ASSIGNMENT-1

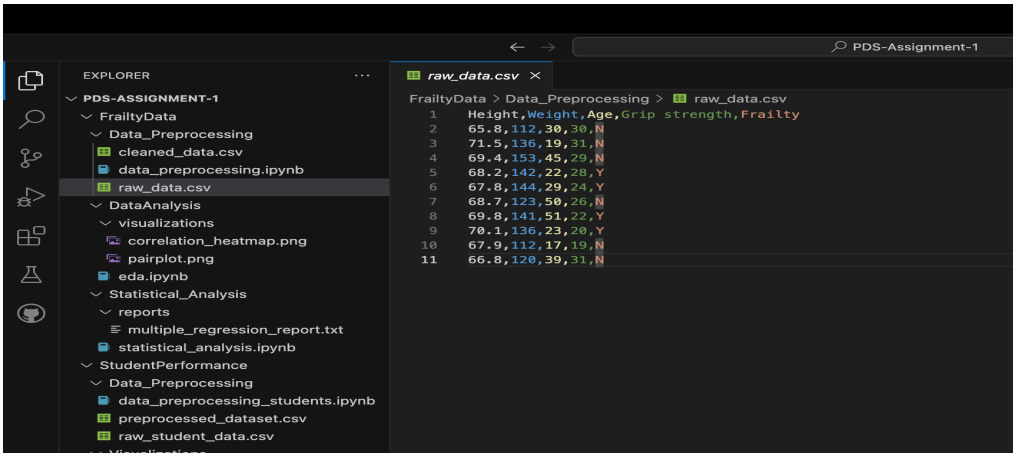
Name: ENUKONDA SAI DEEPTHI
STUDENT ID:16343756

1) Frailty is physical weakness; lack of health or strength. Reduced grip strength in females correlated with higher frailty scores and vice versa. Hand grip strength can be quantified by measuring the amount of static force that the hand can squeeze around a dynamometer. The force has most commonly been measured in kilograms and pounds. The table below represents data from 10 female participants. The Height is measured in inches, Weight in pounds, Age in years, Grip strength in kilograms. Frailty is qualitative attribute indicated the presence or absence of the symptoms. Based on the following table, design the three stages of reproducible workflow, includes the work you can do and the folder structure in each stage (reference study case in chapter 3). (5 points)

Height	Weight	Age	Grip strength	Frailty
65.8	112	30	30	N
71.5	136	19	31	N
69.4	153	45	29	N
68.2	142	22	28	Y
67.8	144	29	24	Y
68.7	123	50	26	N
69.8	141	51	22	Y
70.1	136	23	20	Y
67.9	112	17	19	N
66.8	120	39	31	N

STEP 1: DATA PREPROCESSING

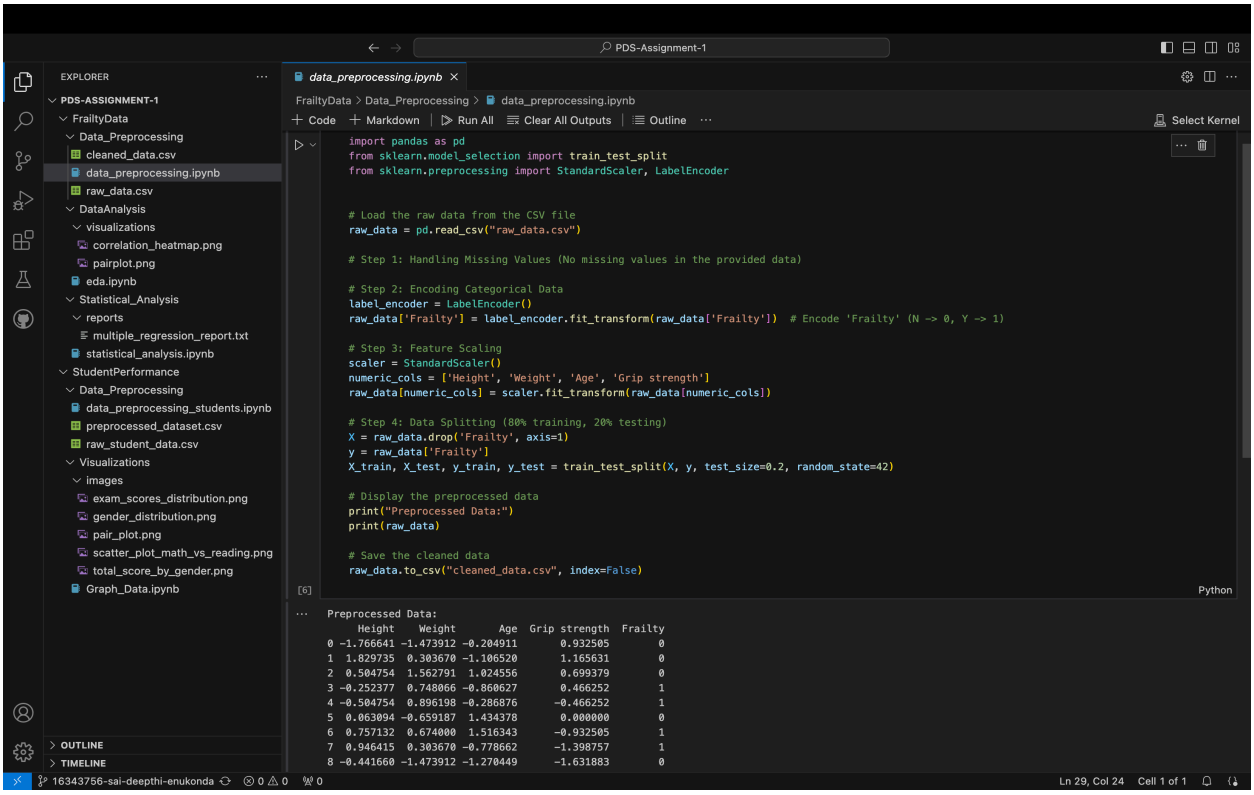
- The below image shows the raw dataset before data preprocessing:



- I've performed several preprocessing steps on the given dataset:

1. **Data Loading**: I created a DataFrame using the provided data, which contains information about individuals' height, weight, age, grip strength, and frailty status.
2. **Encoding Categorical Data**: The 'Frailty' column contains categorical data with values 'N' (No) and 'Y' (Yes). I used label encoding to convert these categorical values to numerical values:
    - 'N' is encoded as 0.
    - 'Y' is encoded as 1.
3. **Standardization of Numeric Features**: I standardized the numeric columns ('Height', 'Weight', 'Age', and 'Grip strength') to ensure that they have similar scales. Standardization transforms the data so that it has a mean of 0 and a standard deviation of 1. This helps prevent certain machine learning algorithms from being sensitive to the scale of input features.
4. **Data Splitting**: I split the data into a training set and a testing set using the **train_test_split** function from scikit-learn. The split was done with 80% of the data used for training and 20% for testing. This separation allows you to train machine learning models on one subset and evaluate their performance on another, helping to assess how well your models generalize to unseen data.

The code concludes by printing the preprocessed data to the console, which you can use for further analysis or modeling tasks. These preprocessing steps help prepare the data for machine learning or statistical analysis by making it more suitable and consistent for modeling techniques.

