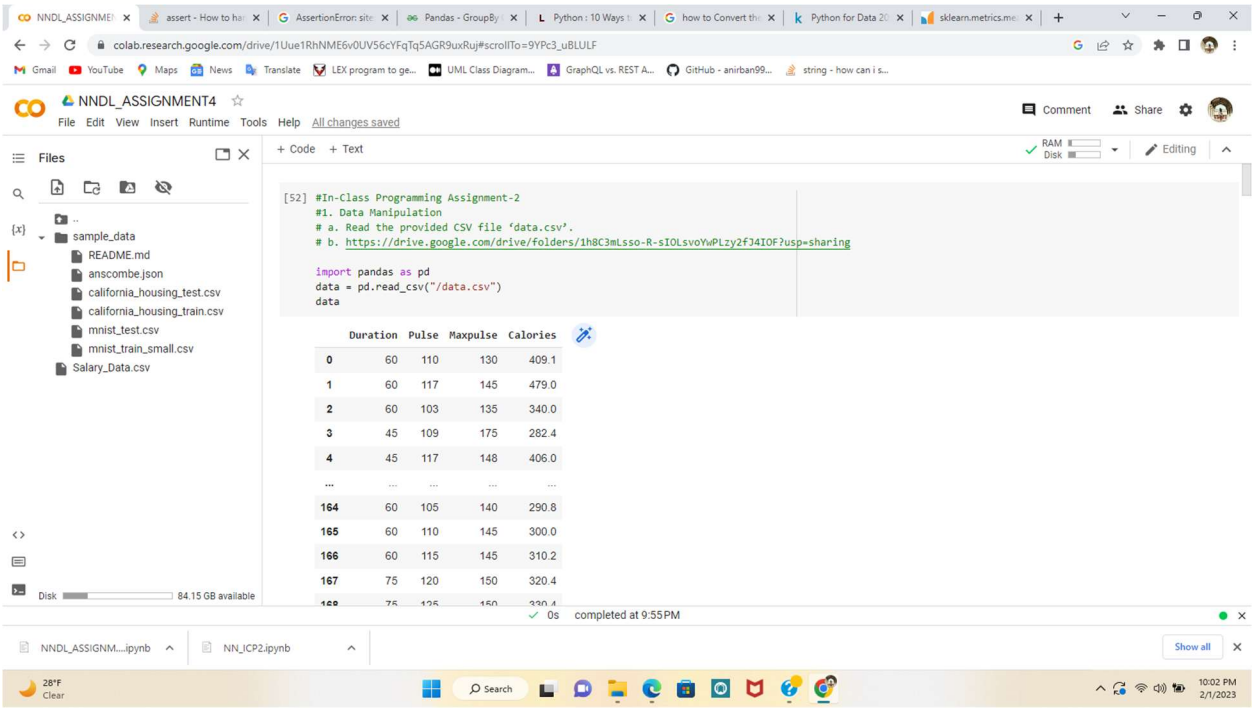
Spring 2023: CS5720 – NN &DL In-Class Programming Assignment-2

1. Data Manipulation

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

b. https://drive.google.com/drive/folders/1h8C3mLsso-R-sIOLsvoYwPLzy2fJ4IOF?usp=sharing

