LAB Logbook

Lab 1:

Code:

import numpy as np

sid\_last\_two = 75

if sid\_last\_two < 10:

num\_elements = sid\_last\_two + 100

else:

num\_elements = sid\_last\_two

a = np.arange(num\_elements)

print("Original array:", a)

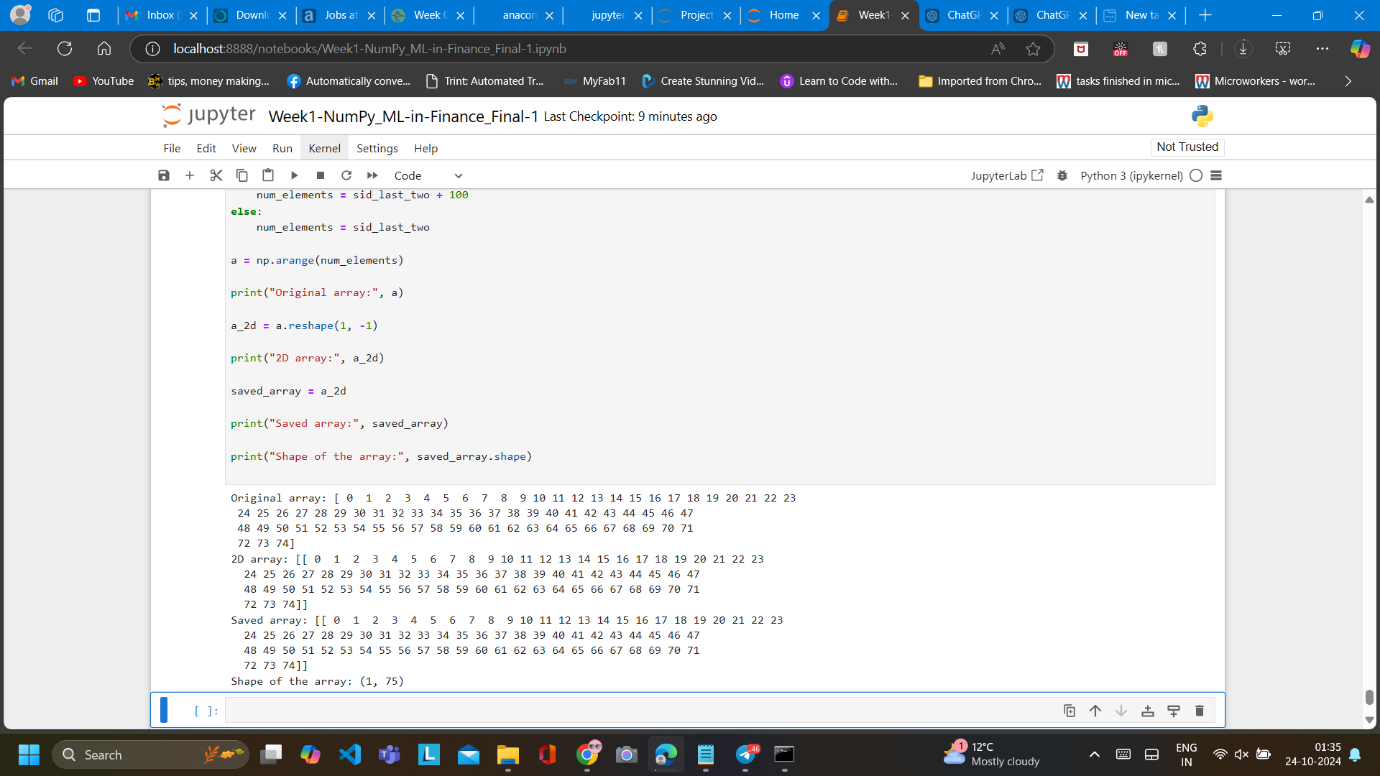
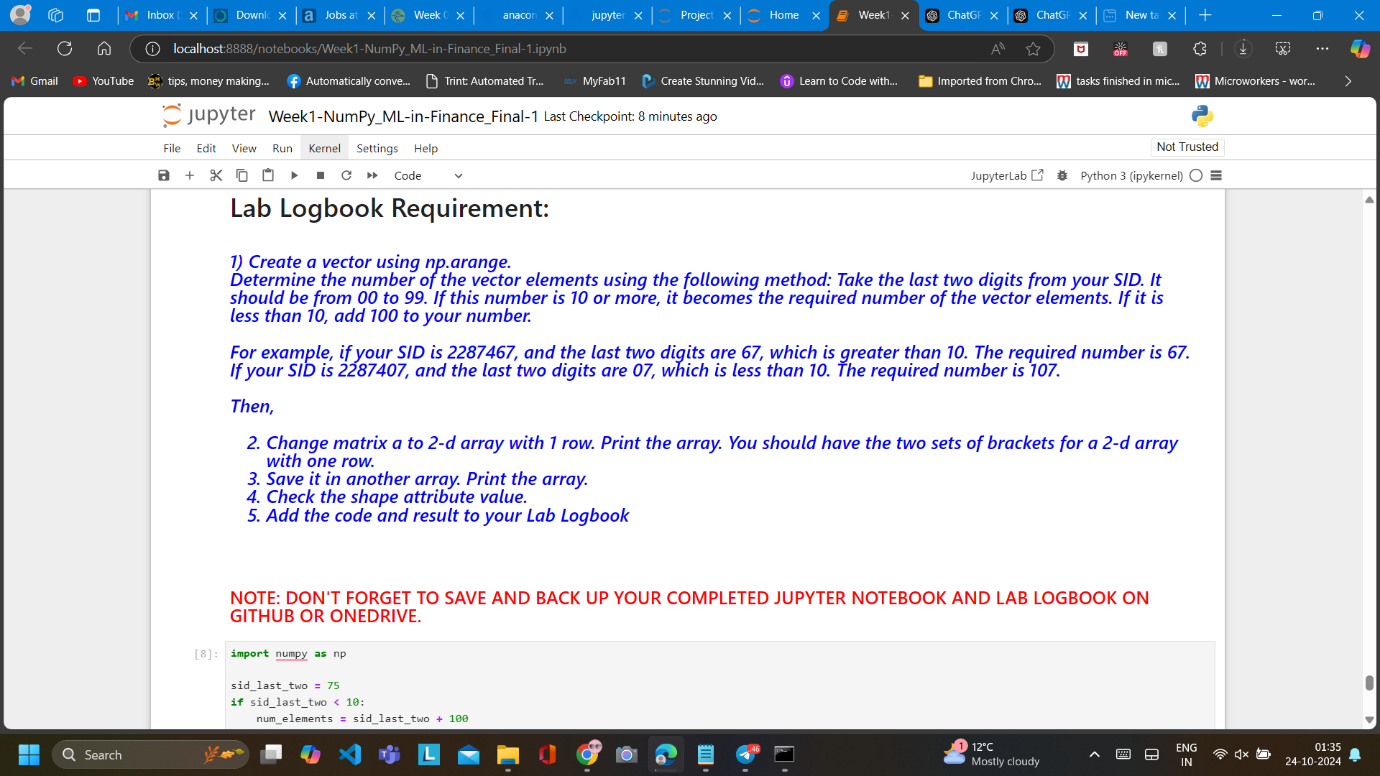
a\_2d = a.reshape(1, -1)