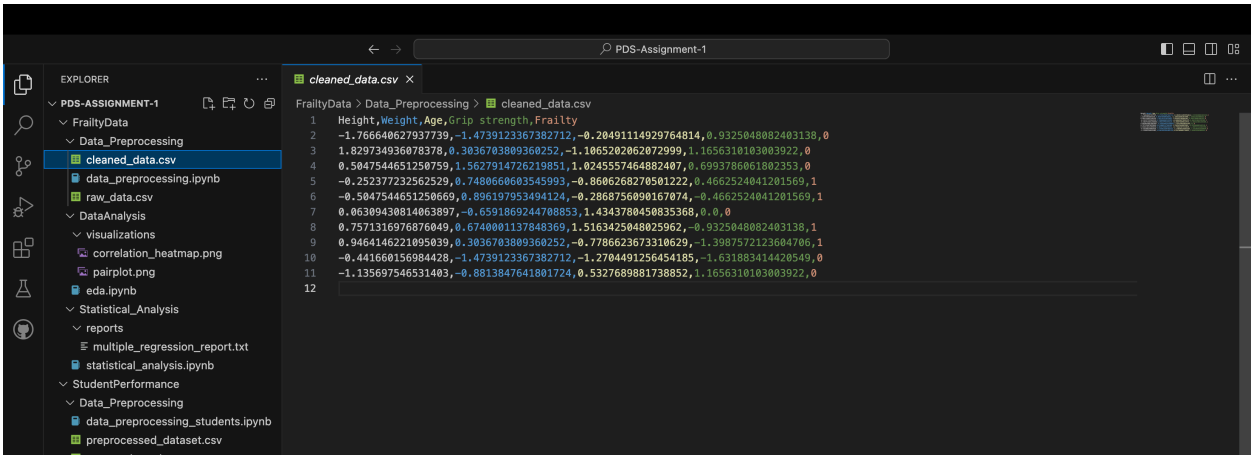


The output obtained here is the preprocessed data. Here's what each column represents:

- **Height:** Standardized height values.
- **Weight:** Standardized weight values.
- **Age:** Standardized age values.
- **Grip strength:** Standardized grip strength values.
- **Frailty:** Encoded frailty status (0 for 'N' and 1 for 'Y').

This output indicates that the preprocessing steps were successfully applied to the dataset. All numeric features have been standardized (scaled to have a mean of 0 and a standard deviation of 1), and the 'Frailty' column has been encoded into numerical values.

- The below is the cleaned data is obtained after the preprocessing

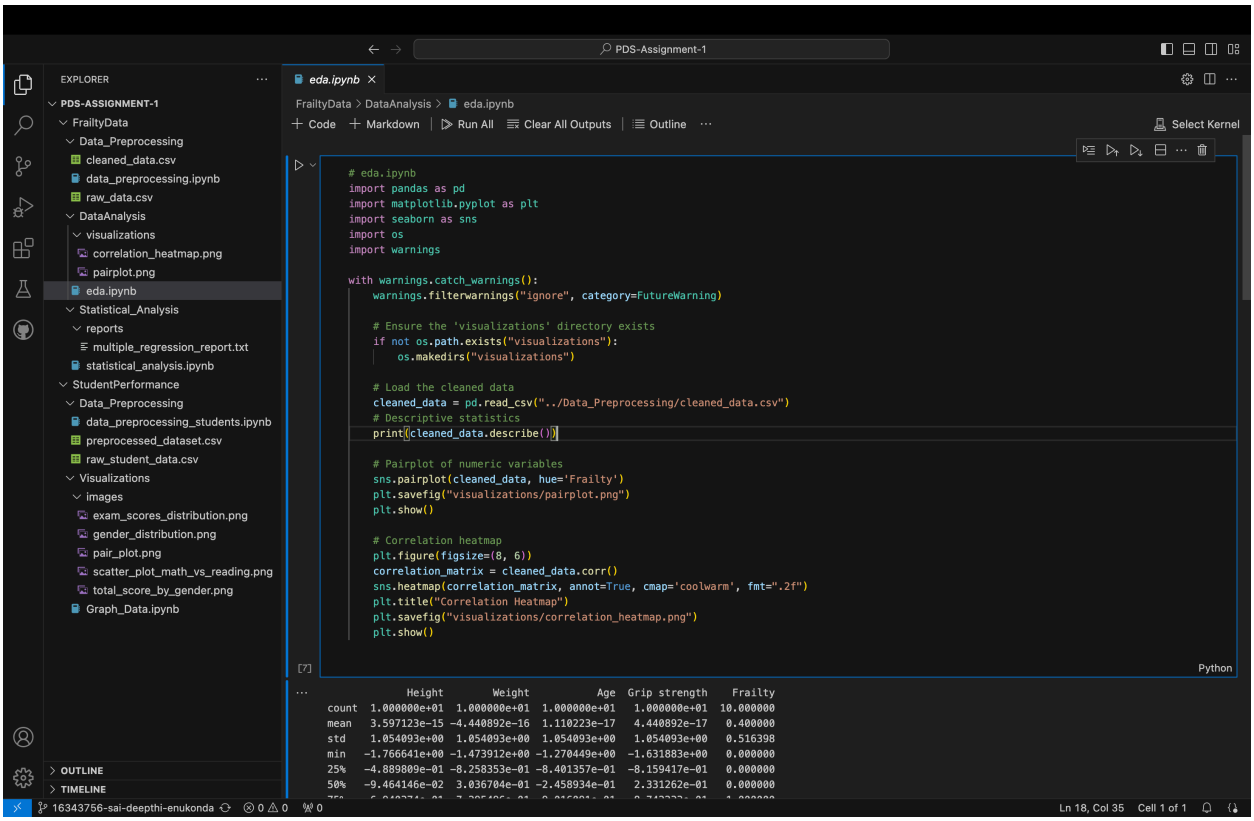


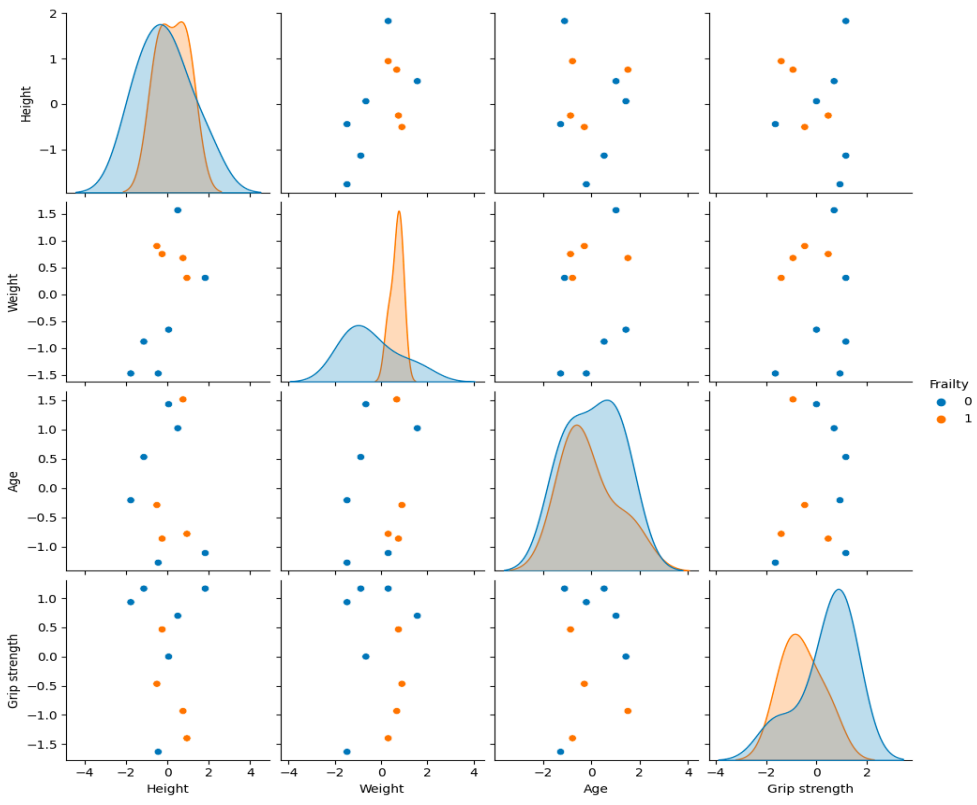
STEP 2: DATA ANALYSIS:

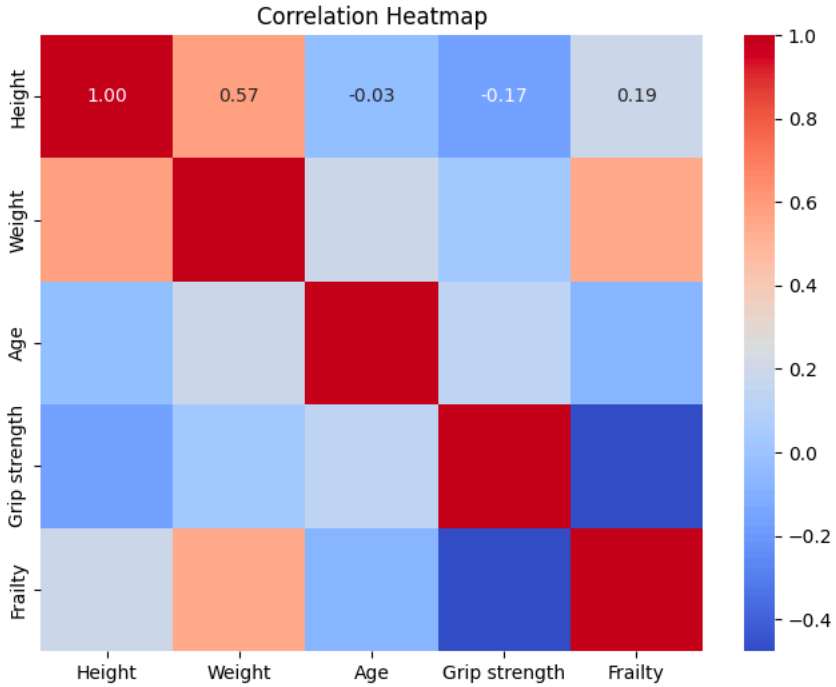
This code is meant for data analysis and report generation, providing visualizations and statistics to help understand the dataset and its relationships.

Steps done as below:

1. Imported necessary libraries for data analysis and visualization.
2. Suppresses future warning messages.
3. Created a directory called 'visualizations' for storing generated images.
4. Loaded cleaned data from a CSV file.
5. Computes and prints descriptive statistics of the data.
6. Generated a pair plot to visualize relationships between numeric variables and saves it as an image.
7. Creates a correlation heatmap to show variable correlations and saves it as an image.


PDS-Assignment-1
EXPLORER
PDS-ASSIGNMENT-1
FrailtyData
Data_Preprocessing
cleaned_data.csv
data_preprocessing.ipynb
raw_data.csv
DataAnalysis
visualizations
correlation_heatmap.png
pairplot.png
eda.ipynb
Statistical_Analysis
reports
multiple_regression_report.txt
statistical_analysis.ipynb
StudentPerformance
Data_Preprocessing
data_preprocessing_students.ipynb
preprocessed_dataset.csv
raw_student_data.csv
Visualizations
images
exam_scores_distribution.png
gender_distribution.png
pair_plot.png
scatter_plot_math_vs_reading.png
total_score_by_gender.png
Graph_Data.ipynb
OUTLINE
TIMELINE
eda.ipynb
FrailtyData > DataAnalysis > eda.ipynb
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
# eda.ipynb
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
import warnings
with warnings.catch_warnings():
warnings.filterwarnings("ignore", category=FutureWarning)
# Ensure the 'visualizations' directory exists
if not os.path.exists("visualizations"):
os.makedirs("visualizations")
# Load the cleaned data
cleaned_data = pd.read_csv("../Data_Preprocessing/cleaned_data.csv")
# Descriptive statistics
print(cleaned_data.describe())
# Pairplot of numeric variables
sns.pairplot(cleaned_data, hue='Frailty')
plt.savefig("visualizations/pairplot.png")
plt.show()
# Correlation heatmap
plt.figure(figsize=(8, 6))
correlation_matrix = cleaned_data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.savefig("visualizations/correlation_heatmap.png")
plt.show()
[7]
Python
Height Weight Age Grip strength Frailty
count 1.000000e+01 1.000000e+01 1.000000e+01 1.000000e+01 10.000000
mean 3.597123e-15 -4.440892e-16 1.110223e-17 4.440892e-17 0.400000
std 1.054093e+00 1.054093e+00 1.054093e+00 1.054093e+00 0.516398
min -1.766641e+00 -1.473912e+00 -1.270449e+00 -1.631883e+00 0.000000
25% -4.889809e-01 -8.258353e-01 -8.401357e-01 -8.159417e-01 0.000000
50% -9.464146e-02 3.036704e-01 -2.458934e-01 2.331262e-01 0.000000
16343756-sai-deepthi-enukonda
Ln 18, Col 35
Cell 1 of 1

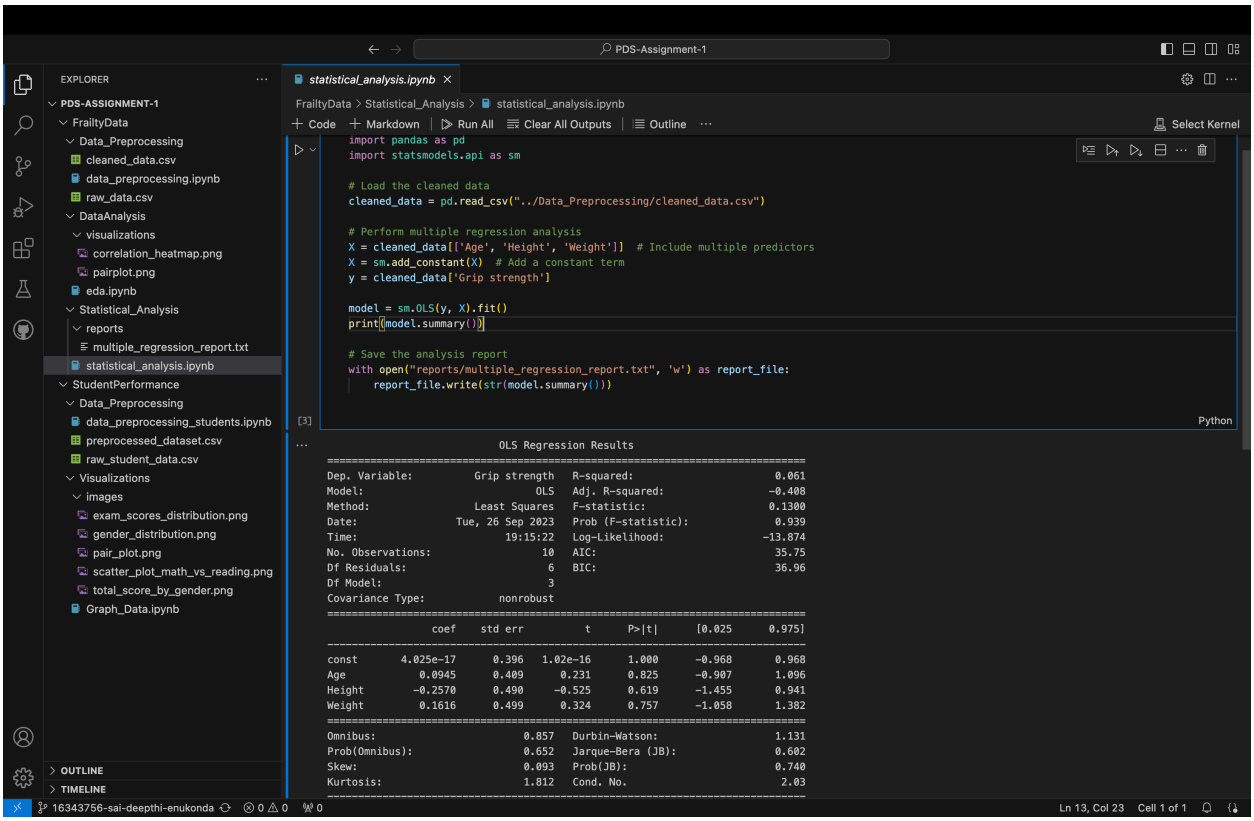

Height
Weight
Age
Grip strength
Frailty
0
1



STEP 3: STATISTICAL ANALYSIS:

This code loads data, performs a multiple regression analysis, and saves the analysis summary as a text report for further reference and reporting.

