Holiday	FunctioningDay
0	12	254	0	-5.2	37	2.2	2000	-17.6	0.0	0.0	0.0	0	0	1
1	12	204	1	-5.5	38	0.8	2000	-17.6	0.0	0.0	0.0	0	0	1
2	12	173	2	-6.0	39	1.0	2000	-17.7	0.0	0.0	0.0	0	0	1
3	12	107	3	-6.2	40	0.9	2000	-17.6	0.0	0.0	0.0	0	0	1
4	12	78	4	-6.0	36	2.3	2000	-18.6	0.0	0.0	0.0	0	0	1

我先針對 object 的資料進行轉換，將其字串轉換成數字。由上圖可以注意到我將 Date 進行修改只擷取其中的月份並改為名叫 Month 的欄位，接著我將 Seasons 從原本的 Winter 、 Spring 、 Summer 、 Autumn 改為數字 0～3。Holiday 、 Functioning Day 也依同樣做法改為 0 、 1。接著將欄位本身內部資料間不該具有權重之分的欄位屬性改為 category。

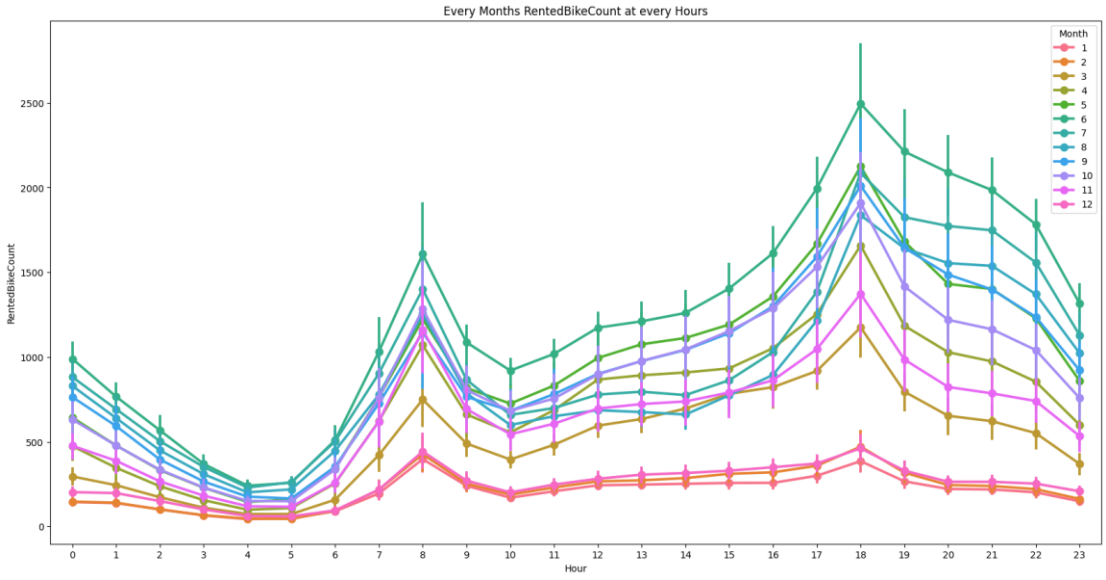
```
Month                category
RentedBikeCount         int64
Hour                 category
Temperature           float64
Humidity                int64
WindSpeed             float64
Visibility              int64
DewPointTemperature   float64
SolarRadiation        float64
Rainfall              float64
Snowfall              float64
Seasons              category
Holiday              category
FunctioningDay       category
```