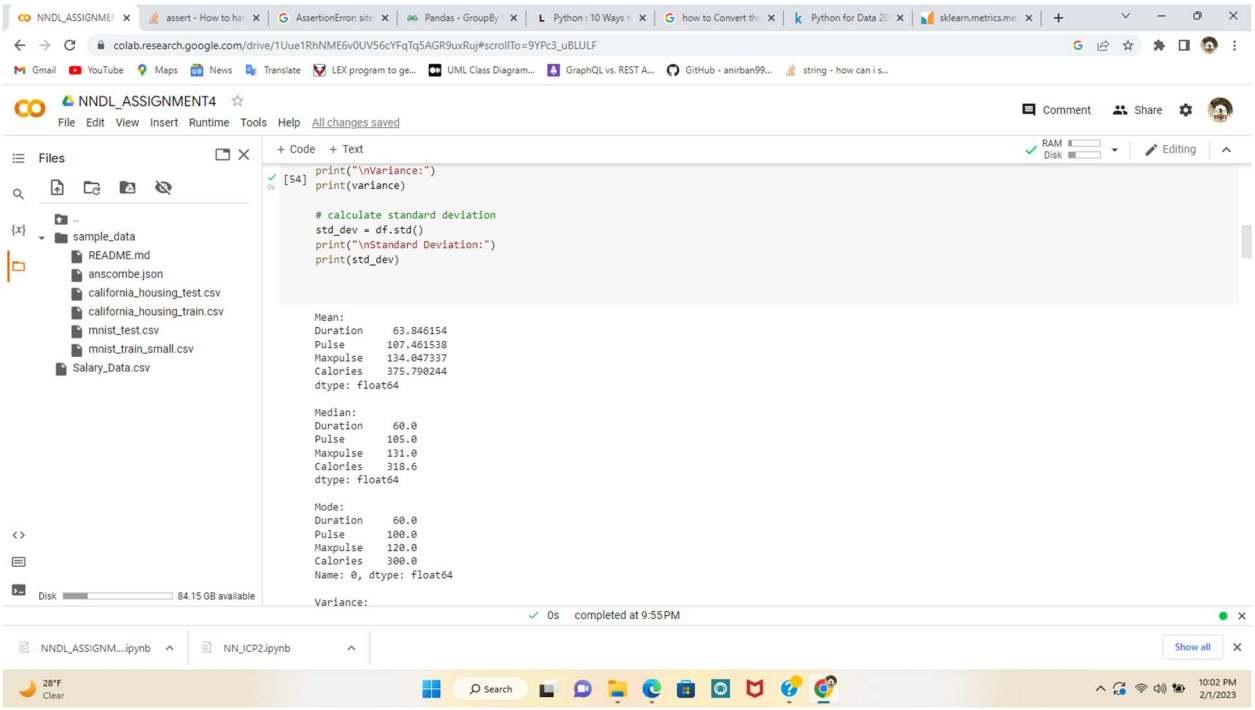
```
import pandas as pd
data = pd.read_csv("/data.csv")
data
```



Using pandas module, I have read the data.csv file and imported it and also uploaded in google collab notebook.

```
# c. Show the basic statistical description about the data.
import pandas as pd
import numpy as np

# create a sample dataframe
#data = ("/data.csv")
df = pd.DataFrame(data)

mean = df.mean()
```

```
print("Mean:")
print(mean)

# calculate median
median = df.median()
print("\nMedian:")
print(median)

# calculate mode
mode = df.mode().iloc[0]
print("\nMode:")
print(mode)

# calculate variance
variance = df.var()
print("\nVariance:")
print(variance)

# calculate standard deviation
std_dev = df.std()
print("\nStandard Deviation:")
print(std_dev)
```

