

📄 > ChatGPT 5.2 Thinking ▾

一、作業目標：

本次作業延伸自 HW1，目標是讓同學能夠以實際資料集進行「多元線性回歸 (Multiple Linear Regression)」的完整分析，並遵循 CRISP-DM 流程完成從資料理解、建模到評估的全過程。

二、作業內容：

1. 資料來源

至 Kaggle 選擇一個具有 10 至 20 個特徵 (features) 的公開資料集。
類型不限 (可為房價預測、醫療、車輛效能等主題)。
請明確標示資料集來源與連結。

2. 分析任務

使用線性回歸 (Linear Regression) 模型進行預測。
可嘗試單純線性回歸、多元線性回歸或 Auto Regression。
必須執行 特徵選擇 (Feature Selection) 與 模型評估 (Model Evaluation)。
結果部分需包含請提供預測圖(加上信賴區間或預測區間)

3. CRISP-DM 流程說明

Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
Deployment

4. AI協助要求

所有與 ChatGPT 的對話請以 pdfCrowd 或其他方式須匯出為 PDF
請使用 NotebookLM 對網路上同主題的解法進行研究，並撰寫一份 100 字以上的摘要，放入報告中。
請在報告中明確標示「GPT 輔助內容」與「NotebookLM 摘要」

5. 繳交內容

主程式：7114056XXX_hw2.py/.ipynb
報告檔：PDF，需包含以下內容：
按照 CRISP-DM 說明的分析流程
GPT 對話過程 (pdfCrowd 匯出)
NotebookLM 研究摘要
網路上主流或更優解法之比較與說明
以上檔案與資料夾請壓縮為學號命名的一個zip (例如 7114056XXX_hw2.zip) 上傳。

(optional) 若上傳至 GitHub，或是以 colab 撰寫，需提供連結，並在 README.md 中整理流程與成果。

三、評分標準

文件說明 (50%)

CRISP-DM 流程完整且邏輯清楚 (25%)
包含 GPT 對話與 NotebookLM 摘要 (15%)
有明確說明資料集來源與研究脈絡 (10%)

結果呈現 (50%)

模型正確可執行，具特徵選擇與評估 (25%)
結果合理、美觀且具有說服力 (15%)
呈現出 Kaggle 名次(若有)/預測結果評估(預測圖、評估指標) (10%)

已思考 1m 25s >

下面我直接給你一套「可照抄就能完成作業」的方案 (含：資料集選擇、CRISP-DM 報告架構、特徵選擇、評估、以及預測圖 + 信賴區間/預測區間)，你只要把學號改掉、把資料放進資料夾就能跑。

1) Kaggle 資料集建議（符合 10–20 features）

推薦：Red Wine Quality（11 個特徵）

Kaggle 連結：Red Wine Quality（Cortez et al. 2009）[kaggle.com](https://www.kaggle.com)

它本身就是經典「多元線性回歸」範例（11 個化學特徵 → 預測品質分數 quality）。UCI 也標示特徵數為 11。

archive.ics.uci.edu

2) 你要交的資料夾/檔名（照這個做）

建議專案長這樣（最後 zip 成 7114056XXX_hw2.zip）：

kotlin

複製程式碼

```
7114056XXX_hw2/  
  7114056XXX_hw2.ipynb    (或 .py)  
  report.pdf              (CRISP-DM + 圖表 + 指標 + 比較 + 標示GPT/NotebookLM)  
  gpt_dialog.pdf          (pdfCrowd 匯出的 ChatGPT 對話)  
  data/  
    winequality-red.csv  
  figures/  
    pred_interval.png
```

3) 主程式（可直接存成 7114056XXX_hw2.py）

特色：

- ✓ Multiple Linear Regression：(1)
- ✓ Feature Selection：LassoCV（自動挑特徵）
- ✓ Model Evaluation：RMSE/MAE/R²
- ✓ 預測圖：同時畫 **confidence interval (mean_ci)** 與 **prediction interval (obs_ci)**（用

statsmodels `get_prediction().summary_frame()`）statsmodels.org +1

python

複製程式碼

```
# 7114056XXX_hw2.py  
# Multiple Linear Regression + Feature Selection + Prediction Interval Plot (CRISP-DM)  
  
from pathlib import Path  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
  
from sklearn.model_selection import train_test_split  
from sklearn.pipeline import Pipeline  
from sklearn.preprocessing import StandardScaler  
from sklearn.linear_model import LassoCV  
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score  
  
import statsmodels.api as sm  
  
# -----  
# 0. Config  
# -----  
DATA_PATH = Path("data/winequality-red.csv")  
FIG_DIR = Path("figures")  
FIG_DIR.mkdir(exist_ok=True)  
  
RANDOM_STATE = 42
```

```

TEST_SIZE = 0.2
ALPHA = 0.05 # 95% intervals

# -----
# 1. Load data
# -----
df = pd.read_csv(DATA_PATH)

# Target / Features
target = "quality"
X = df.drop(columns=[target]).copy()
y = df[target].copy()

print("Shape:", df.shape)
print("Columns:", df.columns.tolist())
print("Missing values:\n", df.isna().sum())

# -----
# 2. Split
# -----
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=TEST_SIZE, random_state=RANDOM_STATE
)

# -----
# 3. Feature Selection (LassoCV)
# -----
selector = Pipeline(
    steps=[
        ("scaler", StandardScaler()),
        ("lasso", LassoCV(cv=5, random_state=RANDOM_STATE, max_iter=20000)),
    ]
)
selector.fit(X_train, y_train)

coef = selector.named_steps["lasso"].coef_
selected_features = X_train.columns[coef != 0].tolist()

# 若 Lasso 太嚴格導致全為 0，至少保留全部特徵避免空模型
if len(selected_features) == 0:
    selected_features = X_train.columns.tolist()

print("\nSelected features (Lasso):", selected_features)
print("Num selected:", len(selected_features))

# -----
# 4. Modeling (OLS)
#  $y = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p + \varepsilon$  (1)
# -----
X_train_sel = sm.add_constant(X_train[selected_features])
X_test_sel = sm.add_constant(X_test[selected_features], has_constant="add")

ols_model = sm.OLS(y_train, X_train_sel).fit()
print("\nOLS summary (truncated):")
print(ols_model.summary())

# -----
# 5. Prediction + Intervals
# mean_ci_* : confidence interval for  $E[y|x]$ 
# obs_ci_* : prediction interval for new observation

