ASSIGNMENT-4

Student Name : Pavan Gandavarapu

Student Id : 700741173

Video link : https://drive.google.com/file/d/1oNqqHKkA9HD\_M89Dl3L2pVuw2Impzfop/view

GIT Hub URL: https://github.com/PavanGandavarapu/ML\_Assignment\_4

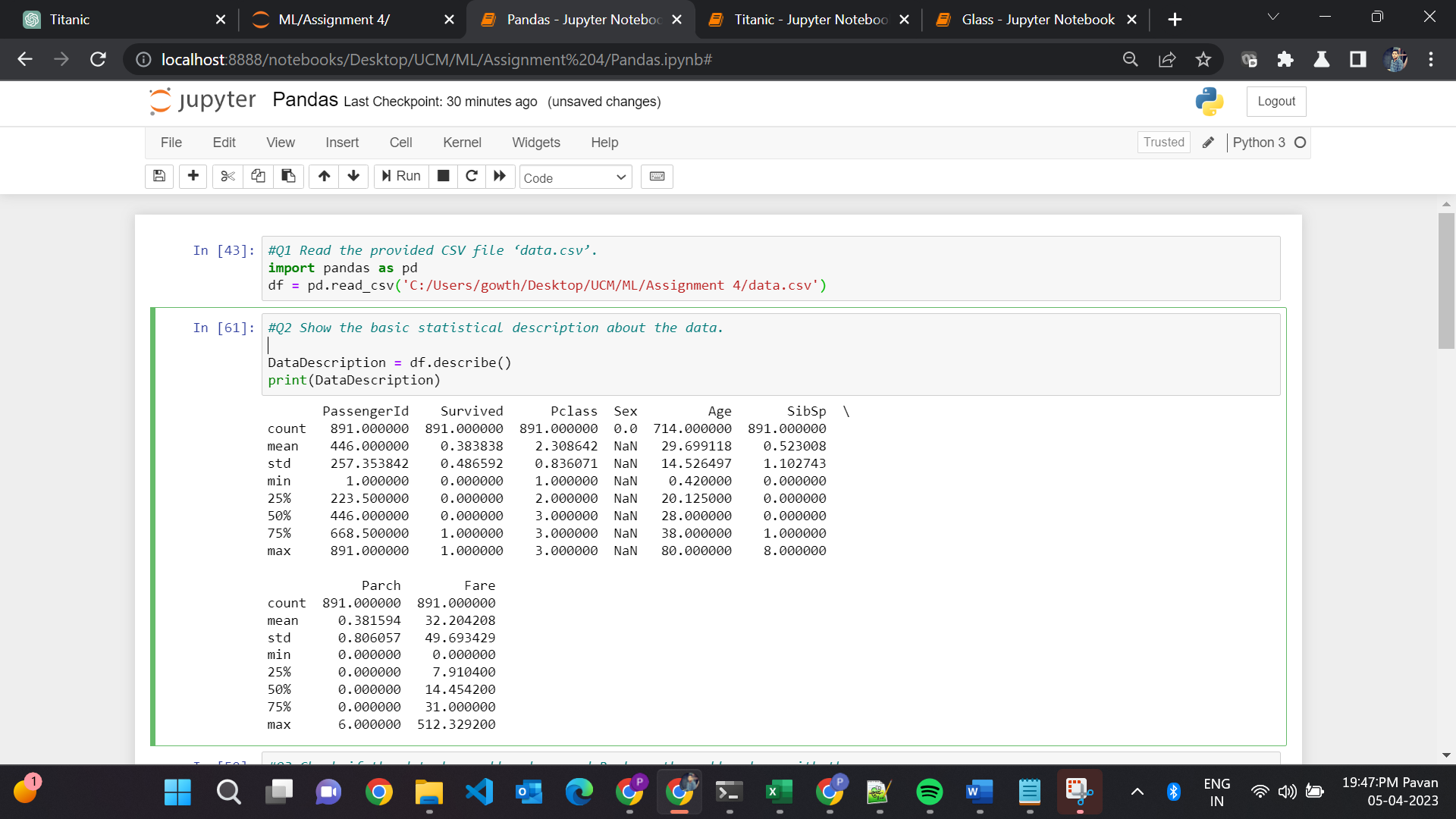
**1. Pandas**

1. Read the provided CSV file ‘data.csv’.



* Import the pandas library, which is used for data manipulation and analysis.
* Use the read\_csv() function from pandas to read the CSV file named data.csv
* The DataFrame object is assigned to the variable df, which allows for further manipulation and analysis of the data in the file.

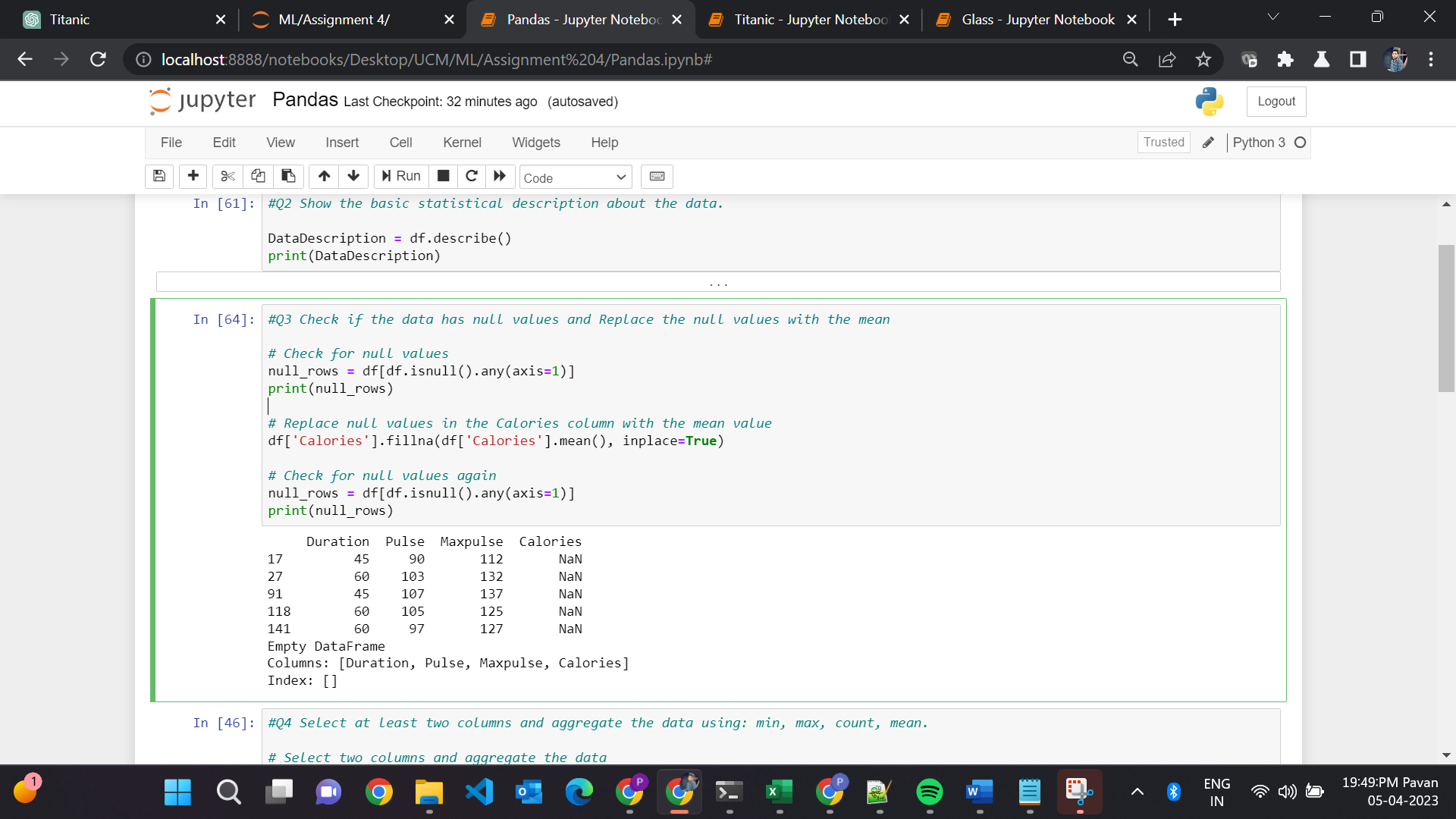
2. Show the basic statistical description about the data.



