**Neural Networks and Deep Learning – Assignment 4**

**GitHub Link: <https://github.com/shanmukha1610/NNDL_ASS4>**

**Video Link:<https://drive.google.com/file/d/1Fcas7KiSa1djTJvzYB7wi7ju8CaBhGN8/view?usp=drive_link>**

**Name: Pasumarthi Shanmukha Laxmi**

**Student ID: 700756736**

1. **Data Manipulation:**
2. Read the provided CSV file ‘data.csv’
3. https://drive.google.com/drive/folders/1h8C3mLsso-R-sIOLsvoYwPLzy2fJ4IOF?usp=sharing
4. Show the basic statistical description about the data.

In this program, I have first imported the libraries required which are pandas, numpy and matplotlib. To read the given csv file I have used pandas pd.read\_csv method into Data Frame. This supports optionally iterating or breaking of the file into chunks too.

For question c. I have used the describe () method which returns description of the data in the Data Frame. As our Data Frame contains numerical data the description contains for each column as count, mean, standard deviation, min, max values as statistical description.

Table

Description automatically generated

1. Check if the data has null values.
2. Replace the null values with the mean

To check if the given data contains null values I have used df.isna().sum() method. This helps in returning the number of Null values in all columns of a pandas DataFrame. It shows the total Null values in a particular colum. As per our data file, the output shows that the Calories column has 5 Null values. Now these must be replaced with the mean values.

To calculate the mean() we use the mean function of the particular column. With the help of fillna() function we will change all ‘Null’ of that particular column for which we have its mean.

The output below checks the null values first and prints then the null values are replaced with mean as shown. We then again verify if columns still have the null values. As shown in the output the Null values are 0 after replacing.

Table

Description automatically generated with medium confidence

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

To do this I have used groupby to group all the data in the dataframe and then by using the aggregate () method which allows to apply a function or a list of function names to be executed along one of the axis of the DataFrame. By using this we aggregate the min, max, mean and count of the given data in dataframe. The output is as shown below:

Table

Description automatically generated

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

The df['calories'] > 500 expression creates a boolean array that is True for the rows where the calories value is greater than 500. The df['calories'] < 1000 expression creates a boolean array that is True for the rows where the calories value is less than 1000. The & operator is used to combine these two arrays into a single boolean array that is True only for the rows where both conditions are True.

1. The df['calories'] > 500 expression creates a boolean array that is True for the rows where the calories value is greater than 500. The df[pulse] < 100 expression creates a boolean array that is True for the rows where the calories value is less than 100. The & operator is used to combine these two arrays into a single boolean array that is True only for the rows where both conditions are True.

Table

Description automatically generated

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

To do this I have used drop method that removes the specified column or from the dataframe. In this case, we're specifying axis=1 to indicate that we want to drop a column, and the "Maxpulse" column will be dropped from the dataframe. The printed output is as shown below:

Graphical user interface, text, application

Description automatically generated

1. To delete the “Maxpulse” column from the df dataframe I have used the drop method. This method removes the column from the original main dataframe and by printing we show the resultant as shown below:

Graphical user interface, text, application

Description automatically generated

1. To Convert the datatype of Calories column to int datatype, I have used the astype method which changes the datatype of values in the column to integer type. Originally the values in our data file were float. The output after conversion looks like this:

Table

Description automatically generated

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

we have already imported the matplotlib.pyplot library as plt, which is used to create the scatter plot. We use the plot method to create a scatter plot. Then the we create the scatter plot using the scatter. The resultant scatter plot is as shown below:

Chart, scatter chart

Description automatically generated

1. **Linear Regression:**
2. Import the given “Salary\_Data.csv”

In this program I have first imported the required libraries which are numpy, matplotlib.pyplot and pandas. For the first question to read the given csv file. I have used the pd.read\_csv method which reads the given Salary\_Data.csv file and stores in saldata. Using the info and head() methods I have printed the info from file.

