

---

**使用者：**

我要用 Kaggle 的 Wine Quality 資料做多元線性回歸，請用 **CRISP-DM** 流程寫一份乾淨的 Notebook：

1. 匯入套件與專案常數、2) 載入/檢視資料、3) 清理與切分、4) 做一個基準模型 (StandardScaler+LinearRegression, 5 折 CV RMSE)、5) 用 **LassoCV** 做特徵選擇 (若全 0 用相關係數備援)、6) 用 **statsmodels** **OLS** 訓練與列印摘要、7) 在測試集計算  $R^2$ 、MAE、RMSE、8) 畫 **預測含 95% 預測區間** 圖、9) 做 **殘差診斷** (Residuals vs Fitted、Q-Q)、10) 另外畫一張 **單變數線性回歸** (預設用 alcohol) 含 95% 信賴區間。圖請存到 ./outputs/。

**GPT：**

好的！先建立專案骨架與常用匯入。

```
# Imports & constants
import os, numpy as np, pandas as pd, matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, KFold,
cross_val_score
from sklearn.linear_model import LinearRegression, LassoCV
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
import statsmodels.api as sm
import scipy.stats as stats

STUDENT_ID = "7114056076"
DATA_PATH = "WineQT.csv" # 原始資料檔
TARGET = "quality"
RANDOM_STATE = 42
TEST_SIZE = 0.2

OUTPUT_DIR = "outputs"
os.makedirs(OUTPUT_DIR, exist_ok=True)
```

---

**使用者：**

載入資料，顯示前幾列與資料形狀。

**GPT：**

```
df = pd.read_csv(DATA_PATH)
display(df.head())
df.shape
```

---

**使用者：**

檢視欄位型態、描述統計與缺失值。

**GPT：**

```
display(df.info())
display(df.describe())
df.isna().sum()
```

---

**使用者：**

做資料清理：去重、移除缺失；若有 type 欄位就轉 0/1 (white=1)；只保留數值欄位；切分訓練/測試集。

**GPT：**

```
# Clean & split
df = df.drop_duplicates().dropna().reset_index(drop=True)

# 若資料同時有紅/白酒型態欄位，轉成 0/1 特徵 (white=1)
if 'type' in df.columns:
    df['type'] =
(df['type'].astype(str).str.lower().str.contains('white')).astype(int
)

num_cols = df.select_dtypes(include=[np.number]).columns.tolist()
features = [c for c in num_cols if c != TARGET]

X = df[features]
y = df[TARGET].astype(float)
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=TEST_SIZE, random_state=RANDOM_STATE
)
```

---

#### 使用者：

先做一個基準模型（StandardScaler + LinearRegression），用 5 折交叉驗證計算 RMSE。

#### GPT：

```
baseline = Pipeline([
    ('scaler', StandardScaler()),
    ('lr', LinearRegression())
])

cv = KFold(n_splits=5, shuffle=True, random_state=RANDOM_STATE)
rmse_cv = -np.mean(cross_val_score(
    baseline, X, y, scoring='neg_root_mean_squared_error', cv=cv
))
rmse_cv
```

---

#### 使用者：

用 **LassoCV** 做特徵選擇；如果係數全是 0，就用和目標的絕對相關係數最高的前 6 個當備援。把選到的特徵列出來。

#### GPT：

```
lasso = Pipeline([
    ('scaler', StandardScaler()),
    ('lasso', LassoCV(cv=5, random_state=RANDOM_STATE, n_alphas=50))
])
lasso.fit(X_train, y_train)

coefs = lasso.named_steps['lasso'].coef_
selected = [f for f, c in zip(features, coefs) if abs(c) > 1e-8]
```

# 備援：若 Lasso 全 0，就用與目標的相關係數 Top-K

```
if not selected:
    corr = df[features +
[TARGET]].corr()[TARGET].abs().drop(TARGET).sort_values(ascending=False)
    selected = corr.head(min(6, len(features))).index.tolist()

selected
```

---

**使用者：**

用 `statsmodels OLS` 在選到的特徵上訓練，並印出回歸摘要。

**GPT：**

```
X_train_sel = sm.add_constant(X_train[selected], has_constant='add')
X_test_sel  = sm.add_constant(X_test[selected], has_constant='add')

ols = sm.OLS(y_train, X_train_sel).fit()
print(ols.summary().as_text())
```

---

**使用者：**

在測試集上評估  $R^2$ 、MAE、RMSE。

**GPT：**

```
pred_test = ols.predict(X_test_sel)

r2 = r2_score(y_test, pred_test)
mae = mean_absolute_error(y_test, pred_test)
rmse = mean_squared_error(y_test, pred_test, squared=False)

r2, mae, rmse
```