* The describe() function is applied to the pandas DataFrame object df.
* The output of the describe() function is stored in a new pandas DataFrame object named DataDescription.
* The contents of DataDescription are printed to the console using the print() function.

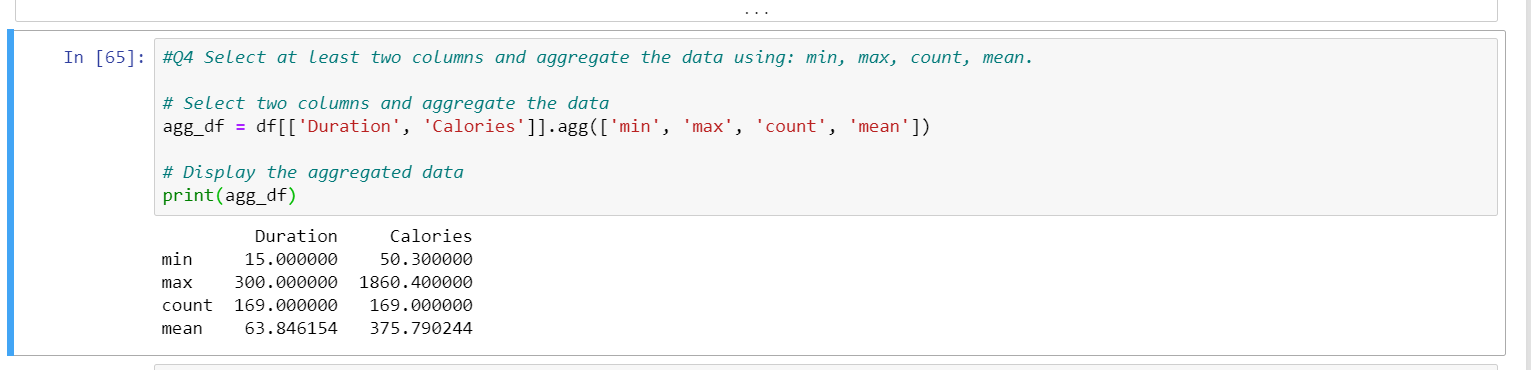
3. Check if the data has null values.

a. Replace the null values with the mean



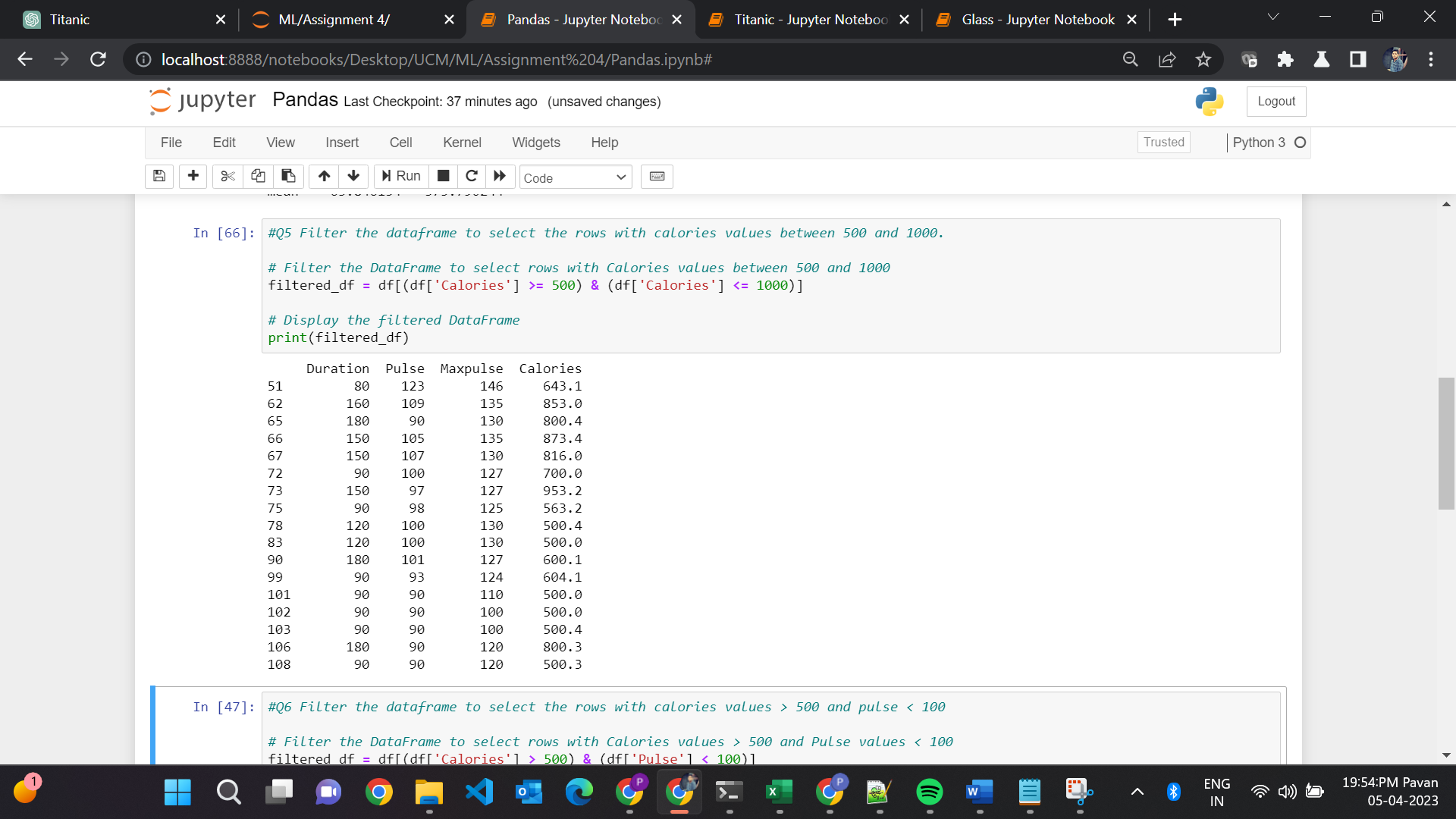
* It checks if the DataFrame object df contains any null values using the isnull() function, and returns the rows containing null values using the any() function with axis=1.
* It prints the rows containing null values to the console using the print() function.
* It replaces the null values in the Calories column with the mean value of the column using the fillna() function with inplace=True.
* It checks for null values again and prints the rows containing null values to the console.

4. Select at least two columns and aggregate the data using: min, max, count, mean



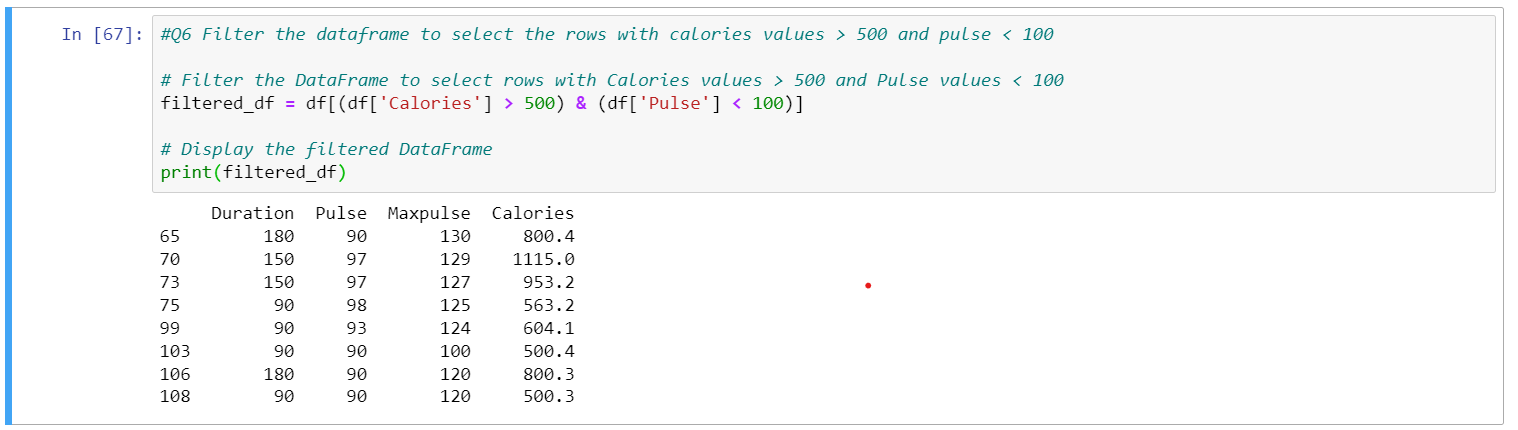
* The DataFrame object df is selected with two columns (Duration and Calories) and assigned to a new DataFrame object.
* The agg() function is used on the new DataFrame object with four aggregate methods (min, max, count, and mean).
* The output of the aggregation is stored in a new DataFrame object agg\_df.
* The contents of agg\_df are printed to the console using the print() function.

