機器學習

第 6 章 線性迴歸成立前提
(Regression Assumptions)

講師：紀俊男

本章大綱

- **樣本點呈線性（Linearity）**
  - 「自變數 X_i 」對「應變數 Y」 的分佈是「線型的」
- **殘差呈常態性（Normality）**
  - 所有殘差（Y_i-Y_h, Residuals）會呈現「常態分佈」
- **殘差獨立性（Independency）**
  - 各殘差之間獨立，前一個殘差不會對下一個殘差造成影響
- **殘差等分散性（Homoscedasticity）**
  - 各殘差的變異數都差不多，沒有離群值
- **自變數無共線性（Non-Multicollinearity）**
  - 自變數之間各自獨立，沒有任何自變數、是另一個自變數的線性組合

樣本點呈線性

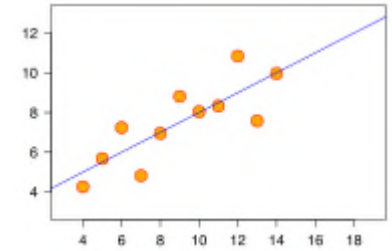
Linearity

何謂「樣本點呈線性（Linearity）」

- **自變數 X_i** 對**應變數 Y** 的分佈是「**線型的**」

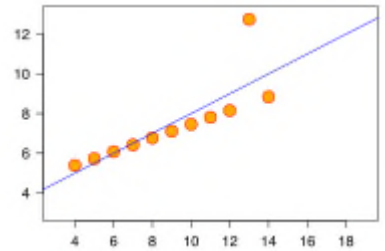
$$y = c_0 + c_1 x_1 + \cdots c_n x_n$$

y



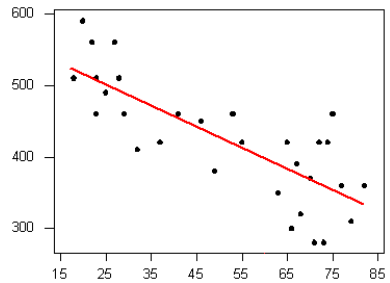
x_1

y



x_2

y



x_n

實作「樣本點呈線性（Linearity）」之檢查

- 原始程式碼

套件引入

類別成員

建構函數

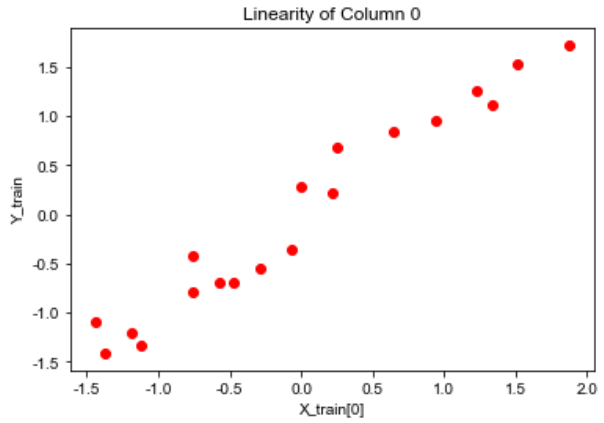
自變數繪製

```python
import matplotlib.pyplot as plt
import scipy.stats as stats
from pandas.plotting import autocorrelation_plot
import pandas as pd
import numpy as np

class AssumptionChecker:
    __x_train = None
    __x_test = None
    __y_train = None
    __y_test = None
    __y_pred = None
    __residuals = None

    def __init__(self, x_train, x_test, y_train, y_test, y_pred):
        self.__x_train = x_train
        self.__x_test = x_test
        self.__y_train = y_train
        self.__y_test = y_test

        self.__y_pred = y_pred
        self.__residuals = (self.__y_test.ravel() - self.__y_pred.ravel())

    def sample_linearity(self):
        print("*** Check for Linearity of Independent to Dependent Variable ***")

        for i in range(self.__x_train.shape[1]):
            plt.scatter(self.__x_train[:, i], self.__y_train, color="red")
            plt.title("Linearity of Column {}".format(i))
            plt.xlabel("X_train[{}]".format(i))
            plt.ylabel("Y_train")
            plt.show()
```

- 呼叫範例

```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.sample_linearity()
```