1. Imports necessary libraries, including pandas for data handling and statsmodels for statistical analysis.
2. Loaded cleaned data from a CSV file named "cleaned_data.csv" located in the "../Data_Preprocessing/" directory.
3. Performs a multiple regression analysis using the statsmodels library:
    - Defines predictor variables (Age, Height, and Weight) and adds a constant term to the predictor matrix.
    - Defines the target variable (Grip strength).
    - Fits an ordinary least squares (OLS) multiple regression model, which estimates the relationship between the predictors and the target variable.
4. Prints a summary of the regression model, including statistics like coefficients, R-squared, and p-values.
5. Saves the analysis report to a text file named "multiple_regression_report.txt" in a directory named "reports." The report contains the same summary information as displayed in the console.

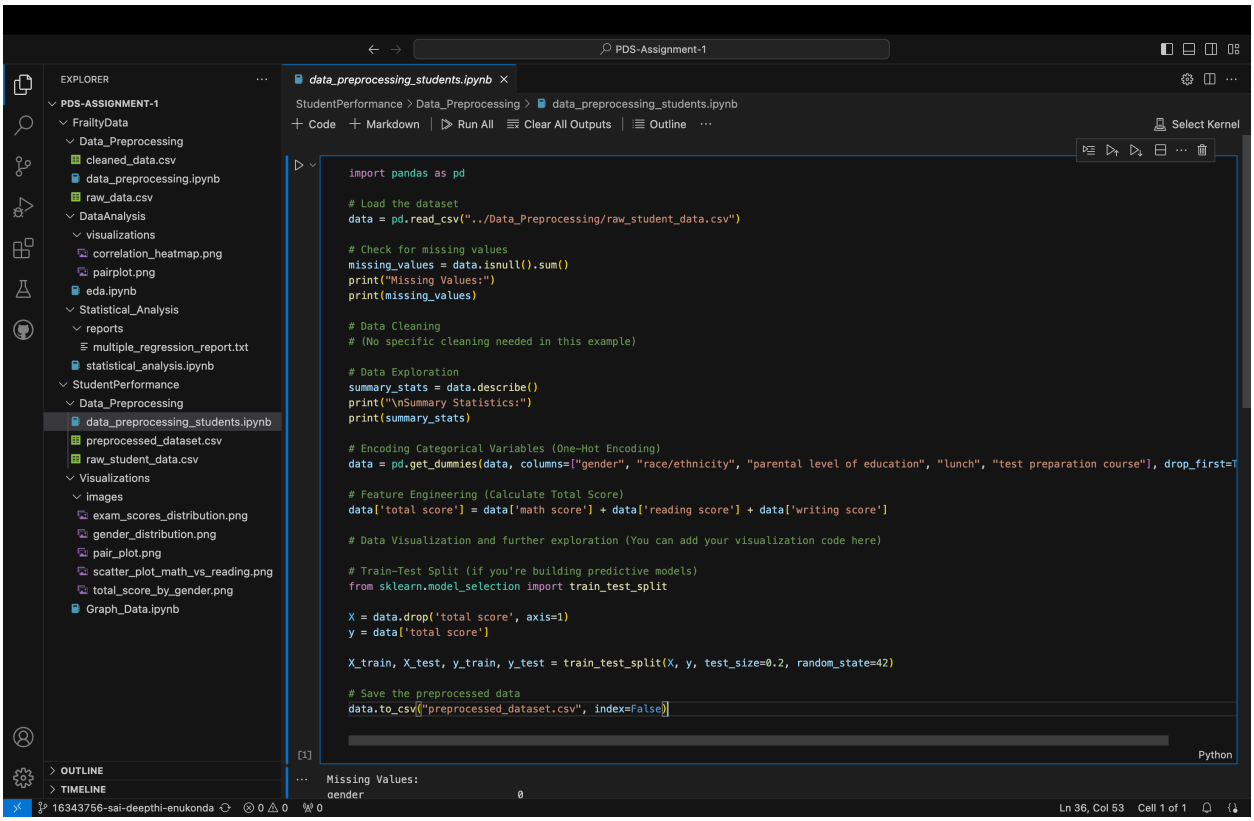


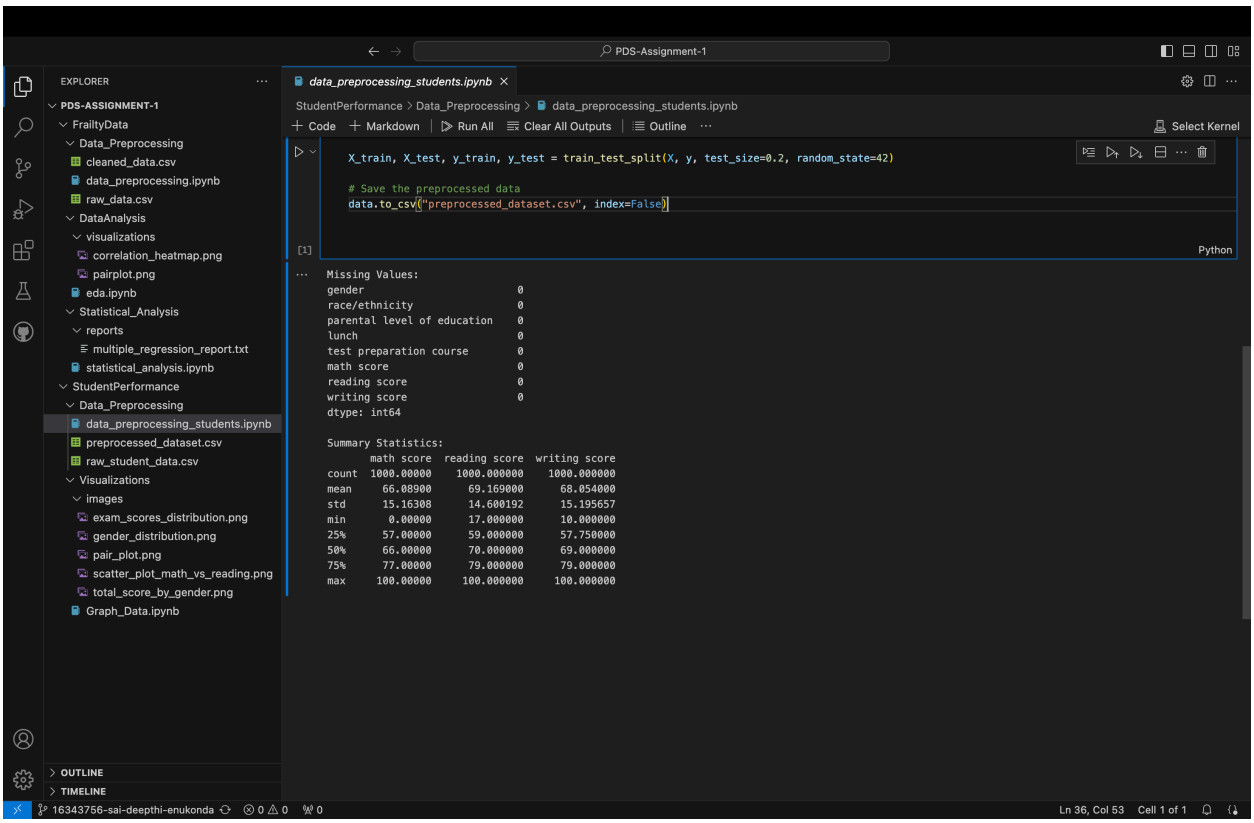
2) Perform 5 data visualization tasks on the student performance dataset given in the link below (create 5 different visualizations). Explain what kind analysis has become easier with each of the visualizations. Create the folder structure for this question similar to question 1.