5. Filter the dataframe to select the rows with calories values between 500 and 1000.



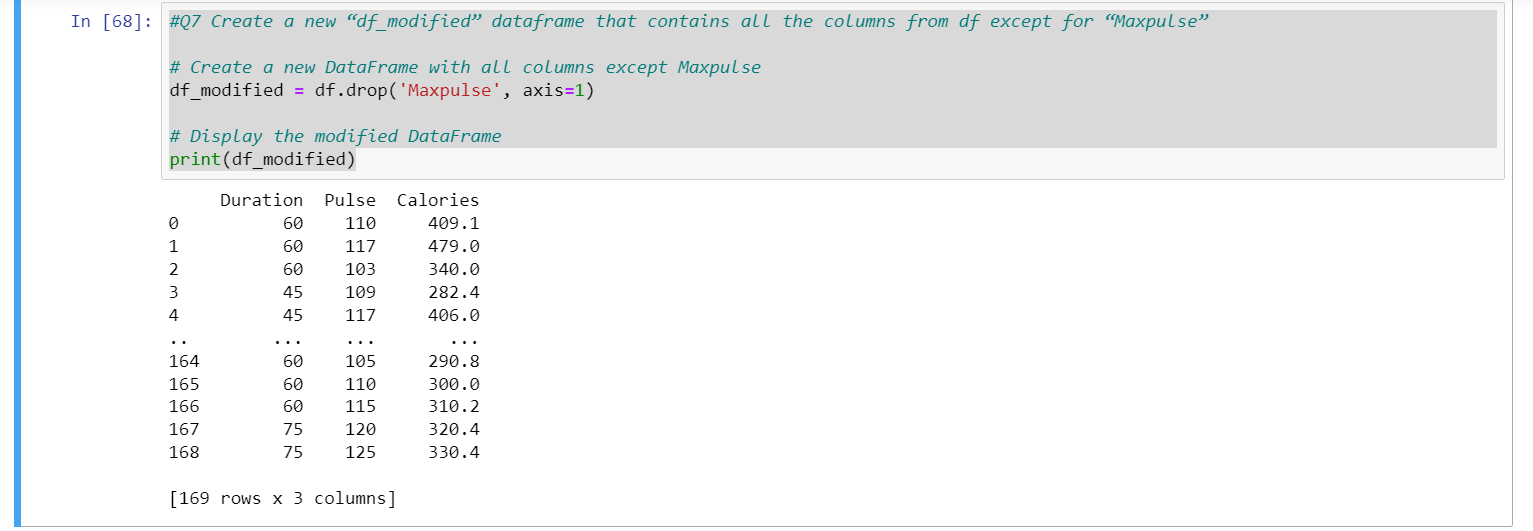
* The DataFrame object df is filtered using boolean indexing to select rows where the Calories column has a value between 500 and 1000.
* The filtered data is assigned to a new DataFrame object named filtered\_df.
* The contents of filtered\_df are printed to the console using the print() function.

6. Filter the dataframe to select the rows with calories values > 500 and pulse < 100.



* Filters the DataFrame object df using boolean indexing to select rows where the Calories column has a value greater than 500 and the Pulse column has a value less than 100.
* Stores the filtered data in a new DataFrame object named filtered\_df.
* Prints the contents of filtered\_df to the console.

7. Create a new “df\_modified” dataframe that contains all the columns from df except for “Maxpulse”.



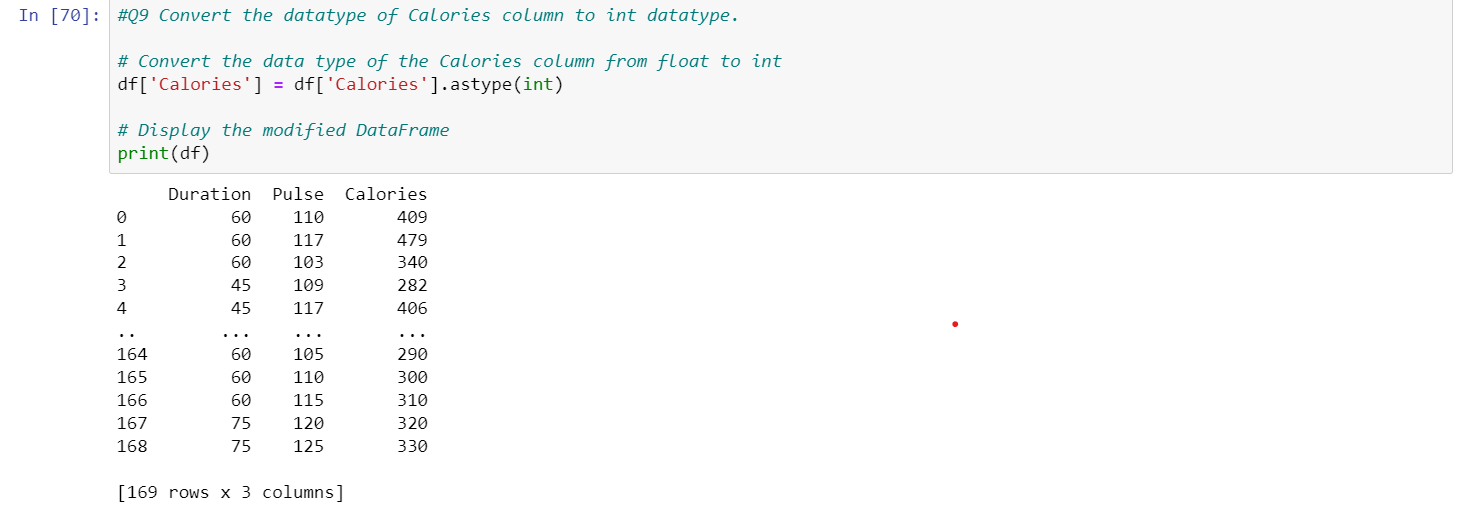
* It creates a new DataFrame object named df\_modified that contains all the columns from the DataFrame object df except for the Maxpulse column.
* It uses the drop() function with axis=1 to drop the Maxpulse column from the DataFrame object df.
* It assigns the modified DataFrame object to the new variable df\_modified.
* It prints the contents of the modified DataFrame object df\_modified to the console using the print() function.