Text

Description automatically generated

1. Split the data in train\_test partitions, such that 1/3 of the data is reserved as test subset.

To split the data in partitions we first import the train\_test\_split function from the sklearn.model\_selection. Then I have split the data into training and testing sets using the train\_test\_split function. The test\_size parameter is set to 1/3, meaning that 1/3 of the data will be reserved for testing and the remaining 2/3 will be used for training. The random\_state parameter is set to 0 to ensure that the same random partition of the data is obtained each time the code is run.

I have then printed the split data as shown below:

A picture containing table

Description automatically generated

Text

Description automatically generated with medium confidence

1. To train and predict the model LinearRegression class is used to create the linear regression model. The fit method is used to train the model on the training data, and the predict method is used to make predictions on the test data. The y\_pred variable contains the predicted values for the test data. As shown below:

Graphical user interface, text

Description automatically generated

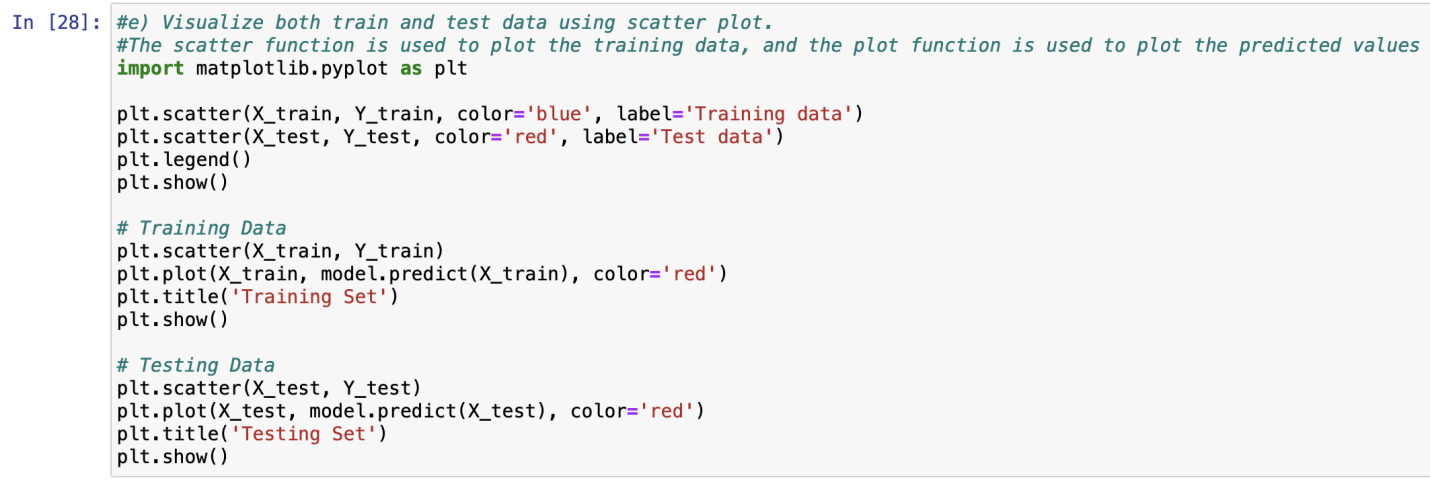
1. To calculate the mean squared error: Here, y\_test is the true values, and y\_pred is the predicted values. The mean\_squared\_error function takes these two arrays as input and returns the mean squared error between the two.

Text

Description automatically generated

1. To visualize both the train and test data using the scatter plot I have used the imported library which is matplotlib.pyplot as plt. Below, X\_train and y\_train are the training data and target variables, and X\_test and y\_test are the test data and target variables, respectively. The scatter function is used to plot the data, and the color argument is used to specify the color of the data points. The label argument is used to specify the label for the legend. The legend function is used to add a legend to the plot, and the show function is used to display the plot.

The resultant scatter plot is as shown below:



Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated