



# HW2：多元線性回歸分析 — AI 協作開發紀錄(實際紀錄版)

## 指令1：

幫我從 Kaggle 上找 3 個符合條件的資料集（10-20 個特徵，適合做多元線性回歸），並幫我挑一個最適合的，附上連結與理由。

## 回覆1：

好的，這是在 Kaggle 上找到的三個適合進行多元線性回歸分析，且特徵數量（欄位）在 10-20 個之間的資料集。

我將為您列出三個選項，並說明推薦哪一個。

### 1. 葡萄酒品質 (Wine Quality)

- 簡介：這個資料集包含約 1600 種葡萄牙紅酒的物理化學特性數據，以及一個從 0 到 10 的品質評分。
- 特徵數量：11 個自變數（如酸度、糖分、酒精濃度等）和 1 個應變數（品質 quality）。總共 12 個欄位。
- 適合性：非常適合。所有特徵都是數值型，目標清晰，是學習回歸分析的經典案例。

### 2. 汽車價格預測 (Car Price Prediction)

- 簡介：包含 200 多輛汽車的各種屬性，目標是預測其價格。
- 特徵數量：包含超過 20 個特徵，但去除 ID 和車名等不相關欄位，並將分類特徵（如燃料類型、車身樣式）進行處理後，可選出 15-20 個核心特徵來建模。
- 適合性：適合，但需要做一些前置的特徵工程（Feature Engineering），例如將文字類別轉換為數字（One-Hot Encoding），是練習真實世界資料處理的好例子。

### 3. 學生表現 (Student Performance)

- 簡介：此資料集包含中學生的個人、家庭和學校相關數據，目標是預測他們的期末成績（G3）。
- 特徵數量：原始特徵超過 30 個，但很多是相關性低的，可以篩選出 10-15 個重要特徵（如讀書時間 studytime、缺席次數 absences、先前成績 G1, G2 等）來建立模型。
- 適合性：適合。與汽車價格資料集類似，需要做一些資料篩選和清理，但主題明確，容易理解。

---

### 最佳推薦：葡萄酒品質 (Wine Quality)

在這三個選項中，我最推薦 葡萄酒品質 (Wine Quality) 資料集。

- Kaggle 連結: Red Wine Quality Dataset  
(<https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009>)
- 推薦理由：
  1. 完全符合條件：它有 11 個預測特徵，完美落在您指定的 10-20 個特徵區間內。
  2. 資料乾淨且格式統一：所有欄位都已是數值型，不需要進行繁瑣的資料清理或轉換，您可以直接投入到模型建立的環節。
  3. 目標明確：要預測的目標 quality 是一個清楚的數值，非常適合做為多元線性回歸的應變數。
  4. 經典且易於上手：這是一個非常知名的入門級資料集，網路上有大量的教學文章和範例程式碼可供參考，如果您遇到問題，很容易找到解決方案。

---

## 指令2：

幫我針對這個資料集撰寫 CRISP-DM 中的『Business Understanding』與『Data Understanding』部分說明，包括變數解釋與問題定義。

---

## 回覆2：

好的，這是一份基於 CRISP-DM 方法論，針對「葡萄酒品質預測」專案的 商業理解 (Business Understanding) 與 資料理解 (Data Understanding) 說明文件。

## 葡萄酒品質預測專案：CRISP-DM 流程說明

### 1. 商業理解 (Business Understanding)

#### 1.1 背景 (Background)

在競爭激烈的葡萄酒市場中，品質是決定產品價格、品牌聲譽和消費者滿意度的核心因素。傳統上，葡萄酒品質多由品酒專家進行感官評估，這個過程不僅成本高昂、耗時，且容易帶入主觀判斷。

如果能透過分析葡萄酒的物理化學特性，建立一個客觀、可量化的品質預測模型，將能為酒莊的生產、品管及行銷環節帶來巨大的商業價值。本專案旨在利用數據科學方法，探索化學成分與葡萄酒品質之間的關聯。