- 執行結果



隨堂練習：樣本點呈線性

- 請先瀏覽、並理解講師提供之 AssumptionChecker 類別內的 sample_linearity() 函數。

- 使用下列程式碼呼叫之，並確認樣本點呈線性：

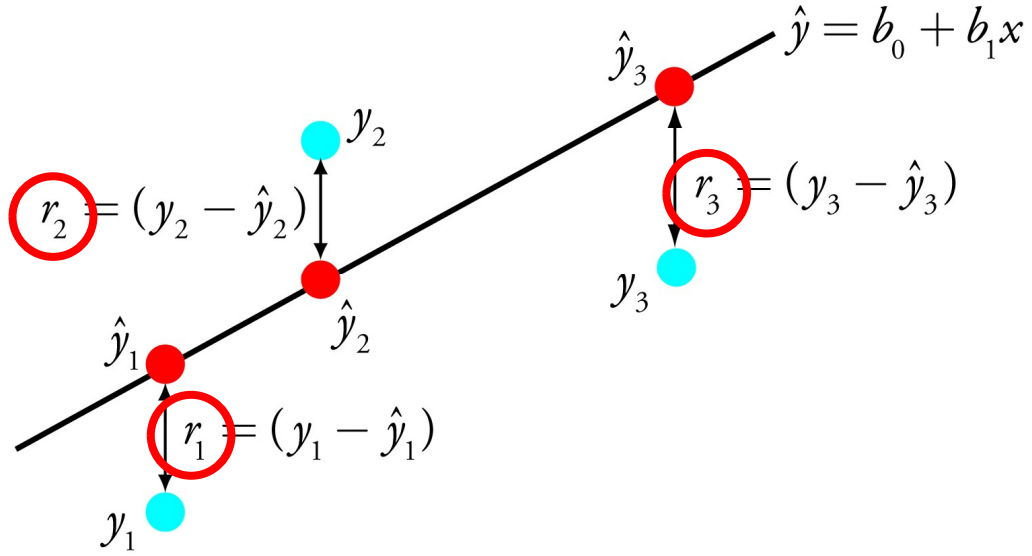
```
1  from HappyML.criteria import AssumptionChecker
2
3  checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
4  checker.sample_linearity()
```

