**NEURAL NETWORKS DEEP LEARNING**

Lavanya Sushma Gorijala

700760704

Github link: <https://github.com/lavanyasushma/ICP3>

import pandas as pd

from io import StringIO

# CSV data as a string

csv\_data = """Duration,Pulse,Maxpulse,Calories

60,110,130,409.1

.

.

.

75,125,150,330.4

"""

# Create DataFrame

df = pd.read\_csv(StringIO(csv\_data))

# Show the first few rows

print(df.head())

Duration Pulse Maxpulse Calories

0 60 110 130 409.1

1 60 117 145 479.0

2 60 103 135 340.0

3 45 109 175 282.4

4 45 117 148 406.0

# Show basic statistical description

print(df.describe())

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

print(df.isnull().sum())

Duration 0

Pulse 0

Maxpulse 0

Calories 5

dtype: int64

#Replace null values with mean

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

# Select two columns for aggregation

columns\_to\_aggregate = ['Calories', 'Duration']

aggregated\_data = df[columns\_to\_aggregate].agg(['min', 'max', 'count', 'mean'])

print(aggregated\_data)

Calories Duration

min 50.300000 15.000000

max 1860.400000 300.000000

count 169.000000 169.000000

mean 375.790244 63.846154

# Filter rows with calories between 500 and 1000

filtered\_df = df[(df['Calories'] >= 500) & (df['Calories'] <= 1000)]

print(filtered\_df)

Duration Pulse Maxpulse Calories

51 80 123 146 643.1

62 160 109 135 853.0

65 180 90 130 800.4

66 150 105 135 873.4

67 150 107 130 816.0

72 90 100 127 700.0

73 150 97 127 953.2

75 90 98 125 563.2

78 120 100 130 500.4

83 120 100 130 500.0

90 180 101 127 600.1

99 90 93 124 604.1

101 90 90 110 500.0

102 90 90 100 500.0

103 90 90 100 500.4

106 180 90 120 800.3

108 90 90 120 500.3

# Create a new dataframe without the 'Maxpulse' column

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

print(df\_modified.head())

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

# Delete the 'Maxpulse' column from the main dataframe

df.drop(columns=['Maxpulse'], inplace=True)

print(df.head())

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

# Convert 'Calories' column to int datatype

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

print(df.dtypes)

Duration int64

Pulse int64

Calories int32

dtype: object

import matplotlib.pyplot as plt

# Scatter plot of Duration vs Calories

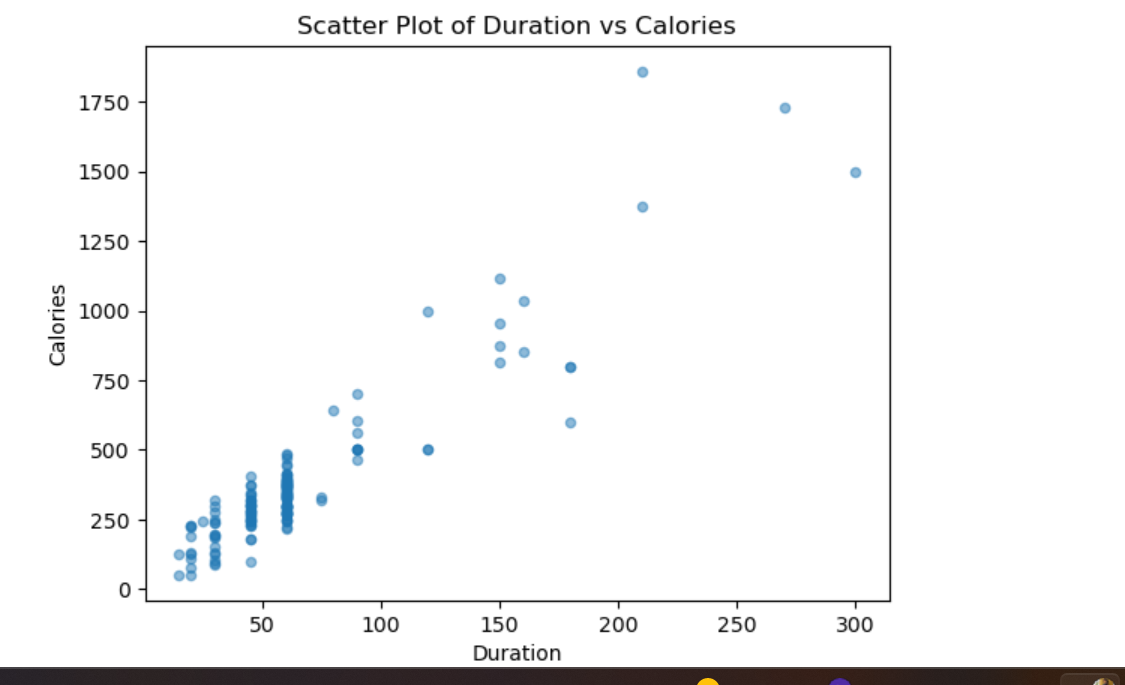
df.plot(kind='scatter', x='Duration', y='Calories', alpha=0.5)

plt.title('Scatter Plot of Duration vs Calories')

plt.xlabel('Duration')

plt.ylabel('Calories')

plt.show()



import pandas as pd

from io import StringIO

# CSV data as a string

csv\_data = """YearsExperience,Salary

1.1,39343.00

3.9,63218.00

10.5,121872.00

"""