8. Delete the “Maxpulse” column from the main df dataframe



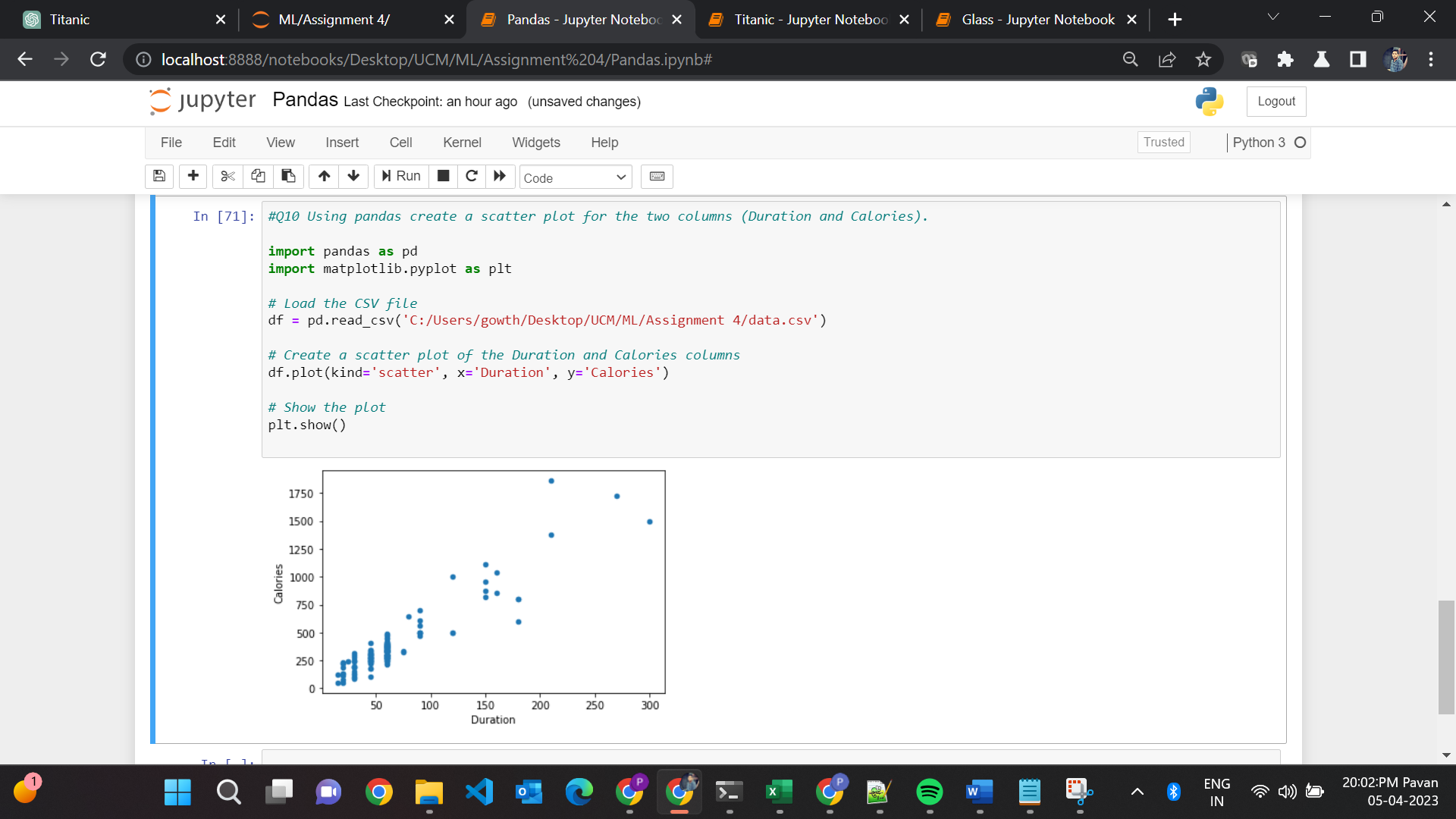
* It uses the drop() function with axis=1 and inplace=True to remove the Maxpulse column from the DataFrame object df.
* It prints the contents of the modified DataFrame object df to the console using the print() function.

9. Convert the datatype of Calories column to int datatype.



* It uses the astype() function to convert the data type of the Calories column in the DataFrame object df from float to integer.
* It assigns the modified DataFrame object back to the df variable to store the modified data.
* It prints the contents of the modified DataFrame object df to the console using the print() function.

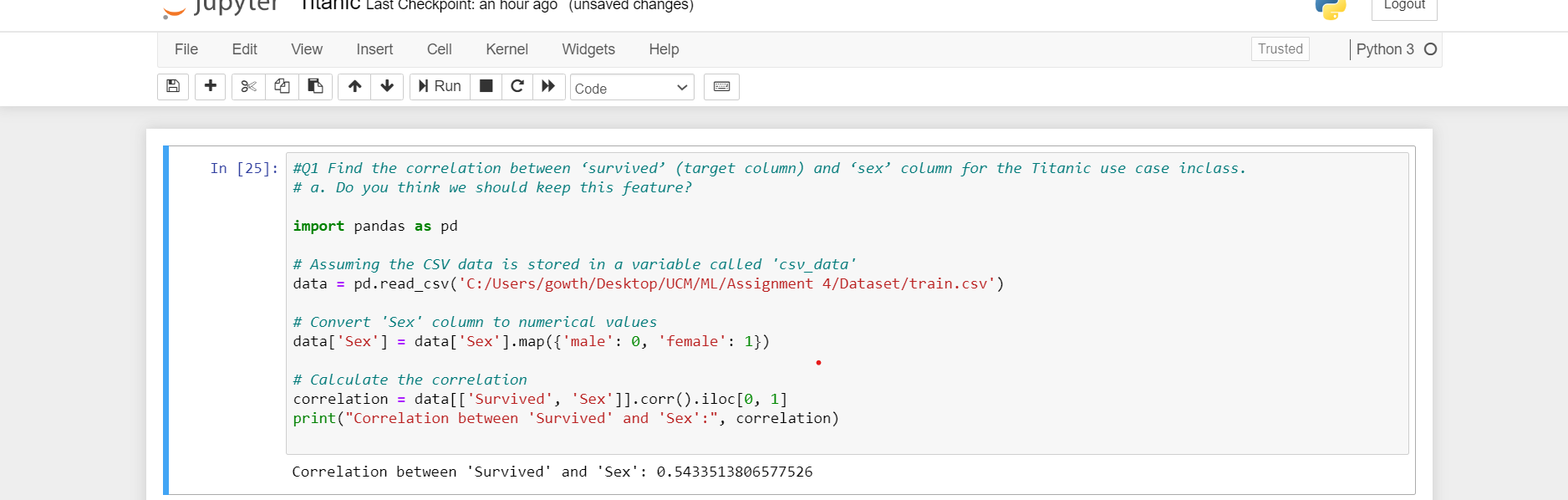
10. Using pandas create a scatter plot for the two columns (Duration and Calories).



* It imports the pandas library as pd and the matplotlib.pyplot library as plt.
* It uses the read\_csv() function to load the CSV file named data.csv and store the contents in a pandas DataFrame object named df.
* It creates a scatter plot of the Duration and Calories columns in the DataFrame object df using the plot() function with kind='scatter', x='Duration', and y='Calories'.
* It displays the plot using the show() function from the plt library.