```

```
# -----
pred = ols_model.get_prediction(X_test_sel)
pred_sf = pred.summary_frame(alpha=ALPHA)

y_pred = pred_sf["mean"].values

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("\nEvaluation on test:")
print(f"RMSE = {rmse:.4f}")
print(f"MAE = {mae:.4f}")
print(f"R^2 = {r2:.4f}")

# -----
# 6. Plot: prediction with intervals
# -----
# Sort by true y for nicer visualization
order = np.argsort(y_test.values)
y_true_sorted = y_test.values[order]
y_mean_sorted = pred_sf["mean"].values[order]

mean_lo = pred_sf["mean_ci_lower"].values[order]
mean_hi = pred_sf["mean_ci_upper"].values[order]
obs_lo = pred_sf["obs_ci_lower"].values[order]
obs_hi = pred_sf["obs_ci_upper"].values[order]

x_axis = np.arange(len(y_true_sorted))

plt.figure(figsize=(10, 6))
plt.plot(x_axis, y_true_sorted, marker="o", linestyle="None", markersize=3, label="True")
plt.plot(x_axis, y_mean_sorted, linewidth=2, label="Predicted mean")

plt.fill_between(x_axis, mean_lo, mean_hi, alpha=0.25, label="95% Confidence Interval (")
plt.fill_between(x_axis, obs_lo, obs_hi, alpha=0.15, label="95% Prediction Interval (of

plt.title("Prediction with Confidence & Prediction Intervals (OLS)")
plt.xlabel("Test samples (sorted by true y)")
plt.ylabel("quality")
plt.legend()
plt.tight_layout()

out_fig = FIG_DIR / "pred_interval.png"
plt.savefig(out_fig, dpi=200)
print("\nSaved figure:", out_fig)
```

4) 報告 (PDF) 建議架構：直接照 CRISP-DM 寫

CRISP-DM 六階段可直接照這樣排（你報告就會很完整）：

CRISP-DM 六階段定義可引用這類整理（六步驟名稱一致）。 Data Science PM +1

1. **Business Understanding**：為什麼要預測 wine quality？（品質評分可協助品管/配方調整）
2. **Data Understanding**：資料來源（Kaggle 連結）、筆數、特徵數、目標欄位、基本統計/缺失值
3. **Data Preparation**：切 train/test、標準化（給 Lasso 用）、必要的清理
4. **Modeling**：
 - Baseline：全特徵 OLS
 - Feature Selection：LassoCV 選出特徵（列出選到哪些）

- Final：用選後特徵做 OLS

5. **Evaluation**：RMSE/MAE/ R^2 + 殘差討論 + 預測圖（含區間）

6. **Deployment**：如何把模型封裝成可重跑流程（.py/.ipynb + README，或 GitHub）

5) 「GPT 輔助內容」與「NotebookLM 摘要」要怎麼放才不會被扣分

(A) GPT 輔助內容（建議你在報告中用框框標）

在報告加一段：

- **【GPT 輔助內容】**本作業程式架構、Lasso 特徵選擇、OLS 預測區間繪圖與 CRISP-DM 報告大綱由 GPT 協助產生；資料分析與結果解讀由本人確認與撰寫。

(B) NotebookLM 摘要（你要真的用 NotebookLM 跑，但我給你「可貼上的格式」）

你用 NotebookLM 丟 2-3 篇網路文章/ Kaggle Notebook（同主題 wine quality prediction），然後在報告貼：

- **【NotebookLM 摘要】**（至少 100 字）
- 內容包含：常見做法、常用模型、大家如何評估、為何比線性回歸更強（例如樹模型抓非線性）

注意：我這裡不假裝幫你「跑過 NotebookLM」，因為那會變成不實內容；但你只要把 NotebookLM 產出的摘要貼進來就滿足規定。

6) 匯出 ChatGPT 對話成 PDF（pdfCrowd）

你只要做到「能附在 zip 裡」，通常老師就收：

- 用瀏覽器把本對話頁面 列印（Print）→ Save as PDF
- 或
- 把對話複製到 pdfCrowd/線上 HTML→PDF 工具再輸出成 `gpt_dialog.pdf`

如果你願意，我也可以順便幫你把**report.pdf 的內容（文字版）**直接生成一份「可貼到 Word/Google Docs」的完整報告草稿（含：CRISP-DM 六段、指標表格位置、圖表插入位置、以及比較段落）。