確認更改完成後，接著要來確認資料內部是否有空值。

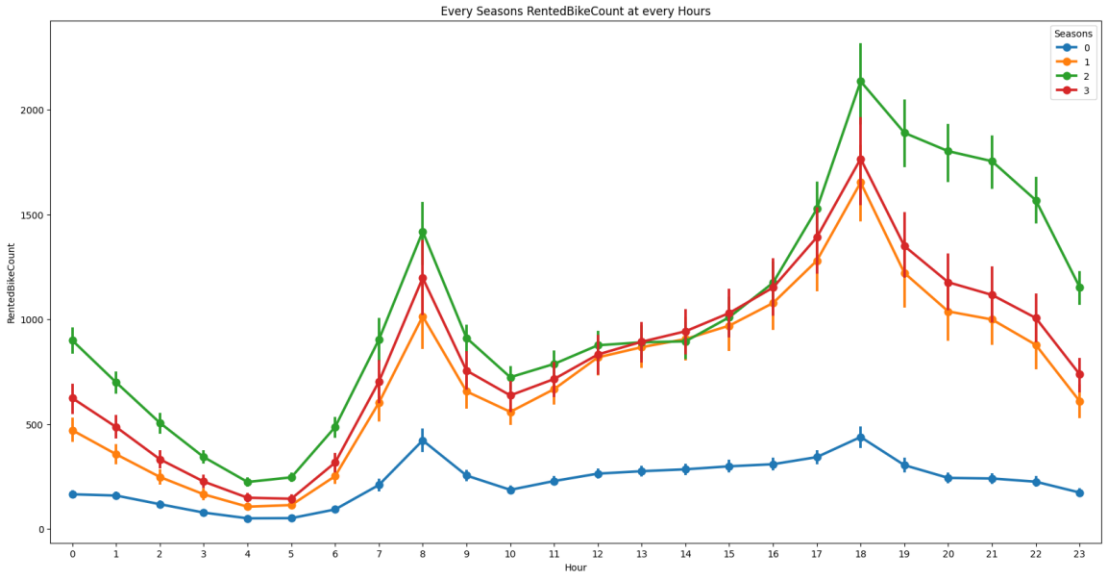
```
Month                  False
RentedBikeCount        False
Hour                   False
Temperature            False
Humidity               False
WindSpeed              False
Visibility             False
DewPointTemperature    False
SolarRadiation         False
Rainfall               False
Snowfall               False
Seasons                False
Holiday                False
FunctioningDay         False
```