1. **(Titanic Dataset)**

1. Find the correlation between ‘survived’ (target column) and ‘sex’ column for the Titanic use case inclass.

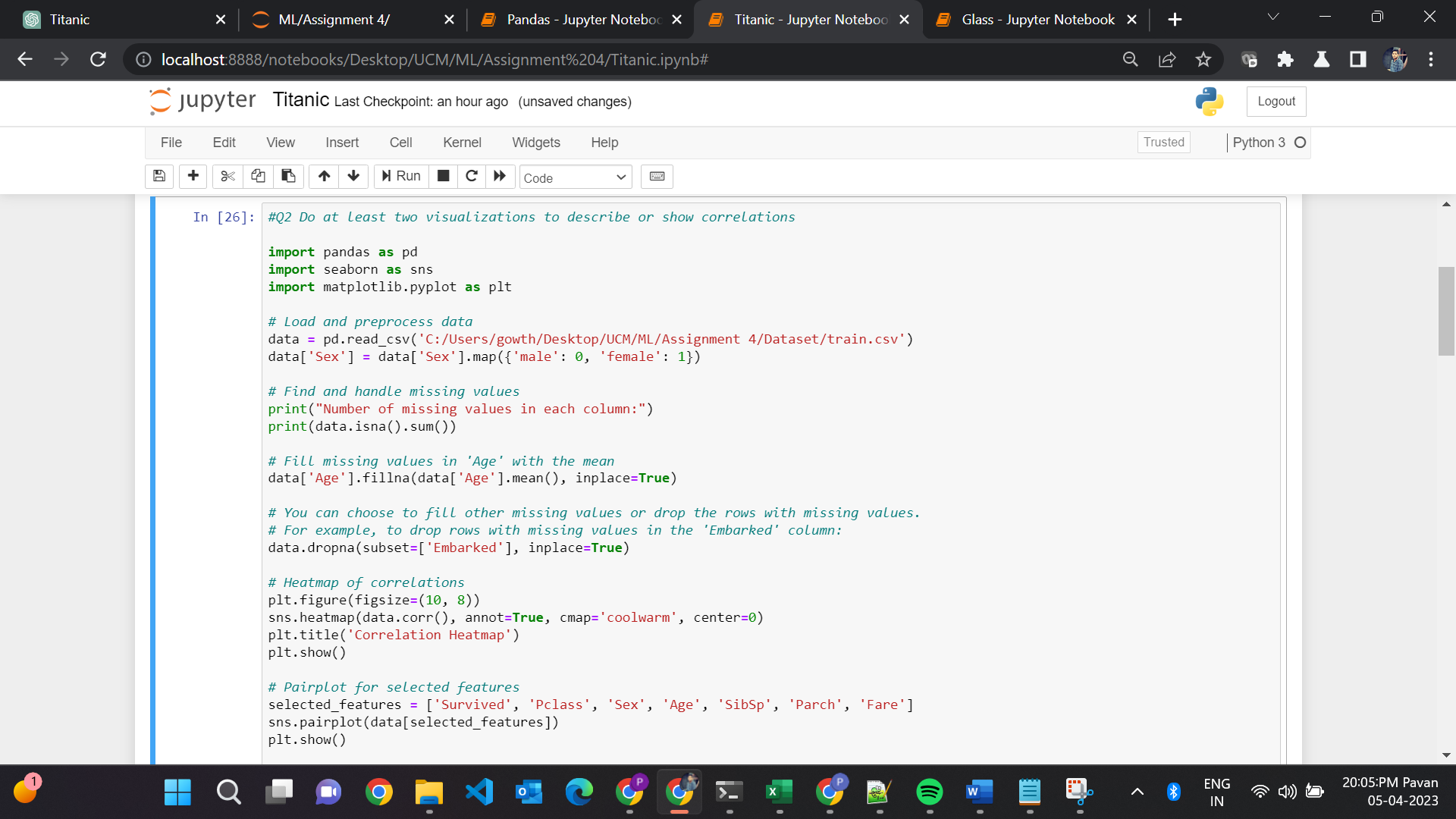
****

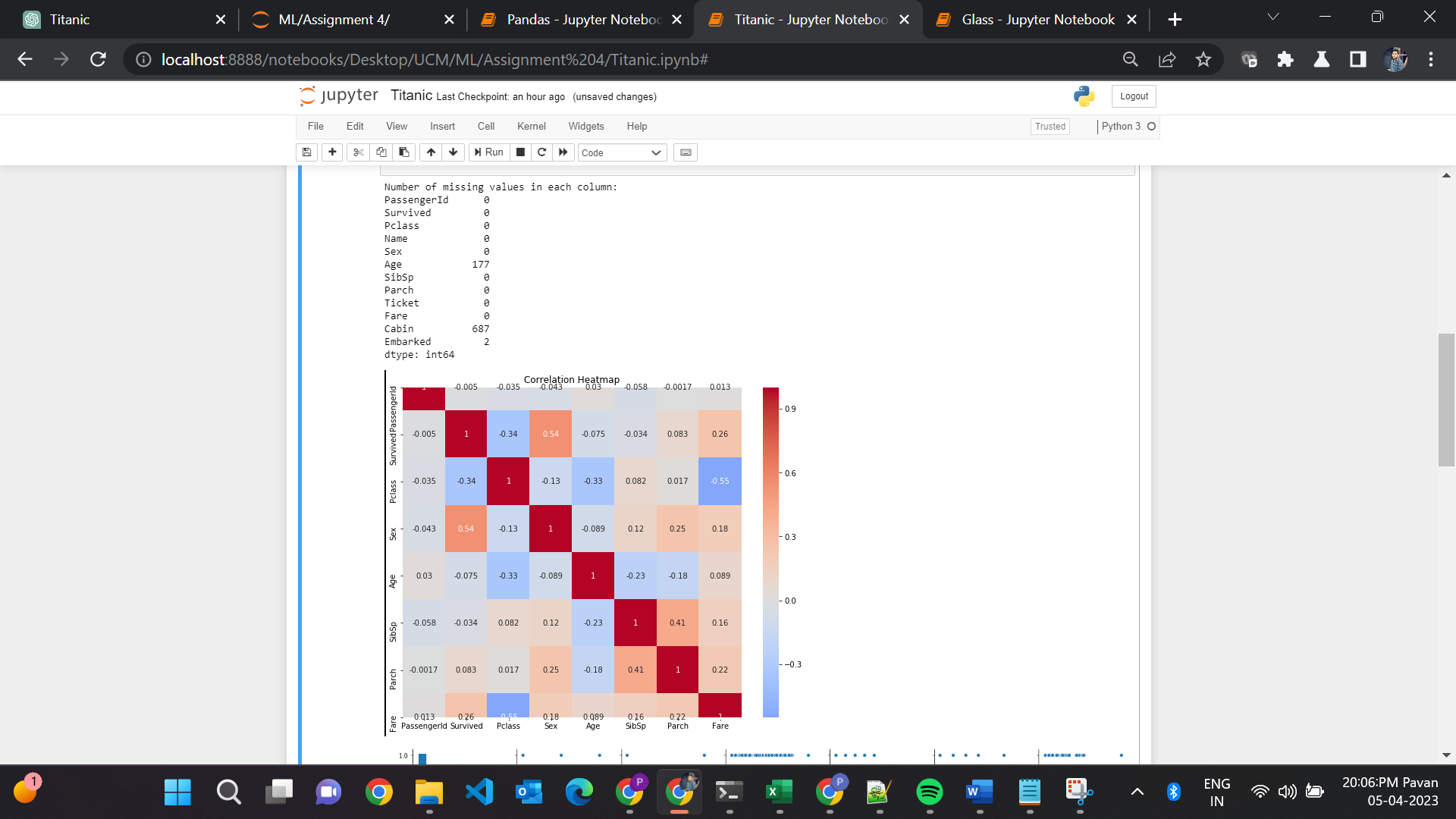
* It imports the pandas library as pd.
* It uses the read\_csv() function to read the Titanic dataset CSV file named train.csv and store the contents in a pandas DataFrame object named data.
* It maps the values in the Sex column to numerical values (0 for male and 1 for female) using the map() function.
* It calculates the correlation between the Survived and Sex columns using the corr() function and selects the correlation value using the iloc[0, 1] syntax.
* It prints the calculated correlation value to the console using the print() function.

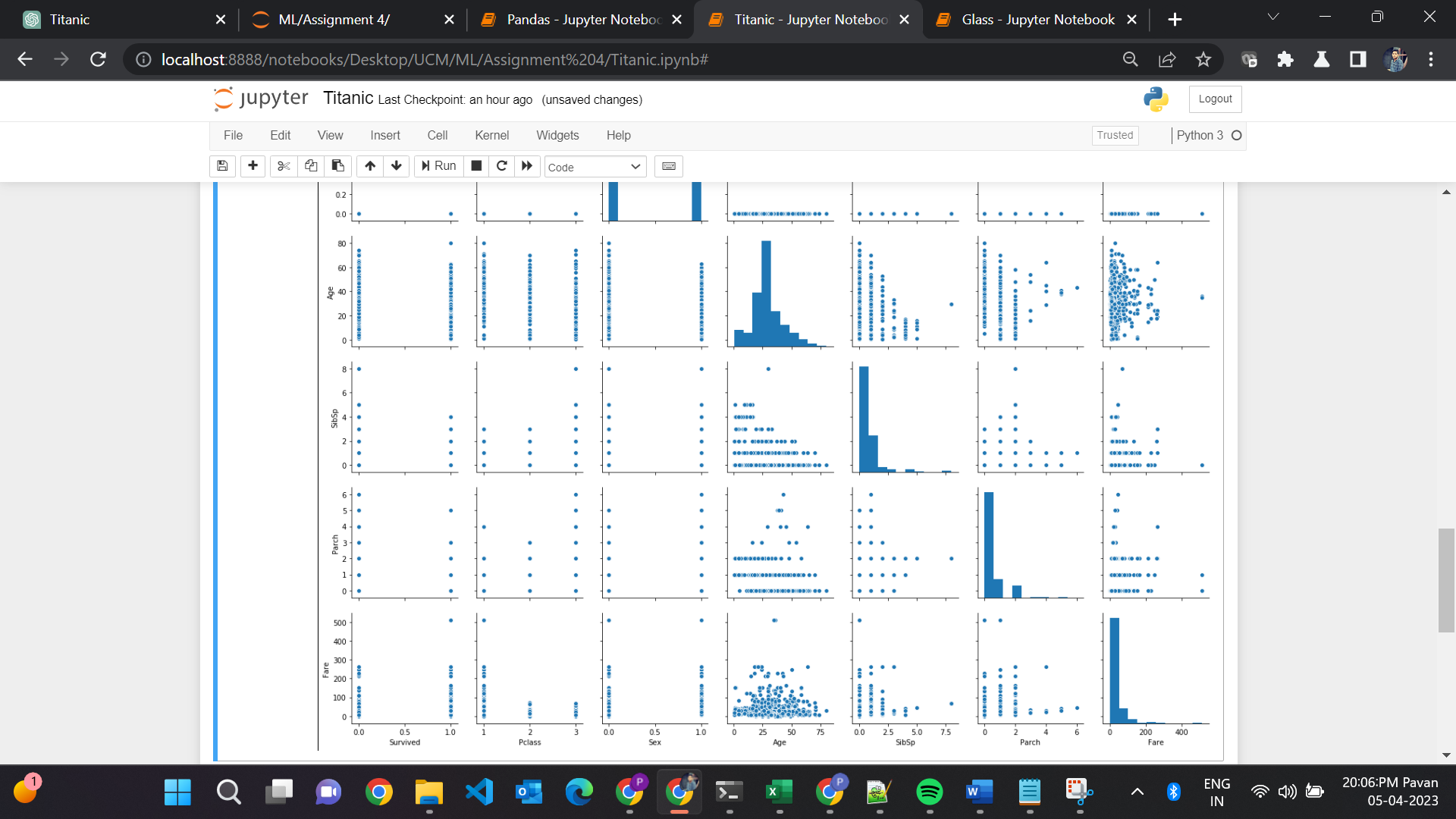
a. Do you think we should keep this feature?