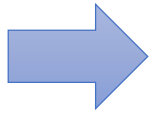
殘差呈常態性

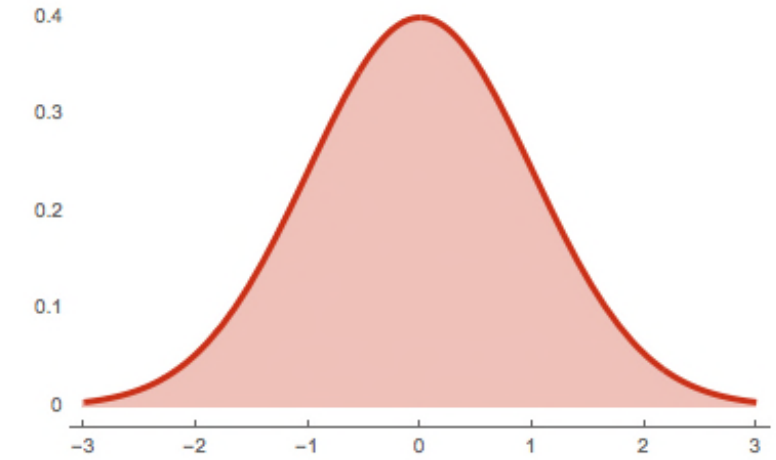
Normality

何謂「殘差呈常態性（Normality）」

- 將所有的**殘差（$R_i=Y_i-Y_h$）**排列起來，會呈**常態分佈**

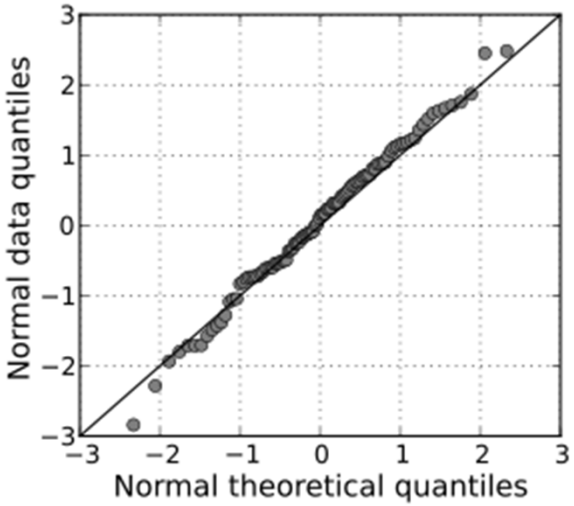




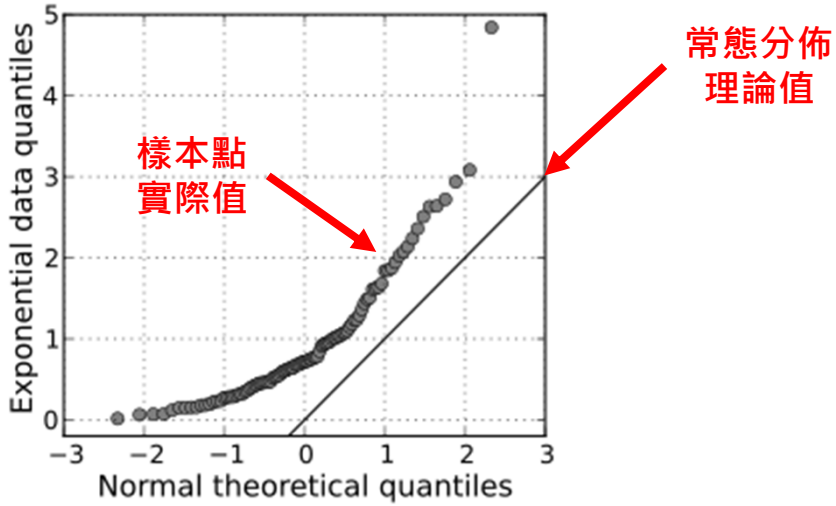


如何檢查「殘差呈常態性（Normality）」

- 利用「**分位圖（ Quantile-Quantile Plot，QQ Plot ）**」
  - 把兩個分佈的各個「**分位數（Quantile）**」排列比較，藉以看兩者分佈的相似性。



兩者分佈相似



兩者分佈不同

實作「殘差呈常態性（Normality）」之檢查

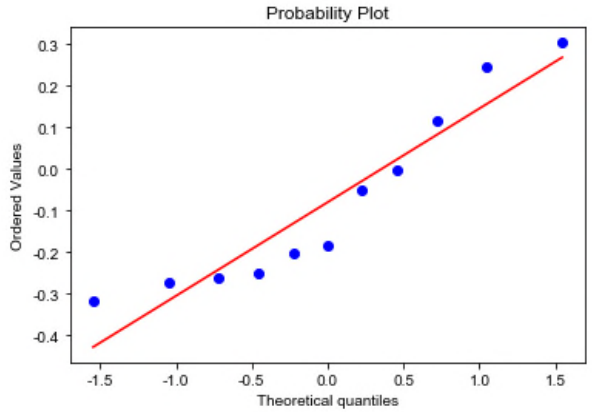
- 原始程式碼

```python
import scipy.stats as stats
import matplotlib.pyplot as plt

def residuals_normality(self):
    print("*** Check for Normality of Residuals ***")

    stats.probplot(self.__residuals, plot=plt)
    plt.show()
```

- 呼叫範例

```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.sample_linearity()

checker.residuals_normality()
```

- 執行結果



- **scipy.stats.probplot(殘差陣列, dist="norm", plot=plt)**
  - 殘差陣列：傳入一個 Y_i-Y_h 的 NDArray
  - **dist="norm"**：對比用、理論機率分佈模型（預設為「常態分佈（Normal Distribution）」）
  - **plot=plt**：想使用的繪圖函式庫

隨堂練習：殘差呈常態性

- 請先瀏覽、並理解講師提供之 AssumptionChecker 類別內的 residuals_normality() 函數。

- 使用下列程式碼呼叫之，並確認殘差呈常態性：

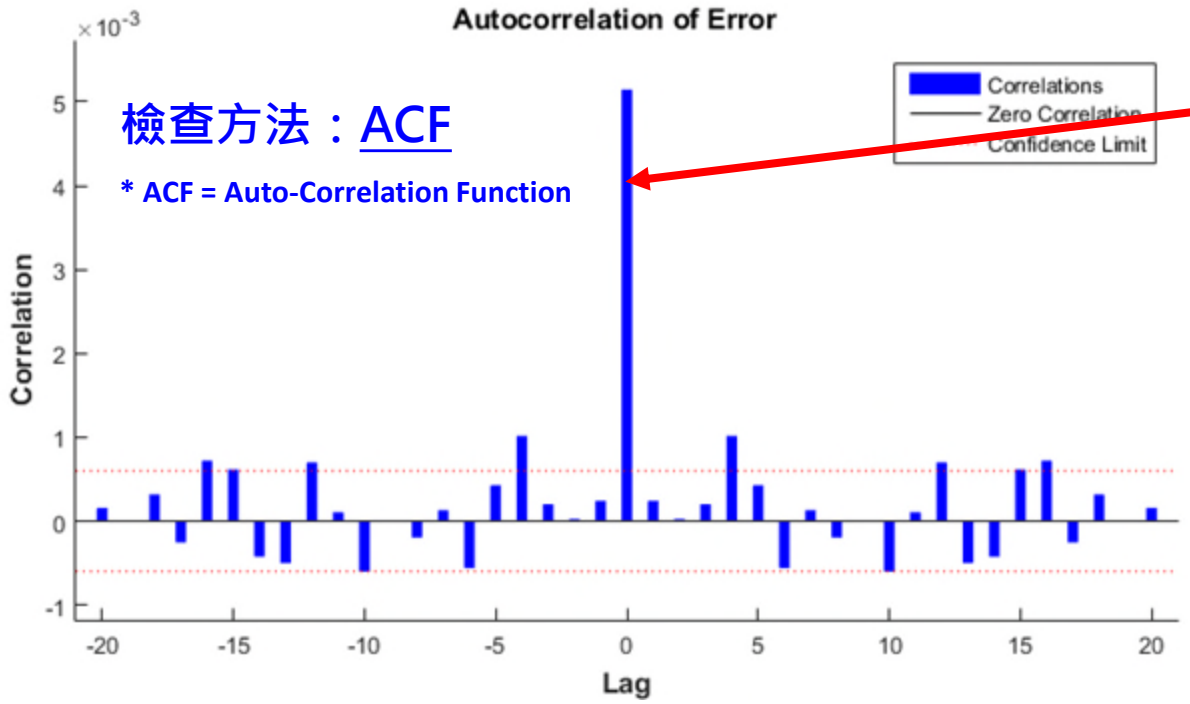
```
6   checker.residuals_normality()
```

殘差獨立性

Independency

何謂「殘差獨立性（Independency）」

- 前一個**誤差**，不會影響到下一個**誤差**，使之**擴大**或**縮小**
  - 反例：股票前一天的**漲跌**，會影響下一天的**漲跌** --> 非**獨立性**！

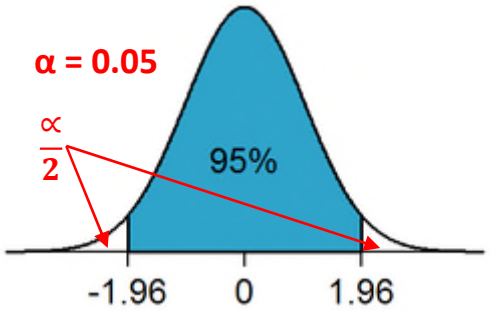


$$ACF = \frac{\frac{1}{N}\sum_{t=1}^{N-h}(Y_t - \bar{Y})(Y_{t+h} - \bar{Y})}{\frac{1}{N}\sum_{t=1}^{N-h}(Y_t - \bar{Y})^2}$$