# Create DataFrame

df = pd.read\_csv(StringIO(csv\_data))

# Show the first few rows

print(df.head())

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

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

# Split the data into features and target variable

X = df[['YearsExperience']] # Feature

y = df['Salary'] # Target

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

# Print the shapes of the resulting datasets

print(f"Training set size: {X\_train.shape[0]}")

print(f"Test set size: {X\_test.shape[0]}")

Training set size: 20

Test set size: 10

from sklearn.linear\_model import LinearRegression

# Create and train the model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Predict on the test set

y\_pred = model.predict(X\_test)

# Print the first few predictions

print("First few predictions:")

print(y\_pred[:5])

First few predictions:

[115814.30756236 71511.92534771 102617.85328566 75282.34085534

55487.65944028]

from sklearn.metrics import mean\_squared\_error

# Calculate the mean squared error

mse = mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error: {mse}")

Mean Squared Error: 35301898.887134895

import matplotlib.pyplot as plt

# Plot training data

plt.scatter(X\_train, y\_train, color='blue', label='Training Data')

# Plot test data and predictions

plt.scatter(X\_test, y\_test, color='green', label='Test Data')

plt.scatter(X\_test, y\_pred, color='red', label='Predictions')

# Add labels and title

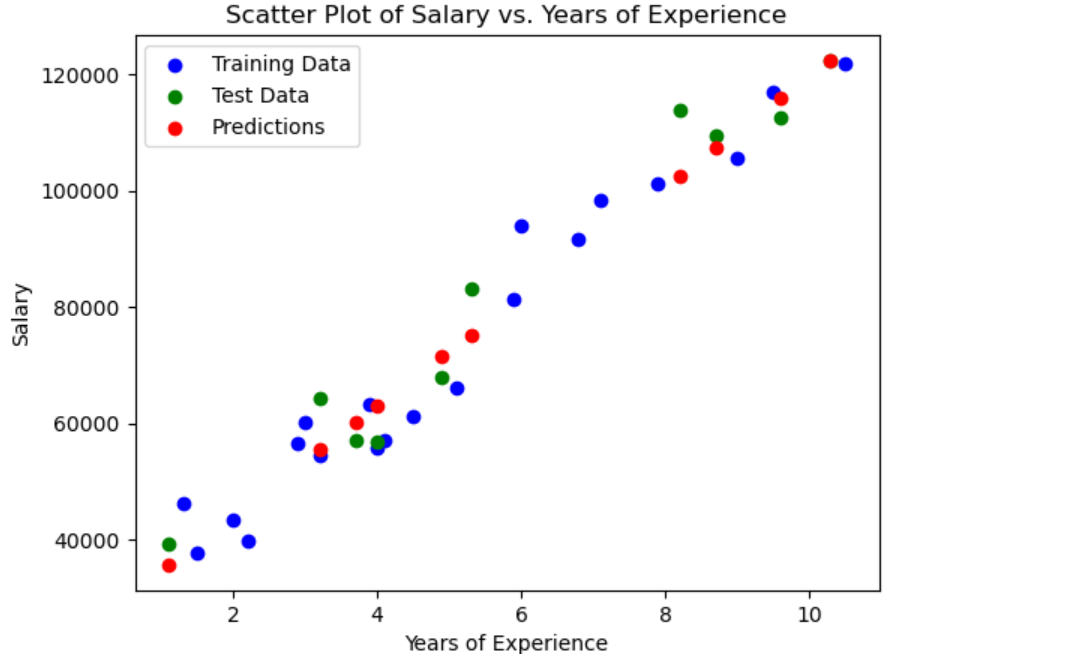
plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.title('Scatter Plot of Salary vs. Years of Experience')

plt.legend()

plt.show()



import pandas as pd

from io import StringIO

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

# CSV data as a string

csv\_data = """YearsExperience,Salary

1.1,39343.00

.

.

.

10.5,121872.00

"""

# Create DataFrame

df = pd.read\_csv(StringIO(csv\_data))

# Split the data into features and target variable

X = df[['YearsExperience']]

y = df['Salary']

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

# Create and train the model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Predict on the test set

y\_pred = model.predict(X\_test)

# Calculate the mean squared error

mse = mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error: {mse}")

# Plot training data

plt.scatter(X\_train, y\_train, color='blue', label='Training Data')

# Plot test data and predictions

plt.scatter(X\_test, y\_test, color='green', label='Test Data')

plt.scatter(X\_test, y\_pred, color='red', label='Predictions')

# Add labels and title

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.title('Scatter Plot of Salary vs. Years of Experience')

plt.legend()

plt.show()

