**INTERNSHIP REPORT**

**On**

**Machine Learning and Data Science**

**with Python**

**By**

**Nitinsai.Avirneni**

**Regd. No.: 322010322007**

**PHN Technology**

**Pune.**

**( Duration: 6th April,2023 to 6th June,2023 )**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Gandhi Institute of Technology and Management**

**(DEEMED TO BE A UNIVERSITY)**

**BENGALURU, KARNATAKA, INDIA**

**SESSION:2020-2024**

# ACKNOWLEDGEMENT

I want to start by expressing my gratitude to Mr. Pradip Narayankar, Director of PHN Technology Pvt. Ltd., for providing me with the chance to complete an internship with the company.

I want to convey my appreciation to everyone that worked with me, demonstrating exceptional patience and an open mindset in creating a nice working environment.

I acknowledge the invaluable help of these extraordinary people with great delight and a deep sense of gratitude.

I extend my thanks to Mr. Pradip Narayankar, the Head of the Department, whose constructive critiques greatly contributed to my growth during my internship.

I am tremendously appreciative of my colleagues, whose support played a pivotal role in this culmination of this internship.

**Nitinsai.Avirneni**

**(322010322007)**

**Internship Objectives**

It is usually believed that college students seeking to obtain experience in a certain field should only apply for this internship. However, training internships can help a wide range of people gain practical experience and develop their abilities.

Acquired real-world expertise in data visualization using the Python matplotlib module.

Acquired knowledge of how to make various plots, including scatter plots, pie charts, histograms, and bar charts.

Acquired knowledge of Python's NumPy module and how to use it to manipulate and analyze data.

Acquired experience performing data analytic tasks on a real-world dataset.

Learn to construct various types of visualizations using Matplotlib to properly depict data.

Using NumPy, be able to manipulate arrays and apply mathematical operations on data.

**WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES**

**WEEK 1:**

**Task 1**: Prepare the sample CSV file for car model with the following column names

**Task 2**: Prepare the sample Excel file car pricing and loan amount with the following column names:

**Task 3**: Write the code to read the CSV file and the Excel file and convert them into DataFrames.

**Task 4**: Merge the two DataFrames on the basis of primary key (Company Name and Model Name).

**Task 5:** Fill the "NA" values in the merged DataFrame (if any).

**Task 6:** Iterate over the merged DataFrame and add the GST value in the "On road pricing" column

**WEEK 2:**

**Task 1**: Prepare the Titanic dataset file (download it from Google)

**Task 2**: Find out the names of passengers younger than 35 years.

**Task 3**: Print the rows from index 10 to 25 and columns 3 to 5

**Task 4**: Find out the statistics aggregate of Age & Fare using the DataFrame.agg() method.

**Task 5:** Find out the mean ticket fare price for each of the sex and cabin class combinations

**WEEK 3:**

**Task:**

Write a Python script that should perform the following tasks:

1. Read the data from the Excel spreadsheet.

2. Calculate the total quantity of each fruit.

3. Write the results to a new Excel spreadsheet with the following format:

4. The first column should contain the name of each fruit.

5. The second column should contain the total quantity of that fruit.

**WEEK 4:**

**Task:**

Write a Python program using NumPy to perform the following tasks on a given array:

1. Create a NumPy array with the following values: [1, 2, 3, 4, 5].

2. Print the shape of the array using the .shape attribute.

3. Reshape the array into a 2D array with 2 rows and 3 columns.

4. Print the shape of the new array.

5. Create a second NumPy array with the following values: [6, 7, 8, 9, 10].

6. Compute the resulting array's mean, median, and standard deviation.

**WEEK 5:**

**Task:**

1.Write a Python code to Create a figure with 2x2 subplots.

2. In the first subplot (top-left), plot a bar chart.

3. In the second subplot (top-right), plot a histogram.

4. In the third subplot (bottom-left), plot a pie chart.

5. In the fourth subplot (bottom-right), plot a scatter plot.

6. Add appropriate titles and labels to each subplot.

7. Adjust the spacing between subplots to avoid overlapping.

**Requirements**

**Software :**

* Operating system : Windows 11.
* Languages : PYTHON
* Front-End : Google Colab
* Other : GitHub

**TECHNOLOGY**

**Overview of Python:**

Python is a high-level programming language known for its simplicity, readability, and versatility.

**Readable Syntax:** Python's syntax is designed to be easy to read and write, using indentation instead of explicit braces or keywords to define code blocks. This leads to clean and organized code.

**Interpreted Language:** Python is an interpreted language, meaning that code is executed line by line by the Python interpreter. This allows for quick development and testing.

**Dynamic Typing:** Python uses dynamic typing, where variable types are determined at runtime. This allows for flexible coding but also requires careful handling of data types.

**Object-Oriented Programming (OOP):** Python supports OOP principles, allowing developers to create classes and objects, encapsulate data, and implement inheritance and polymorphism.

**High-Level Data Types**: Python offers built-in data structures like lists, dictionaries, sets, and tuples, making it easy to work with various types of data efficiently.

**Ease of Learning**: Python's simple and consistent syntax makes it an excellent language for beginners. Its readability and the availability of resources, tutorials, and documentation contribute to a gentle learning curve.

**Overview of Machine Learning:**

Machine learning is a subset of artificial intelligence (AI) that focuses on creating algorithms and models that enable computers to learn from and make predictions or decisions based on data.

It involves the development of techniques that allow systems to improve their performance over time without being explicitly programmed.

Steps in the Machine Learning Process:

Data Collection: Gathering relevant data for training and testing.

Data Preprocessing: Cleaning, transforming, and preparing data for model training.

Feature Engineering: Selecting, transforming, or creating features to improve model performance.

Model Selection: Choosing an appropriate algorithm or model architecture.

Model Training: Feeding the training data into the model to let it learn patterns.

Model Evaluation: Testing the model on new, unseen data to assess its performance.

Hyperparameter Tuning: Adjusting model parameters to optimize performance.

Model Deployment: Integrating the trained model into real-world applications for making predictions.

Applications:

1. Image and Speech Recognition: Recognizing objects, faces, and speech patterns.
2. Natural Language Processing: Analyzing and generating human language text.
3. Recommendation Systems: Providing personalized recommendations based on user behavior.

**Overview of Data Science:**

* Data science is a field that studies data and how to extract meaning from it. This includes collecting, cleaning, analyzing, and visualizing data.
* Data science is a multidisciplinary field that combines elements of statistics, machine learning, computer science, and business analytics.
* Data scientists use a variety of tools and techniques to analyze data, including statistical analysis, machine learning algorithms, and visualization software.
* Data science is used in a wide variety of industries, including healthcare, finance, retail, and manufacturing.
* Data scientists are in high demand, as businesses are increasingly looking for ways to use data to make better decisions.

Some of the most popular machine learning algorithms that can be used in Python:

Linear regression: This algorithm is used to predict continuous values, such as the price of a house or the number of sales.

Logistic regression: This algorithm is used to predict binary values, such as whether a customer will click on an ad or not.

Decision trees: This algorithm is used to make decisions based on a set of rules. Random forests: This algorithm is a combination of many decision trees.

Support vector machines: This algorithm is used to find the best hyperplane that separates two classes of data.

**Week\_1:**

**Code:**

import pandas as pd

df = pd.read\_csv('/content/carmodel.csv')

print(df)

df1 = pd.read\_excel('/content/carpricing.xlsx')

print(df1)

merged\_df = pd.merge(df, df1, on='Model Name')

merged\_df = merged\_df.fillna(0)

print(merged\_df)

gst = 0.18

for index, row in merged\_df.iterrows():

