Spring 2024: CS5720 – NN &DL

BHAVANA BILLA -700756590-ICP4

https://github.com/BillaBhavana7/neuralN/tree/main

1. Data Manipulation

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

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

c. Show the basic statistical description about the data.

d. Check if the data has null values.

   i. Replace the null values with the mean

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

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

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

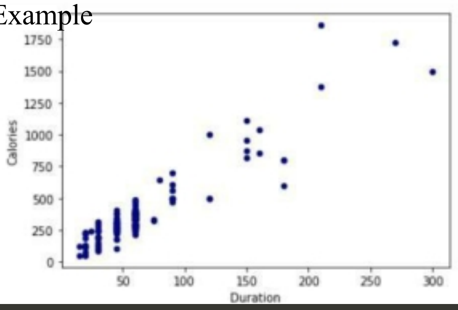
h. Create a new “df_modified” dataframe that contains all the columns from df except for “Maxpulse”.

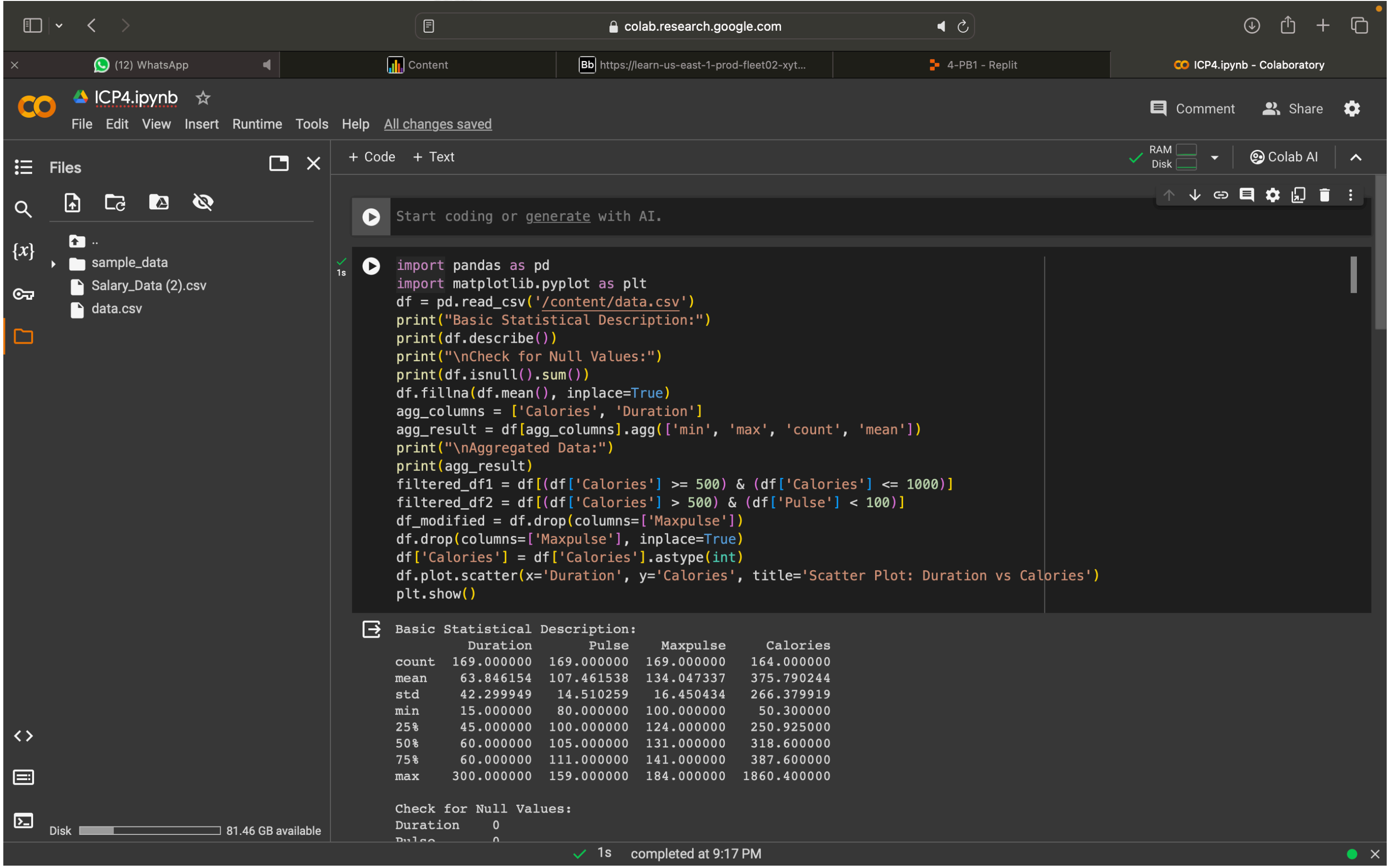
i. Delete the “Maxpulse” column from the main df dataframe

