Machine Learning: Linear Regression with Scikit-Learn (Unit 4)

Overview

This unit introduced linear regression in Python using Scikit-Learn, covering both single and multivariable models. We explored how it works with libraries like NumPy and SciPy for data analysis (Pedregosa et al., 2011), while also progressing with team projects and updating our e-portfolios.

What I Have Learned

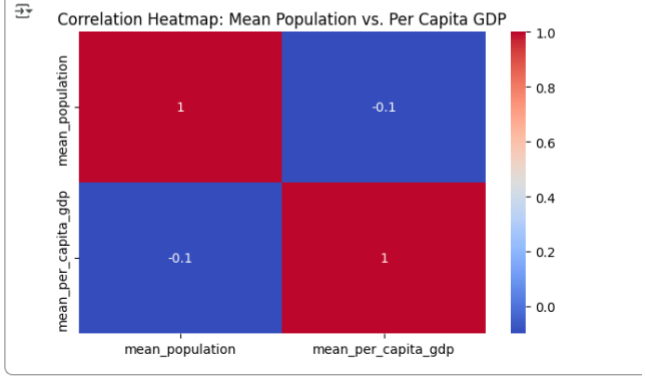
This week boosted my confidence using Scikit-Learn for regression. I learned how to fit models, interpret outputs, and evaluate performance in both simple and multivariate cases. The practical tasks and readings helped connect theory to real-world applications, making the process feel more structured and useful.

Activity

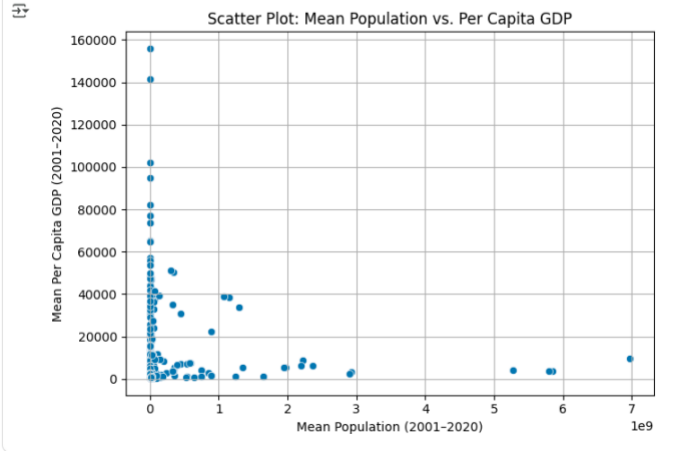
a. Correlation task

```
3]
0s
print(f"Pearson Correlation Coefficient: {corr:.3f}")

Pearson Correlation Coefficient: -0.100
```

```
7]
0s
import matplotlib.pyplot as plt
import seaborn as sns

# Create a heatmap of the correlation matrix
plt.figure(figsize=(6, 4))
sns.heatmap(final_data[['mean_population', 'mean_per_capita_gdp']].corr(),
            annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap: Mean Population vs. Per Capita GDP")
plt.tight_layout()
plt.show()
```



```python
# Create a scatter plot to visualise the relationship
plt.figure(figsize=(7, 5))
sns.scatterplot(data=final_data,
                x='mean_population',
                y='mean_per_capita_gdp')
plt.title("Scatter Plot: Mean Population vs. Per Capita GDP")
plt.xlabel("Mean Population (2001–2020)")
plt.ylabel("Mean Per Capita GDP (2001–2020)")
plt.grid(True)
plt.tight_layout()
plt.show()
```

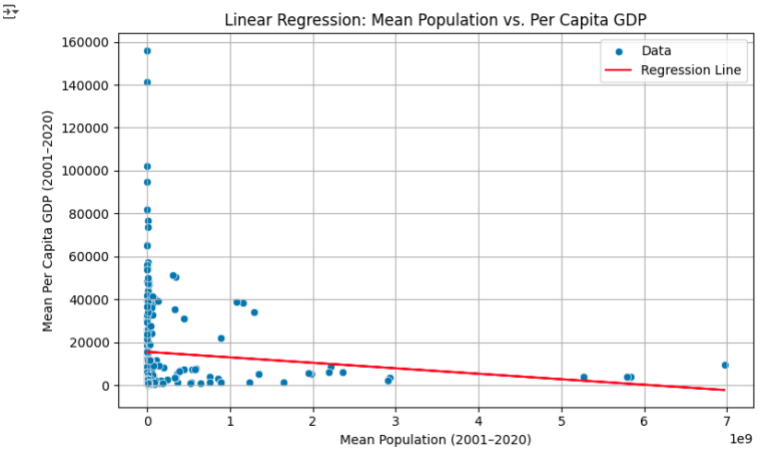


b. Regression task

Using linear regression to model the relationship between **mean population** and **mean per capita GDP** (2001–2020):

- **Slope:** -2.55×10^{-6}
- **Intercept:** 15,532.32
- **R^2 Score:** 0.0099

```python
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import seaborn as sns

# Prepare variables
X = final_data[['mean_population']].values
y = final_data['mean_per_capita_gdp'].values

# Fit linear regression model
model = LinearRegression()
model.fit(X, y)

# Extract model parameters
slope = model.coef_[0]
intercept = model.intercept_
r_squared = model.score(X, y)
y_pred = model.predict(X)

# Plot regression
plt.figure(figsize=(8, 5))
sns.scatterplot(x=final_data['mean_population'], y=final_data['mean_per_capita_gdp'], label="Data")
plt.plot(final_data['mean_population'], y_pred, color='red', label="Regression Line")
plt.title("Linear Regression: Mean Population vs. Per Capita GDP")
plt.xlabel("Mean Population (2001–2020)")
plt.ylabel("Mean Per Capita GDP (2001–2020)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



The negative slope suggests a slight decrease in per capita GDP as population rises. However, the R^2 value of 0.0099 shows that population explains less than 1% of the variation—confirming it's a very weak predictor, as seen in the correlation analysis.

Reference

Field, A. (2018) *Discovering Statistics Using IBM SPSS Statistics*. 5th edn. London: SAGE.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, pp.2825–2830.