#### 1.2 商業目標 (Business Objectives)

- 主要目標：協助釀酒師理解哪些關鍵化學成分對葡萄酒品質有顯著影響，從而優化釀造過程（如調整發酵參數、控制原料品質）以穩定或提升產品質量。
- 次要目標：
  - 輔助品質控管：建立一個初步的自動化篩選系統，快速識別潛在的低品質批次，減少對人工品鑑的依賴。
  - 提供定價參考：模型的預測分數可作為新酒定價的參考依據之一。
  - 指導行銷策略：了解高分酒的化學特徵，有助於在行銷時突顯產品的特定優點（例如，「果香濃郁」可能與較高的檸檬酸有關）。

#### 1.3 問題定義 (Problem Definition)

本專案的數據科學問題定義為：

我們能否利用葡萄酒的 11 個客觀物理化學指標，建立一個多元線性回歸模型，來有效預測其最終的感官品質評分？

這是一個典型的監督式學習中的回歸問題。我們將以物理化學數據作為自變數 (Independent Variables)，並以品質評分作為應變數 (Dependent Variable)。

#### 1.4 成功標準 (Success Criteria)

- 技術標準：建立的回歸模型在測試集上的 R-squared ( $R^2$ ) 值應達到一個可接受的水平（例如  $> 0.35$ ），表示模型能解釋一部分品質變異。同時，均方根誤差 (RMSE) 應盡可能低，代表預測值與實際值的差距較小。

- 商業標準：模型分析結果需具備可解釋性，能夠明確指出哪些化學指標與品質呈正相關或負相關，為釀酒師提供具體、可執行的改善建議。

---

## 1. 資料理解 (Data Understanding)

### 2.1 資料來源 (Data Source)

本專案使用的資料集來自 Kaggle 上的「Red Wine Quality」資料集，原始數據由葡萄牙維尼奧韋爾德 (Vinho Verde) 葡萄酒的認證機構收集。

### 2.2 資料描述 (Data Description)

- 資料格式：CSV (逗號分隔值) 檔案。
- 資料筆數：共 1,599 筆觀測值（代表 1,599 款紅酒樣本）。
- 欄位數量：共 12 個欄位，全部為數值型。其中 11 個為自變數，1 個為應變數。
- 資料完整性：初步檢視資料無明顯的遺失值 (Missing Values)。

### 2.3 變數解釋 (Variable Explanations)

以下是資料集中各欄位的詳細說明：

自變數 (Features / Independent Variables):

1. fixed acidity (固定酸度): 影響葡萄酒口感的主要酸類，不會輕易揮發。
2. volatile acidity (揮發性酸度): 葡萄酒中易揮發的乙酸含量，過高會產生類似醋的難聞氣味。
3. citric acid (檸檬酸): 微量存在，能增加葡萄酒的清新感與風味。
4. residual sugar (殘糖): 發酵結束後剩餘的糖分，決定葡萄酒的甜度。
5. chlorides (氯化物): 葡萄酒中的鹽分含量。
6. free sulfur dioxide (游離二氧化硫): 作為抗氧化劑和抗菌劑，防止葡萄酒變質。
7. total sulfur dioxide (總二氧化硫): 游離和結合形式的二氧化硫總和，是重要的化學指標。
8. density (密度): 葡萄酒的密度，與酒精和糖分含量有關。
9. pH (pH值): 衡量葡萄酒的酸鹼度，影響其口感和化學穩定性。
10. sulphates (硫酸鹽): 一種添加劑，有助於防止氧化並殺滅微生物。
11. alcohol (酒精濃度): 葡萄酒的酒精含量百分比，是影響酒體和口感的重要因素。

應變數 (Target / Dependent Variable):

1. quality (品質評分): 由品酒專家給出的感官評分，範圍在 0 到 10 之間，是我們希望預測的目標。

---

### 指令3：

請幫我新增一個app.py的檔案，寫出 Data Preparation 的 Python 程式碼（用 streamlit、pandas、numpy、seaborn、matplotlib、scikit-learn），包括缺失值處理、離群值偵測、特徵標準化與訓練/測試切分。

---

### 回覆3：

好的，這是一個 `app.py` 檔案的完整程式碼。

它會執行您指定的資料準備所有步驟，並使用 Streamlit 將每一步的結果與說明視覺化地呈現出來。

我將為您建立這個檔案。

---

### 指令4：

請幫我對app.py撰寫多元線性回歸模型的訓練程式碼，並輸出模型係數、截距與回歸方程式。

---


### 回覆4：

好的，我已經分析了您的 `app.py` 檔案。它目前是一個 Streamlit 應用程式，負責載入「紅酒品質」資料集並進行完整的資料前處理，最後將資料切分為訓練集與測試集。

