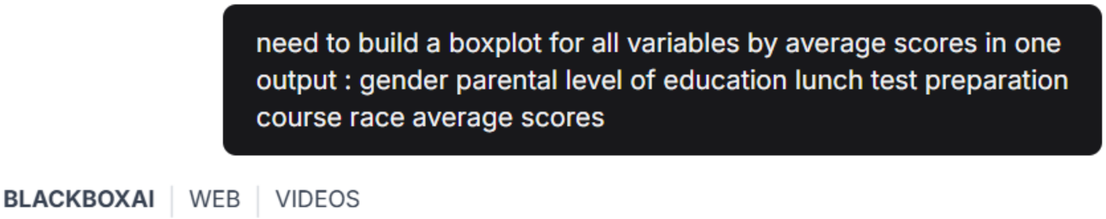
Section 6: Option A: Prompt Engineering

I used Blackbox AI in most of my work. In this project, I used it to help me recall certain code snippets I couldn't remember. For example, when I needed to build five boxplots and display them in a single output, so I could comment on them all at once, I used the following prompt:



need to build a boxplot for all variables by average scores in one output : gender parental level of education lunch test preparation course race average scores

the response from the AI was:

Example Code

Here's an example of how to create boxplots for each of the categorical variables against average scores:

python

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
data = {
    'gender': ['male', 'female', 'female', 'male', 'female', 'male', '
    'parental level of education': ['high school', 'some college', 'as
    'lunch': ['standard', 'free/reduced', 'standard', 'free/reduced',
    'test preparation course': ['completed', 'none', 'completed', 'non
    'race': ['group A', 'group B', 'group A', 'group B', 'group A', 'g
    'average scores': [85, 78, 90, 65, 88, 70, 92, 75]
}

df = pd.DataFrame(data)

# Set the style of seaborn
sns.set(style="whitegrid")

# Create a figure with subplots
fig, axes = plt.subplots(3, 2, figsize=(15, 15))  # Adjust the size as
fig.suptitle('Boxplots of Average Scores by Categorical Variables', fo

# Create boxplots for each categorical variable
sns.boxplot(x='gender', y='average scores', data=df, ax=axes[0, 0])
axes[0, 0].set_title('By Gender')

sns.boxplot(x='parental level of education', y='average scores', data=
axes[0, 1].set_title('By Parental Level of Education')
axes[0, 1].tick_params(axis='x', rotation=45)  # Rotate x labels for b

sns.boxplot(x='lunch', y='average scores', data=df, ax=axes[1, 0])
axes[1, 0].set_title('By Lunch Type')

sns.boxplot(x='test preparation course', y='average scores', data=df,
axes[1, 1].set_title('By Test Preparation Course')

sns.boxplot(x='race', y='average scores', data=df, ax=axes[2, 0])
axes[2, 0].set_title('By Race')

# Hide the last subplot (if not needed)
fig.delaxes(axes[2, 1])

# Adjust layout
plt.tight_layout(rect=[0, 0.03, 1, 0.95])  # Adjust the layout to make
plt.show()
```

What did the AI do well?

The AI performed very well — it generated all the code correctly and even included a data example. It also explained why each part of the code was used, which helped me understand the logic behind it.

What did you have to modify or improve?

I only had to adjust a few things to fit my IDE, like changing the figsize to (15, 15) and modifying the title and axis labels. Since I provided a clear and simple prompt with specific requirements, the AI's response was almost perfect, and very little editing was needed.