不精準的白話文 $=$ $\frac{\text{自己跟別人的離散程度}}{\text{自己跟自己的離散程度}}$

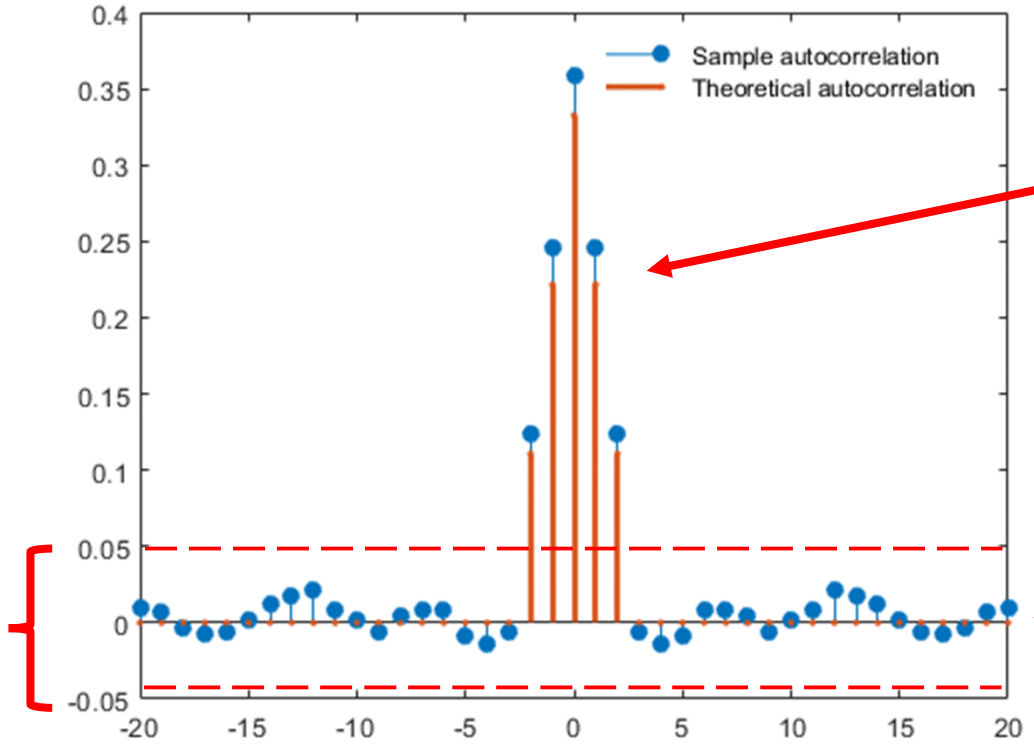
$$B = \pm \frac{Z_{1-\frac{\propto}{2}}}{\sqrt{N}}$$

Z：標準分佈機率密度函數
α：顯著水準
N：樣本數



如何檢查「殘差獨立性（Independency）」

- 使用 Auto-Correlation Plot 可以判斷殘差有無**獨立性**



$$B = \pm \frac{Z_{1-\frac{\propto}{2}}}{\sqrt{N}}$$

無獨立性

有獨立性

實作「殘差獨立性（Independency）」之檢查

- 原始程式碼

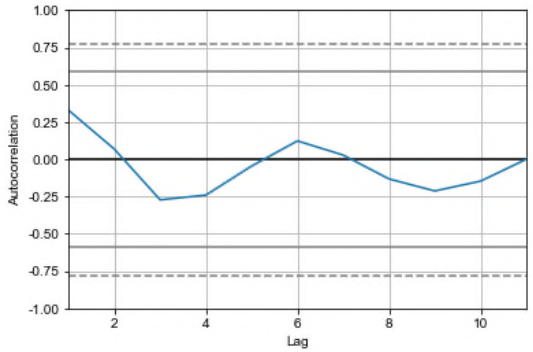
```python
import pandas as pd
from pandas.plotting import autocorrelation_plot

def residuals_independence(self):
    print("*** Check for Independence of Residuals ***")

    df_res = pd.DataFrame(self.__residuals)
    autocorrelation_plot(df_res)
    plt.show()
```

- pandas.plotting.**autocorrelation_plot**(殘差陣列)
  - 能夠繪製出 Auto-Correlation Function 圖
  - 殘差陣列必須是 DataFrame 型態

- 呼叫範例

```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.sample_linearity()
checker.residuals_normality()

checker.residuals_independence()
```

- 執行結果



隨堂練習：殘差獨立性

- 請先瀏覽、並理解講師提供之 **AssumptionChecker** 類別內的 residuals_independence() 函數。

- 使用下列程式碼呼叫之，並確認殘差的獨立性：

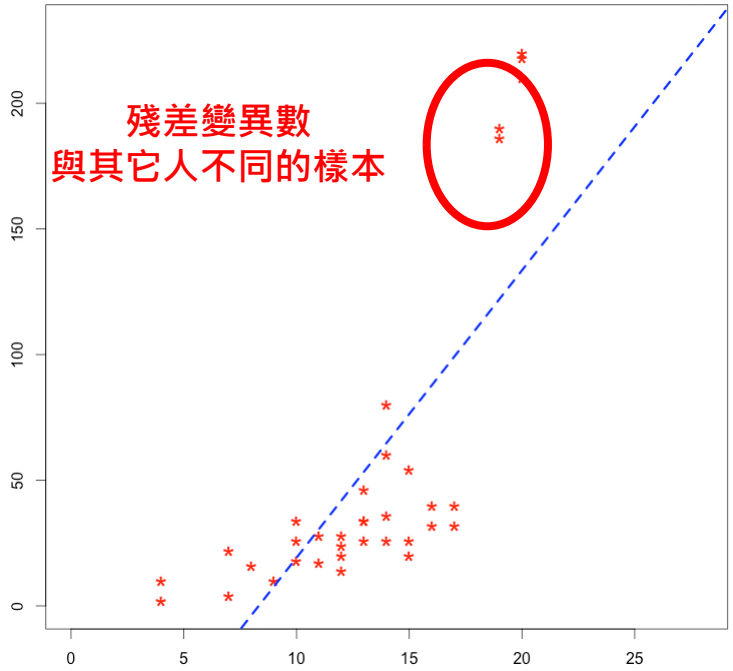
```
7   checker.residuals_independence()
```