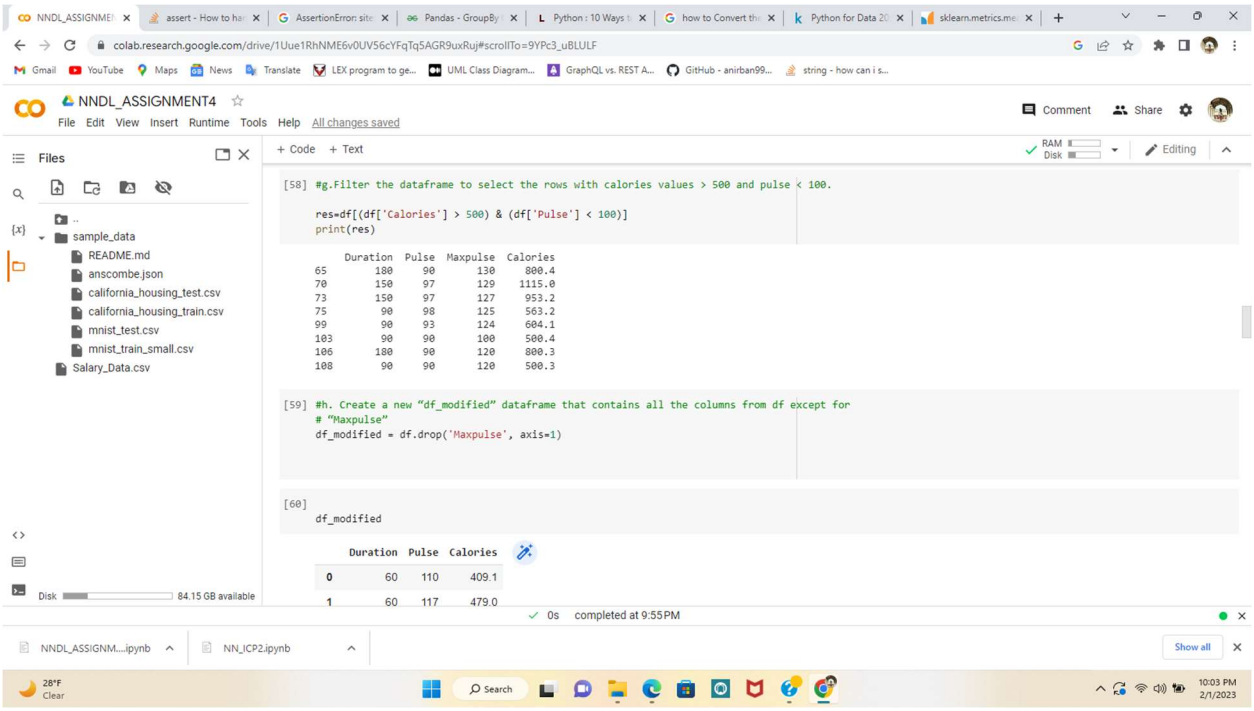


Using pandas and numpy I have shown the `basic statistical description`

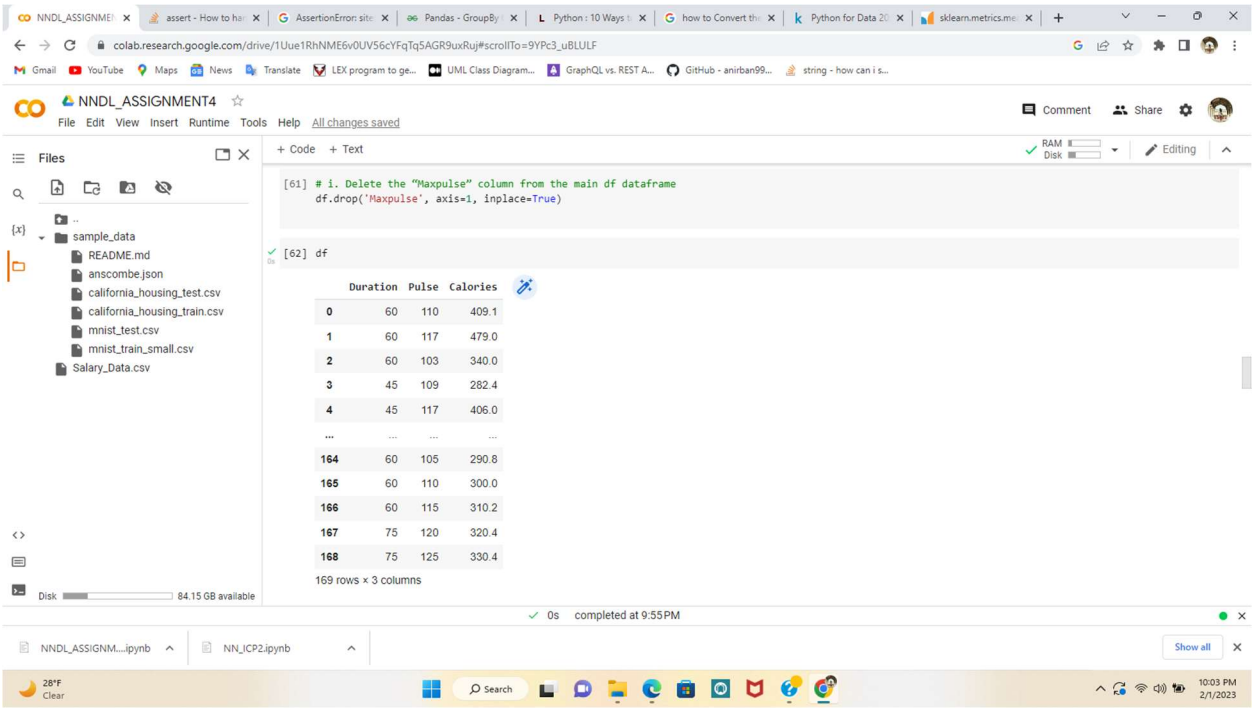
```
#d.Check if the data has null values.
# i. Replace the null values with the mean
df.isnull().sum()
df = df.fillna(df.mean())



#e. Select at least two columns and aggregate the data using: min, max, co
unt, mean
agg_df = df[['Duration', 'Calories']].agg({'Duration': ['min', 'max', 'cou
nt', 'mean'], 'Calories': ['min', 'max', 'count', 'mean']})


#f.Filter the dataframe to select the rows with calories values between 50
0 and 1000
result = df[df['Calories'].between(500,1000 )]
print(result)
```