4. 確認沒有空值後，接下來就是來利用 python 的 matplotlib、seaborn 函式庫的視覺化呈現需要的欄位的關係圖。

首先針對租借數量、時辰、月份來就行分析觀察

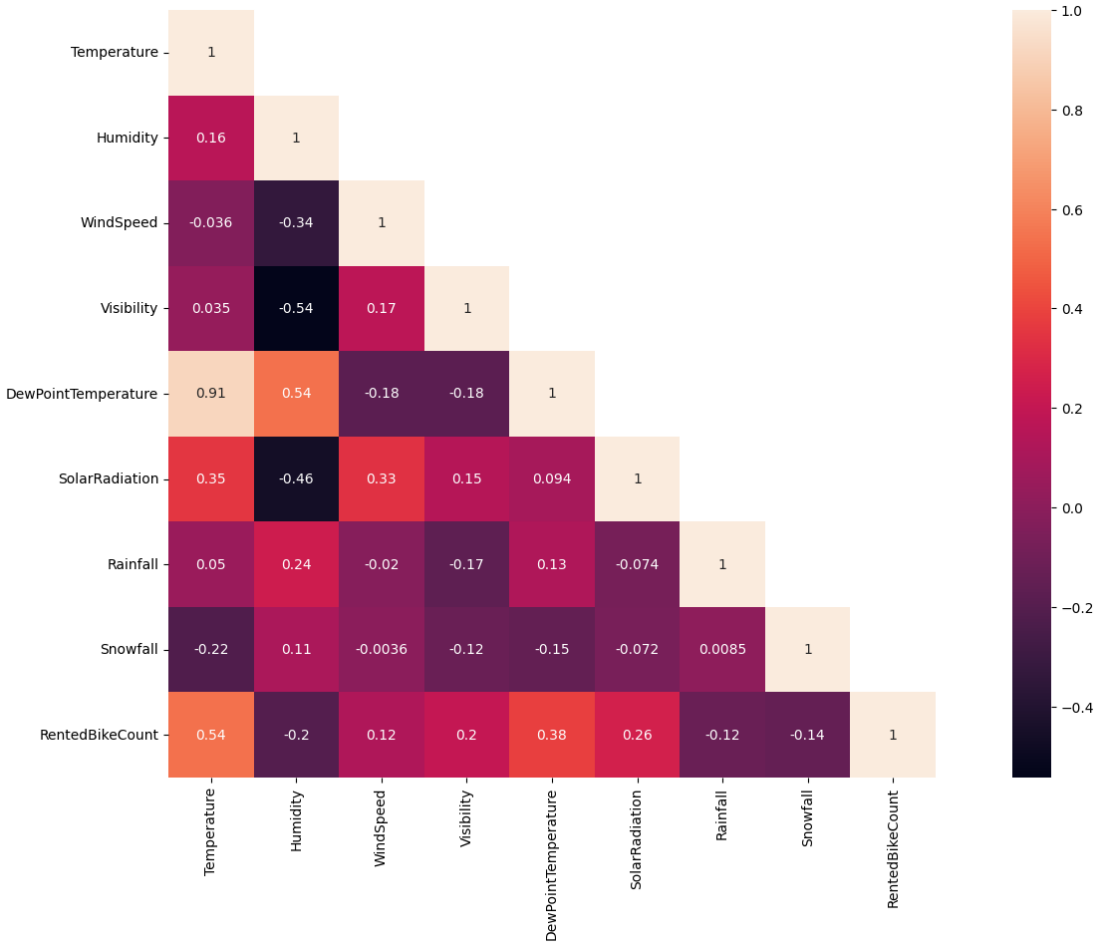


由上圖可以發現不管是哪個月分幾乎都是在接近每天的 8 點跟 18 點時，租借人潮都會呈現高峰。



把月份改為季節呈現也得到類似的樣貌，從這裡可以得知月分與季節在對於租借次數上的關聯性很高。

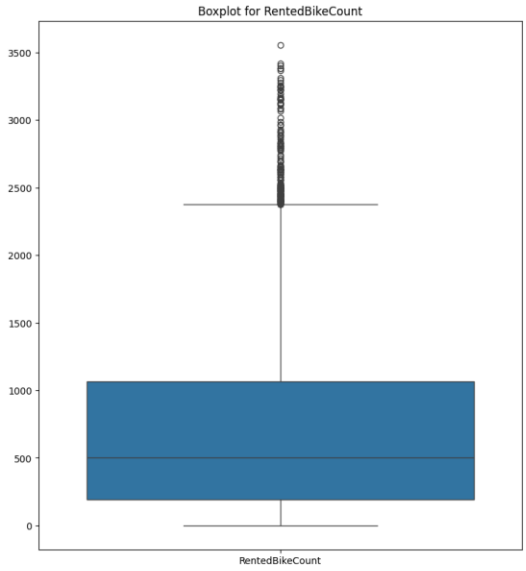
主要的欄位看完後，接著就利用 seaborn 函式庫的 heatmap()就來呈現剩餘的欄位的相關性，範圍是[-1,1]，絕對值越靠近 0，表示不相關，絕對值越靠近 1，表示相關性越強 小於 0 表示負相關；大於 0 表示正相關。

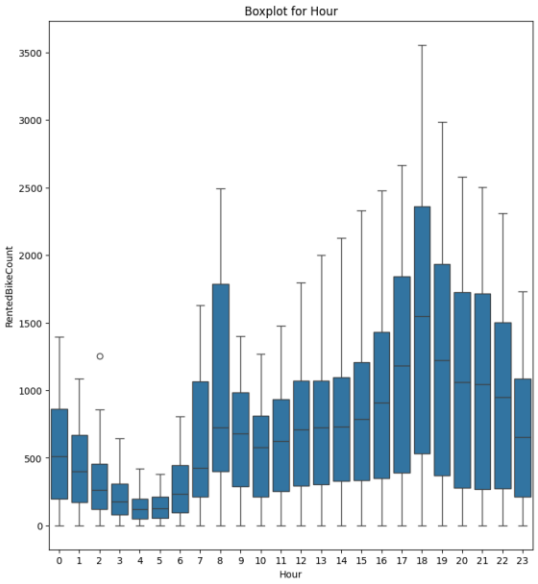


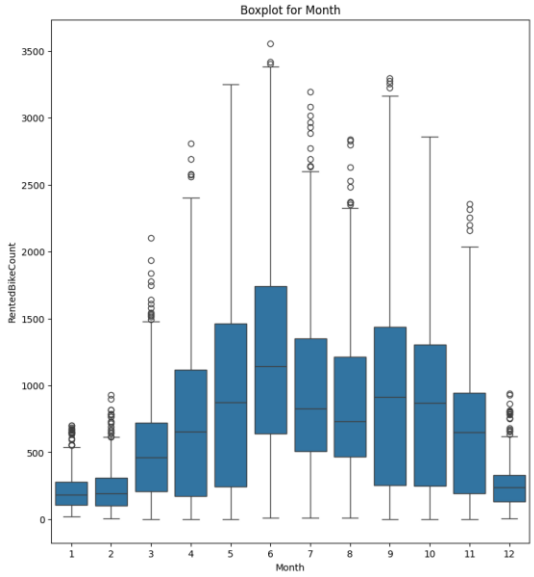
由上圖可以發現 Temperature 與 DewPointTemperature 的相關性極高，

