#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)

import pandas as pandas

import pandas as pd

df = pd.read\_csv("data.csv")

df.describe()

A screenshot of a calculator

Description automatically generated

null\_values = df.isnull().sum()

#is null is to identify null values

print(null\_values)

A screenshot of a computer

Description automatically generated

df.fillna(df.mean(), inplace=True)

#fillna is to fill the null position

print(df)

A table of numbers with text

Description automatically generated with medium confidence

df = df[["Duration", "Pulse" ]]

agg\_dict = {"Duration": ["max", "min", "count", "mean"],

            "Pulse": ["max", "min", "count", "mean"]}

agg\_df = df.agg(agg\_dict)

print(agg\_df)

A white background with black numbers

Description automatically generated

df = pd.read\_csv("data.csv")

Calories\_filter = (df["Calories"] >= 500) & (df["Calories"] <= 1000)

filtered\_df = df[Calories\_filter]

print(filtered\_df)

A table of numbers with black text

Description automatically generated

df = pd.read\_csv("data.csv")

Calories\_filter = (df["Calories"] > 500) & (df["Pulse"] < 100)

filtered\_df = df[Calories\_filter]

print(filtered\_df)

A table with numbers and text

Description automatically generated

df\_modified = df.drop(columns=["Maxpulse"])

print(df\_modified)

A screenshot of a calories table

Description automatically generated

df['Calories'] = df['Calories'].fillna(0).astype(int)

print(df)

A screenshot of a calculator

Description automatically generated

import matplotlib.pyplot as plt

df.plot(kind='scatter', x='Duration', y='Calories', figsize=(6,3))

plt.show()

A graph with blue dots

Description automatically generated

#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.

import pandas as pd

from sklearn.metrics import mean\_squared\_error

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

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

salariesData = pd.read\_csv('Salary\_Data.csv') #importing data from the CSV file

#Here we are splitting the data in to 2 categories: training and testing

X = salariesData.iloc[:, :-1].values

Y= salariesData.iloc[:, 1].values

#splitting the 1/3 of the data

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 1/3, random\_state = 0)

# using reg.fit i am fitting Simple Linear Regression to my training set

reg = LinearRegression()

reg.fit(X\_train, Y\_train)

# used reg.Predict to predict the test set result

pred = reg.predict(X\_test)

# Calculating the Mean\_squared\_error

mse = mean\_squared\_error(Y\_test, pred)

print(mse)

#Visualisation of Training set results and Test set results

plt.scatter(X\_train, Y\_train, color = 'blue')

plt.scatter(X\_test, Y\_test, color = 'red')

plt.title('Salary Data')

plt.xlabel('Experience (Years)')

plt.ylabel('Salary')

plt.show()

A graph with red and blue dots

Description automatically generated