

新專案HW2 Prompt

依據下列prompt產出新專案#HW2的prompt ## 1. 專案概觀 本專案旨在建立一個互動式的網頁應用程式，用於展示簡單的線性迴歸。 **核心功能需求：** - **模型：** 使用 Python 實作簡單線性迴歸 ($y = ax + b$)。 - **互動性：** 允許使用者透過網頁介面即時調整以下參數： - 斜率 a - 雜訊 (noise) - 資料點數量 - **流程：** 遵循 CRISP-DM 的步驟來建構。 - **框架：** 使用 Streamlit 建立網頁介面。

--- ## 2. 部署說明 本節將引導您完成此 Streamlit 應用程式的部署流程。 ### 2.1. 前置準備 請確認以下項目皆已完成：

- **應用程式檔案**：主要的應用程式檔案為 `app.py`。
- **相依套件 (Dependencies)**：所有需要的 Python 套件都已列在 `requirements.txt` 檔案中。
- **本機驗證**：已透過 `streamlit run app.py` 指令，確認應用程式可以在本機正常執行。
- **版本控制**：專案程式碼已上傳至 GitHub 儲存庫 (repository)，並且為最新版本。

2.2. 部署步驟 (使用 Streamlit Community Cloud)

- **前往 Streamlit Community Cloud**：在瀏覽器中開啟 [\[https://share.streamlit.io/\]](https://share.streamlit.io/) (<https://share.streamlit.io/>)。
- **登入**：使用您的 GitHub 帳號登入。
- **建立新應用程式 (New app)**：點擊頁面上的 "New app" 按鈕。
- **選擇儲存庫 (Repository)**：從列表中選擇您存放此專案的 GitHub 儲存庫。
- **設定部署選項**：
 - **Branch**：確認分支是否正確 (例如 `main` 或 `master`)。
 - **Main file path**：確認主要檔案路徑為 `app.py`。
- **部署 (Deploy!)**：點擊 "Deploy!" 按鈕。

以下是依據你提供的內容，為「#HW2」設計的新專案 Prompt：

HW2：互動式多項式迴歸網頁應用程式

1. 專案概觀

本專案旨在建立一個互動式的網頁應用程式，用於展示多項式迴歸 (Polynomial Regression) 的概念與模型擬合過程。

核心功能需求：

- **模型：** 使用 Python 實作多項式迴歸模型，形式為

$$y = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

- **互動性：** 使用者可透過網頁介面即時調整以下參數：

- 多項式階數 `degree` (n)
- 雜訊強度 `noise`
- 資料點數量 `num_points`
- 是否顯示真實函數與模型擬合曲線

- **視覺化：**

- 動態顯示資料點與模型擬合曲線。
- 顯示訓練誤差 (MSE)。

- **流程**：將開發過程對應到 **CRISP-DM** 的各階段（業務理解、資料理解、資料準備、建模、評估、部署）。
 - **框架**：使用 **Streamlit** 建立互動式網頁介面。
-

2. 部署說明

本節將指引如何在 **Streamlit Community Cloud** 上部署此多項式迴歸應用程式。

2.1. 前置準備

請確認以下項目皆已完成：

- **主要應用程式檔案**：app.py
- **相依套件 (Dependencies)**：所有需要的 Python 套件已列於 requirements.txt
- **本機測試**：在本機成功執行：

bash

```
streamlit run app.py
```

- **版本控制**：程式碼已上傳至 GitHub，並確保版本為最新。
-

2.2. 部署步驟（使用 Streamlit Community Cloud）

1. **開啟平台**：前往 <https://share.streamlit.io/>
2. **登入帳號**：使用 GitHub 帳號登入。
3. **建立新應用程式 (New app)**：點擊「New app」按鈕。
4. **選擇儲存庫 (Repository)**：選擇包含本專案的 GitHub 儲存庫。
5. **設定部署選項**：
 - **Branch**：確認分支（例如 main）
 - **Main file path**：輸入 app.py
6. **部署 (Deploy!)**：點擊「Deploy!」按鈕。

成功部署後，平台將提供應用程式的公開連結，可供他人即時操作互動。

是否要我幫你擴充這個 HW2 prompt，包括 範例介面設計 (如 Streamlit 介面區塊布局) 和 CRISP-DM 各階段的說明模板？

這樣會讓作業架構更完整、可直接實作。

新專案#HW2 **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（資安主題）。 請明確標示資料集來源與連結。
<https://www.kaggle.com/datasets/teamincrisbo/cyber-security-attacks> cybersecurity_attacks.csv

[README.md](http://readme.md/) 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

以下是根據你提供的內容，整理與優化後的 **新專案 #HW2 Prompt** (格式清晰、可直接作為作業說明或提交文件使用 👉)