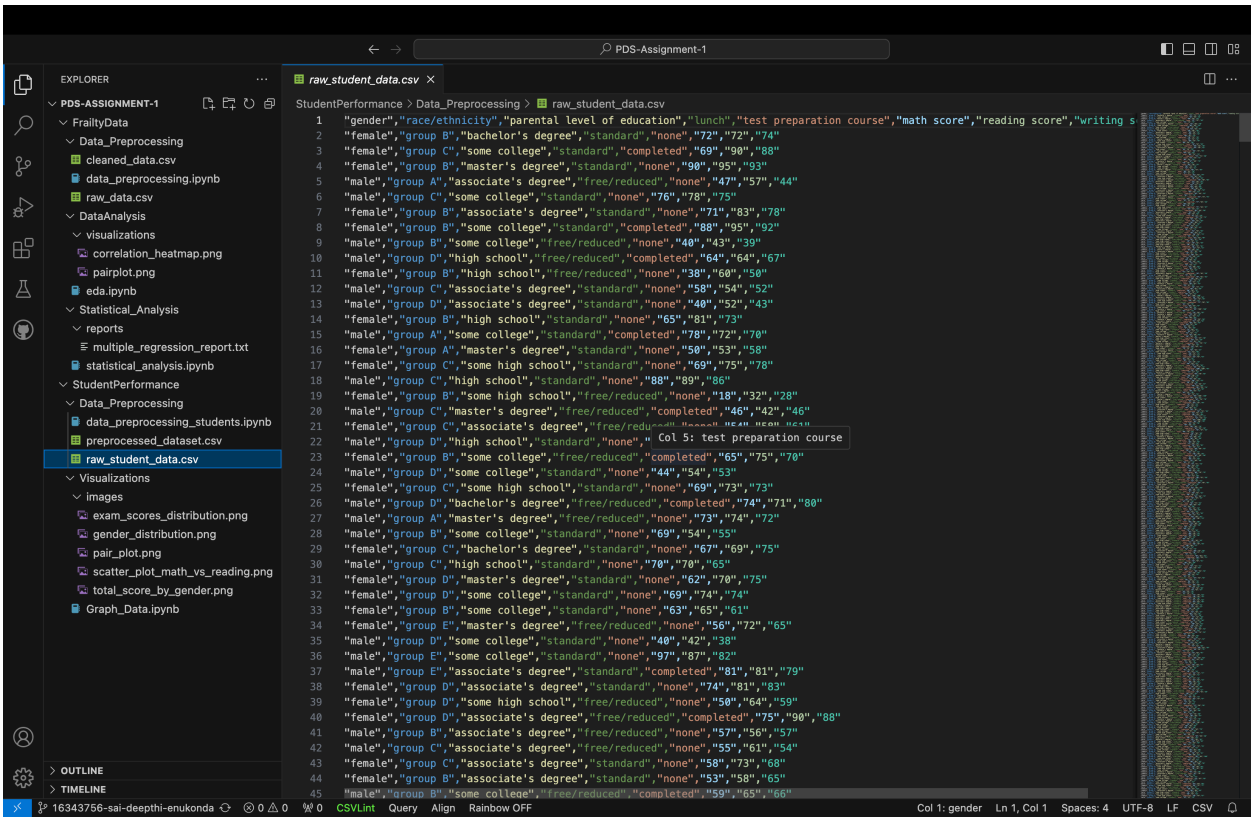
STEP 1: DATA PREPROCESSING:

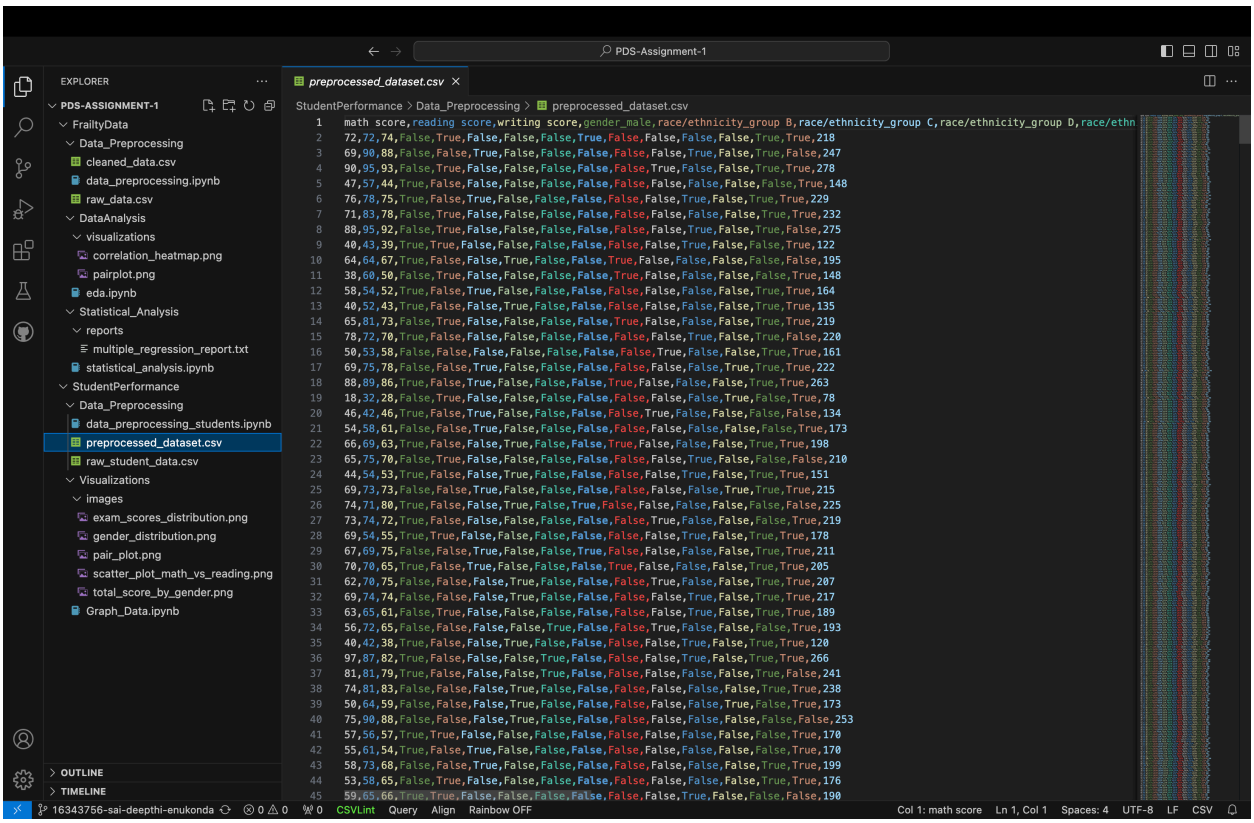
The following steps are taken for the student performance dataset :

1. Loaded a dataset from a CSV file.
2. Checked for missing values in the dataset.
3. Performed one-hot encoding on categorical variables.
4. Calculated a 'total score' feature by summing individual scores.
5. Splitted the data into training and testing sets Define feature variables ('X') and the target variable ('y').Use **train_test_split** from scikit-learn to split the data into training and testing sets (80% training, 20% testing),Set a random seed ('random_state') for reproducibility.
6. Saved the preprocessed dataset to a new CSV file.