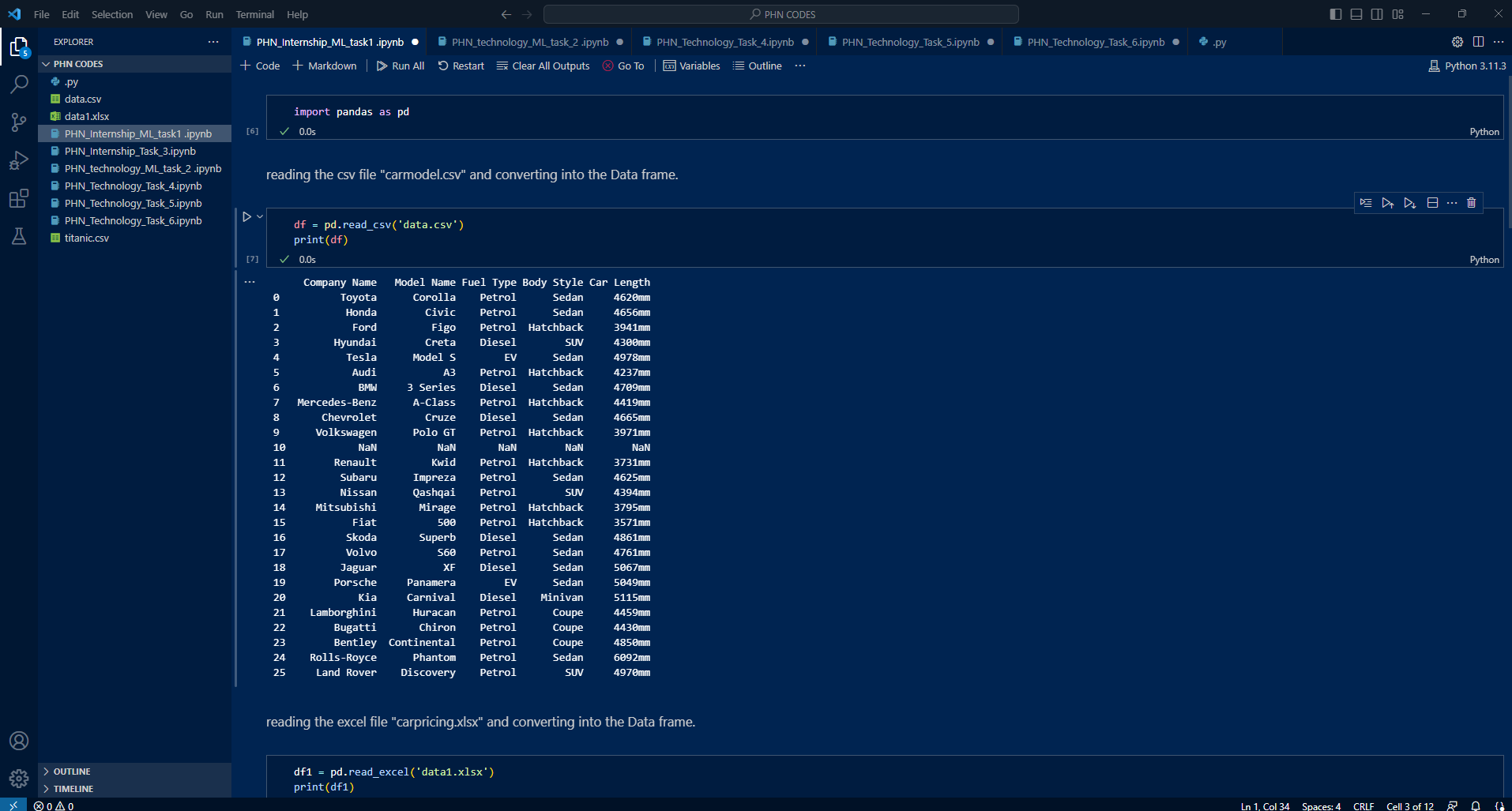
    on\_road\_price = row['On Road Pricing']

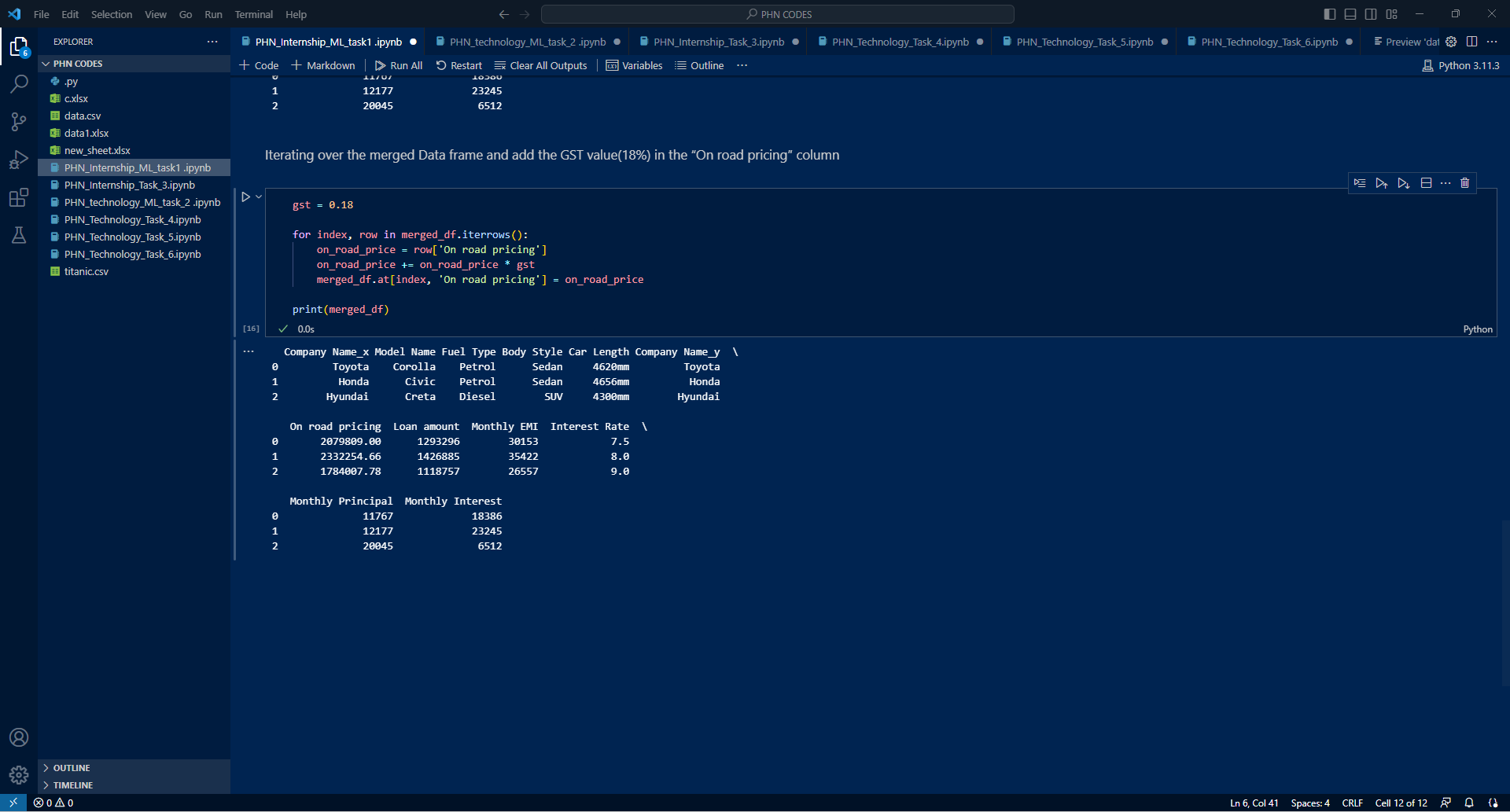
    on\_road\_price += on\_road\_price \* gst