HW2：多元線性回歸分析與 CRISP-DM 流程實作

一、作業目標

本次作業為 HW1 的延伸，目標是讓同學能夠以實際資料集進行完整的 **多元線性回歸 (Multiple Linear Regression)** 分析。

學生需遵循 CRISP-DM 流程，完成從資料理解、特徵選擇、建模、評估到部署的完整流程，並以程式與報告呈現分析結果。

二、作業內容

(1) 資料來源

- 從 Kaggle 選擇一個具有 10 至 20 個特徵 (features) 的公開資料集，主題不限（建議選擇資安相關資料集）。
- 需於報告中明確標示資料集名稱與來源連結。

範例資料集：

[Cyber Security Attacks Dataset \(Kaggle\)](#)

檔案名稱：cybersecurity_attacks.csv

可附上：

- 資料集摘要 (特徵數量、資料筆數)
 - [README.md](#) 連結或自行整理資料描述
-

(2) 分析任務

請使用 **線性回歸 (Linear Regression)** 模型進行資料分析與預測，並完成以下要求：

A. 模型選擇

- 可嘗試以下任一或多種模型進行比較：
 - 單一線性迴歸 (Simple Linear Regression)

- 多元線性迴歸 (Multiple Linear Regression)
- 自迴歸模型 (Auto Regression, 適用於時間序列資料)

B. 特徵選擇 (Feature Selection)

- 採用至少一種特徵選擇方法：
 - 皮爾森相關係數 (Correlation)
 - 逐步回歸 (Stepwise Regression)
 - Lasso / Ridge Regularization
 - 特徵重要度 (Feature Importance)
- 請說明選擇特徵的依據與結果。

C. 模型評估 (Model Evaluation)

- 至少包含以下評估指標：
 - R² (決定係數)
 - RMSE (均方根誤差)
 - MAE (平均絕對誤差)
- 比較不同模型的表現並討論原因。

D. 視覺化成果

- 繪製 實際值 vs 預測值圖。
- 顯示 預測區間或信賴區間。
- 可視化重要特徵對目標變數的影響。

(3) CRISP-DM 流程說明

請在報告中完整呈現以下六大步驟內容：

流程階段	說明與作業內容
Business Understanding	說明分析目的與資料應用情境。例：預測某類網路攻擊次數或風險程度。
Data Understanding	說明資料來源、特徵摘要、遺漏值、異常值、資料型態、初步探索統計與圖表。
Data Preparation	進行資料清理（遺漏值處理、編碼、標準化等）與特徵選擇。
Modeling	建立迴歸模型、選擇適當超參數，並說明訓練與驗證方法（如 train/test split 或 k-fold cross validation）。
Evaluation	比較模型表現，使用評估指標 (R ² 、RMSE、MAE)，並解釋結果。
Deployment	展示最終模型的應用方式（例如在 Streamlit 上互動展示、輸入新資料進行預測等）。

三、提交內容

請於期限內提交以下項目：

1. 程式碼 (.ipynb 或 .py)

- 含完整分析流程與程式註解。
- 若有互動式應用 (Streamlit / Gradio)，請附上部署連結或截圖。

2. 報告檔 (HW2_Report.pdf)

- 條列 CRISP-DM 各階段內容。
- 包含模型比較、圖表與結果分析。

3. 資料集來源與說明

- 附上 Kaggle 連結與資料摘要。

四、加分項目（選做）

- 使用 Streamlit 建立互動式回歸分析介面。
- 實作 多模型比較儀表板（可切換不同模型與特徵）。
- 使用 Pipeline 或 AutoML 工具（如 PyCaret）自動化分析流程。

是否要我幫你接著生成一份這個 HW2 專案的 程式架構範例 (例如 app.py 與資料分析流程範例) ? 可以直接用來開始開發或提交作業。

yes

太好了 

以下是根據你提供的 HW2 要求，我為你設計的一個完整可執行專案架構範例。

這個範例使用 Streamlit + Scikit-learn 實作多元線性回歸分析，並遵循 CRISP-DM 流程，同時可以直接在 Kaggle 或本地執行。

專案結構建議

bash

```
HW2_MLR_Project/
|
├── app.py                      # Streamlit 主應用程式
├── requirements.txt             # 相依套件
└── data/
    └── cybersecurity_attacks.csv
├── README.md                    # 專案說明 (資料來源、操作說明)
└── HW2_Report.pdf              # 報告 (CRISP-DM 各階段分析)
```