---

**使用者：**

請產生 每筆測試資料的 95% 預測區間，並畫「Actual vs Predicted（含區間）」圖存檔。

**GPT：**

```

# 取得預測與 95% 預測區間（觀測值區間）
pred_res = ols.get_prediction(X_test_sel).summary_frame(alpha=0.05)

plot_df = pd.DataFrame({
    'actual': y_test.values,
    'pred': pred_res['mean'].values,
    'pi_low': pred_res['obs_ci_lower'].values,
    'pi_high': pred_res['obs_ci_upper'].values
})

plt.figure(figsize=(8, 4.8))
plt.scatter(plot_df['actual'], plot_df['pred'], s=18, alpha=0.7,
            label='Predicted')

mn = float(np.min(plot_df[['actual', 'pred']].values))
mx = float(np.max(plot_df[['actual', 'pred']].values))
plt.plot([mn, mx], [mn, mx], lw=1, label='y = x')

# 用垂直線標出預測區間
for a, lo, hi in zip(plot_df['actual'], plot_df['pi_low'],
                    plot_df['pi_high']):
    plt.vlines(a, lo, hi, alpha=0.15)

plt.title('Test Predictions with 95% Prediction Interval')
plt.xlabel('Actual'); plt.ylabel('Predicted'); plt.legend();
plt.tight_layout()
plt.savefig(os.path.join(OUTPUT_DIR, 'pred_vs_actual_with_PI.png'),
            dpi=150)

```

---

**使用者：**

做 **殘差診斷**：Residuals vs Fitted 與 **Q-Q plot**，都存檔。

**GPT：**

```
residuals = y_test.values - pred_res['mean'].values
```

```
# Residuals vs Fitted
```

```
plt.figure(figsize=(8, 4.8))
```

```

plt.scatter(pred_res['mean'].values, residuals, s=18, alpha=0.7)
plt.axhline(0, lw=1)
plt.title('Residuals vs Fitted')
plt.xlabel('Fitted'); plt.ylabel('Residuals')
plt.tight_layout()
plt.savefig(os.path.join(OUTPUT_DIR, 'residuals_vs_fitted.png'),
            dpi=150)

# Q-Q plot
plt.figure(figsize=(8, 4.8))
stats.probplot(residuals, dist='norm', plot=plt)
plt.title('Residual Q-Q Plot')
plt.tight_layout()
plt.savefig(os.path.join(OUTPUT_DIR, 'residuals_qq.png'), dpi=150)

```

---

**使用者：**

另外畫一張 **單變數線性回歸** 圖（優先用 `alcohol`，否則就用 Lasso 選到的第一個特徵），顯示 **95% 信賴區間（均值區間）**，也存檔。

**GPT：**

```

one_feat = 'alcohol' if 'alcohol' in features else selected[0]

X1_train = sm.add_constant(X_train[[one_feat]])
X1_test  = sm.add_constant(X_test[[one_feat]])

ols1 = sm.OLS(y_train, X1_train).fit()
pred1 = ols1.get_prediction(X1_test).summary_frame(alpha=0.05)

# 排序後畫線+區間更清楚
order = np.argsort(X_test[one_feat].values)
x_sorted = X_test[one_feat].values[order]
mean_sorted = pred1['mean'].values[order]
l_sorted = pred1['mean_ci_lower'].values[order]
u_sorted = pred1['mean_ci_upper'].values[order]

plt.figure(figsize=(8, 4.8))
plt.plot(x_sorted, mean_sorted, label='Fitted line')

```

```
plt.fill_between(x_sorted, l_sorted, u_sorted, alpha=0.2, label='95%
CI (mean)')
plt.scatter(X_test[one_feat], y_test, s=18, alpha=0.7,
label='Actual')

plt.title(f'Simple Linear Regression: {one_feat} → {TARGET}')
plt.xlabel(one_feat); plt.ylabel(TARGET)
plt.legend(); plt.tight_layout()
plt.savefig(os.path.join(OUTPUT_DIR, f'SLR_{one_feat}_with_CI.png'),
dpi=150)
```

---

### 使用者：

最後請列出本實驗的指標與輸出的圖檔清單，方便寫在 README。

### GPT：

- 指標（測試集）： $R^2 = r^2$ 、MAE = mae、RMSE = rmse（皆由上方程式碼計算）。
- 主要圖檔輸出於 ./outputs/：
  - pred\_vs\_actual\_with\_PI.png：Actual vs Predicted（含 95% **預測區間**）。
  - residuals\_vs\_fitted.png：殘差 vs 擬合值。
  - residuals\_qq.png：殘差 Q-Q Plot。
  - SLR\_<fea