In d,e,f I Have shown the required data using manupulations.

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

res=df[(df['Calories'] > 500) & (df['Pulse'] < 100)]
print(res)

#h. Create a new “df_modified” dataframe that contains all the columns fro
m df except for
# “Maxpulse”
df_modified = df.drop('Maxpulse', axis=1)
```




```
# i. Delete the "Maxpulse" column from the main df dataframe
df.drop('Maxpulse', axis=1, inplace=True)

#j. Convert the datatype of Calories column to int datatype
df['Calories'] = df['Calories'].astype(int)

df.dtypes
```

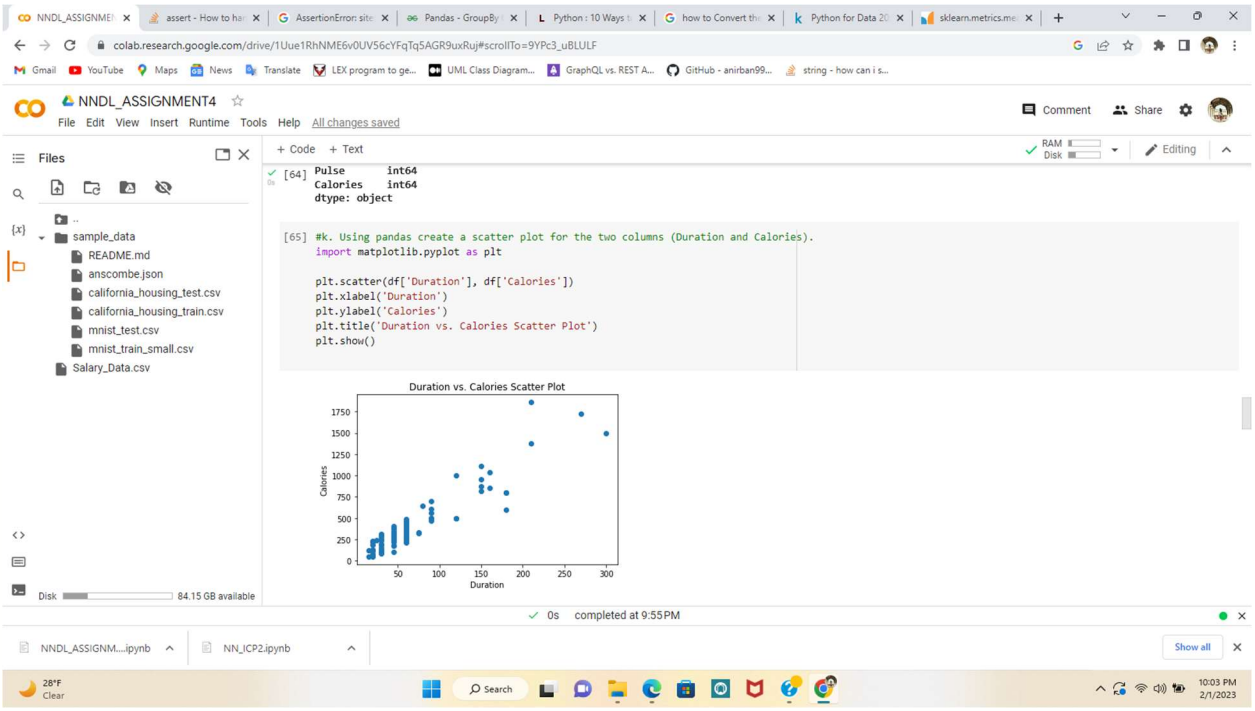


	Duration	Pulse	Calories
0	60	110	409.1
1	60	117	479.0
2	60	103	340.0
3	45	109	282.4
4	45	117	406.0
...	...	...	...
164	60	105	290.8
165	60	110	300.0
166	60	115	310.2
167	75	120	320.4
168	75	125	330.4

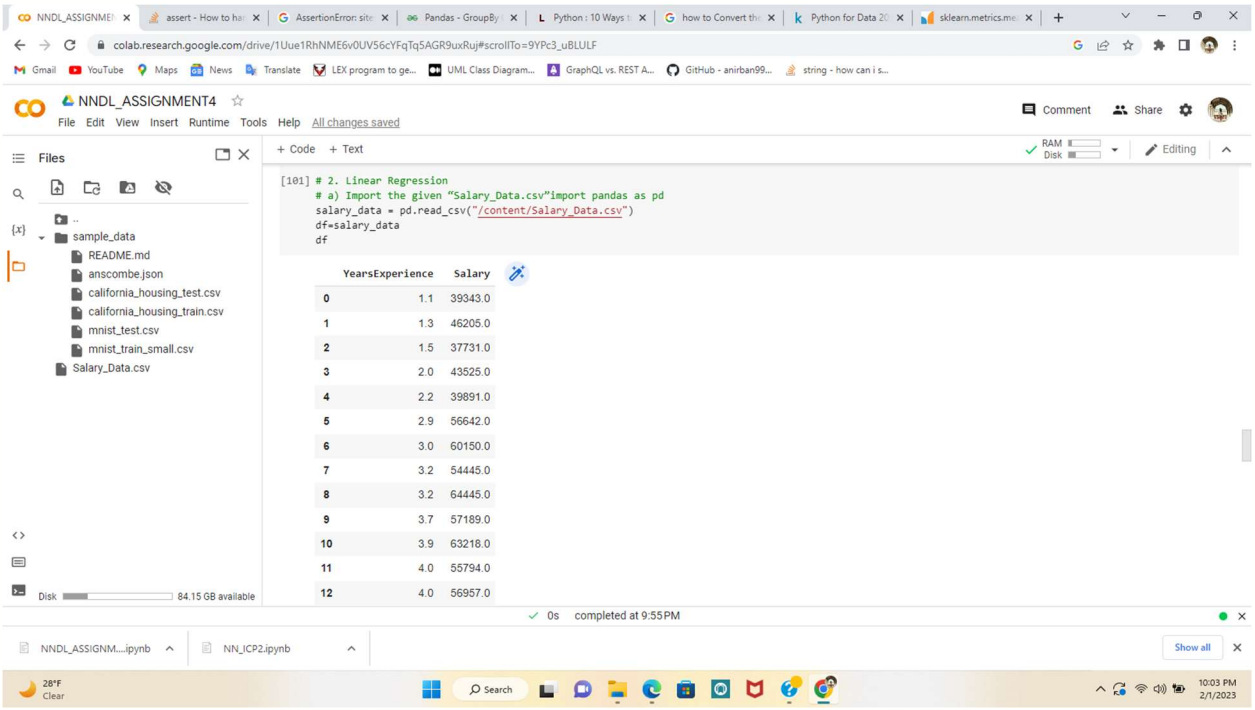
169 rows × 3 columns

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