print("2D array:", a\_2d)

saved\_array = a\_2d

print("Saved array:", saved\_array)

print("Shape of the array:", saved\_array.shape)



Lab 2

Code :

import pandas as pd

df = pd.read\_csv('adult\_data\_mini.csv')

print("Original DataFrame:")

#print(df)

#n as the last digit of my SID 2331175

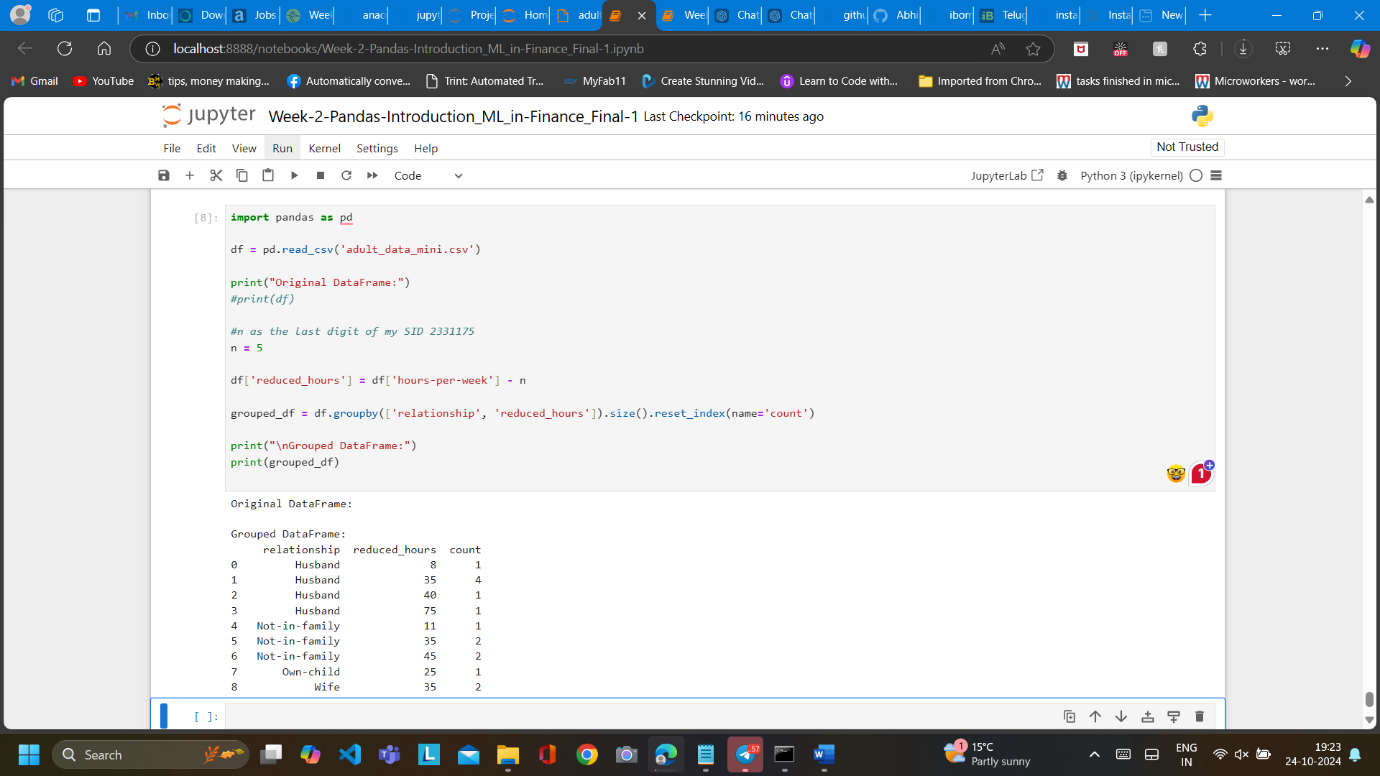
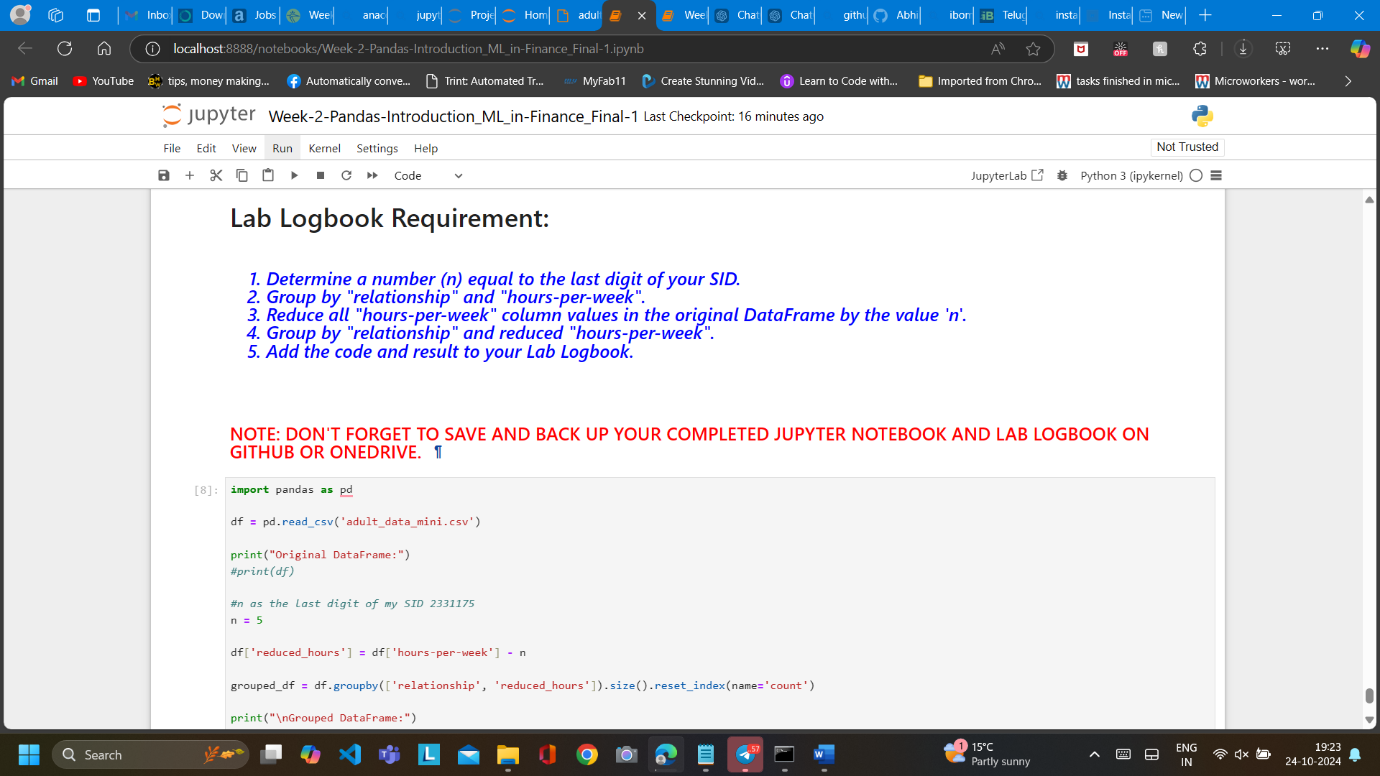
n = 5

df['reduced\_hours'] = df['hours-per-week'] - n

grouped\_df = df.groupby(['relationship', 'reduced\_hours']).size().reset\_index(name='count')

print("\nGrouped DataFrame:")

print(grouped\_df)