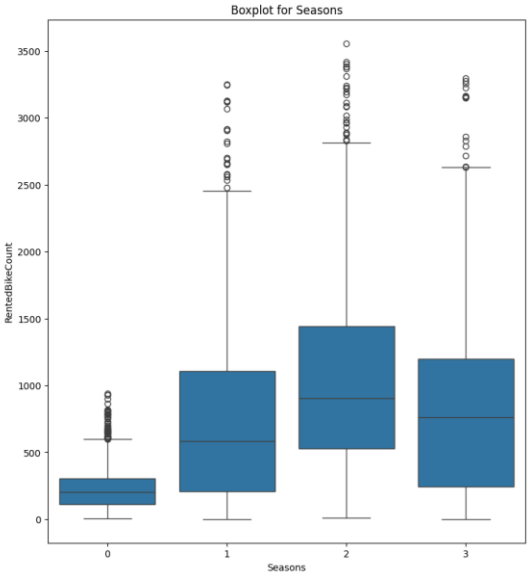
這樣一來可以考慮把 DewPointTemperature 替除掉。

接下來利用 seaborn 的 boxplot 來檢查異常值

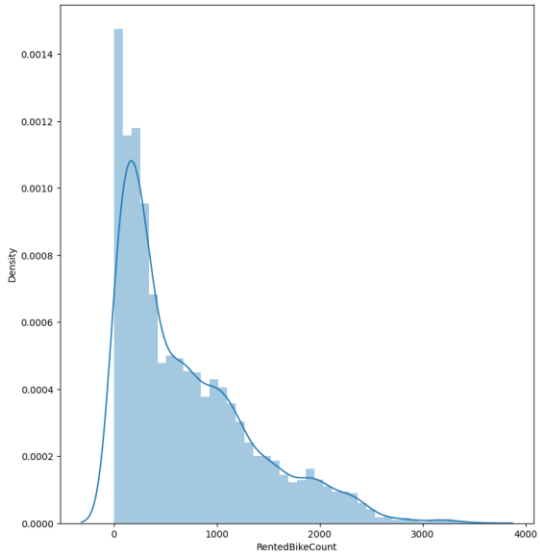


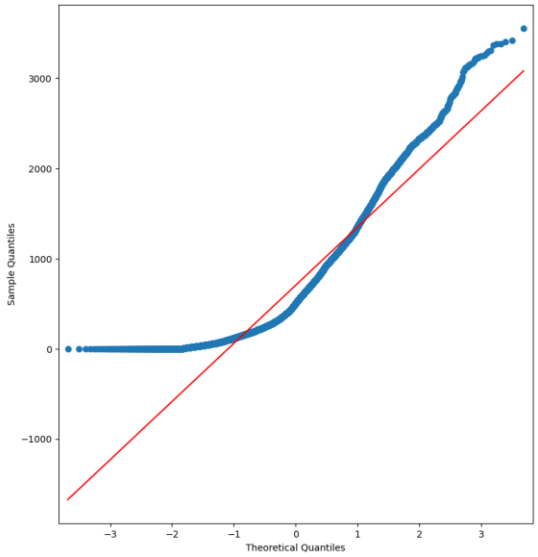




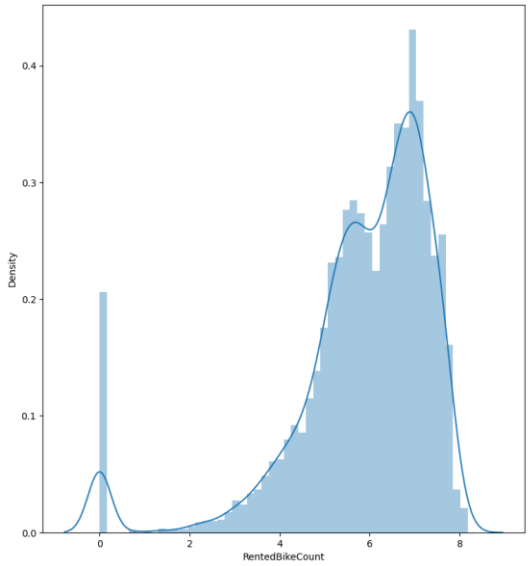


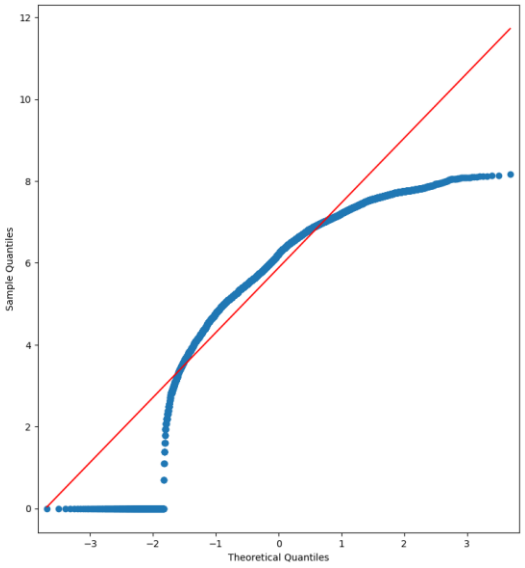
可以發現 RentedBikeCount 的圖結果有點不太好，那麼就來利用 python seaborn 的 distplot() 與 statesmodels 函式庫的、qqplot() 來看看 RentedBikeCount 是否呈現正態分布。





可以發現結果不太正常，所以要來利用 log 的方式把偏差範圍縮小固定。





這樣一來放進 model 時就會有比較好的結果。

接著就是進行將剛剛上面所提到的欄位本身內部資料相互之間不具有權重的欄位，利用 **One Hot Encoding** 熱編碼來實現。

	RentedBikeCount	Temperature	Humidity	Visibility	SolarRadiation	Rainfall	Month_2	Month_3	Month_4	Month_5	...	Hour_19	Hour_20	Hour_21	Hour_22	Hour_23	Seasons_1	Seasons_2	Season
0	5.537334	-5.2	37	2000	0.0	0.0	False	False	False	False	...	False	False	False	False	False	False	False	Fa
1	5.318120	-5.5	38	2000	0.0	0.0	False	False	False	False	...	False	False	False	False	False	False	False	Fa
2	5.153292	-6.0	39	2000	0.0	0.0	False	False	False	False	...	False	False	False	False	False	False	False	Fa
3	4.672829	-6.2	40	2000	0.0	0.0	False	False	False	False	...	False	False	False	False	False	False	False	Fa
4	4.356709	-6.0	36	2000	0.0	0.0	False	False	False	False	...	False	False	False	False	False	False	False	Fa