```
import matplotlib.pyplot as plt

plt.scatter(df['Duration'], df['Calories'])
plt.xlabel('Duration')
plt.ylabel('Calories')
plt.title('Duration vs. Calories Scatter Plot')
plt.show()
```

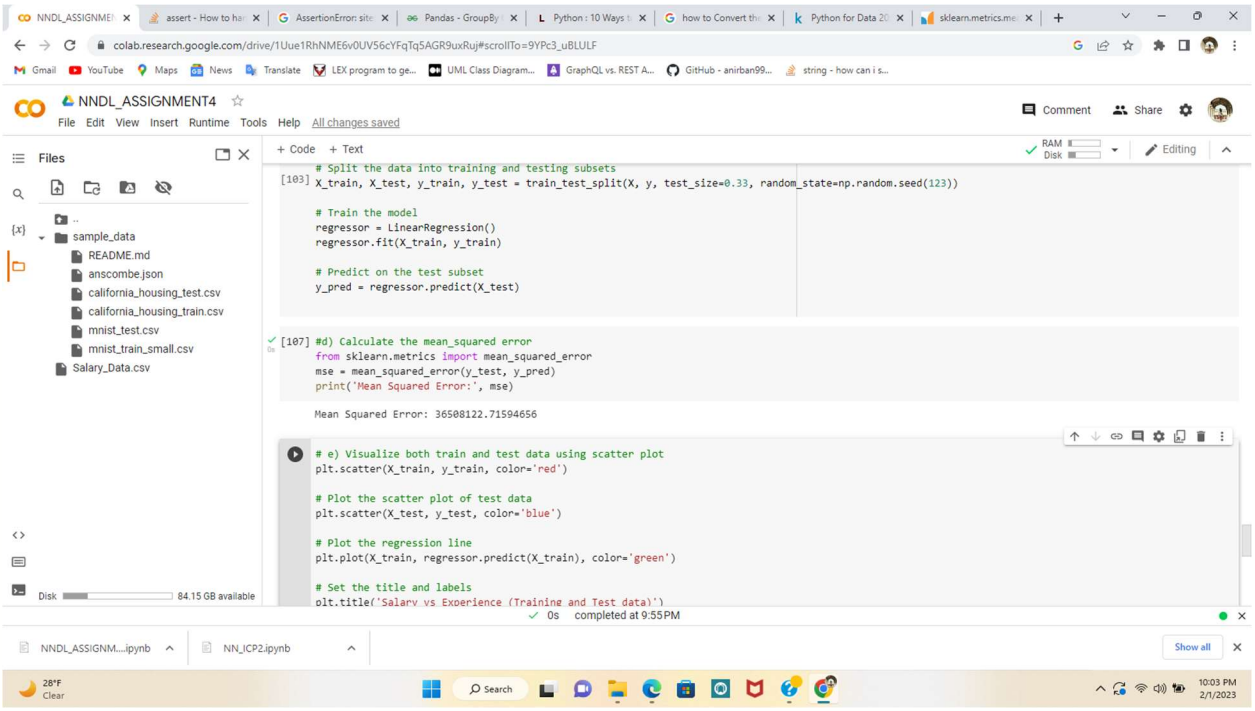