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

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

Example



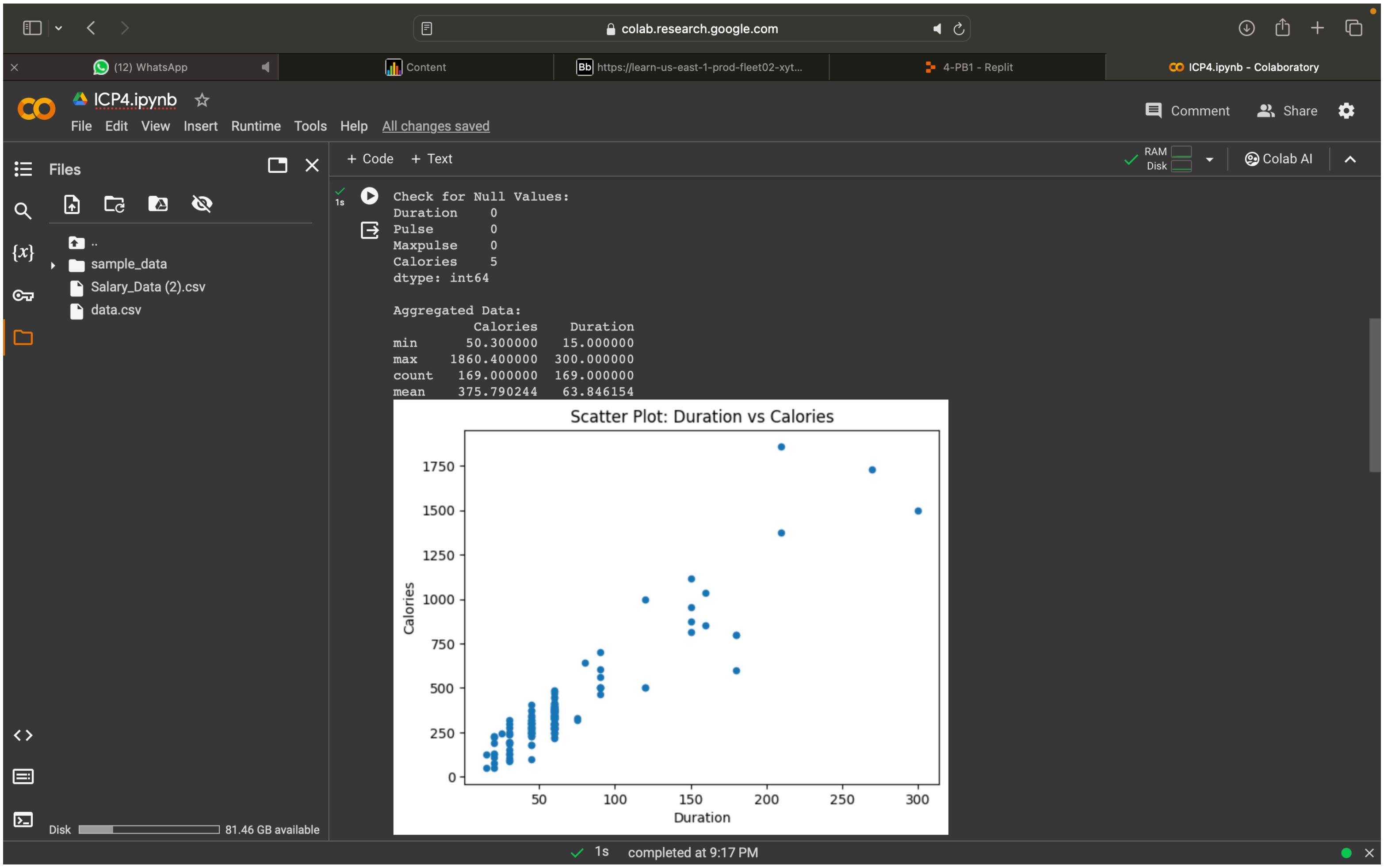

colab.research.google.com
(12) WhatsApp
Content
https://learn-us-east-1-prod-fleet02-xyt...
4-PB1 - Replit
ICP4.ipynb - Colaboratory
ICP4.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
sample_data
Salary_Data (2).csv
data.csv
+ Code + Text
RAM
Disk
Colab AI
Start coding or generate with AI.
Disk 81.46 GB available

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('/content/data.csv')
print("Basic Statistical Description:")
print(df.describe())
print("\nCheck for Null Values:")
print(df.isnull().sum())
df.fillna(df.mean(), inplace=True)
agg_columns = ['Calories', 'Duration']
agg_result = df[agg_columns].agg(['min', 'max', 'count', 'mean'])
print("\nAggregated Data:")
print(agg_result)
filtered_df1 = df[(df['Calories'] >= 500) & (df['Calories'] <= 1000)]
filtered_df2 = df[(df['Calories'] > 500) & (df['Pulse'] < 100)]
df_modified = df.drop(columns=['Maxpulse'])
df.drop(columns=['Maxpulse'], inplace=True)
df['Calories'] = df['Calories'].astype(int)
df.plot.scatter(x='Duration', y='Calories', title='Scatter Plot: Duration vs Calories')
plt.show()
```

```
Basic Statistical Description:
         Duration       Pulse    Maxpulse     Calories
count  169.000000  169.000000  169.000000   164.000000
mean    63.846154  107.461538  134.047337   375.790244
std     42.299949   14.510259   16.450434   266.379919
min     15.000000   80.000000  100.000000    50.300000
25%     45.000000  100.000000  124.000000   250.925000
50%     60.000000  105.000000  131.000000   318.600000
75%     60.000000  111.000000  141.000000   387.600000
max    300.000000  159.000000  184.000000  1860.400000

Check for Null Values:
Duration    0
Pulse       0
```

1s completed at 9:17 PM



```
Check for Null Values:
Duration     0
Pulse        0
Maxpulse     0
Calories     5
dtype: int64

Aggregated Data:
          Calories    Duration
min      50.300000   15.000000
max    1860.400000  300.000000
count   169.000000  169.000000
mean    375.790244   63.846154
```

2. Linear Regression

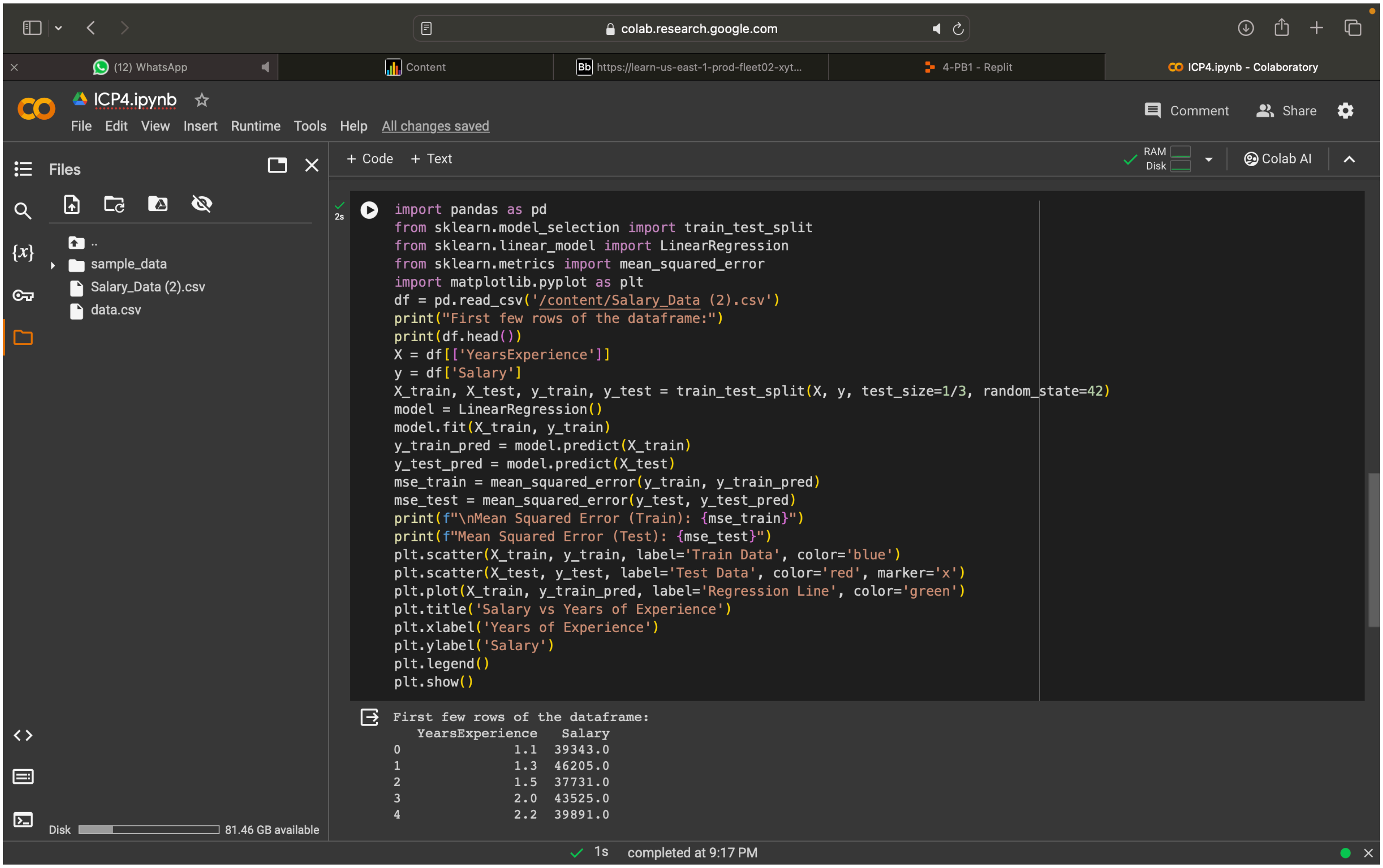
a) Import the given “Salary_Data.csv”

b) Split the data in train_test partitions, such that 1/3 of the data is reserved as test subset.

c) Train and predict the model.

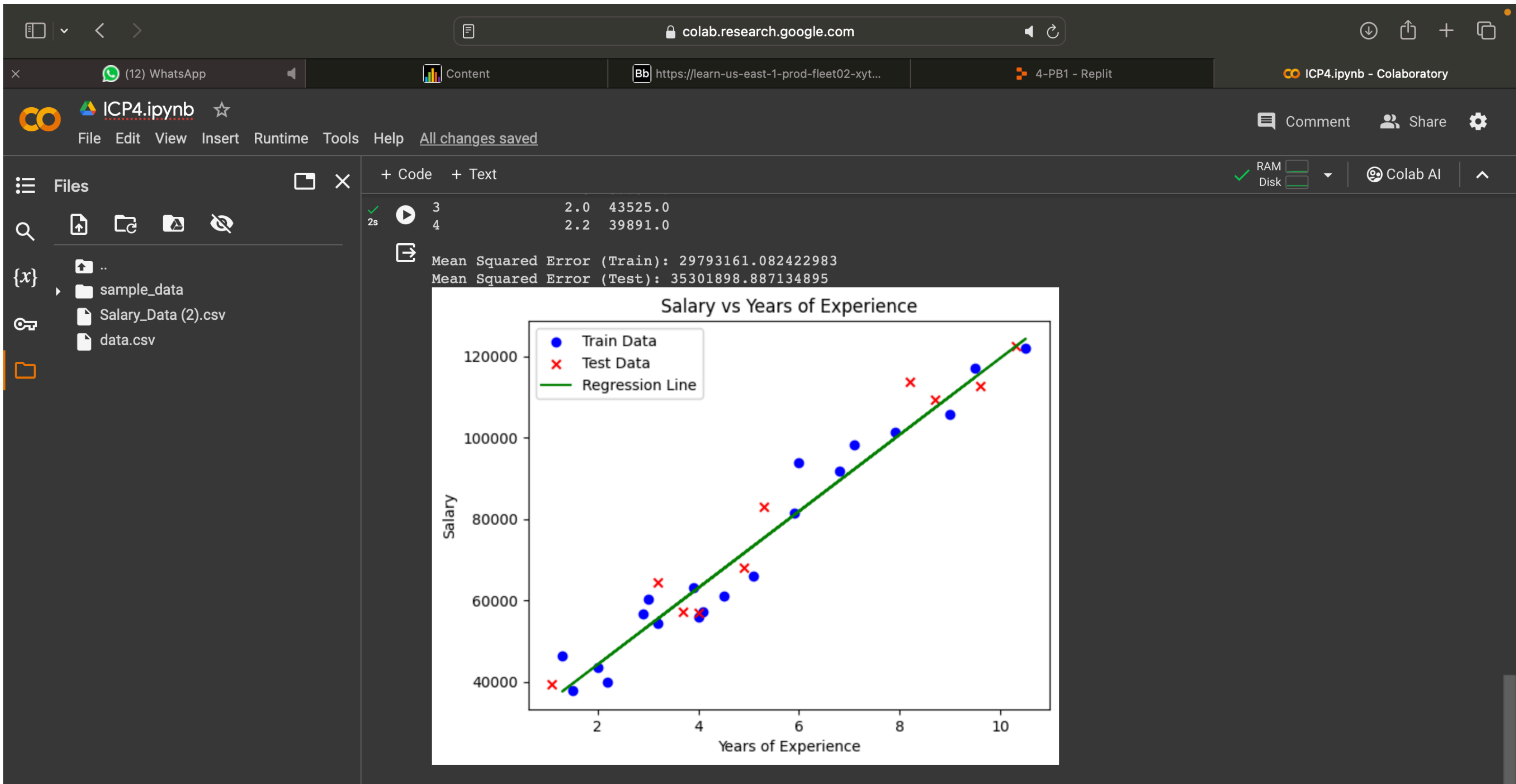
d) Calculate the mean_squared error

e) Visualize both train and test data using scatter plot.



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
df = pd.read_csv('/content/Salary_Data (2).csv')
print("First few rows of the dataframe:")
print(df.head())
X = df[['YearsExperience']]
y = df['Salary']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)
mse_train = mean_squared_error(y_train, y_train_pred)
mse_test = mean_squared_error(y_test, y_test_pred)
print(f"\nMean Squared Error (Train): {mse_train}")
print(f"Mean Squared Error (Test): {mse_test}")
plt.scatter(X_train, y_train, label='Train Data', color='blue')
plt.scatter(X_test, y_test, label='Test Data', color='red', marker='x')
plt.plot(X_train, y_train_pred, label='Regression Line', color='green')
plt.title('Salary vs Years of Experience')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.legend()
plt.show()
```

```
First few rows of the dataframe:
   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
```



```
3              2.0  43525.0
4              2.2  39891.0

Mean Squared Error (Train): 29793161.082422983
Mean Squared Error (Test): 35301898.887134895
```