結果如上圖，可以看到他將需要更動的欄位都進行了轉換，轉變成了每個

行位的加總都為 1，具有相同的權重值類似標籤的功能。

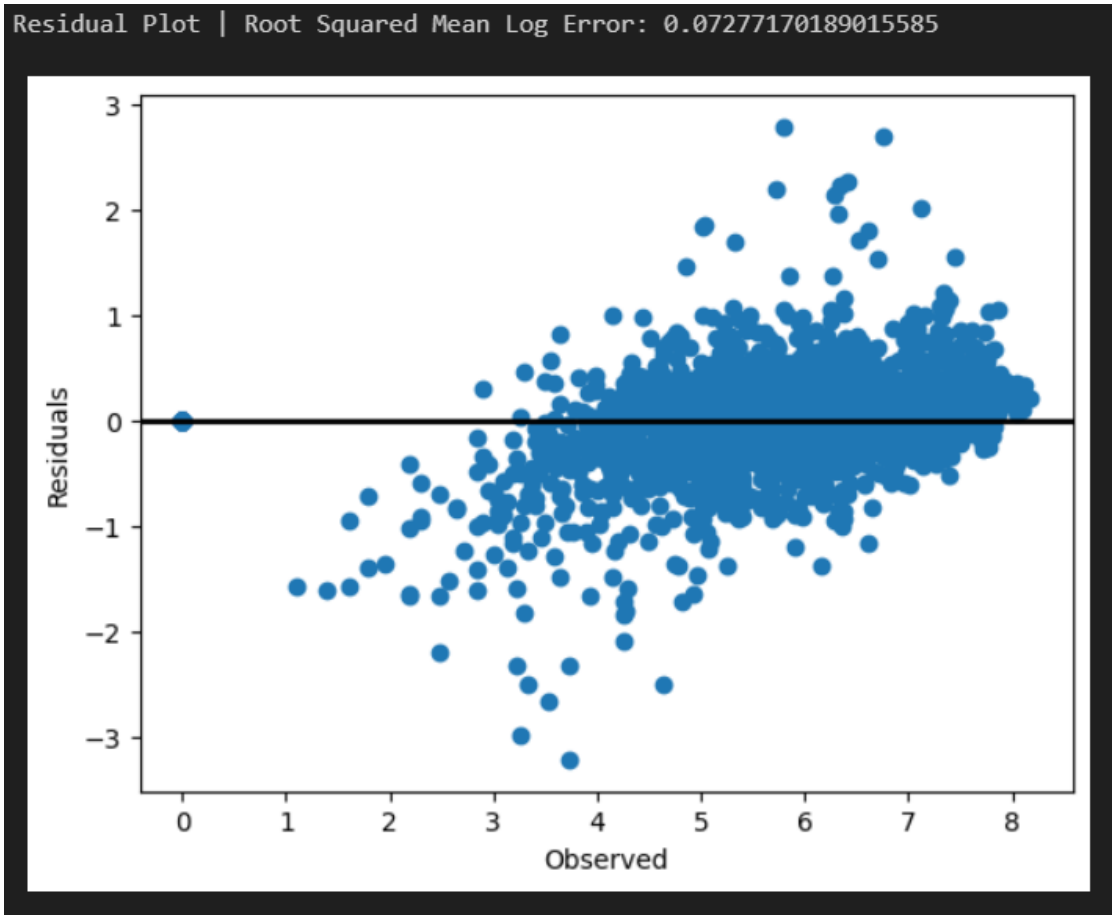
5. 接下來最後一部就是切割出 Train Data、Test Data 來進行多種 model 的比較並挑選出最好的結果來進行 Test。

```
models = [LinearRegression(),
          Ridge(),
          HuberRegressor(),
          ElasticNetCV(),
          DecisionTreeRegressor(),
          ExtraTreesRegressor(),
          GradientBoostingRegressor(),
          RandomForestRegressor(),
          BaggingRegressor()]
```

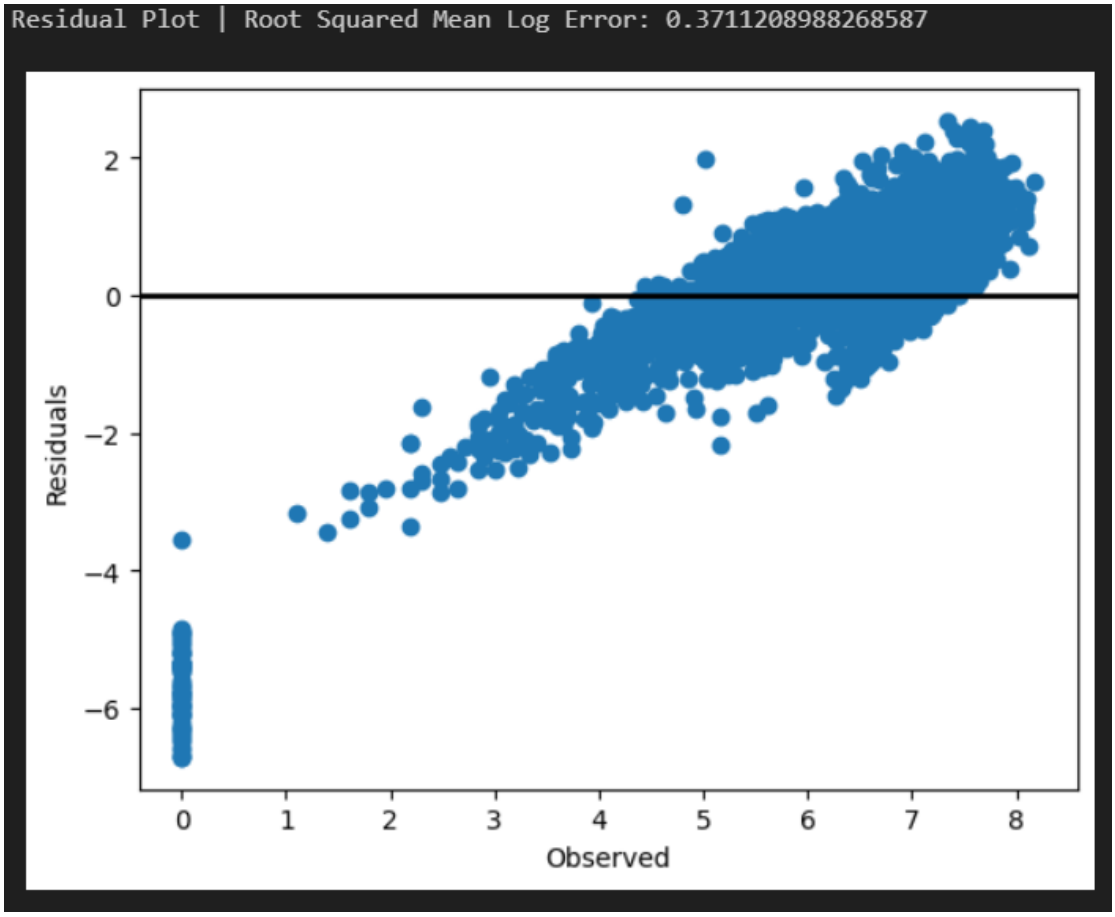
我使用了 sklearn 函式庫裡的這 9 種算法來分別得出各自結果

```
-0.3843184829230969
-0.3843332911663266
-1.2784259916449274
-1.9019786923545823
-0.3759636911840925
-0.19781719485833482
-0.2440645250203078
-0.1961060582143495
-0.21732543925567266
```