PDS-Assignment-1
EXPLORER
PDS-ASSIGNMENT-1
FrailtyData
Data_Preprocessing
cleaned_data.csv
data_preprocessing.ipynb
raw_data.csv
DataAnalysis
visualizations
correlation_heatmap.png
pairplot.png
eda.ipynb
Statistical_Analysis
reports
multiple_regression_report.txt
statistical_analysis.ipynb
StudentPerformance
Data_Preprocessing
data_preprocessing_students.ipynb
preprocessed_dataset.csv
raw_student_data.csv
Visualizations
images
exam_scores_distribution.png
gender_distribution.png
pair_plot.png
scatter_plot_math_vs_reading.png
total_score_by_gender.png
Graph_Data.ipynb
OUTLINE
TIMELINE
data_preprocessing_students.ipynb
StudentPerformance > Data_Preprocessing > data_preprocessing_students.ipynb
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
import pandas as pd
# Load the dataset
data = pd.read_csv("../Data_Preprocessing/raw_student_data.csv")
# Check for missing values
missing_values = data.isnull().sum()
print("Missing Values:")
print(missing_values)
# Data Cleaning
# (No specific cleaning needed in this example)
# Data Exploration
summary_stats = data.describe()
print("\nSummary Statistics:")
print(summary_stats)
# Encoding Categorical Variables (One-Hot Encoding)
data = pd.get_dummies(data, columns=["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"], drop_first=T
# Feature Engineering (Calculate Total Score)
data['total score'] = data['math score'] + data['reading score'] + data['writing score']
# Data Visualization and further exploration (You can add your visualization code here)
# Train-Test Split (if you're building predictive models)
from sklearn.model_selection import train_test_split
X = data.drop('total score', axis=1)
y = data['total score']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Save the preprocessed data
data.to_csv("preprocessed_dataset.csv", index=False)
[1]
Python
Missing Values:
gender 0
16343756-sai-deepthi-enukonda
Ln 36, Col 53
Cell 1 of 1


PDS-Assignment-1
EXPLORER
PDS-ASSIGNMENT-1
FrailtyData
Data_Preprocessing
cleaned_data.csv
data_preprocessing.ipynb
raw_data.csv
DataAnalysis
visualizations
correlation_heatmap.png
pairplot.png
eda.ipynb
Statistical_Analysis
reports
multiple_regression_report.txt
statistical_analysis.ipynb
StudentPerformance
Data_Preprocessing
data_preprocessing_students.ipynb
preprocessed_dataset.csv
raw_student_data.csv
Visualizations
images
exam_scores_distribution.png
gender_distribution.png
pair_plot.png
scatter_plot_math_vs_reading.png
total_score_by_gender.png
Graph_Data.ipynb
OUTLINE
TIMELINE
data_preprocessing_students.ipynb
StudentPerformance > Data_Preprocessing > data_preprocessing_students.ipynb
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Save the preprocessed data
data.to_csv("preprocessed_dataset.csv", index=False)
[1]
Python
Missing Values:
gender 0
race/ethnicity 0
parental level of education 0
lunch 0
test preparation course 0
math score 0
reading score 0
writing score 0
dtype: int64
Summary Statistics:
math score reading score writing score
count 1000.00000 1000.000000 1000.000000
mean 66.08900 69.169000 68.054000
std 15.16308 14.600192 15.195657
min 0.00000 17.000000 10.000000
25% 57.00000 59.000000 57.750000
50% 66.00000 70.000000 69.000000
75% 77.00000 79.000000 79.000000
max 100.00000 100.000000 100.000000
16343756-sai-deepthi-enukonda
Ln 36, Col 53
Cell 1 of 1

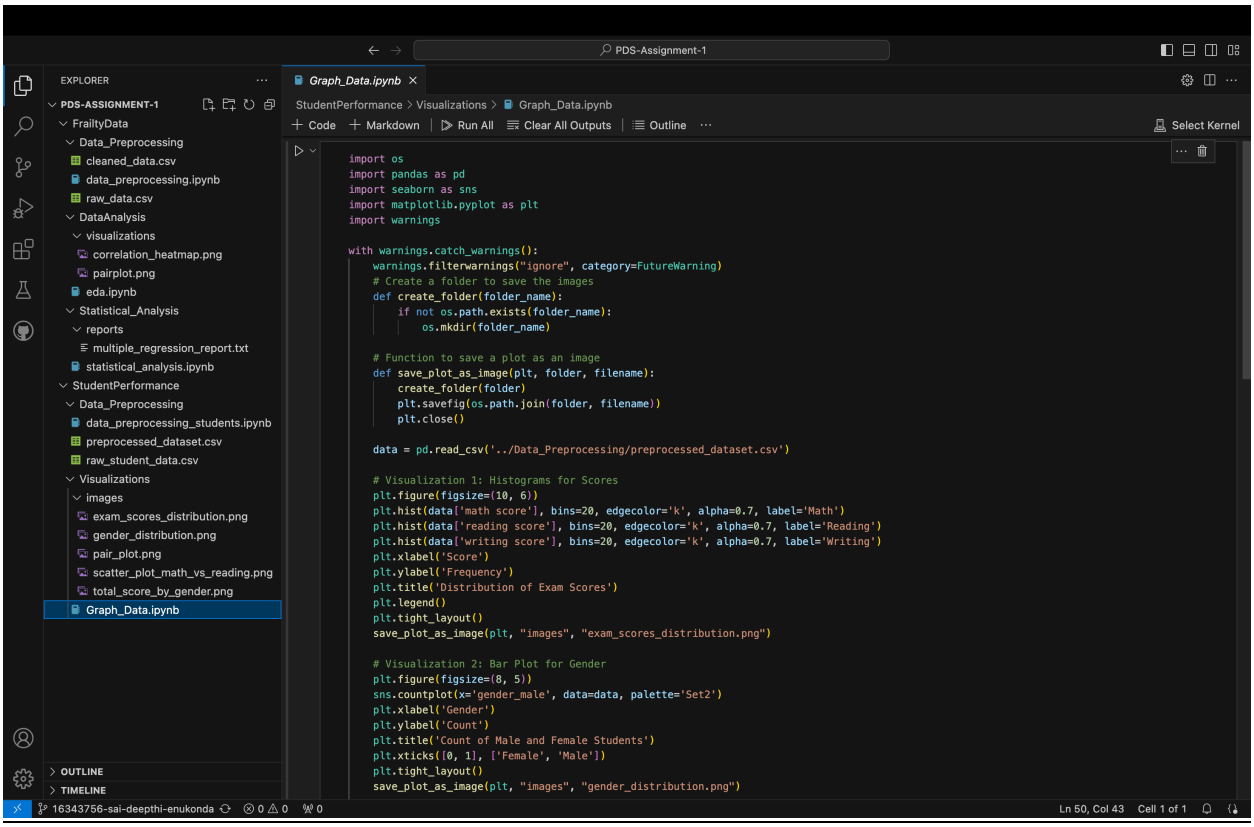

PDS-Assignment-1
EXPLORER
raw_student_data.csv
StudentPerformance > Data_Preprocessing > raw_student_data.csv
Col 5: test preparation course
Col 1: gender
Ln 1, Col 1
Spaces: 4
UTF-8
LF
CSV