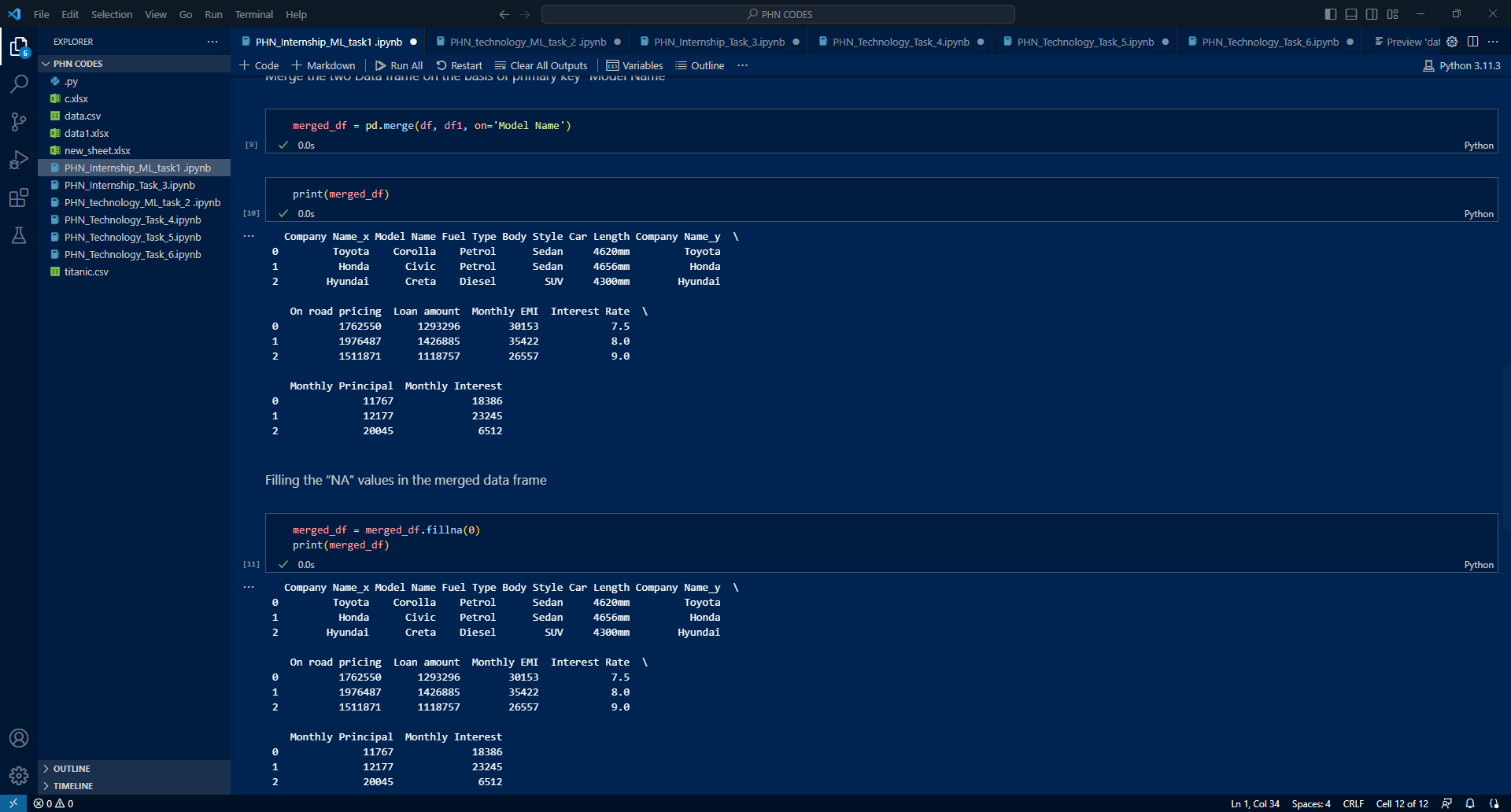
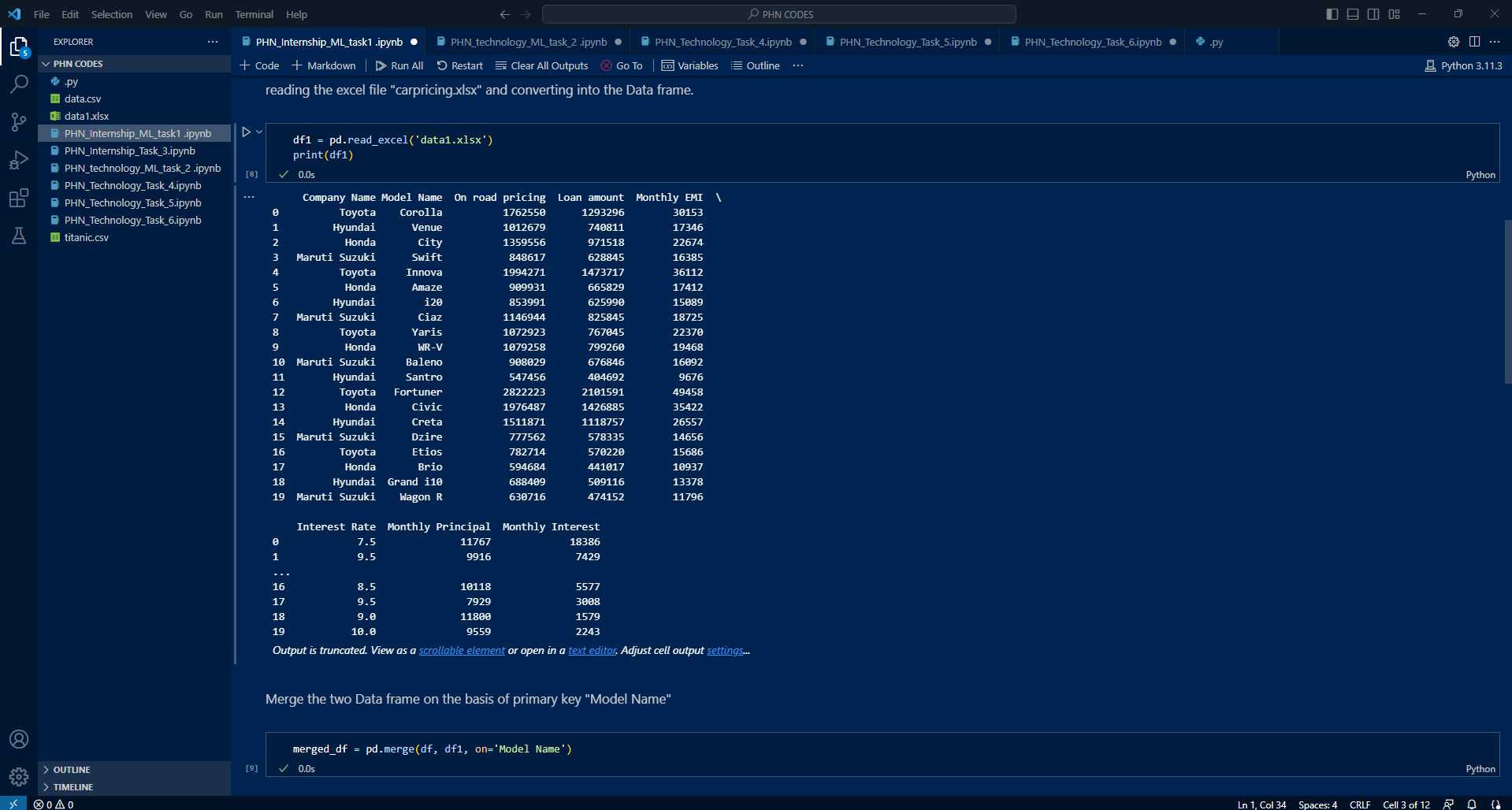
    merged\_df.at[index, 'On Road Pricing'] = on\_road\_price

print(merged\_df)

**Output:**

****

****

****

**Week\_2:**

**Code:**

import pandas as pd

titanic\_df = pd.read\_csv('/content/titanic.csv')

print(titanic\_df.head())

young\_passengers = titanic\_df[titanic\_df['Age'] < 35]

names = young\_passengers['Name']

print(names)

subset = titanic\_df.iloc[10:26, 3:6]

print(subset)

stats = titanic\_df[['Age', 'Fare']].agg(['mean', 'median', 'min', 'max'])

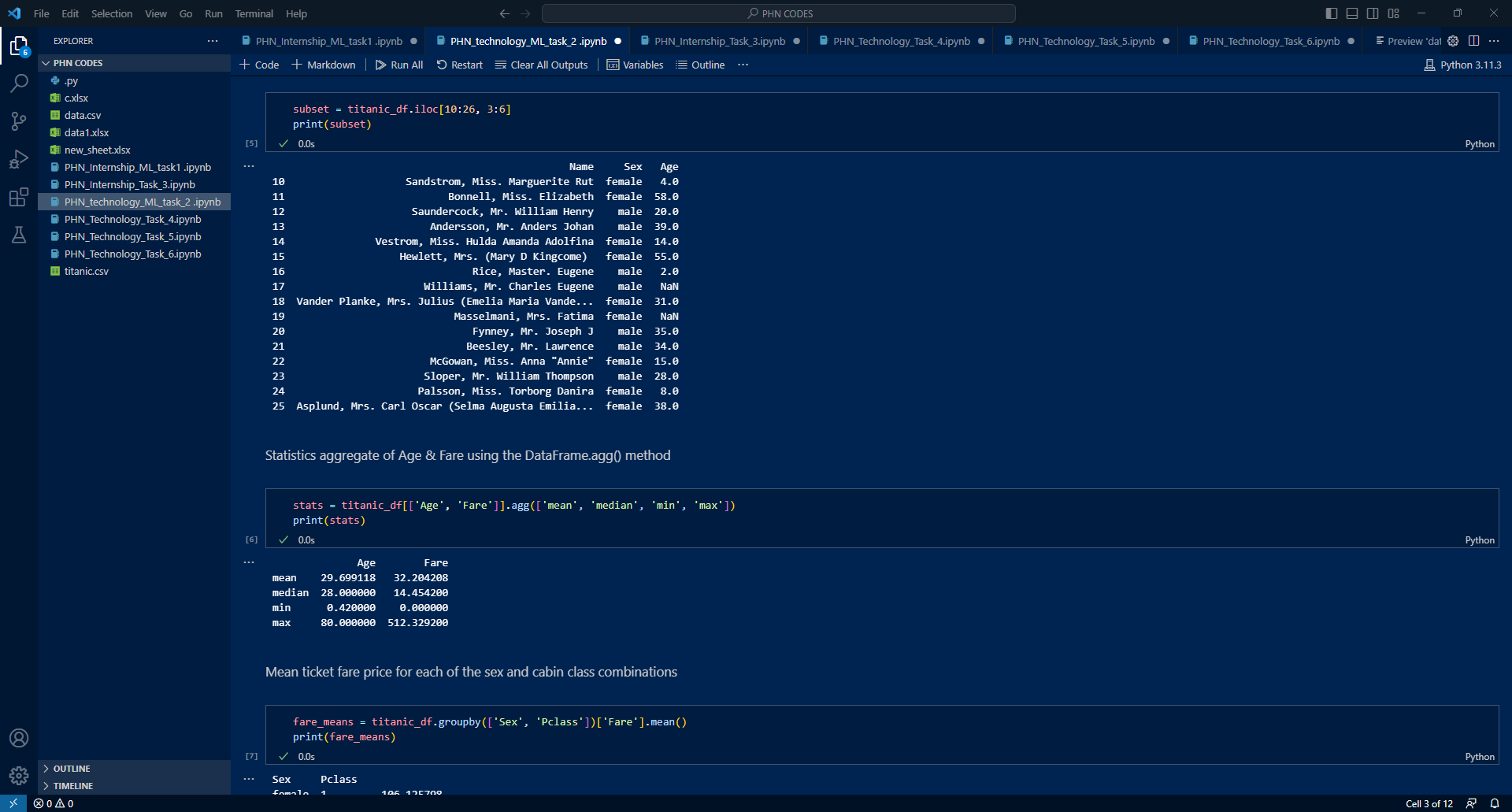
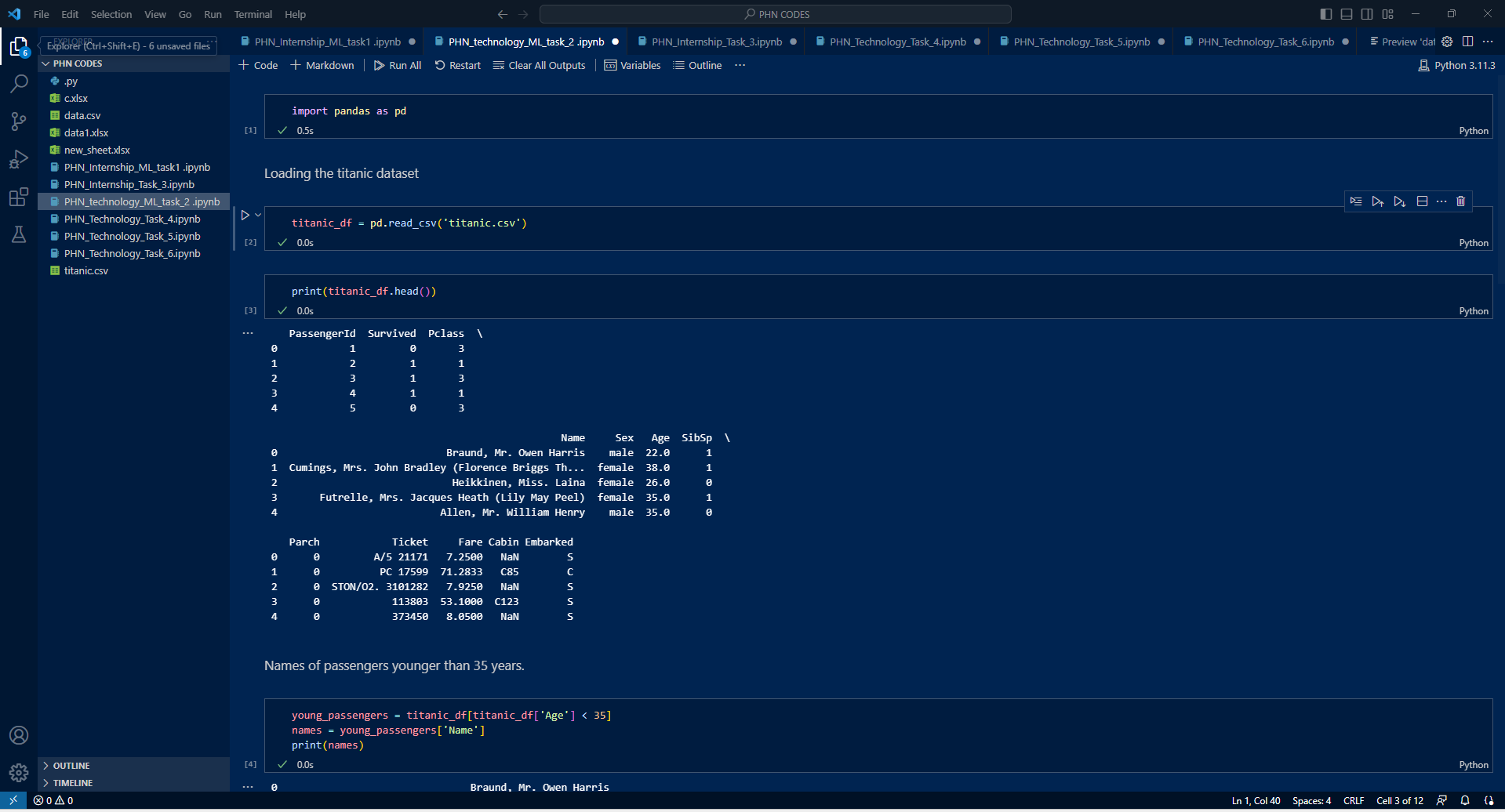
print(stats)