残差等分散性

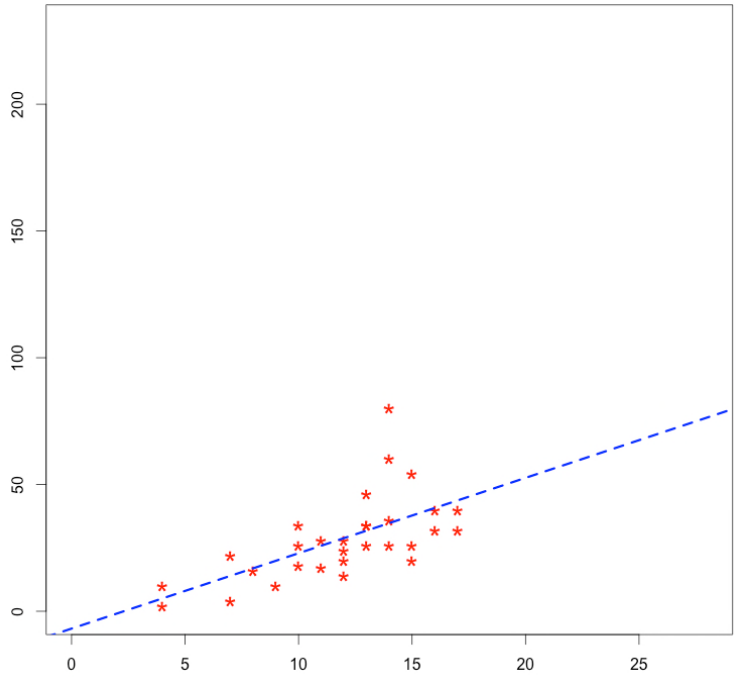
Homoscedasticity

何謂「殘差等分散性（Homoscedasticity）」

- 各殘差的**變異數**都差不多，沒有離群值（Outliers）



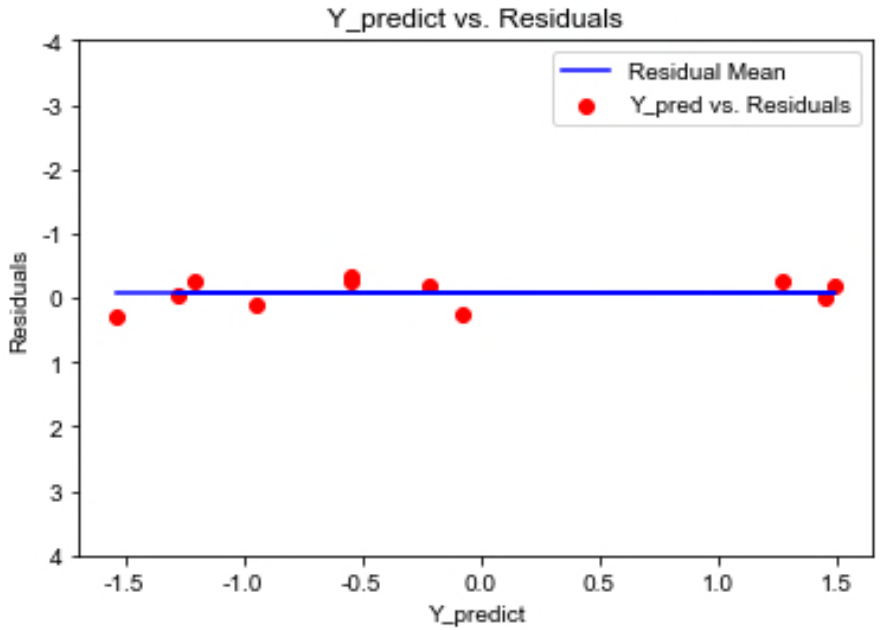
因離群值導致
迴歸結果不準確



假設沒有那些離群值
迴歸結果就會準確多了

如何檢查「殘差等分散性（Homoscedasticity）」

- 判斷方法
  - 用預測值 Y_h 針對殘差 Y_i-Y_h 作畫
  - 若沒有離群值，各點殘差 Y_i-Y_h 會圍繞在殘差的平均值 Y_u 那條水平線附近
  - 若經過特徵縮放的正規化後， Y_u 應該很靠近 0



實作「殘差等分散性（Homoscedasticity）」之檢查

- 原始程式碼

樣本點繪製

平均

圖標

座標範圍

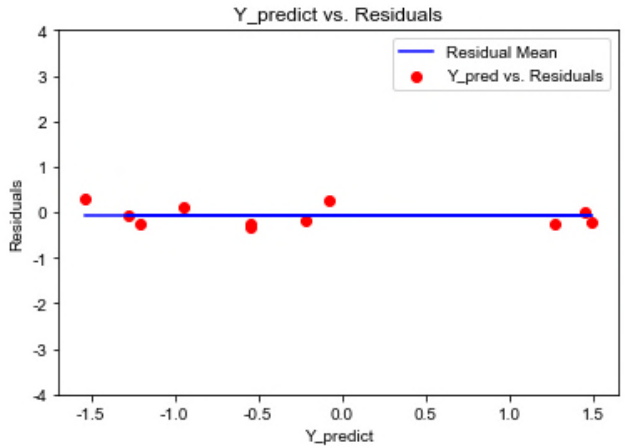
```python
import matplotlib.pyplot as plt
import numpy as np

def residuals_homoscedasticity(self, x_lim=None, y_lim=None):
    print("*** Check for Homoscedasticity of Residuals ***")

    plt.scatter(self.__y_pred, self.__residuals, color="red", label="Y_pred vs. Residuals")

    dimension = self.__y_pred.shape[0]
    residual_mean = self.__residuals.mean()
    plt.plot(self.__y_pred, np.full(dimension, residual_mean), color="blue", label="Residual Mean")

    plt.title("Y_predict vs. Residuals")
    plt.xlabel("Y_predict")
    plt.ylabel("Residuals")
    plt.legend(loc="best")
    if x_lim != None:
        plt.xlim(x_lim)
    if y_lim != None:
        plt.ylim(y_lim)
    plt.show()
```

實作「殘差等分散性（Homoscedasticity）」之檢查

- 呼叫範例

```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.sample_linearity()
checker.residuals_normality()
checker.residuals_independence()

checker.residuals_homoscedasticity(y_lim=(-4, 4))
```

- 執行結果



檢查重點

- 殘差是否圍繞在一條水平線上
- 若特徵縮放過，水平線（Y 平均）是否靠近 0

隨堂練習：殘差等分散性

- 請先瀏覽、並理解講師提供之 **AssumptionChecker** 類別內的 **residuals_homoscedasticity**() 函數。

- 使用下列程式碼呼叫之，並確認殘差的**等分散性**：
  （**等分散性** = 殘差是否距離**平均值**水平線不遠，亦即，**離散**相當）

```
1   checker.residuals_homoscedasticity(y_lim=(-4, 4))
```

自變數
無共線性

Non-Multicollinearity

何謂「自變數無共線性（Non-Multicollinearity）」

- **自變數**之間各自**獨立**，沒有任何**自變數**、是另一個**自變數**的**線性組合**
  （註：單變數的簡單迴歸不需滿足這條）

判別方法：相關矩陣（**Correlation Matrix**）

特徵（Features）0 - 4

特徵（Features）0 - 4

	0	1	2	3	4
0	1.000000	-0.509175	0.150194	0.122271	0.109855
1	-0.509175	1.000000	0.027107	-0.038313	0.069360
2	0.150194	0.027107	1.000000	0.303119	0.771734
3	0.122271	-0.038313	0.303119	1.000000	0.029133
4	0.109855	0.069360	0.771734	0.029133	1.000000

判斷標準：$|\rho_{x,y}| >= 0.8$ --> 有「共線性」

相關係數

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y}$$

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\,\sqrt{E(Y^2) - E^2(Y)}}$$

ρ：相關係數（-1 負相關～ 1 正相關）

X, Y：樣本點

cov()：共變異數

σ：標準差

E()：期望值

μ：平均值

實作「自變數無共線性（Non-Multicollinearity）」之檢查

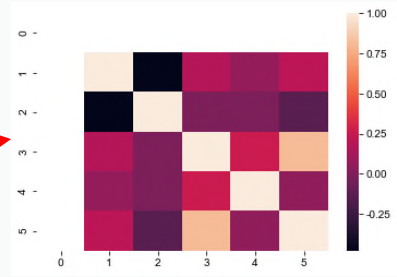
- 原始程式碼

```python
import pandas as pd
import seaborn as sns

def features_correlation(self, heatmap=False):
    print("*** Check for Correlation of Features ***")

    df = pd.DataFrame(self.__x_train)
    corr = df.corr()
    print("--- Features Correlation Matrix ---")
    print(corr)
    if heatmap:
        sns.heatmap(corr)

    corr_ary = corr.to_numpy()
    corr_bool = False
    for i in range(corr_ary.shape[0]):
        for j in range(corr_ary.shape[1]):
            if i != j:
                if corr_ary[i, j] >= 0.8:
                    corr_bool = True
                    print("Correlation Found at[{}, {}] = {}".format(i, j, corr_ary[i, j]))
    if not corr_bool:
        print("No Correlation (>=0.8) Found!")
```

很棒的「統計視覺化套件」

印出「相關矩陣」

印出「熱區圖」



檢查是否 >=0.8

對角線不予計算

實作「自變數無共線性（Non-Multicollinearity）」之檢查

- 呼叫範例

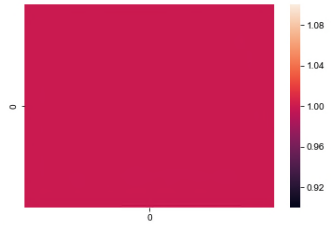
```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.sample_linearity()
checker.residuals_normality()
checker.residuals_independence()
checker.residuals_homoscedasticity(y_lim=(-4, 4))

checker.features_correlation()
```

- 執行結果

```python
checker.features_correlation(heatmap=True)
```

```
--- Features Correlation Matrix ---
     0
0  1.0
No Correlation (>=0.8) Found!
```

相關矩陣（Correlation Matrix）



隨堂練習：自變數無共線性

- 請先瀏覽、並理解講師提供之 AssumptionChecker 類別內的 features_correlation() 函數。

- 使用下列程式碼呼叫之，看看「**相關矩陣**」中，除了對角線，是否相關係數都小於 0.8？

```
1   checker.features_correlation()
```

AI

一口氣執行所有檢查

Check All Criteria

執行所有檢查的方法

- 原始程式碼

```python
class AssumptionChecker:
    ......
    def check_all(self):
        self.sample_linearity()
        self.residuals_normality()
        self.residuals_independence()
        self.residuals_homoscedasticity()
        self.features_correlation()
```

- 呼叫範例

```python
from HappyML.criteria import AssumptionChecker

checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
checker.check_all()
```

隨堂練習：一口氣執行所有檢查

- 請先瀏覽、並理解講師提供之 **AssumptionChecker** 類別內的 check_all() 函數。
- 將原先五種檢查的程式碼註解掉。
- 把程式碼改成下面的樣子，看看是否能一口氣執行五種檢查？

```
1  from HappyML.criteria import AssumptionChecker
2
3  checker = AssumptionChecker(X_train, X_test, Y_train, Y_test, Y_pred)
4  checker.check_all()
```

重點整理

- 影響線性迴歸的五個因素
  - 樣本點呈線性（Linearity）
  - 殘差呈常態性（Normality）
  - 殘差獨立性（Independency）
  - 殘差等分散性（Homoscedasticity）
  - 自變數無共線性（Non-Multicollinearity）

- 何時要檢查線性迴歸五大前提
  - 想找出是哪個因素影響擬合準確率時。
  - 證明自己套用「線性迴歸」師出有名時（如：論文撰寫）。

- 如何實作線性迴歸五大前提之檢查
  - 套件引入：**HappyML**.criteria.AssumptionChecker
  - 樣本線性：.sample_linearity()
  - 殘差常態：.residuals_normality()
  - 殘差獨立：.residuals_independence()
  - 等分散性：.residuals_homoscedasticity()
  - 無共線性：.features_correlation()
  - 全部檢查：.check_all()