```
# 2. Linear Regression
# a) Import the given “Salary_Data.csv”import pandas as pd
salary_data = pd.read_csv("/content/Salary_Data.csv")
df=salary_data
df
```

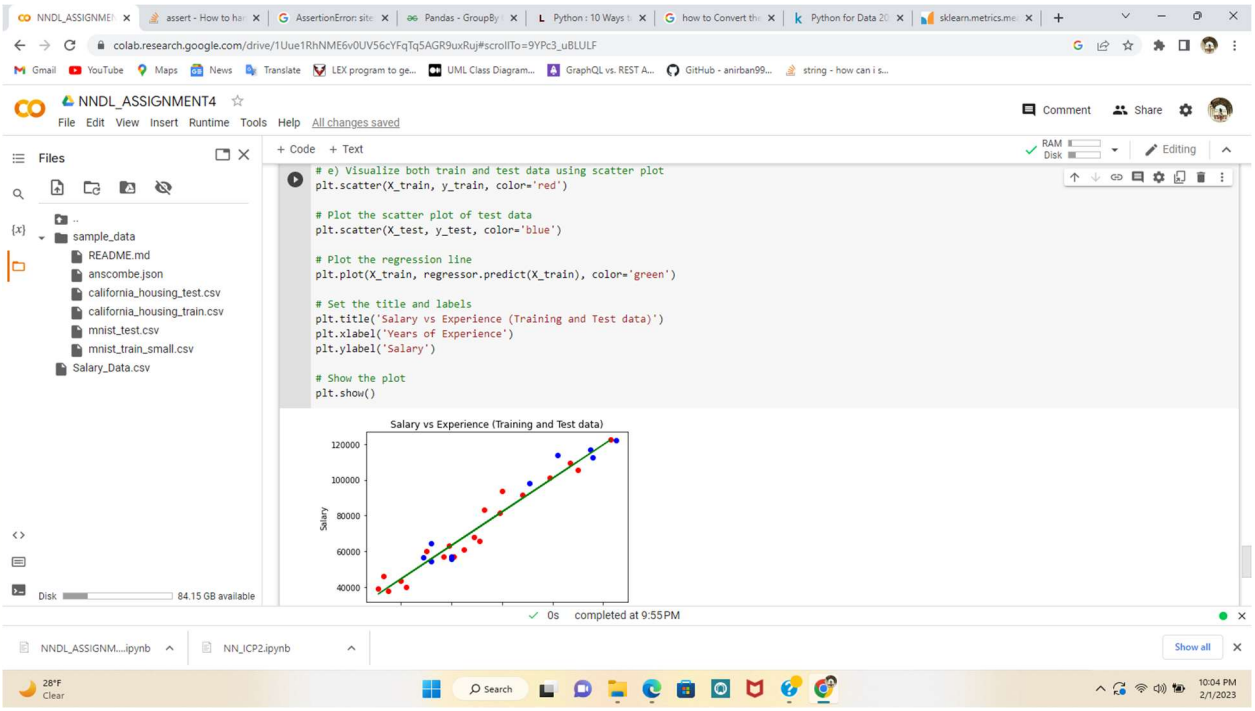


	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0

```
#b) Split the data in train_test partitions, such that 1/3 of the data is
reserved as test subset
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split



# Split the data into features (predictors) and target variables
X = df.iloc[:, :-1]
y = df.iloc[:, -1]

# Split the data into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=np.random.seed(123))


# c) Train and predict the model.

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

# Load the data


# Split the data into features (predictors) and target variables
X = df.iloc[:, :-1].values
y = df.iloc[:, -1].values

# Split the data into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=np.random.seed(123))

# Train the model
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predict on the test subset
y_pred = regressor.predict(X_test)


#d) Calculate the mean_squared error
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, y_pred)
print('Mean Squared Error:', mse)
```




```
# e) Visualize both train and test data using scatter plot
plt.scatter(X_train, y_train, color='red')

# Plot the scatter plot of test data
plt.scatter(X_test, y_test, color='blue')

# Plot the regression line
plt.plot(X_train, regressor.predict(X_train), color='green')

# Set the title and labels
plt.title('Salary vs Experience (Training and Test data)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')

# Show the plot
plt.show()
```



REPO LINK: https://github.com/Goli18/NNDL_ASS4.git

VIDEO LINK: https://www.veed.io/view/6689707c-8dd3-4833-b0a1-b274e45da358?source=compressor-sharing