fare\_means = titanic\_df.groupby(['Sex', 'Pclass'])['Fare'].mean()

print(fare\_means)

**Output:**

****

****

**Week\_3:**

**Code:**

import openpyxl

workbook = openpyxl.load\_workbook('/content/original\_sheet.xlsx')

worksheet = workbook.active

worksheet = workbook.active

# Iterate through all cells in a column

for cell in worksheet['A']:

    print(cell.value)

# Iterate through all rows in a worksheet

for row in worksheet.iter\_rows():

    for cell in row:

        print(cell.value)

fruit\_totals = {}

fruit\_totals = {}

# Iterate through all rows in the worksheet (skipping the header row)

for row in worksheet.iter\_rows(min\_row=2, values\_only=True):

    fruit = row[0]

    quantity = row[1]

    if fruit in fruit\_totals:

        fruit\_totals[fruit] += quantity

    else:

        fruit\_totals[fruit] = quantity

new\_workbook = openpyxl.Workbook()

new\_worksheet = new\_workbook.active

# Write the headers to the new worksheet

new\_worksheet['A1'] = 'Fruit'

new\_worksheet['B1'] = 'Total Quantity'

# Write the fruit totals to the new worksheet

row\_index = 2

for fruit, total\_quantity in fruit\_totals.items():