我將在此基礎上，直接擴充您的 `app.py`，加入多元線性回歸模型的訓練、係數與截距的計算，以及最終回歸方程式的輸出。這會成為您 Streamlit 應用程式的第六個步驟。

我將執行以下操作：

1. 在檔案頂端加入 `from sklearn.linear_model import LinearRegression`。
2. 在檔案末尾新增一個區塊，用來訓練模型並在頁面上顯示結果。

 Gemini CLI update available! 0.9.0 → 0.10.0

Installed with npm. Attempting to automatically update now...

我將更新 `app.py` 檔案。

我已成功將多元線性回歸模型的訓練程式碼新增至 `app.py`。

現在，您的 Streamlit 應用程式包含了完整的流程：從資料載入、前處理、模型訓練到結果展示。

您可以透過在終端機中執行以下指令來啟動應用程式：

```
streamlit run app.py
```

執行後，您將能在瀏覽器中看到完整的應用程式，包括模型係數、截距與回歸方程式。

---

## 指令5：

請幫我對`app.py`使用 RFE 或 SelectKBest 做特徵選擇，列出被選出的重要特徵與其顯著性分析。

---

## 回覆5：

好的，我將為您在 `app.py` 中加入使用 SelectKBest 進行特徵選擇的功能，並分析特徵的顯著性。

我會執行以下步驟：

1. 在現有的 Streamlit 應用中，於模型訓練前新增一個「特徵選擇」的區塊。
2. 匯入 SelectKBest 和 `f_regression` 函式。
3. 使用 SelectKBest 搭配 `f_regression` 來計算每個特徵對於目標變數（酒的品質 quality）的 F-score 和 p-value。
4. 在應用程式中以表格形式展示所有特徵的顯著性分析結果。
5. 讓您可以透過滑桿 (slider) 互動式地選擇要保留的最佳特徵數量 (k)。
6. 列出根據 k 值所選出的重要特徵。
7. 最後，僅使用這些被選出的特徵來訓練您的多元線性回歸模型，並更新模型結果。

我將開始修改 `app.py` 檔案。

我已成功更新 `app.py`。現在您的應用程式加入了互動式的特徵選擇功能。

更新內容如下：

- 顯著性分析：程式會使用 SelectKBest 方法，計算並展示每個特徵的 F-score 與 p-value，讓您判斷其重要性。
- 互動式選擇：您可以使用滑桿來動態調整要保留的特徵數量 (k)，並即時查看選擇結果。
- 模型更新：線性回歸模型現在會根據您選擇的特徵子集進行訓練，並更新其係數、截距與方程式。

請再次執行 `streamlit run app.py` 來體驗新版的應用程式。

## 指令6：

請幫我計算模型評估指標 (MAE、MSE、RMSE、 $R^2$ )，並撰寫一段文字說明模型好壞與可能改進方向。

## 回覆6：

好的，我將為您在 `app.py` 中加入模型評估的最後一個步驟。

我會執行以下操作：

1. 在應用程式的末尾新增一個「模型評估」區塊。
2. 匯入 `scikit-learn` 中用於計算評估指標的函式。
3. 使用訓練好的模型對「測試集」進行預測。
4. 計算並展示四個關鍵的迴歸模型評估指標：MAE、MSE、RMSE 和  $R^2$ 。
5. 根據這些指標，撰寫一段綜合性的分析，說明當前模型的表現好壞，並提供幾個具體的可能改進方向。

我將開始修改 `app.py` 檔案。

哎呀，看來我剛才的自動修改失敗了。這通常是因為檔案的實際內容與我預期的有些微差異，導致無法精確匹配。

別擔心，我將採用一個更穩健的方法：

1. 我會先重新讀取一次 `app.py` 的完整內容，確保我拿到的是最新的版本。
2. 然後，我會將「模型評估」的程式碼區塊和所需的 `import` 語句添加到我讀取到的內容中。
3. 最後，我會用這個全新的、完整的內容直接覆蓋掉舊的 `app.py` 檔案。

這樣可以確保修改的準確性。我馬上開始操作。

| ✓ ReadFile `app.py` |

◆ 好的，我已經重新讀取了檔案的最新內容。現在我將把模型評估的程式碼和分析文字加入其中，並一次性更新 `app.py`。

我已成功將模型評估指標與分析建議的程式碼更新至 `app.py`。

現在您的應用程式已經是一個包含資料處理、特徵選擇、模型訓練、模型評估與改進建議的完整專案。

請再次執行 `streamlit run app.py` 指令，您就可以在頁面最下方看到新增的模型評估指標（MAE、MSE、RMSE、 $R^2$ ）以及對模型表現的詳細分析。

## 指令7：

幫我產生預測 vs 實際值的散佈圖、殘差圖，並加上 95% 信賴區間或預測區間的圖。

## 回覆7：

好的，我將為您在 `app.py` 中加入更多視覺化圖表，以更深入地分析模型表現。

我會執行以下操作：

1. 在應用程式末尾新增一個「模型視覺化分析」區塊。
2. 建立「預測值 vs. 實際值」散佈圖：此圖可以直觀地看出模型的預測與真實情況的貼合程度。我會加上一條 45 度線作為「完美預測」的參考。
3. 建立「殘差圖」：此圖將預測誤差（殘差）與預測值進行對比，有助於診斷模型是否存在系統性偏差，例如非線性或異方差性。
4. 在殘差圖上加入 95% 預測區間：我會計算殘差的標準差，並在殘差圖上繪製出  $\pm 1.96$  個標準差的水平線，這約略代表了 95% 的預測區間。落在這個區間外的點可被視為離群值，這有助於我們評估模型的預測穩定性。

我將開始讀取並修改 `app.py` 檔案。

| ✓ ReadFile `app.py` |

◆ 好的，我已經讀取了最新的 `app.py` 檔案。現在我將把您要求的視覺化分析功能加



入其中，並一次性更新檔案。

我已經成功將您要求的視覺化圖表（預測 vs. 實際散佈圖、殘差圖）及其說明文字加入 `app.py` 中。

現在，您的 Streamlit 應用程式已經非常完整了。請再次執行 `streamlit run app.py`，您就可以在頁面最下方看到新增的視覺化分析圖表與其詳細解讀。

這應該是您所需要的最後一步了。如果您還有其他問題，隨時都可以提出！

---

## 指令8：

請幫我新增CRISP-DM.txt，並撰寫完整的 CRISP-DM 六大步驟（Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, Deployment）報告內容草稿。

---

## 回覆8：

好的，沒有問題。

我將根據我們在 `app.py` 中完成的所有步驟，為您撰寫一份完整的 CRISP-DM 報告草稿，並將其儲存為 `CRISP-DM.txt` 檔案。這份報告將涵蓋從業務理解到最終部署的全部六個階段。

我將開始撰寫報告內容並建立檔案。好的，我將為您建立一個名為 `CRISP-DM.txt` 的檔案，並根據我們在 `app.py` 中完成的所有步驟，撰寫一份完整的 CRISP-DM 六大階段報告草稿。