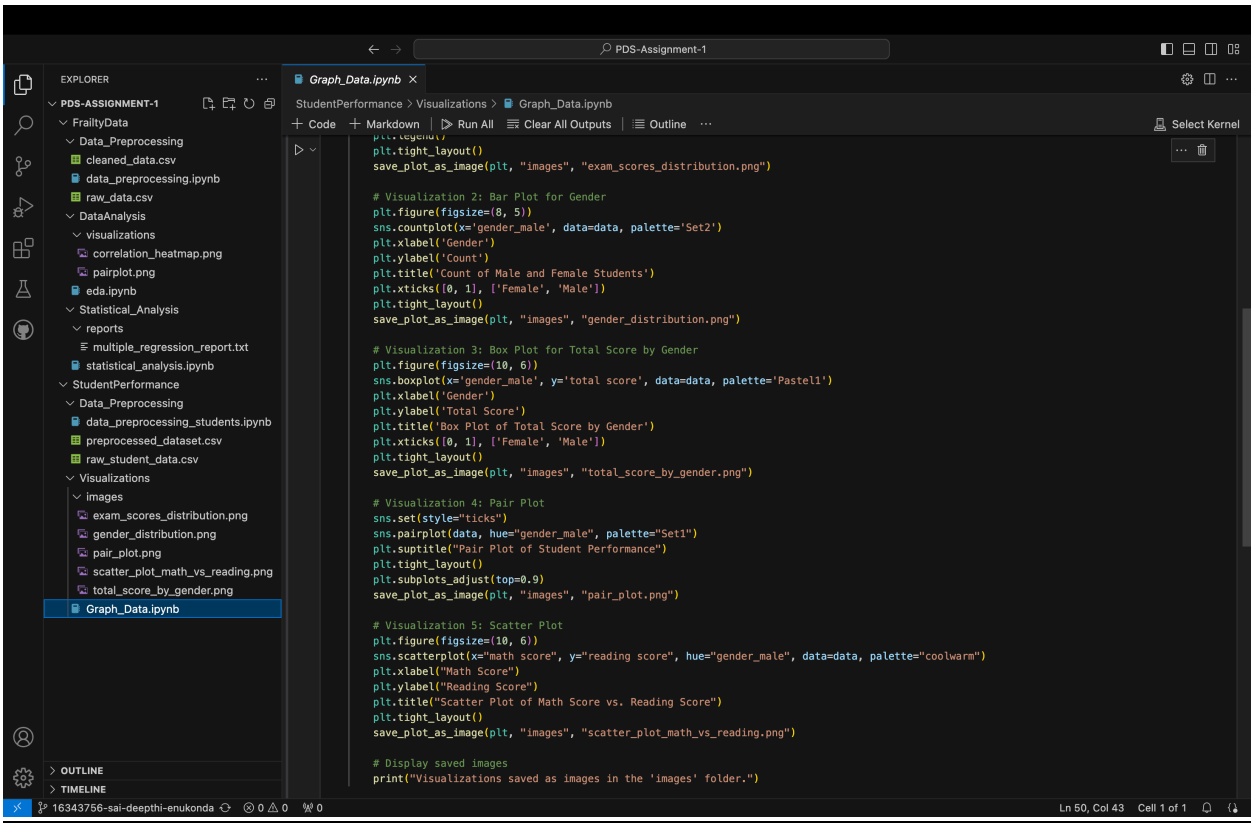

PDS-Assignment-1
EXPLORER
preprocessed_dataset.csv
StudentPerformance > Data_Preprocessing > preprocessed_dataset.csv
Col 1: math score
Ln 1, Col 1
Spaces: 4
UTF-8
LF
CSV

STEP2 : VISUALIZATIONS:

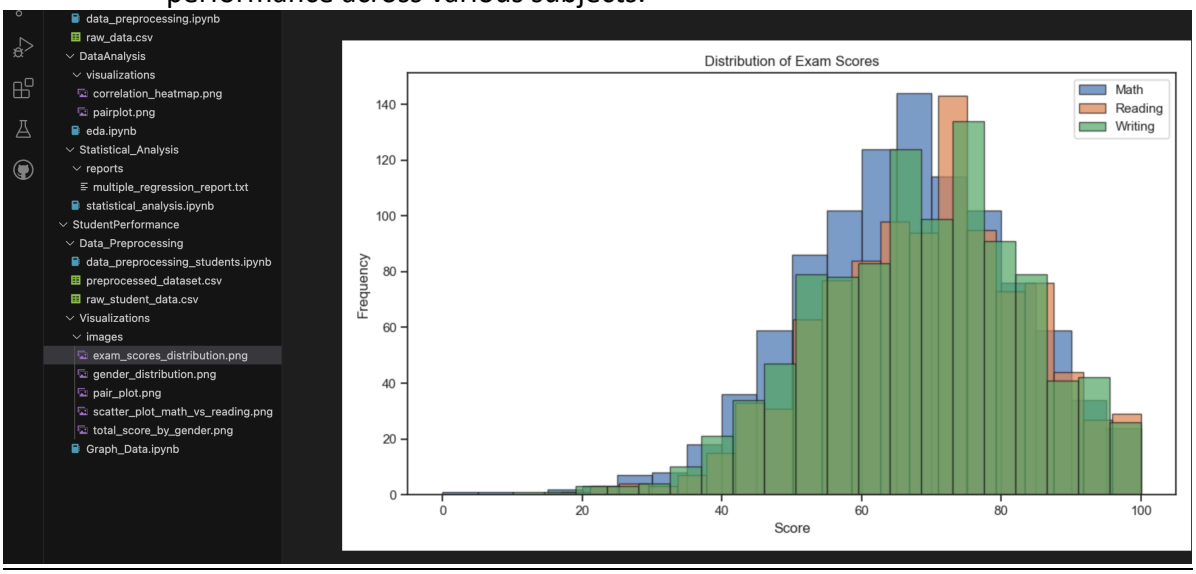
Performed 5 data visualization tasks on the student performance dataset by using these steps:

1. Imports necessary libraries for data manipulation and visualization.
2. Defines functions to create folders and save plots as images.
3. Loaded a preprocessed dataset from a CSV file.
4. Created and saved various data visualizations, including histograms, bar plots, box plots, pair plots, and scatter plots.
5. The saved images are organized in an 'images' folder.
6. A message is printed to indicate that the visualizations have been saved as images.