requirements.txt

txt

```
streamlit
pandas
numpy
matplotlib
seaborn
scikit-learn
plotly
```

app.py — 主程式範例

python

```
import streamlit as st
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_

# -----
# 🏁 1. Business Understanding
# -----
st.title("HW2 : 多元線性回歸分析 (CRISP-DM流程) ")
st.markdown("""
本應用示範如何以 **Cyber Security Attacks Dataset** 建立多元線性回歸模型，  

並依據 CRISP-DM 流程進行資料理解、建模與評估。
""")

# -----
# 📈 2. Data Understanding
# -----
st.header("📊 資料理解 (Data Understanding)")
```

```

uploaded_file = st.file_uploader("請上傳 CSV 檔案", type=["csv"])
if uploaded_file is not None:
    df = pd.read_csv(uploaded_file)
    st.write("資料筆數與欄位：", df.shape)
    st.dataframe(df.head())

    st.subheader("特徵摘要")
    st.write(df.describe())

    st.subheader("相關矩陣 (Correlation Matrix)")
    corr = df.corr(numeric_only=True)
    fig_corr, ax = plt.subplots(figsize=(8, 5))
    sns.heatmap(corr, annot=True, cmap="coolwarm", ax=ax)
    st.pyplot(fig_corr)

# -----
# 🖌 3. Data Preparation
# -----
st.header("💡 資料準備 (Data Preparation)")
st.markdown("請選擇目標變數 (Target) 與特徵 (Features) :")

all_columns = df.columns.tolist()
target_col = st.selectbox("目標變數 (y)", all_columns)
feature_cols = st.multiselect("特徵變數 (X)", [c for c in all_columns])

if len(feature_cols) > 0:
    X = df[feature_cols].select_dtypes(include=[np.number]).fillna(0)
    y = df[target_col]

    st.write("選擇的特徵：", feature_cols)
    st.write("資料形狀：", X.shape)

    test_size = st.slider("測試集比例", 0.1, 0.5, 0.2)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size)

# -----
# 🛠 4. Modeling
# -----
st.header("⚙️ 模型建構 (Modeling)")

```

```

model_type = st.selectbox("選擇模型類型", ["Linear Regression", "F

if model_type == "Linear Regression":
    model = LinearRegression()
elif model_type == "Ridge Regression":
    alpha = st.slider("Ridge 正則化參數 alpha", 0.0, 10.0, 1.0)
    model = Ridge(alpha=alpha)
else:
    alpha = st.slider("Lasso 正則化參數 alpha", 0.0, 10.0, 1.0)
    model = Lasso(alpha=alpha)

model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# -----
# 5. Evaluation
# -----
st.header("📈 模型評估 (Evaluation)")

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

st.metric("R² (決定係數)", round(r2, 3))
st.metric("RMSE", round(rmse, 3))
st.metric("MAE", round(mae, 3))

# 可視化實際值 vs 預測值
fig_pred = px.scatter(x=y_test, y=y_pred, labels={'x': '實際值',
fig_pred.add_shape(type="line", x0=y_test.min(), y0=y_test.min()
                    line=dict(color="red", dash="dash"))
st.plotly_chart(fig_pred, use_container_width=True)

# 信賴區間 (簡化示範)
residuals = y_test - y_pred
st.subheader("殘差分佈圖 (Residual Plot)")
fig_res, ax = plt.subplots()
sns.histplot(residuals, kde=True, ax=ax)
st.pyplot(fig_res)

```

```
# -----
# 🚀 6. Deployment
# -----
st.header("🚀 模型部署 (Deployment)")
st.markdown("""
您可以輸入新的資料以進行預測。
""")

input_data = {}
for f in feature_cols:
    input_data[f] = st.number_input(f"輸入 {f} 的值", float(X[f].min()), float(X[f].max()))

if st.button("進行預測"):
    input_df = pd.DataFrame([input_data])
    pred_result = model.predict(input_df)[0]
    st.success(f"🔮 預測結果 : {pred_result:.3f}")
```

📘 README.md 範例

markdown

```
# HW2：多元線性回歸分析（CRISP-DM 流程）
```

本專案示範如何以 Kaggle 的 ****Cyber Security Attacks Dataset**** 建立多元線性回歸並依據 ****CRISP-DM 流程****進行完整的資料分析、建模與評估。

```
## 📁 專案結構
```

- `app.py` : 主程式 (Streamlit)
- `requirements.txt` : 環境套件
- `data/cybersecurity_attacks.csv` : 原始資料
- `HW2_Report.pdf` : 報告文件 (含 CRISP-DM 各階段說明)

```
## 🚀 執行方式
```

```
```bash
pip install -r requirements.txt
streamlit run app.py
```

```

📊 資料來源

[Kaggle: Cyber Security Attacks Dataset](#)

yaml

是否希望我接著幫你：

1.  生成 `HW2_Report.pdf` 的 **內容大綱模板 (CRISP-DM 六階段報告範本)** ,
2. 或  幫你改寫上面 Streamlit 程式成 **Jupyter Notebook (`HW2_MLR.ipynb`)