* The correlation value between 'Survived' and 'Sex' is 0.5434, which indicates a moderate positive correlation between the two variables. This means that as the value of 'Sex' (0 for male, 1 for female) increases, the probability of survival also increases.
* In this case, we can conclude that the 'Sex' feature is relevant for predicting survival, as it has a moderate correlation with the target variable. Therefore, it may be useful to keep this feature in our analysis.

2.Do at least two visualizations to describe or show correlations.

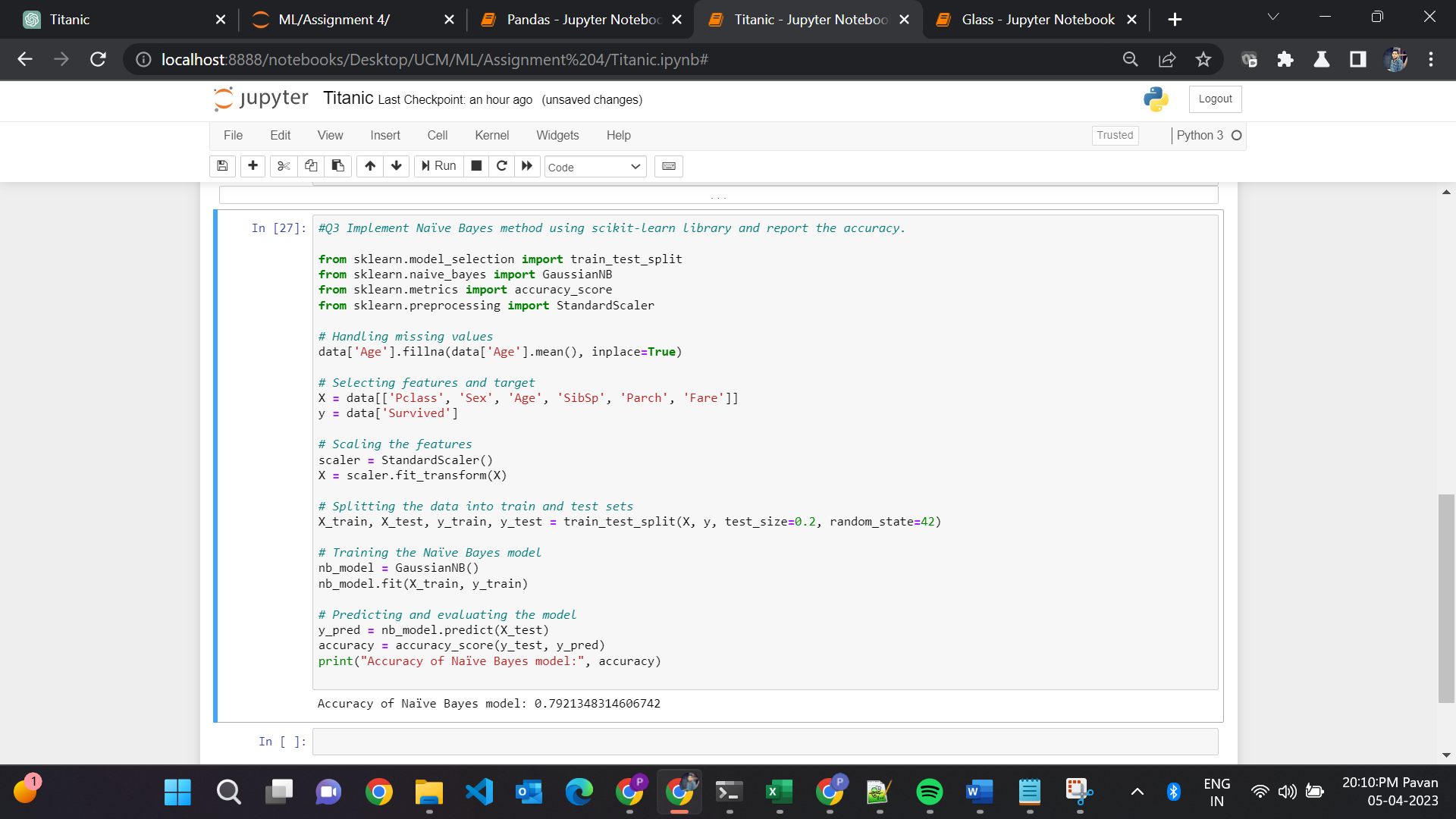






* Load and preprocess the Titanic dataset using pandas.
* Handle missing values in the dataset.
* Create a heatmap of correlations using seaborn's sns.heatmap() function.
* Create a pairplot for selected features using seaborn's sns.pairplot() function.
* Display the visualizations using matplotlib's plt.show() function.It creates a pairplot for selected features using the sns.pairplot() function and the list of selected features.

3. Implement Naïve Bayes method using scikit-learn library and report the accuracy.



* Handle missing values in the dataset.
* Select features and target variables and scale the features using StandardScaler().
* Split the dataset into train and test sets.
* Train the Naïve Bayes model using GaussianNB()and make predictions on the test set and evaluate the model using accuracy\_score().
* Print the accuracy of the Naïve Bayes model using print().