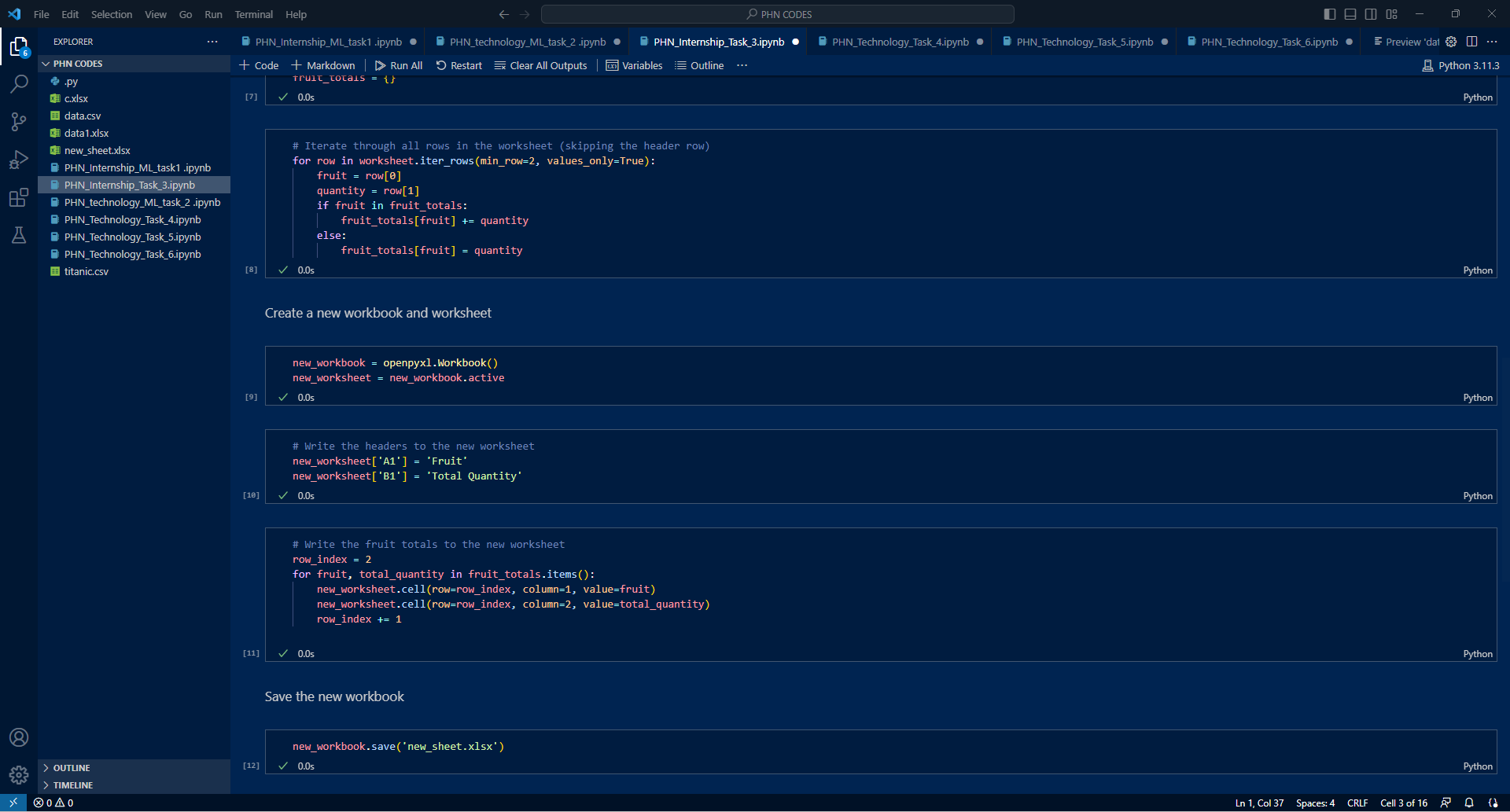
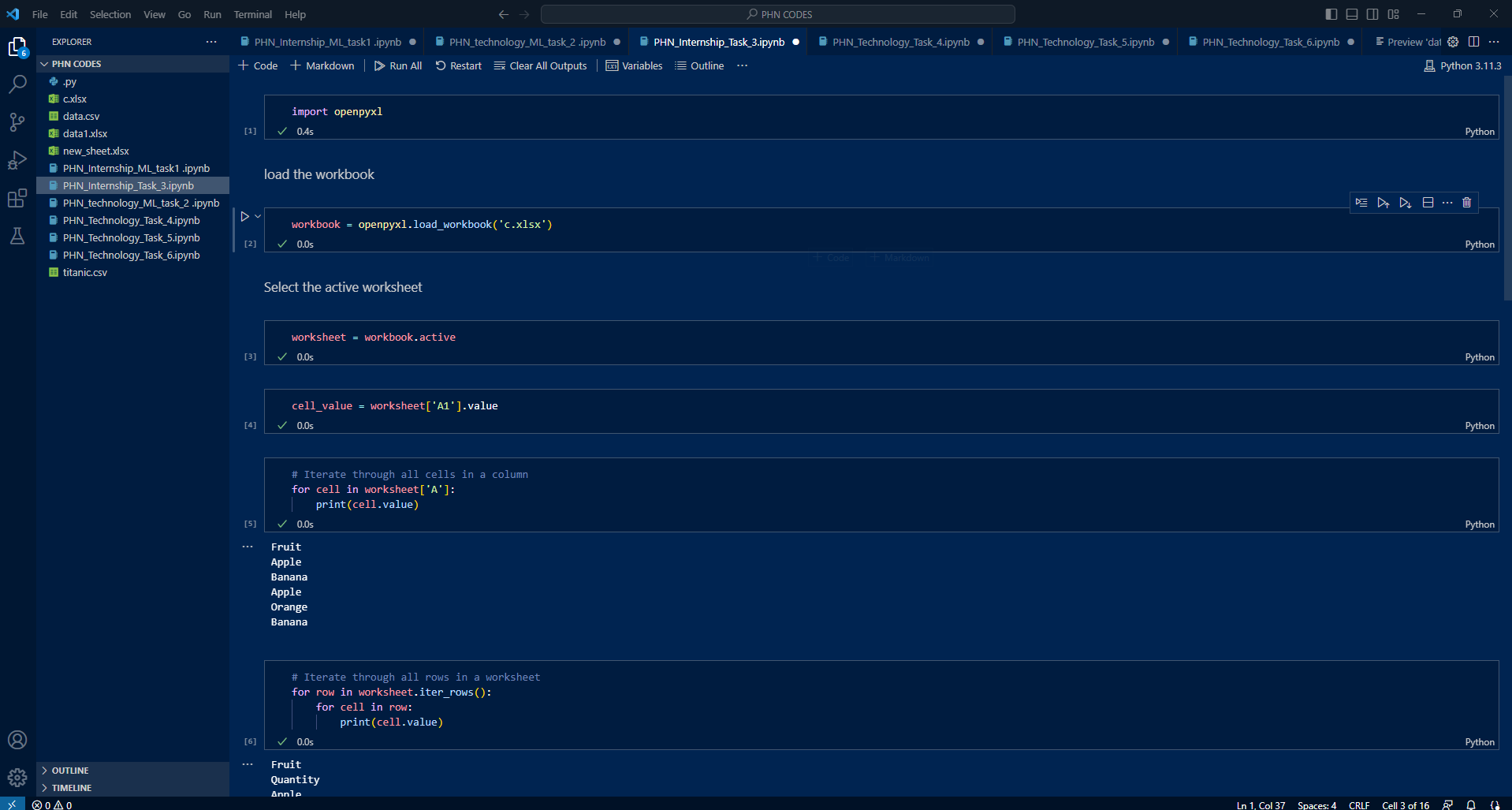
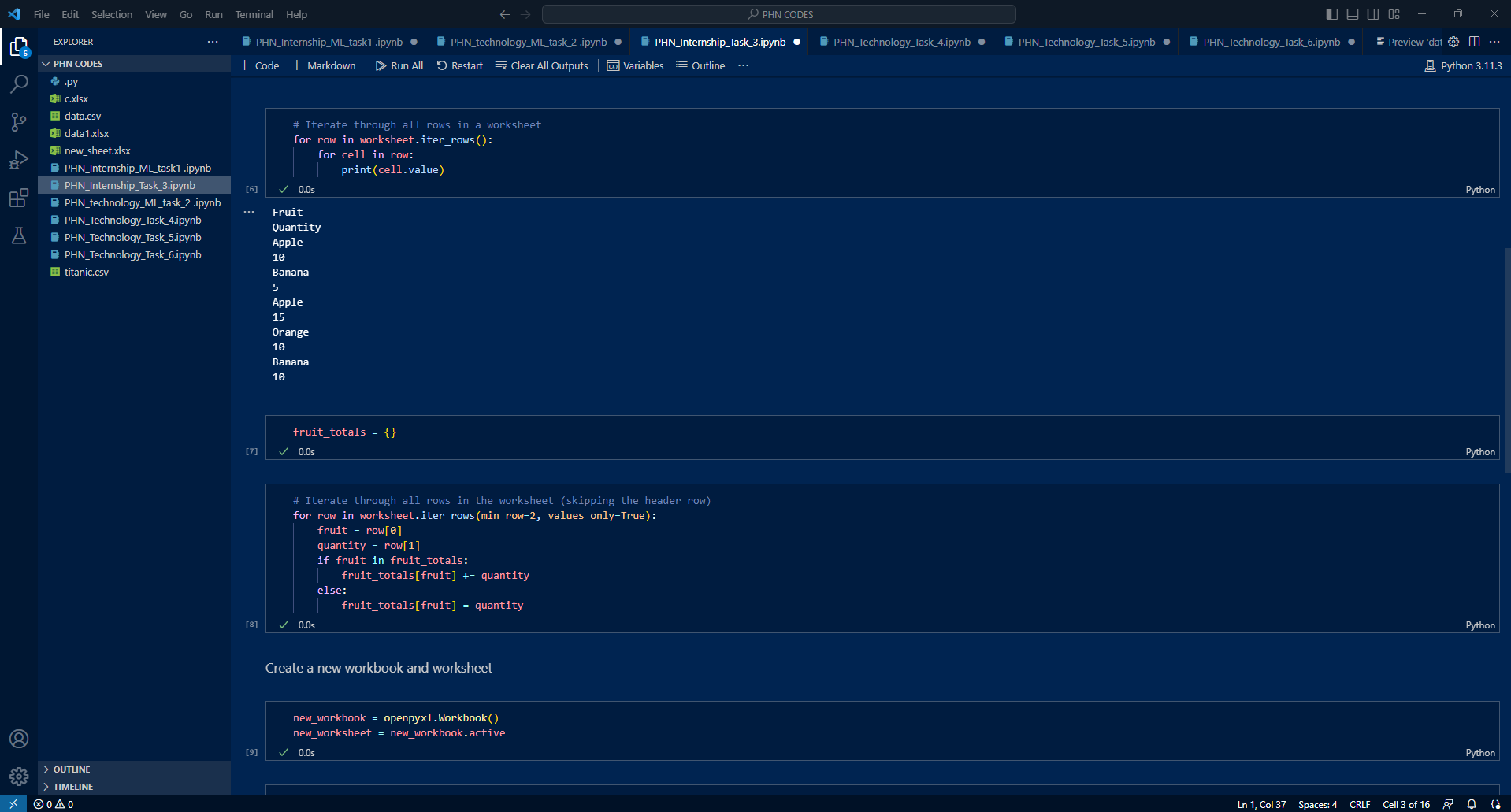
    new\_worksheet.cell(row=row\_index, column=1, value=fruit)

    new\_worksheet.cell(row=row\_index, column=2, value=total\_quantity)

    row\_index += 1

new\_workbook.save('new\_sheet.xlsx')

**Output:**

****

**Week\_4:**

**Code:**

import numpy as np

ar1 = np.array([1,2,3,4,5])

print(ar1)

print(ar1.shape)

ar1 = np.array([0,1,2,3,4,5])

reshaped\_ar1 = ar1.reshape(2, 3)

print(reshaped\_ar1)

print(reshaped\_ar1.shape)

ar1=np.array([1,2,3,4,5])

ar2=np.array([6,7,8,9,10])

print(ar2)

concatenated\_ar = np.concatenate((ar1,ar2), axis=0)

print(concatenated\_ar)

mean\_value = np.mean(concatenated\_ar)

median\_value = np.median(concatenated\_ar)

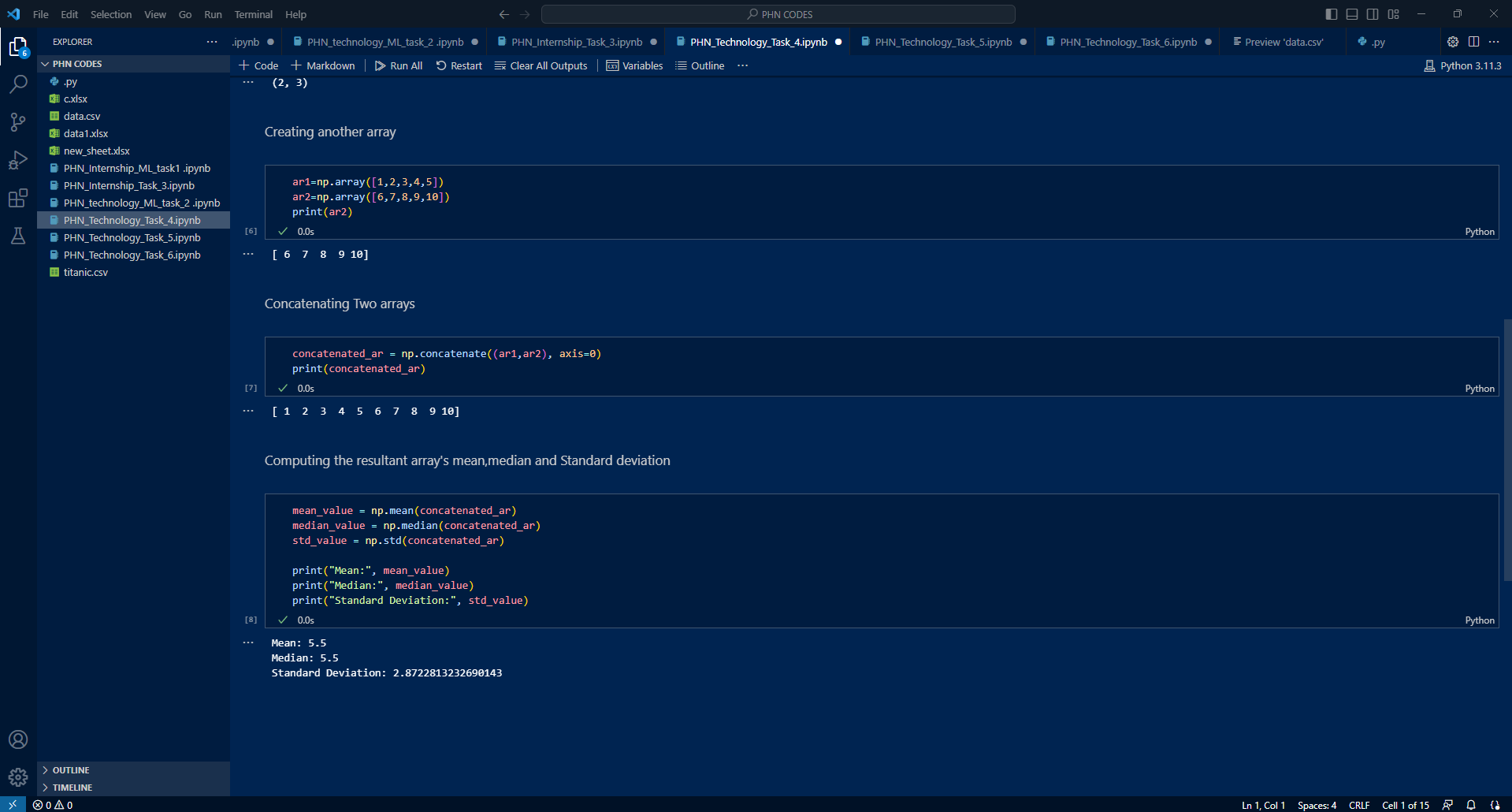
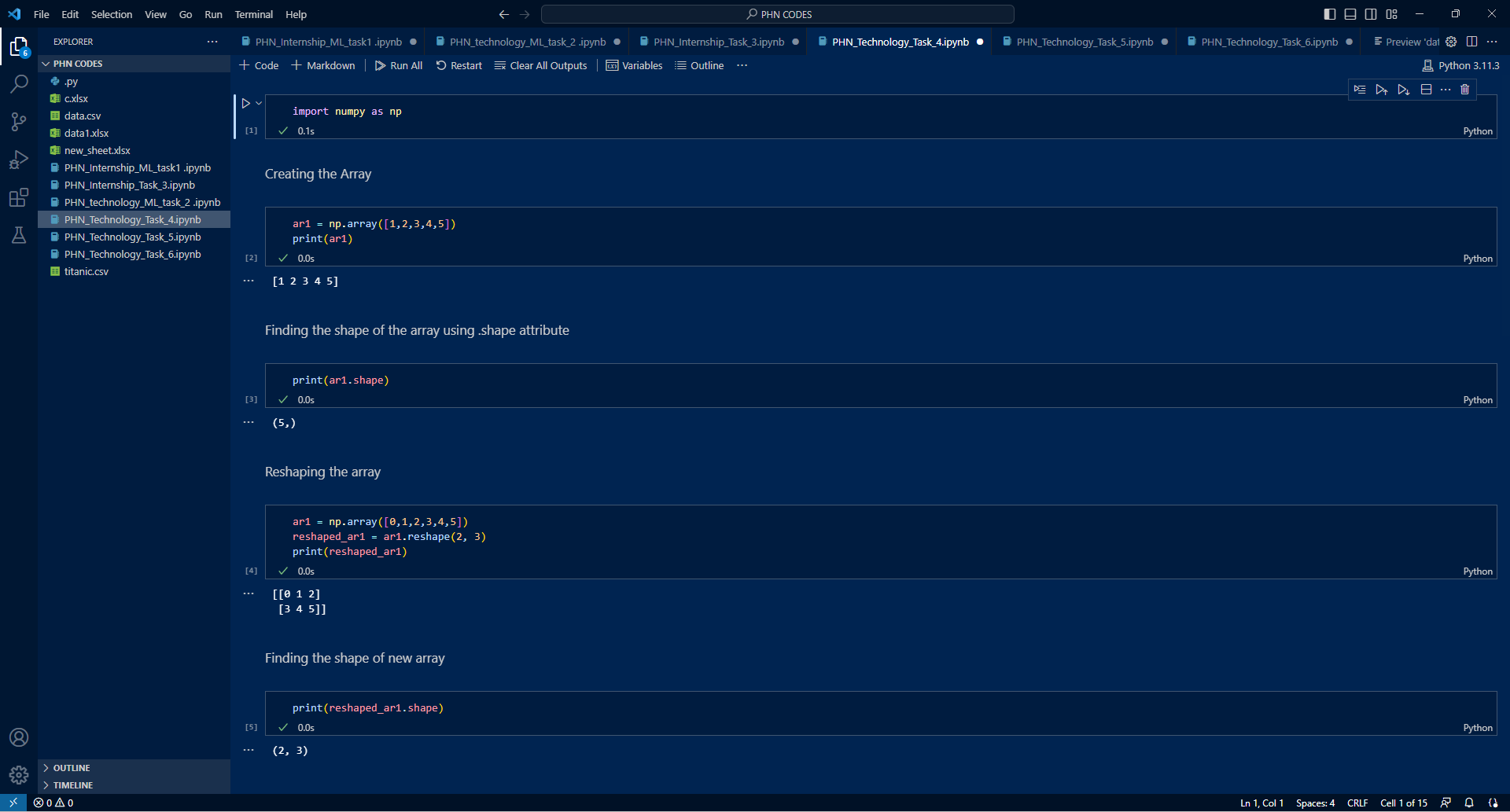
std\_value = np.std(concatenated\_ar)

print("Mean:", mean\_value)

print("Median:", median\_value)

print("Standard Deviation:", std\_value)

**Output:**

****

**Week\_5:**

**Code:**

import matplotlib.pyplot as plt

# Data for bar chart

categories = ['Category A', 'Category B', 'Category C', 'Category D', 'Category E']

values = [23, 56, 41, 62, 19]

# Data for histogram

data = [12, 17, 21, 18, 14, 13, 16, 9, 12, 15, 19, 11, 14, 16, 20, 18, 15, 13, 16, 11, 10]

# Data for pie chart

labels = ['Apple', 'Banana', 'Orange', 'Mango']

sizes = [30, 25, 15, 30]

# Data for scatter plot

x\_values = [1, 2, 3, 4, 5]

y\_values = [2, 5, 3, 6, 4]

# Create the figure and subplots

fig, axs = plt.subplots(2, 2, figsize=(10, 8))

# Plot the bar chart

bar\_width =0.3 # Adjust the width of the bars

axs[0, 0].bar(categories, values, width=bar\_width)

axs[0, 0].set\_title('Bar Chart')

axs[0, 0].set\_xlabel('Categories')

axs[0, 0].set\_ylabel('Values')

# Plot the histogram

axs[0, 1].hist(data, bins=5)

axs[0, 1].set\_title('Histogram')

axs[0, 1].set\_xlabel('Data')

axs[0, 1].set\_ylabel('Frequency')

# Plot the pie chart

axs[1, 0].pie(sizes, labels=labels, autopct='%1.1f%%')

axs[1, 0].set\_title('Pie Chart')

# Plot the scatter plot

axs[1, 1].scatter(x\_values, y\_values)

axs[1, 1].set\_title('Scatter Plot')

axs[1, 1].set\_xlabel('X values')

axs[1, 1].set\_ylabel('Y values')