Lab 3

Code:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

data = pd.read\_csv('telecom\_churn.csv')

df = pd.DataFrame(data)

# SID:2331175 and selecting 5 (Number vmail messages) and 7 (Total day calls)

columns\_to\_plot = df[['Number vmail messages', 'Total day calls']]

plt.figure(figsize=(8, 6))

sns.scatterplot(x='Number vmail messages', y='Total day calls', data=columns\_to\_plot, palette='coolwarm', hue='Total day calls', size='Number vmail messages', sizes=(20, 200), edgecolor="black")

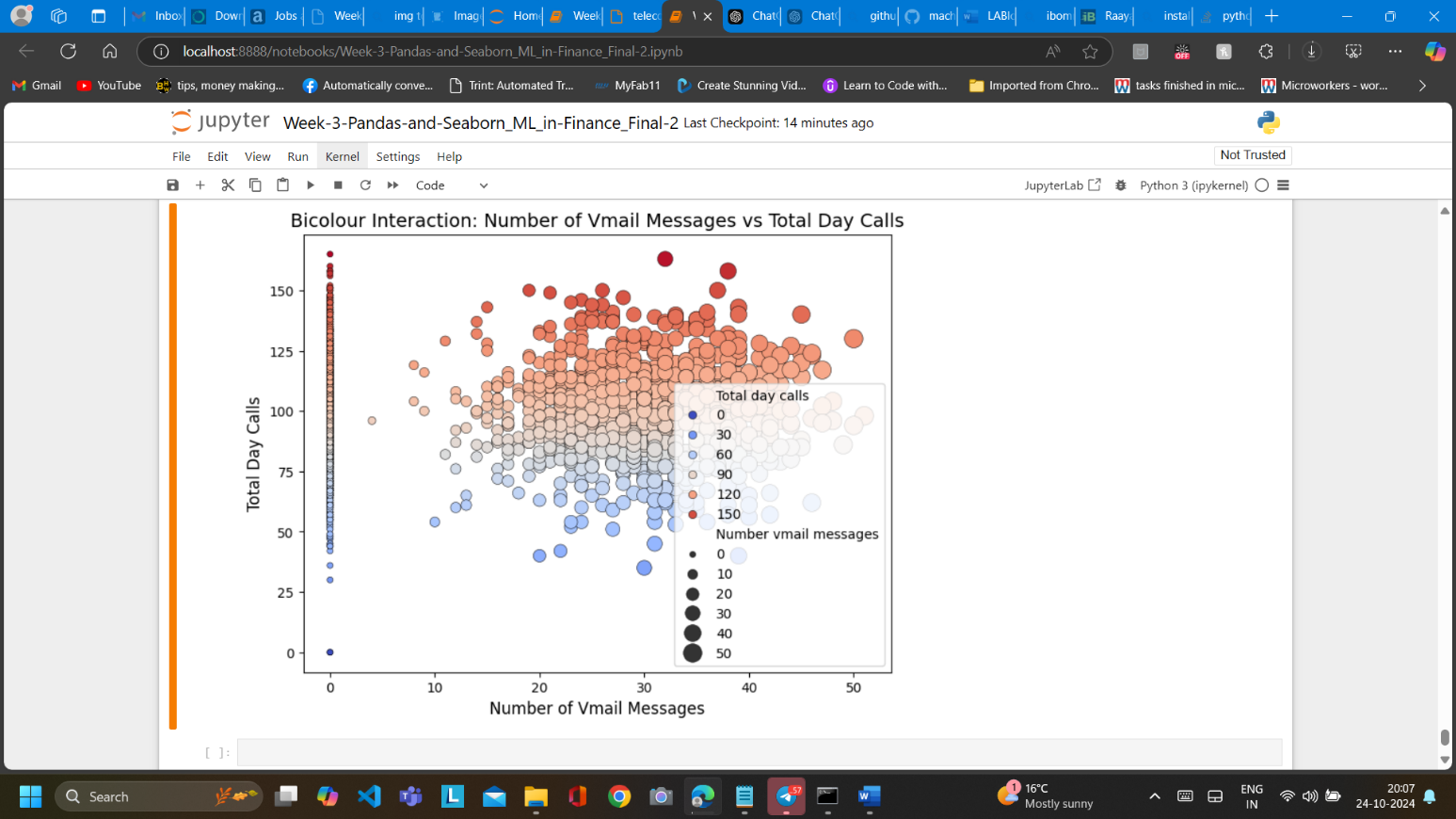
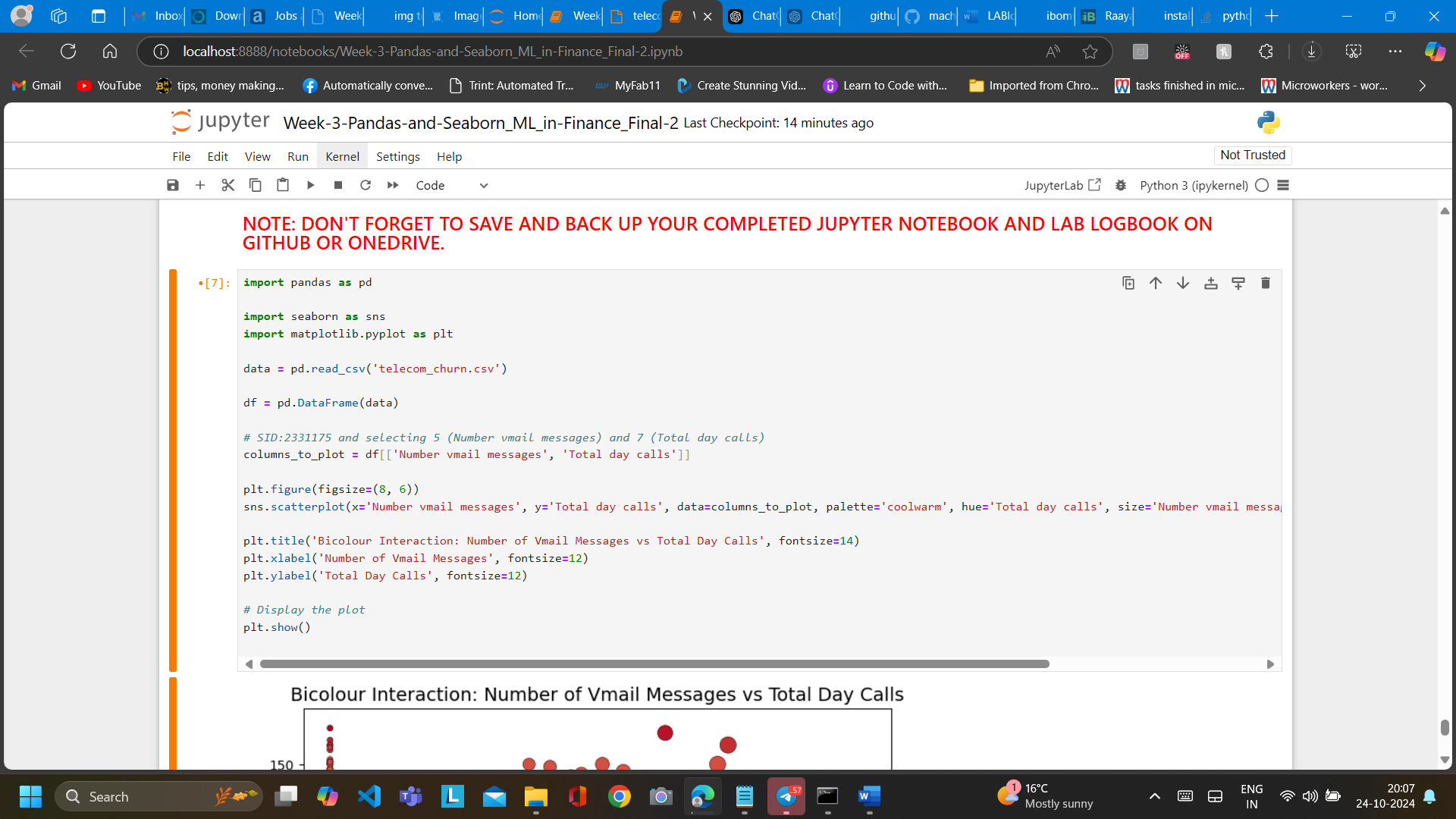
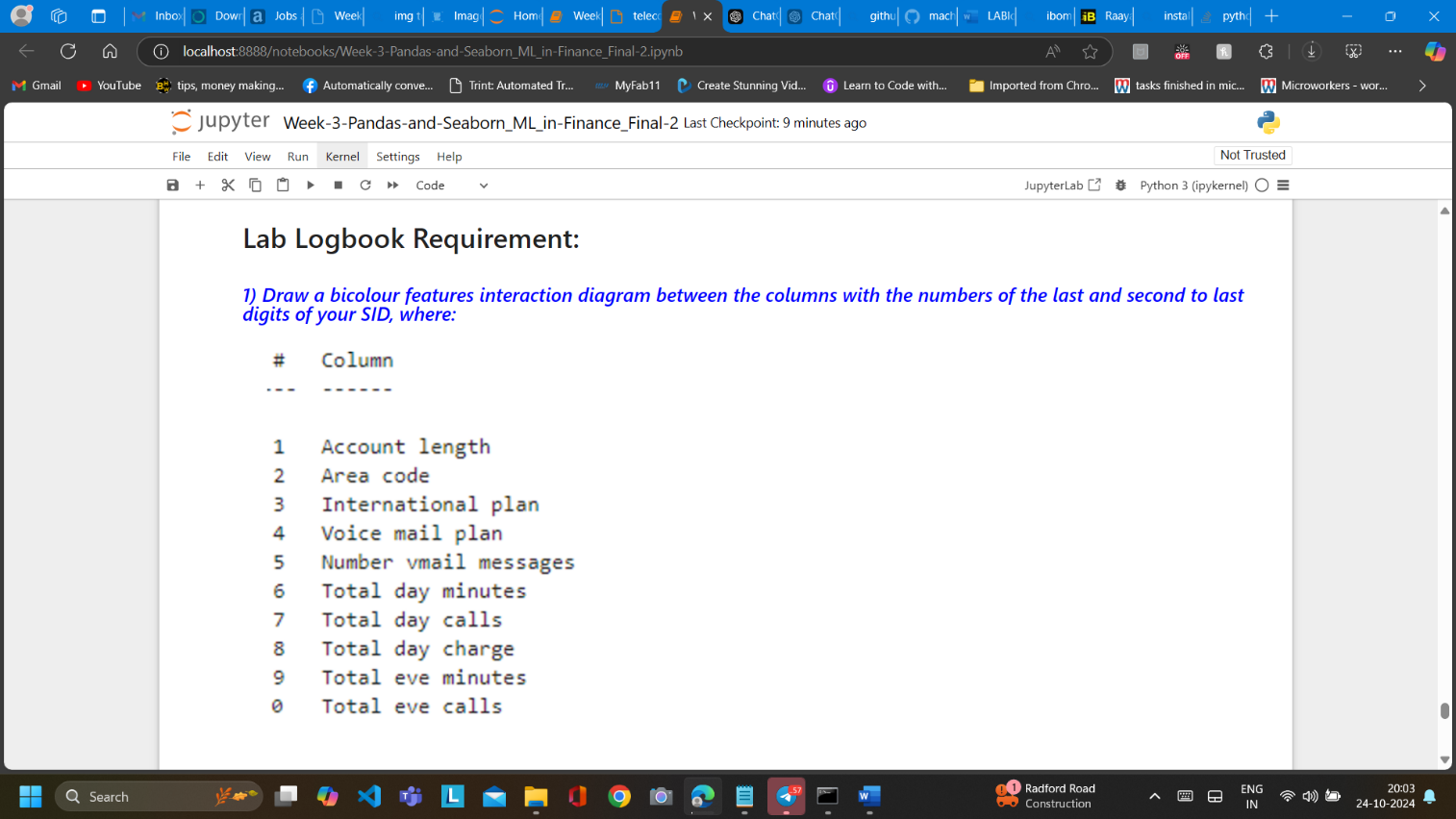
plt.title('Bicolour Interaction: Number of Vmail Messages vs Total Day Calls', fontsize=14)