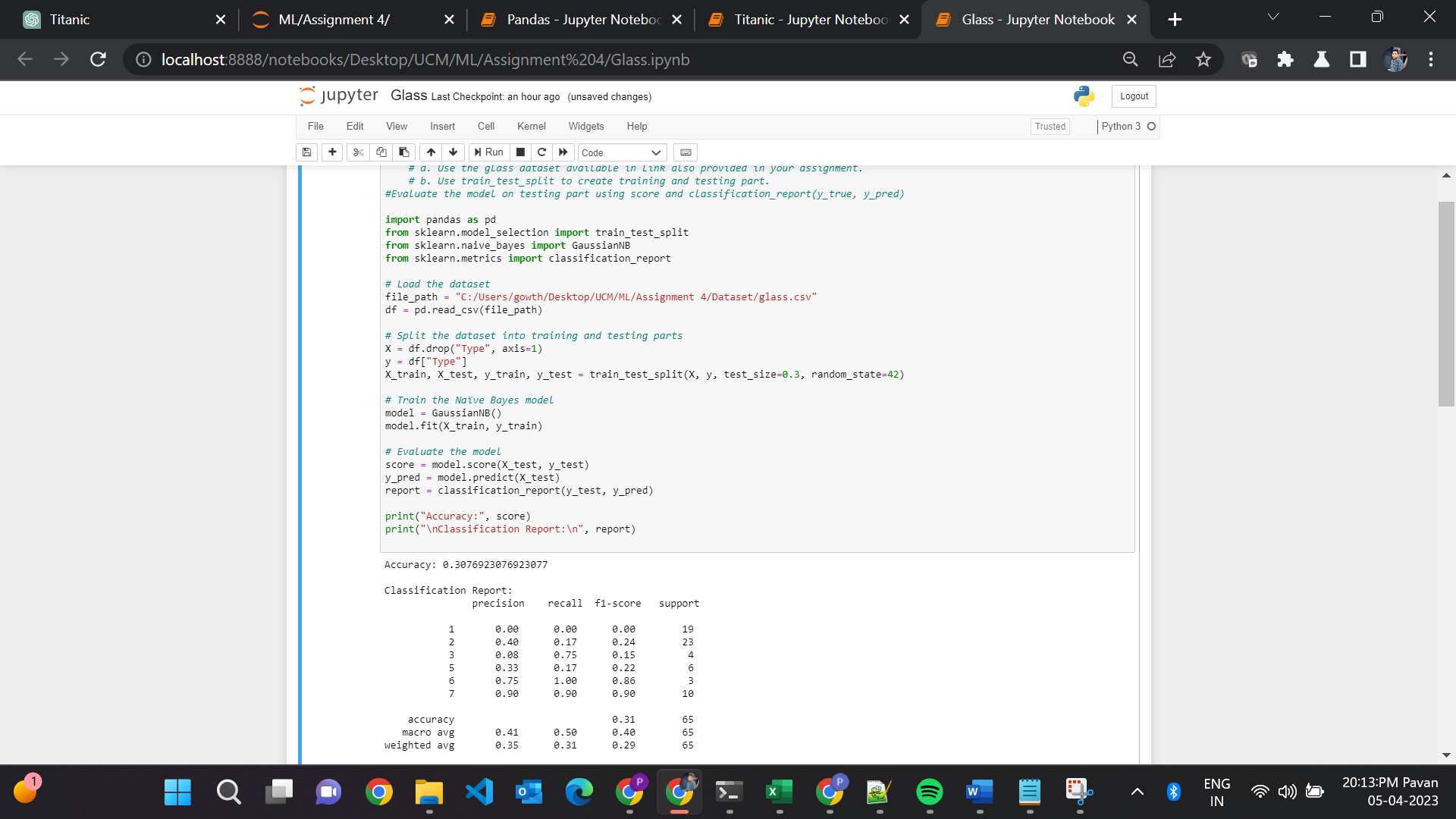
**2. (Glass Dataset)**

1. Implement Naïve Bayes method using scikit-learn library.

a. Use the glass dataset available in Link also provided in your assignment.

b. Use train\_test\_split to create training and testing part.

2. Evaluate the model on testing part using score and classification\_report(y\_true, y\_pred)



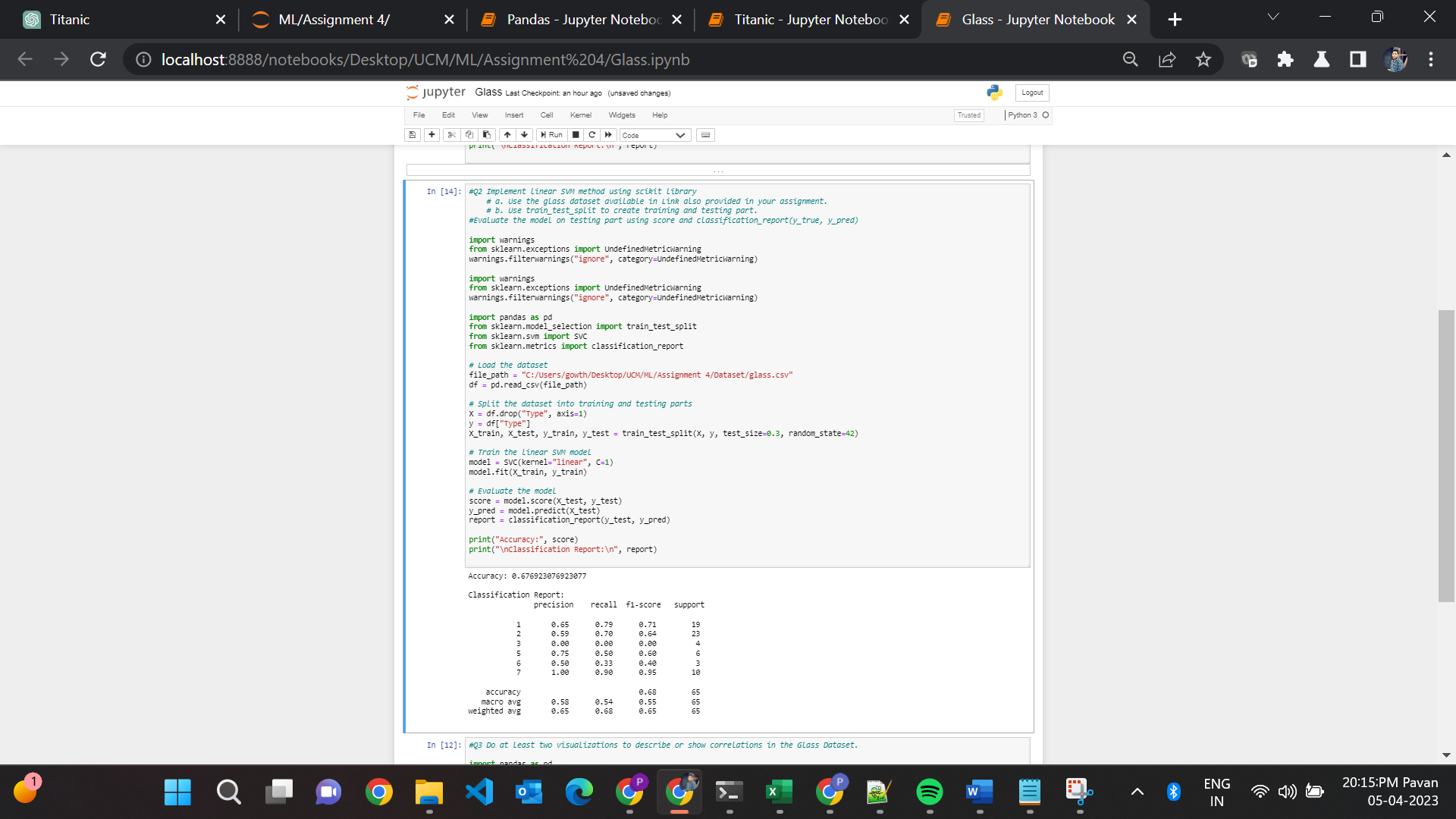
* It imports the necessary libraries including pandas, train\_test\_split, GaussianNB, and classification\_report.
* It loads the glass.csv dataset into a pandas DataFrame object df.
* It splits the dataset into training and testing parts using train\_test\_split().
* It trains the Naïve Bayes model using GaussianNB().
* It evaluates the model using the score() method to calculate the accuracy and the classification\_report() function to generate a classification report.
* It prints the accuracy and classification report to the console using the print() function.

1. Implement linear SVM method using scikit library

a. Use the glass dataset available in Link also provided in your assignment.