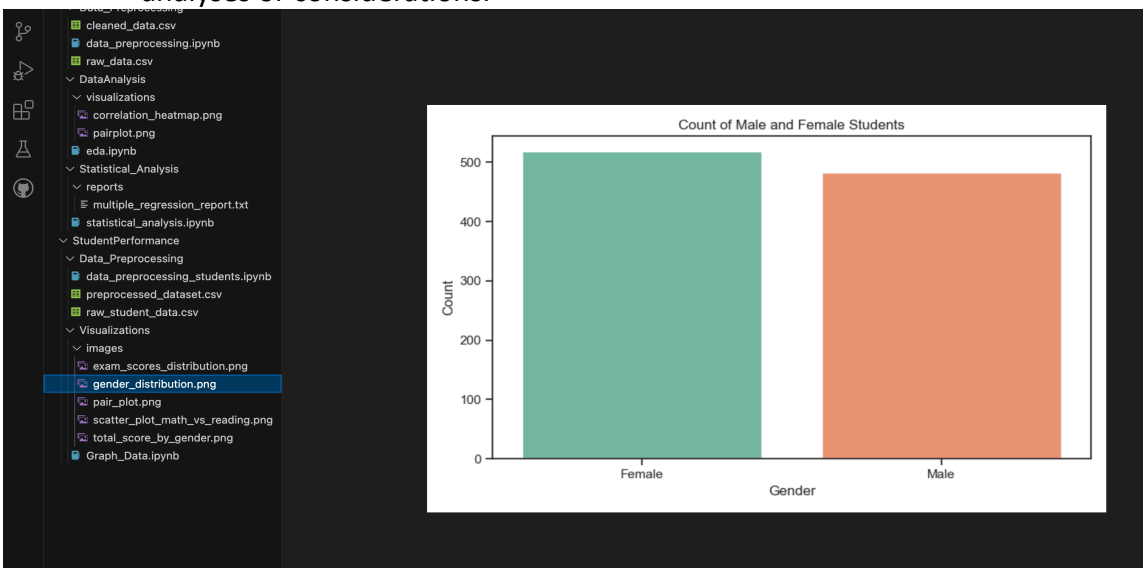




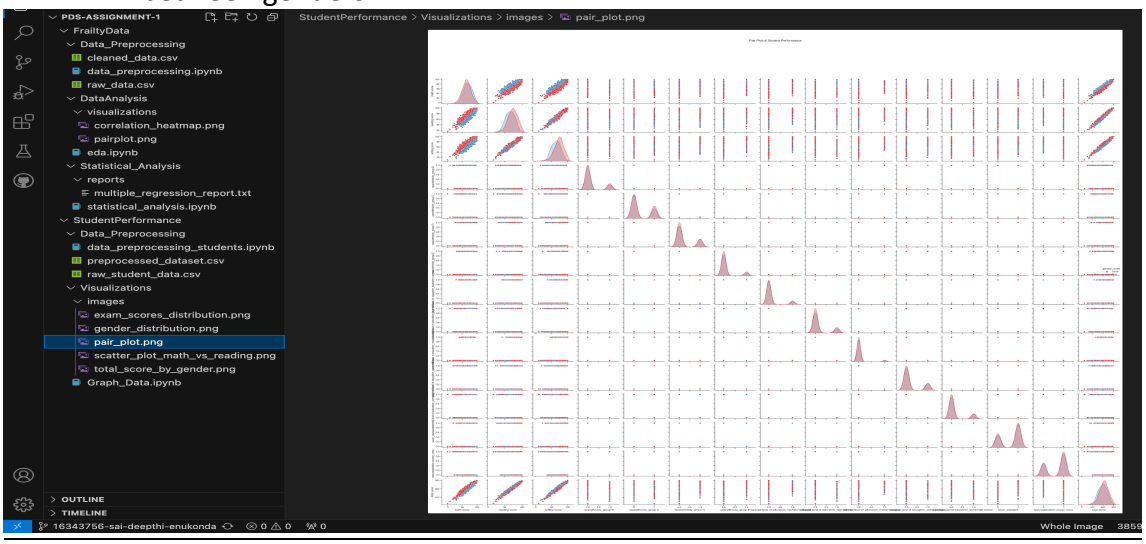
1. **Visualization 1: Histograms for Scores**:
    - Analysis Made Easier: This visualization simplifies the examination of score distributions (math, reading, and writing) by presenting them as histograms. It facilitates the assessment of score patterns, such as whether they follow a normal distribution or exhibit skewness. This aids in gaining insights into student performance across various subjects.



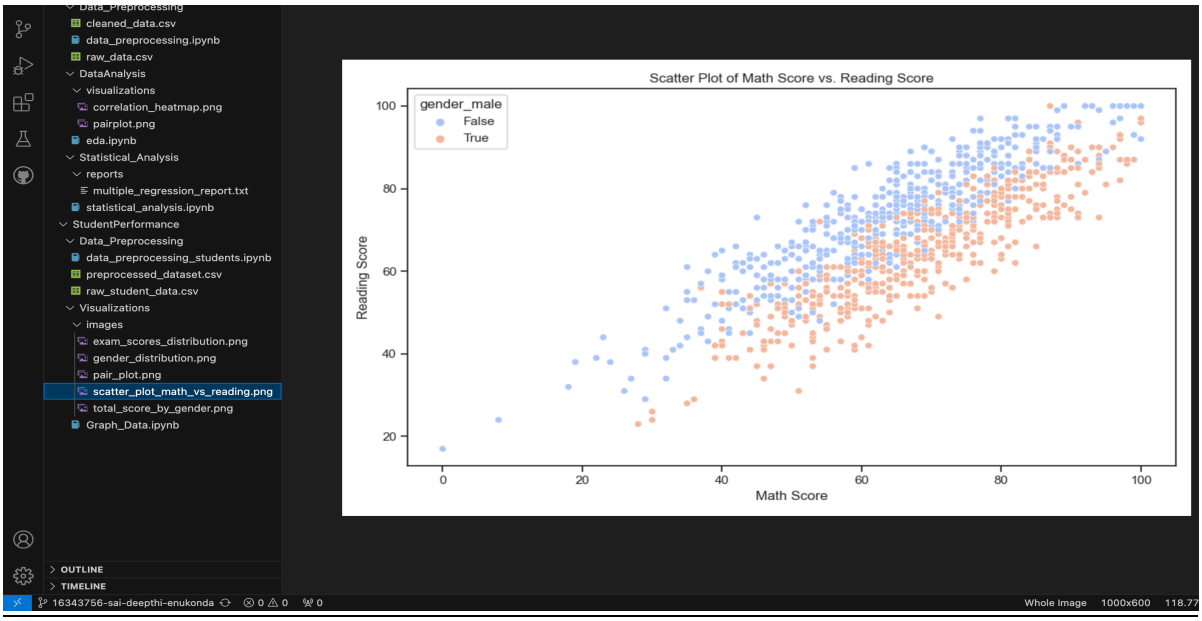
2. **Visualization 2: Bar Plot for Gender**:
    - Analysis Made Easier: By using a bar plot, it becomes straightforward to compare the counts of male and female students. This visual representation clarifies which gender is predominant in the dataset, making it convenient for gender-related analyses or considerations.



3. **Visualization 3: Box Plot for Total Score by Gender**:
    - Analysis Made Easier: The box plot offers a concise visual summary of total score distributions for each gender (male and female). It simplifies the identification of disparities in score distributions, including the presence of outliers and quartiles, between genders.

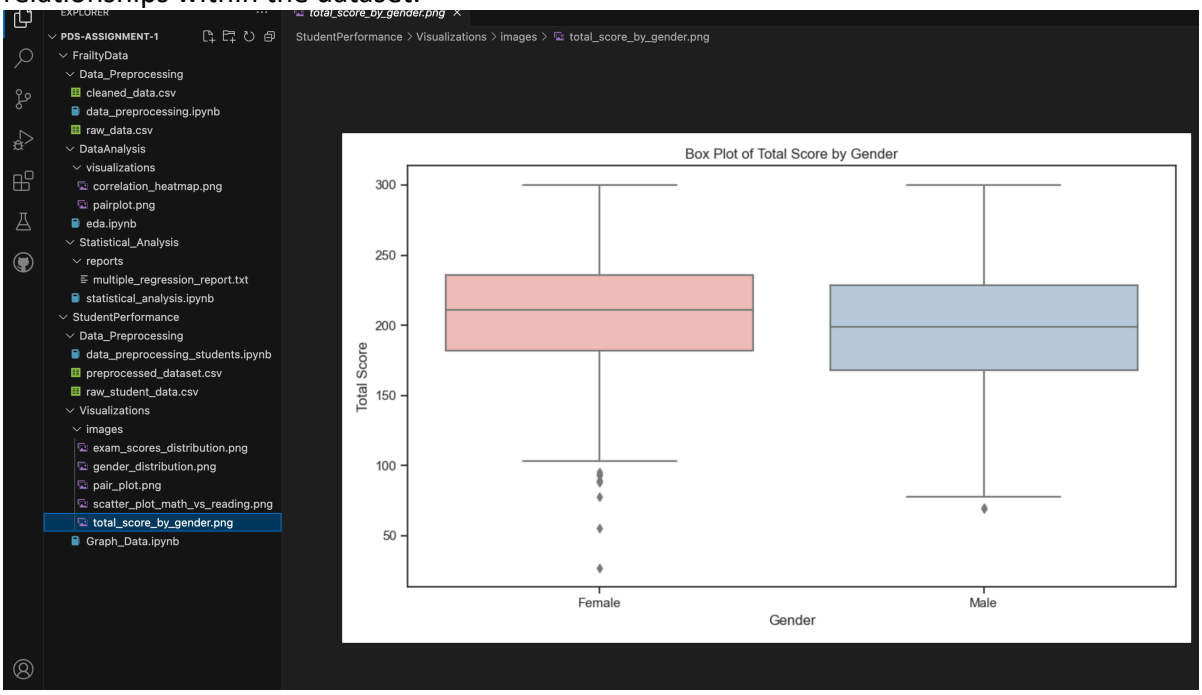


4. **Visualization 4: Pair Plot**:
    - Analysis Made Easier: This comprehensive visualization simplifies the exploration of relationships between various variables. It streamlines the process of identifying correlations and patterns across multiple variables while also allowing for an examination of how gender relates to these patterns.



5. **Visualization 5: Scatter Plot**:
    - Analysis Made Easier: The scatter plot illustrates the connection between math scores and reading scores, with data points categorized by gender. It aids in visualizing the relationship between these two specific variables and highlights any gender-based differences in performance within these subjects.

In summary, each visualization in the code serves a distinct purpose and enhances different aspects of data analysis, making it more accessible and insightful. These visualizations provide valuable insights into score distributions, gender-related trends, correlations, and variable relationships within the dataset.



Submission:

https://github.com/deepthi978/PDS-Assignment-1

Repository Details