plt.xlabel('Number of Vmail Messages', fontsize=12)

plt.ylabel('Total Day Calls', fontsize=12)

# Display the plot

plt.show()



Lab 4

Code:

import pandas as pd

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input

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

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import mean\_absolute\_error

data = pd.read\_csv('data\_stocks.csv')

X = data.drop(columns=['SP500'])

y = data['SP500']

scaler = MinMaxScaler()

X\_scaled = scaler.fit\_transform(X)

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

model = Sequential([

Input(shape=(X\_train.shape[1],)),

Dense(175, activation='relu'),

Dense(88, activation='relu'),

Dense(1)

])

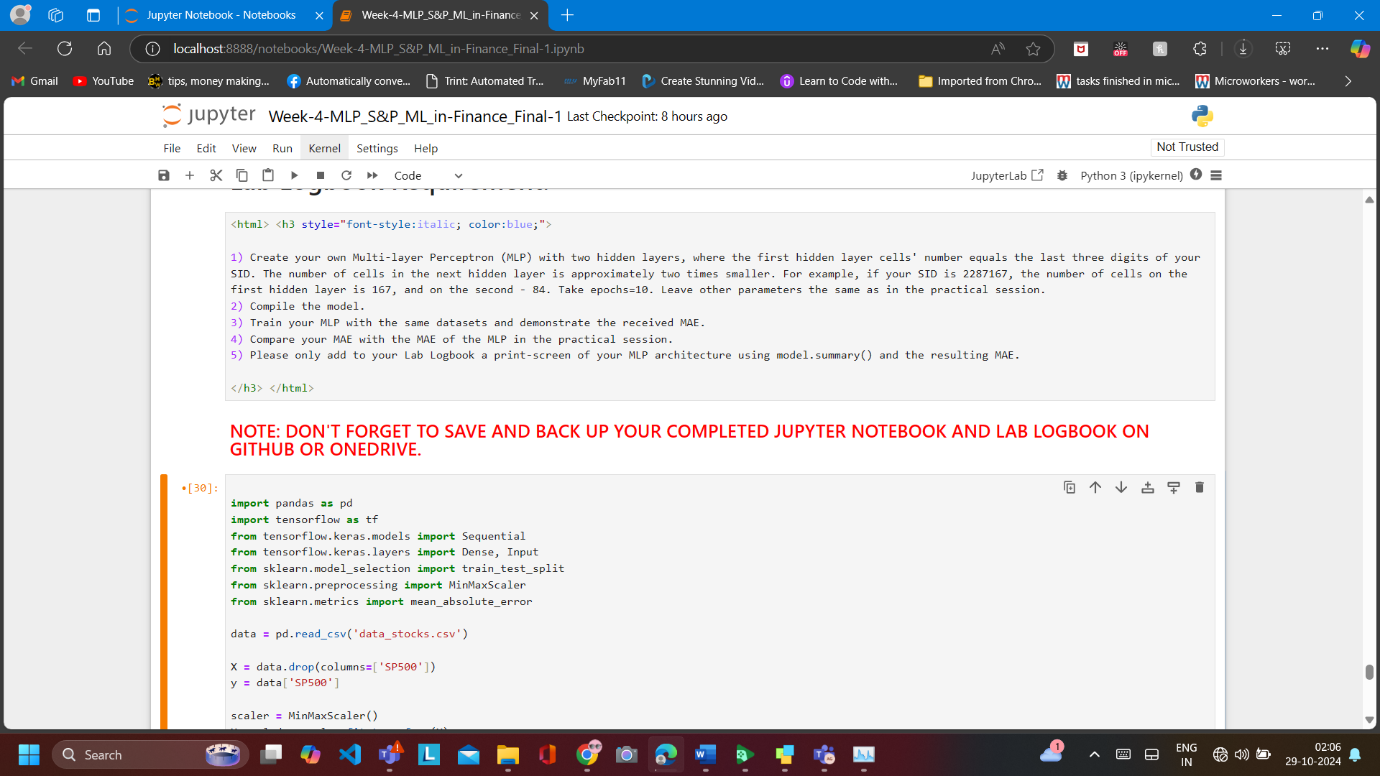
model.compile(optimizer='adam', loss='mse', metrics=['mae'])

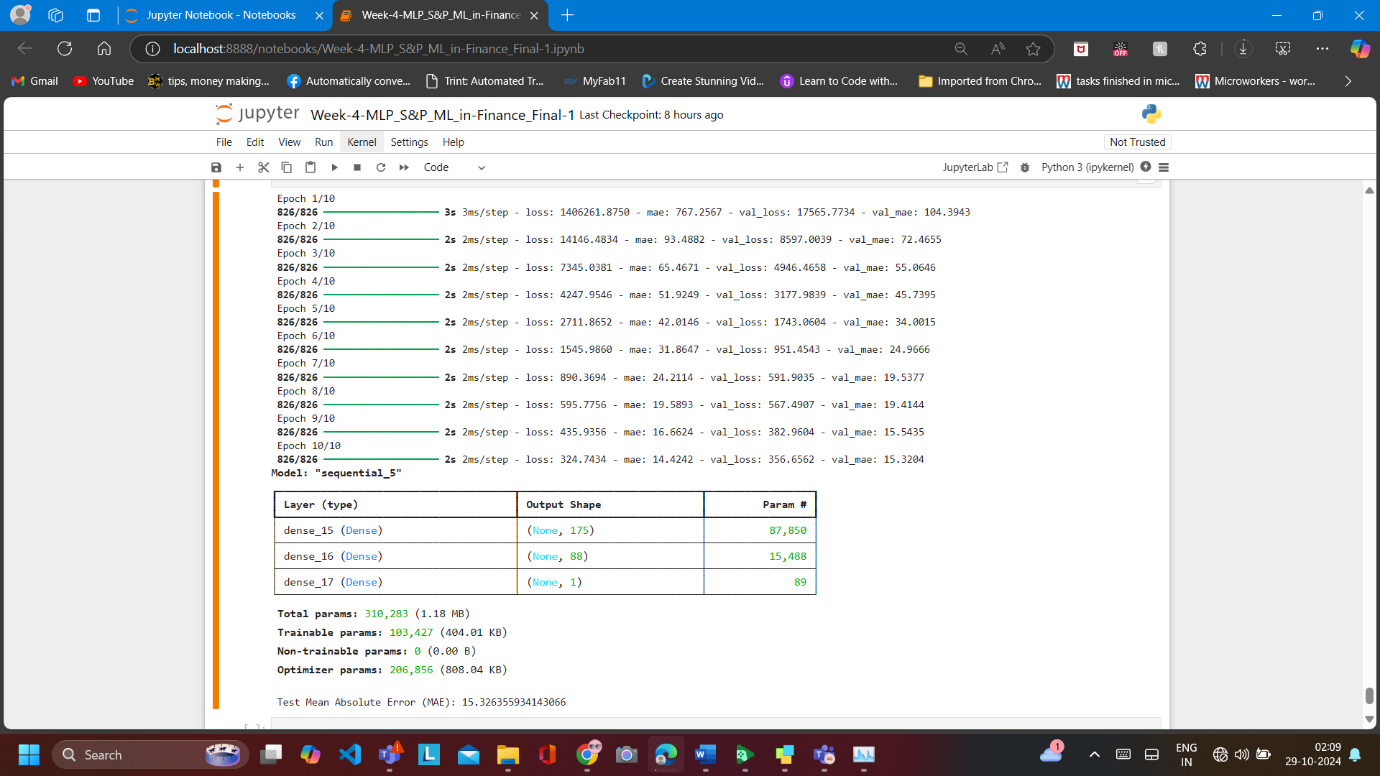
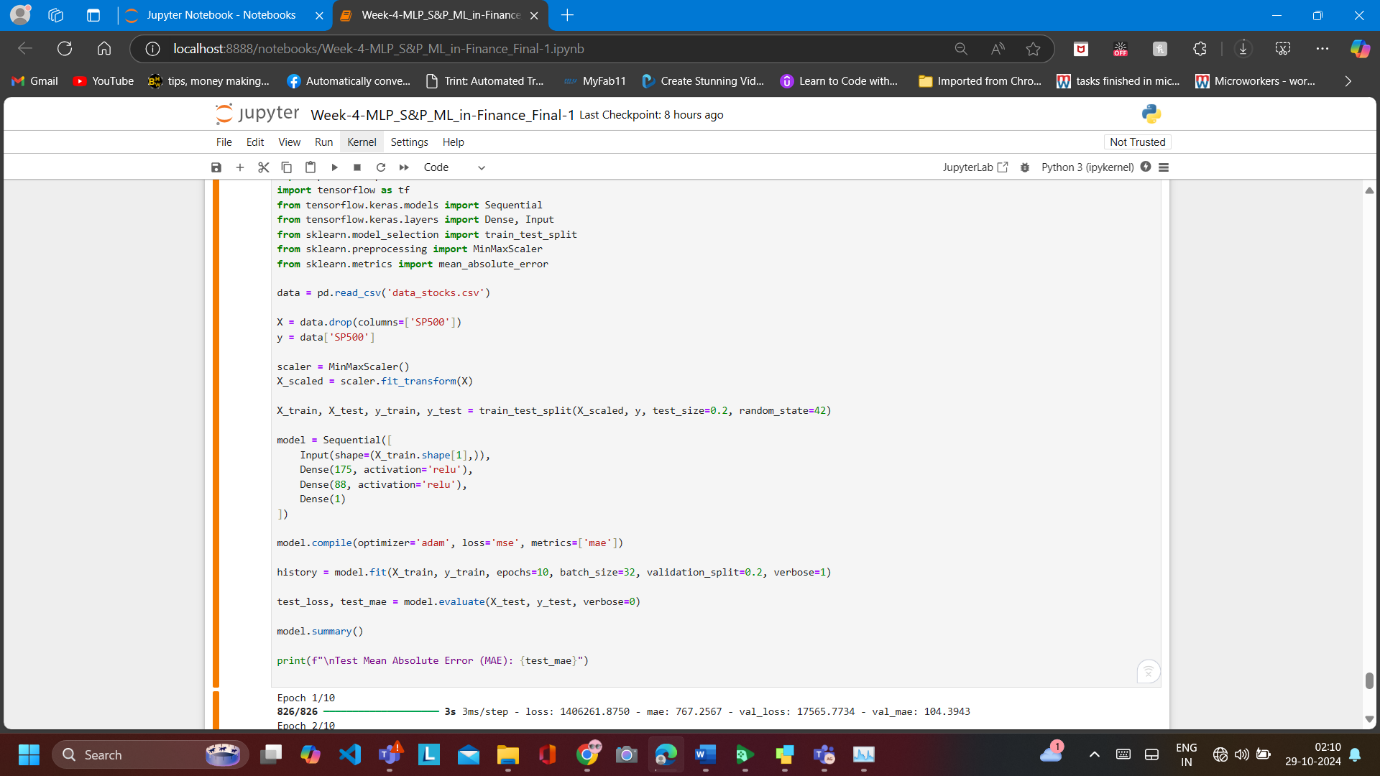
history = model.fit(X\_train, y\_train, epochs=10, batch\_size=32, validation\_split=0.2, verbose=1)

test\_loss, test\_mae = model.evaluate(X\_test, y\_test, verbose=0)

model.summary()

print(f"\nTest Mean Absolute Error (MAE): {test\_mae}")





Lab 5

Lab 6

Lab 7

Lab 8

Lab 9

Lab 10

Lab 11

Lab 12