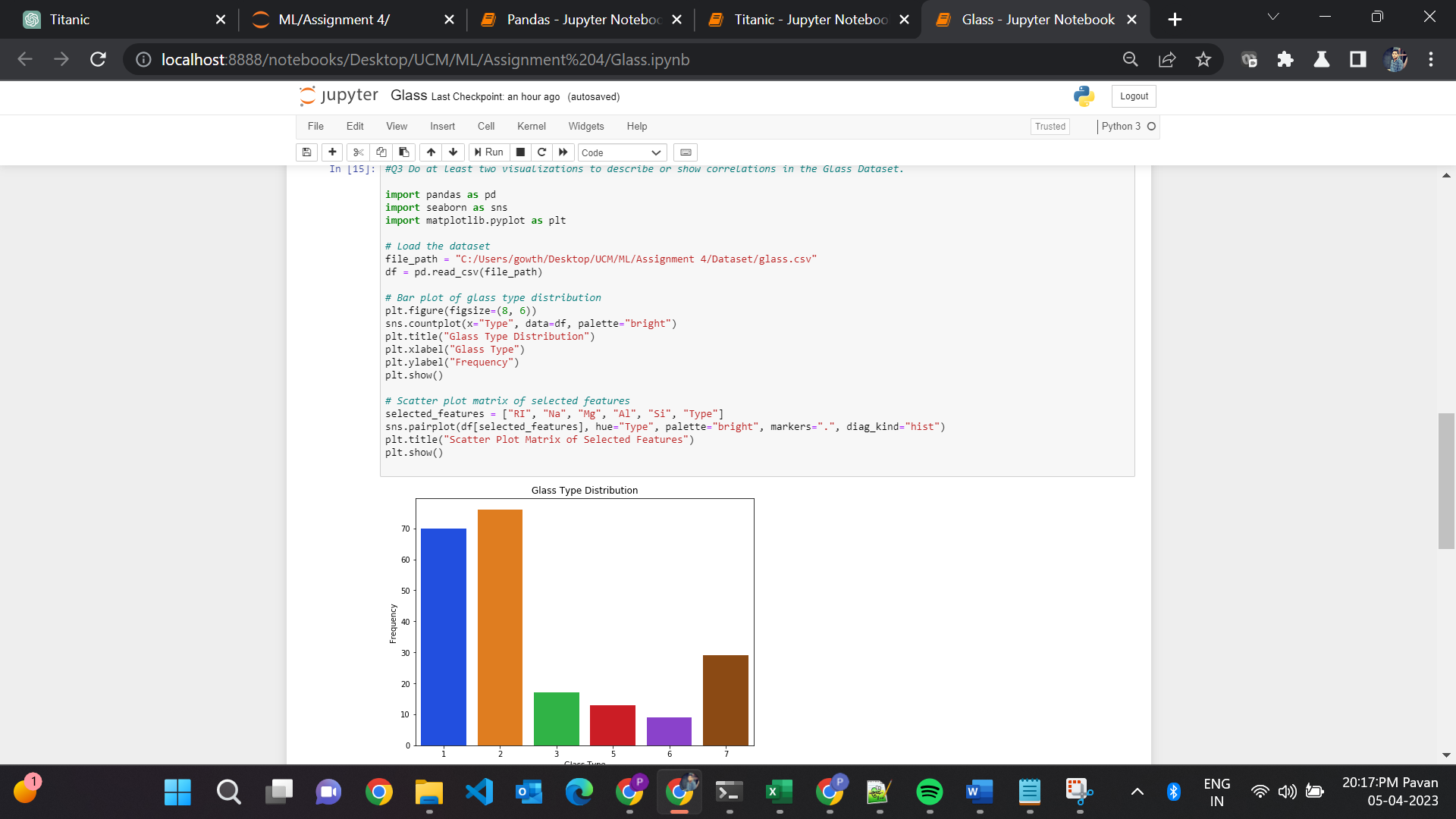
b. Use train\_test\_split to create training and testing part.

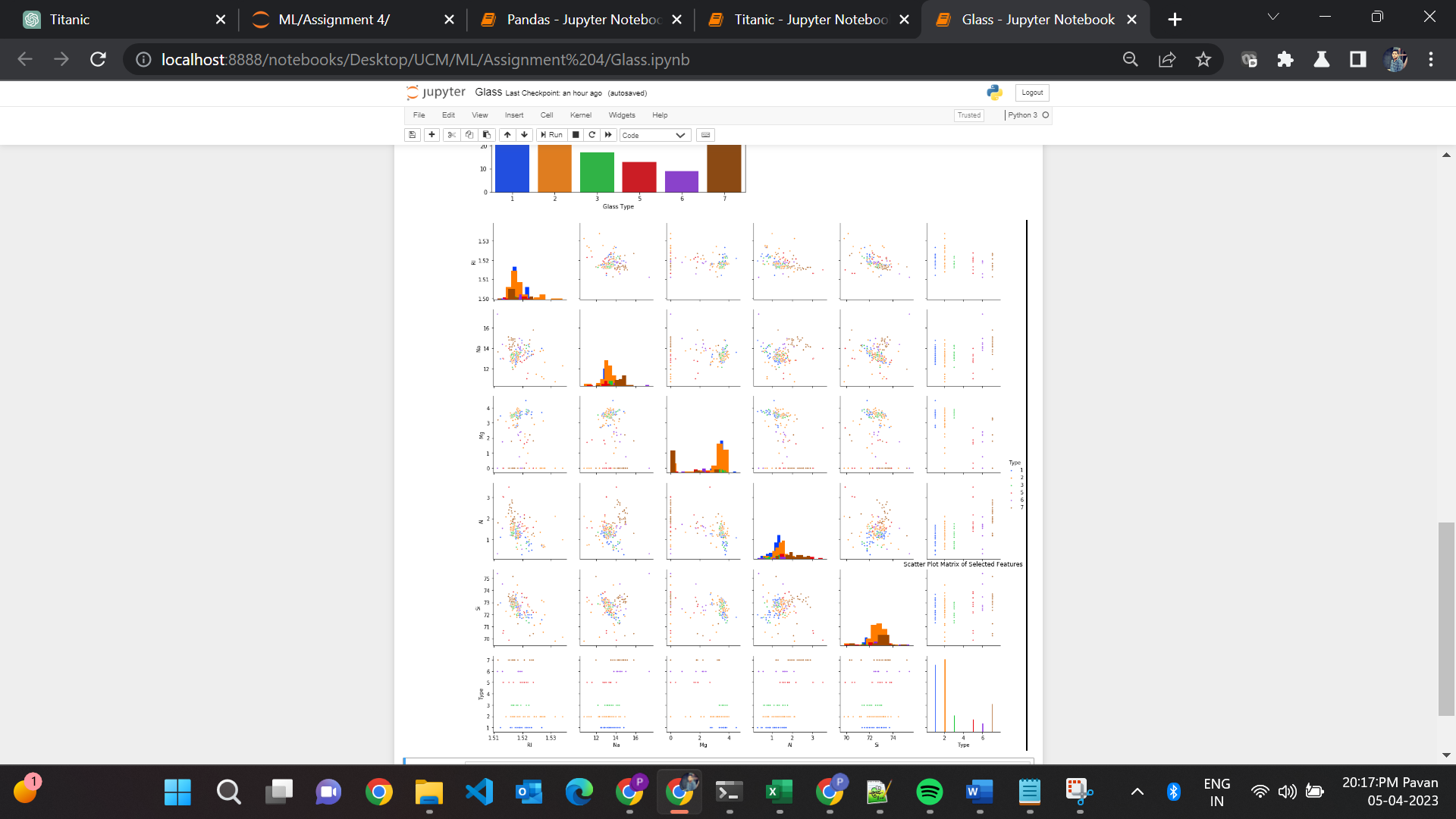
2. Evaluate the model on testing part using score and classification\_report(y\_true, y\_pred)



* It imports the necessary libraries including warnings, train\_test\_split, SVC, and classification\_report.
* It loads the glass.csv dataset into a pandas DataFrame object df.
* It splits the dataset into training and testing parts using train\_test\_split().
* It trains the linear SVM model using SVC() with a linear kernel and regularization parameter C=1.
* It evaluates the model using the score() method to calculate the accuracy and the classification\_report() function to generate a classification report.
* It prints the accuracy and classification report to the console using the print() function.

**Do at least two visualizations to describe or show correlations in the Glass Dataset.**





* It imports the necessary libraries including pandas, seaborn, and matplotlib.pyplot.
* It loads the glass.csv dataset into a pandas DataFrame object df.
* It creates a bar plot of the glass type distribution using countplot() from seaborn.
* It creates a scatter plot matrix of selected features using pairplot() from seaborn.
* It shows the visualizations using show() from matplotlib.pyplot.

**Which algorithm you got better accuracy? Can you justify why?**

* The linear SVM model performs better than the Naïve Bayes model on the Glass dataset, with an accuracy of 0.6769 compared to 0.3077. This can be attributed to:
* Naïve Bayes' assumption of feature independence, which may not hold for the Glass dataset.
* SVM's ability to handle complex feature relationships and find an optimal decision boundary.
* The Glass dataset's class imbalance or overlapping classes, which SVM handles better.
* The choice of hyperparameters, train-test split, and other factors also affect model performance, but overall, linear SVM is a better choice for this dataset.