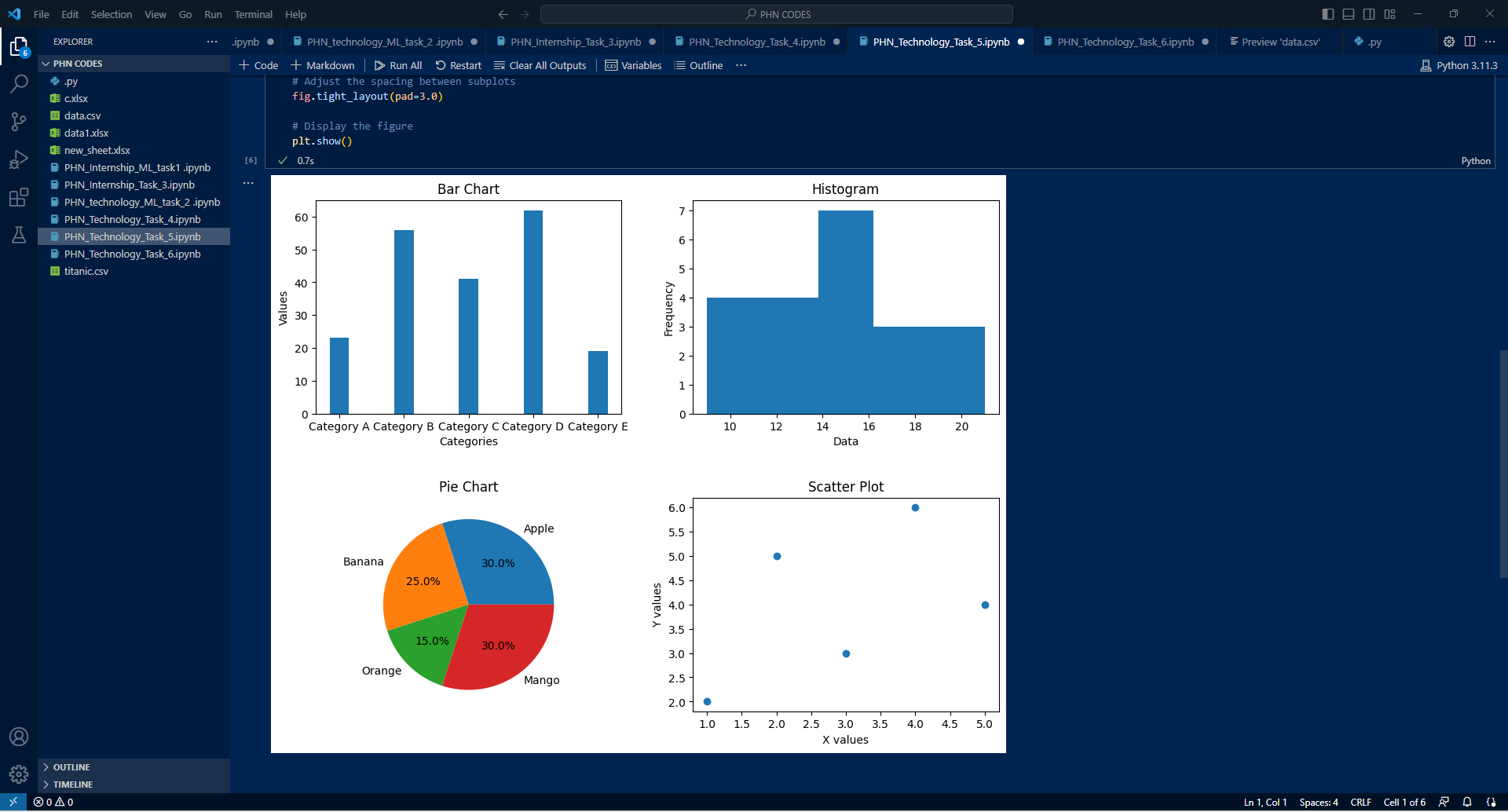
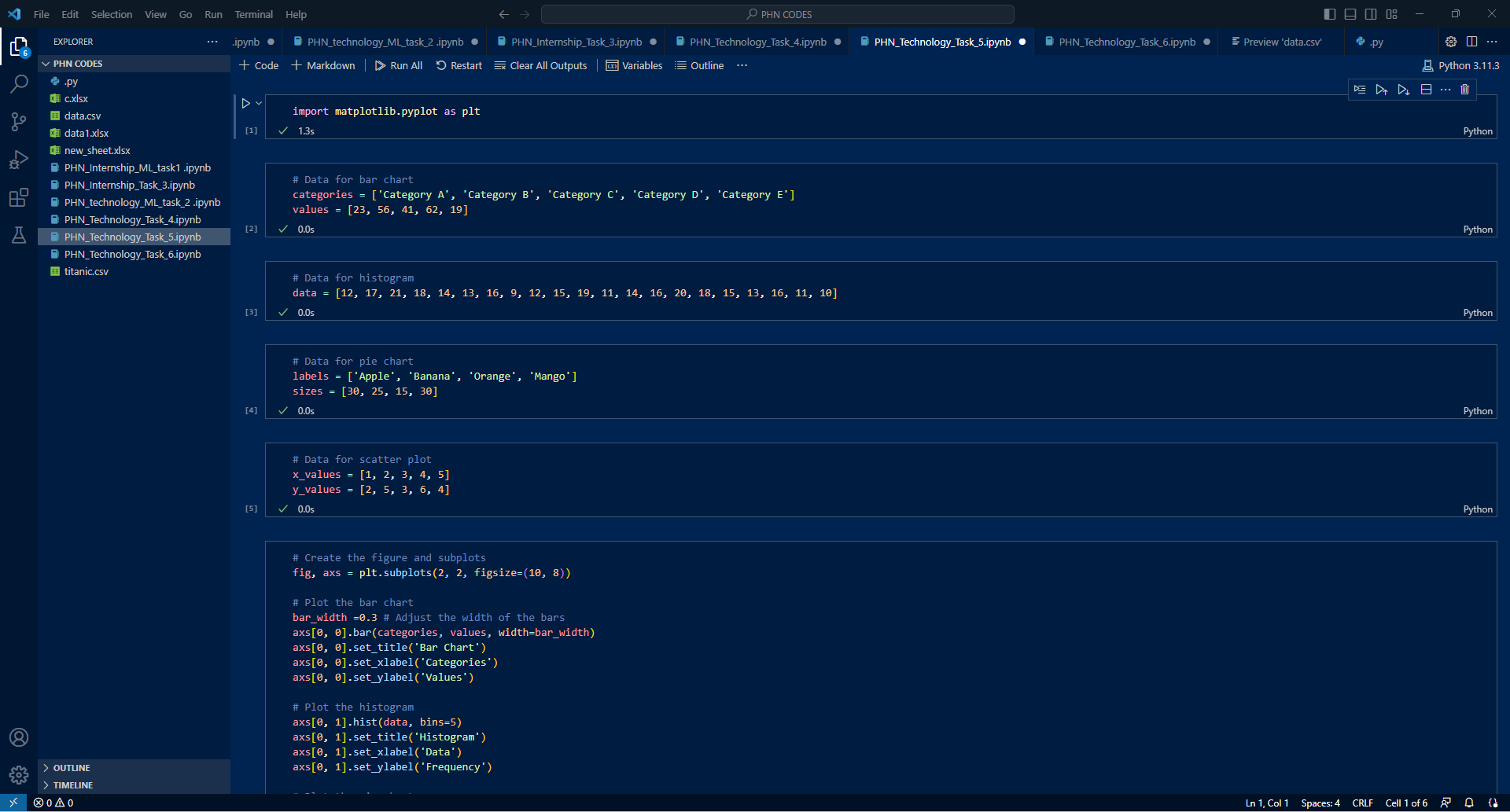
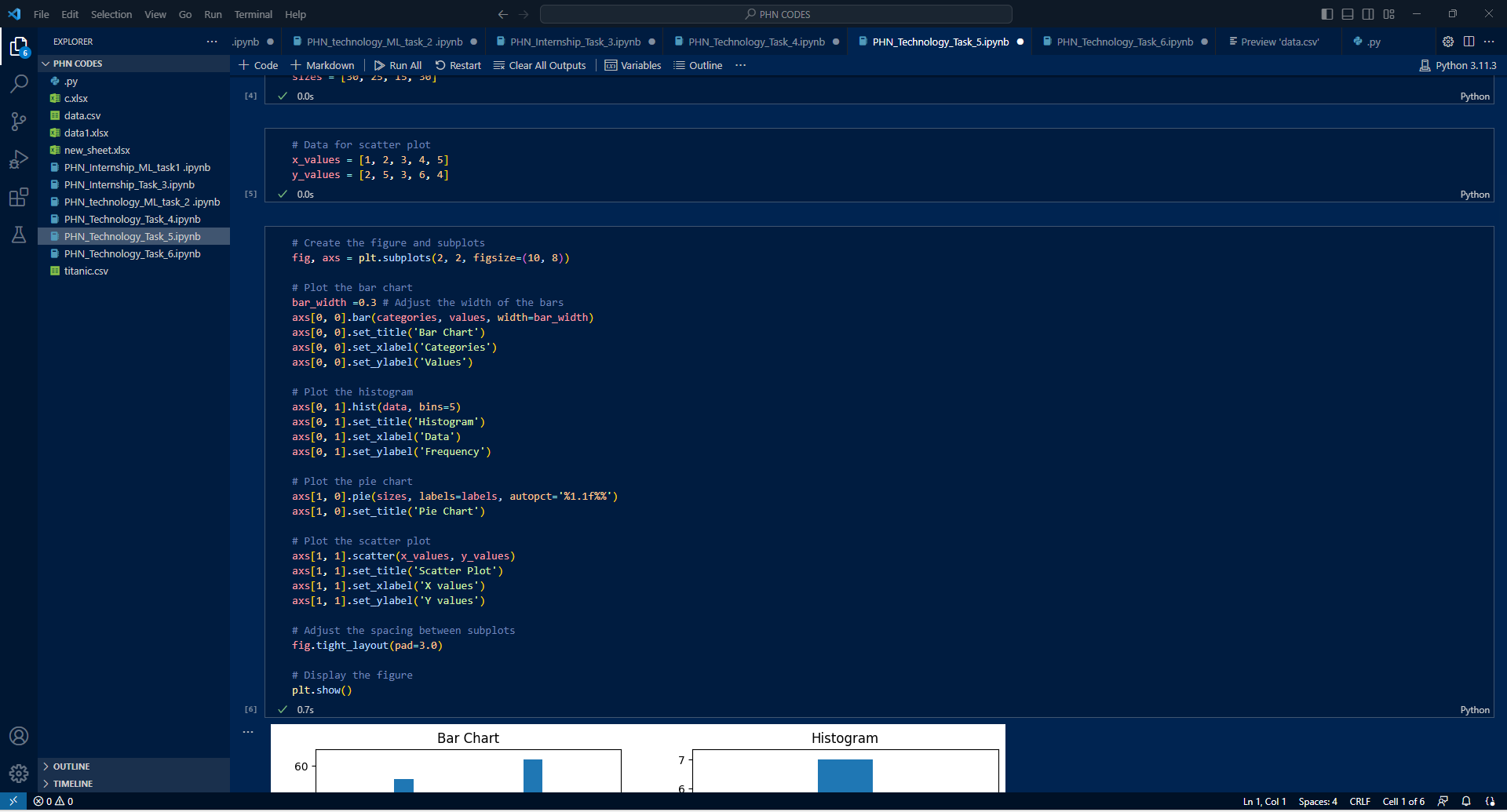
# Adjust the spacing between subplots

fig.tight\_layout(pad=3.0)