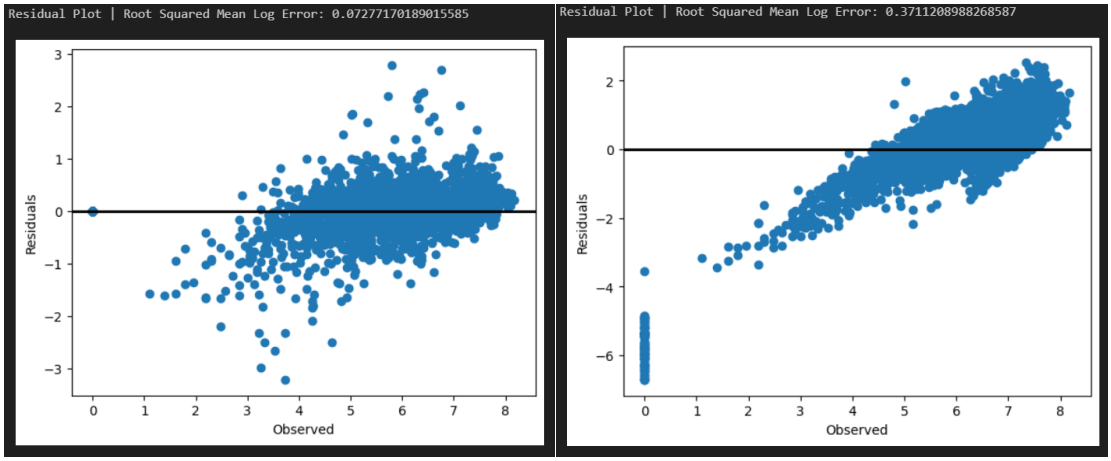
由結果我們可以發現 RandomForestRegressor 的性能是最好的，而相對的 ElasticNetCV 的性能是最差的。



上圖為 RandomForestRegressor 的結果圖，錯誤為 0.073。



上圖為 ElasticNetCV 的結果圖，錯誤為 0.37。



可以看到 RandomForestRegressor 與 ElasticNetCV 的結果差了約 0.3。

這樣一來就能利用此方式去預測出自行車量的狀況，並對其結果去考量設想應對方法，以避免無法處裡的突發狀況。

6. 這次的期末專案在實作方面上除了要知道對於資料預處理的知識方面外，其餘剩下的困難就只剩理解演算法及想法跟毅力了。

7. 對於這次的期末作業我覺得受益良多，受益的點在於能夠知曉自己的能力以及知識廣度。起初在得知老師並不會給予時做方面的範例時，說實話有點慌了手腳，不管怎麼查都不清楚到底該如何呈現，還曾經想過是否放棄。但是在多災多難過程中我意識到了現在在做的事情，正是我未來一定會用到的技能，所以我努力堅持到了最後。雖然不知道結果是否符合老師與助教的期望，但不論結果如何，我認為經過這次的磨練我一定能變得更好，並且永不畏懼未來的挑戰。