這份報告將總結從商業目標的定義到最終模型評估與部署的整個過程。我將開始撰寫並生成檔案。

我已經成功為您建立了 `CRISP-DM.txt` 檔案，並在其中撰寫了詳盡的報告草稿。

這份報告完整涵蓋了 CRISP-DM 的六大步驟，並將我們在 `app.py` 中所做的所有工作——從資料理解到模型評估與部署——都進行了系統性的總結。

您可以直接打開該檔案查看報告內容。如果您還有其他需求，請隨時告訴我。

---

## 指令9：

請幫我新增NotebookLM.txt，並撰寫 100 字以上的 NotebookLM 研究摘要，整理網路上對相同主題（多元線性回歸分析）的主流方法與優化建議。

---

## 回覆9：

好的，我將為您建立 NotebookLM.txt 檔案。

我會為您整理一份關於「多元線性回歸分析」的研究摘要，內容將涵蓋目前業界與學界在應用此方法時的主流作法，以及當模型表現不如預期時，常見的優化策略與建議。

我將開始撰寫摘要並生成檔案。

我已成功為您建立 NotebookLM.txt 檔案，並在其中撰寫了超過 100 字的多元線性回歸分析研究摘要。

該摘要整理了此領域的主流方法（如 Ridge、Lasso 回歸）與常見的模型優化建議（如處理非線性關係、特徵工程等）。

您可以直接打開檔案查閱。如果您還有其他問題，請隨時提出。

## 指令10：

請幫我撰寫報告中『Gemini 輔助內容』章節，包括這次的對話過程摘要、我如何利用 Gemini 完成本次作業，以及與 NotebookLM 摘要的整合建議，可以參考我在 hw2-log.txt 的紀錄。

## 回覆10：

好的，我已經詳細閱讀了您在 hw2-log.txt 中的記錄。這份記錄清晰地展示了我們如何透過一系列的指令與回應，逐步完成整個專案。

我將基於這份記錄，為您撰寫報告中的「Gemini 輔助內容」章節。

報告章節：Gemini 輔助內容

前言

本專案從資料集的選擇、程式碼的開發、分析報告的撰寫，到最終的 Streamlit 應用程式部署，全程在 Google Gemini 的輔助下完成。本章節旨在記錄並總結在此次人機協作過程中，我是如何利用 Gemini

作為一個強大的開發與分析夥伴，以及這個協作模式如何極大地提升了專案的執行效率與深度。

### 1. 互動式開發過程摘要

本次作業的完成過程，可以被視為一個敏捷且高效的「對話式開發」(Conversational Development) 流程。整個專案的推進並非一次性的指令，而是透過與 Gemini

的一系列漸進式對話來逐步建構，完整體現了從無到有的過程。

根據 hw2-log.txt 的記錄，其開發流程如下：

1. 專案啟動與規劃 (Prompt 1-2): 我首先向 Gemini 提出高層次需求，要求推薦適合進行多元線性回歸分析的資料集。Gemini 提供了三個選項並附上詳細理由，我據此選定了「紅酒品質」資料集。接著，我要求 Gemini 依據 CRISP-DM 框架，撰寫專案的「商業理解」與「資料理解」部分，為整個專案確立了清晰的分析方向。
2. 核心應用程式搭建 (Prompt 3): 在確立目標後，我指令 Gemini 開始實作，要求建立 `app.py` 檔案並撰寫「資料準備」階段的 Python 程式碼。Gemini 產出了包含離群值處理、特徵標準化和資料切分的完整、可執行的 Streamlit 程式碼。
3. 模型功能迭代增強 (Prompt 4-7): 在應用程式的基礎上，我透過一系列指令，逐步為其增加新功能，每一項都是一個獨立的分析模組：
  - 模型訓練: 加入基礎的多元線性回歸模型。
  - 特徵選擇: 引入 SelectKBest 讓模型更具解釋性。
  - 模型評估: 新增  $R^2$ 、MAE、MSE 等關鍵效能指標的計算與分析。
  - 視覺化分析: 產生預測值與殘差的散佈圖，使模型評估更為直觀。
4. 文件與研究報告生成 (Prompt 8-9): 最後，我要求 Gemini 根據已完成的所有步驟，撰寫完整的 CRISP-DM.txt 報告，並額外生成一份關於多元線性回歸優化方法的 NotebookLM.txt 研究摘要。

這個過程顯示，Gemini 不僅僅是一個程式碼生成器，更是一個能夠參與專案規劃、執行、評估到文件產出的全程夥伴。

### 1. 我的 Gemini 應用策略：從指令到成果

在這次協作中，我扮演的角色更像是一位「專案經理」或「產品負責人」，而 Gemini 則是高效的「AI 開發者與資料分析師」。我的主要策略如下：

- 拆分任務，循序漸進: 我沒有一開始就提出「幫我完成所有作業」的模糊指令，而是遵循 CRISP-DM 的清晰框架，將複雜的任務拆解成一個個具體、可執行的步驟。hw2-log.txt 中的每一條 prompt 都對應了分析流程中的一個邏輯單元。
- 提供上下文，明確目標: 在每一次請求中，我都讓 Gemini 基於已存在的 `app.py` 檔案進行修改與擴充。這種提供上下文的方式，讓 Gemini 能夠準確理解我的意圖，並在現有基礎上進行無縫的功能疊加。

- 專注於「What」，而非「How」：我將精力集中在定義「需要完成什麼」(What)，例如「幫我做特徵選擇」、「幫我計算評估指標」。而具體的「如何實現」(How)，包括選擇哪個函式庫、撰寫 Streamlit 的版面配置、處理程式錯誤（如日誌中記錄的一次 replace 指令失敗後，Gemini 自動切換策略），都由 Gemini 高效完成。

這種人機協作模式，讓我能從繁瑣的語法查詢和程式碼撰寫中解放出來，更專注於專案的宏觀架構、分析邏輯與結果解讀，從而極大地加速了從概念到成品的轉換過程。

### 1. 與 NotebookLM 摘要的整合建議

Gemini 產出的 NotebookLM.txt 研究摘要，不僅是一份獨立的文件，它更為本專案的下一步優化提供了清晰的藍圖。摘要中提到的「主流進階方法」與「模型優化建議」可以直接轉化為 app.py 的新功能。

整合建議如下：

#### 1. 實現進階迴歸模型:

- 在 Streamlit 應用中新增一個模型選擇器（如下拉式選單），讓使用者可以在「線性回歸」、「嶺迴歸 (Ridge)」和「套索迴歸 (Lasso)」之間切換。
- 執行後，並排比較不同模型的  $R^2$  和 RMSE，讓使用者直觀感受正規化技術對模型效能的影響。

#### 2. 引入特徵工程選項:

- 在「特徵選擇」區塊後，新增一個「特徵工程」區塊。
- 提供核取方塊 (Checkbox)，允許使用者動態生成「多項式特徵」（例如，對 alcohol 產生二次方項）或「交互作用項」（例如，alcohol \* sulphates）。
- 重新訓練模型並更新評估指標，驗證這些新特徵是否能如摘要所說，有效捕捉非線性關係並提升模型表現。

#### 3. 將問題重新定義為分類任務:

- 在應用程式的最後，新增一個「嘗試分類模型」的實驗性區塊。
- 將 quality 分數轉換為類別（例如，quality  $\geq 7$  為「優質」，quality  $\leq 4$  為「劣質」，其餘為「普通」）。
- 訓練一個簡單的分類模型（如邏輯迴歸），並展示其準確率 (Accuracy) 和混淆矩陣 (Confusion Matrix)，驗證 NotebookLM 摘要中「重新定義問題」的策略是否可行。

透過以上整合，本專案將從一個單一模型的分析，演變為一個功能豐富、可供比較和探索的互動式機器學習實驗平台，完美展現了從理論研究到實踐應用的閉環。

---

## 指令11：

```
streamlit run app.py 檢查app.py，有沒有錯誤訊息與是否符合作業
ModuleNotFoundError: No module named 'seaborn'
Traceback:
File "C:\Users\user\Desktop\5114056012 - HW2\app.py", line 4, in <module>
import seaborn as sns
```

## 回覆11：

您好，從您提供的錯誤訊息 `ModuleNotFoundError: No module named 'seaborn'` 來看，問題很清楚：執行 `app.py` 所需的 `seaborn` 函式庫尚未安裝在您目前的 Python 環境中。

為了解決這個問題，我將使用 `pip` 為您安裝 `seaborn`。

```
Shell pip install seaborn (使用 pip 安裝 seaborn 函式庫，以解決
ModuleNotFoundError。)
```

---

## 指令12：

```
streamlit run app.py
```

完成！

---

## 指令13：

```
請幫我生成README.md
```

## 回覆13：

好的，我會根據您的專案內容建立一個 `README.md` 檔案，其中將包含專案概述、安裝說明、使用方法等重要資訊。

已建立 README.md

我已經建立了一個完整的 README.md 檔案，內容涵蓋了：

專案概述與功能特色

檔案結構說明

安裝需求與步驟

使用方法

資料集說明

分析流程概要

作者與授權資訊

您可以根據需要修改其中的部分內容，例如：

更新 GitHub 專案的實際 URL

填入您的姓名

調整授權方式

如果您需要進行任何修改或有其他需求，請隨時告訴我。