# Display the figure

plt.show()

**Output:**

****

**Week\_6:**

**Code:**

import numpy as np

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, r2\_score

# Step 1: Dataset Generation

# Generate input variables x and corresponding output values y using the equation y = mx + c

m = np.random.uniform(-5, 5)  # slope

c = np.random.uniform(-10, 10)  # intercept

num\_samples = 1000  # number of samples in the dataset

x = np.random.uniform(-10, 10, num\_samples)

y = m \* x + c

# Step 3: Model Training

# Split the dataset into training and test sets

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=42)

# Reshape the input variables for scikit-learn's LinearRegression

x\_train = x\_train.reshape(-1, 1)

x\_test = x\_test.reshape(-1, 1)

# Create and train the Linear Regression model

model = LinearRegression()

model.fit(x\_train, y\_train)

# Step 4: Model Evaluation

# Make predictions on the test set

y\_pred = model.predict(x\_test)

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error (MSE):", mse)

print("R-squared (R2):", r2)

# Step 5: Prediction

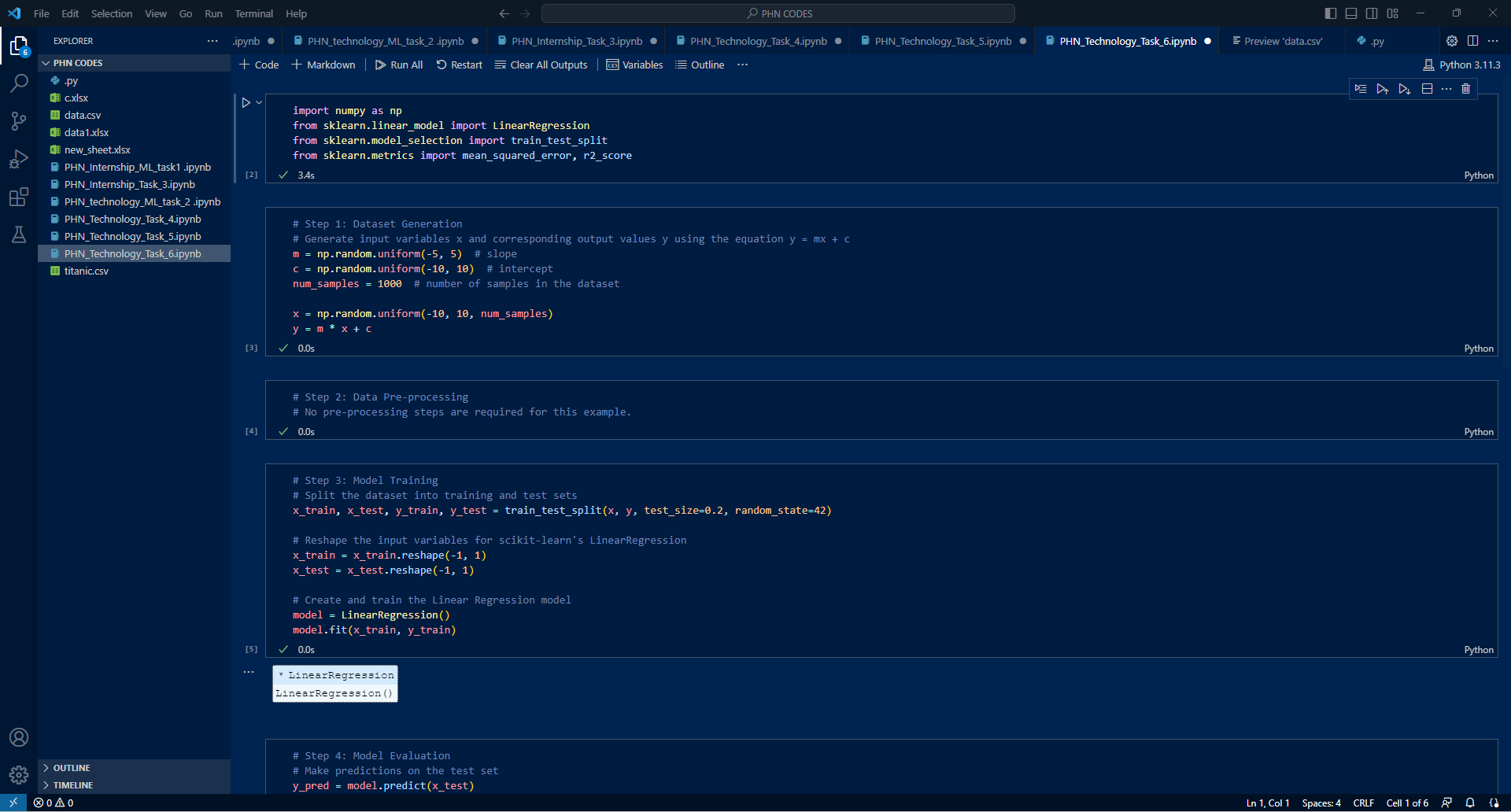
# Predict coefficients for a new input value

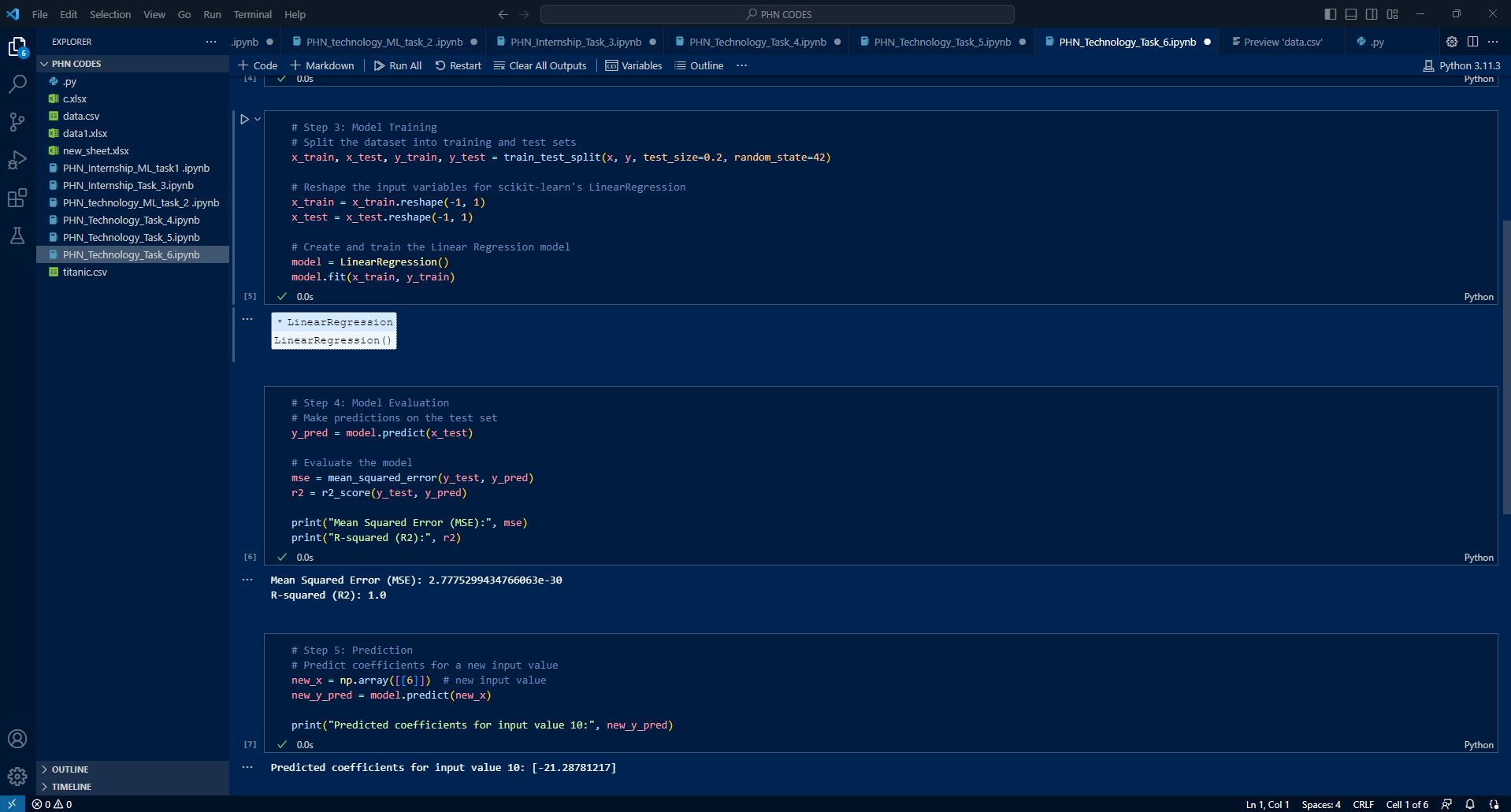
new\_x = np.array([[6]])  # new input value

new\_y\_pred = model.predict(new\_x)

print("Predicted coefficients for input value 10:", new\_y\_pred)

**Output:**

****

****

## Certification:

## 119483-1