Department of Electrical Engineering

Faculty Member: LE Munadi Sial **Date: 29/12/2024**
Semester: 7th **Group: 1**

CS471 Machine Learning

Lab 14: Anomaly Detection

		PLO4 - CLO4	PLO 4 -CLO4	PLO5 - CLO5	PLO 8 -CLO6	PLO 9 -CLO7
Name	**Reg. No**	**Viva /Quiz / Lab Performance** **5 Marks**	**Analysis of data in Lab Report** **5 Marks**	**Modern Tool Usage** **5 Marks**	**Ethics** **5 Marks**	**Individual and Team Work** **5 Marks**
Muhammad Anser Sohaib	**367628**					
Muhammad Talha	**370860**					
Muhammad Taqi	**372071**					
Zuha Fatima	**385647**					
Hashaam Zafar	**395508**					

Introduction

This laboratory exercise will focus on the concept on the technique of anomaly detection which is an unsupervised learning approach to detect outliers.

Objectives

The following are the main objectives of this lab:

- Perform anomaly detection from scratch to detect outliers
- Perform anomaly detection from Sci-kit Learn toolkit

Lab Conduct

- Respect faculty and peers through speech and actions
- The lab faculty will be available to assist the students. In case some aspect of the lab experiment is not understood, the students are advised to seek help from the faculty.
- In the tasks, there are commented lines such as #YOUR CODE STARTS HERE# where you have to provide the code. You must put the code/screenshot/plot between the #START and #END parts of these commented lines. Do NOT remove the commented lines.
- Use the tab key to provide the indentation in python.
- When you provide the code in the report, keep the font size at 12

Theory

Anomaly detection is an unsupervised learning approach used to detect outliers in a dataset. Anomaly detection can be used as preprocessing task for removing unwanted outliers from the dataset and is also useful for fraud detection as fraud activity differs from user activity and can serve as a potential outlier.

A brief summary of the relevant keywords and functions in python is provided below:

print()	output text on console
input()	get input from user on console
range()	create a sequence of numbers
len()	gives the number of characters in a string
if	contains code that executes depending on a logical condition
else	connects with **if** and **elif**, executes when conditions are not met
elif	equivalent to **else if**
while	loops code as long as a condition is true
for	loops code through a sequence of items in an iterable object
break	exit loop immediately
continue	jump to the next iteration of the loop
def	used to define a function
pd.read_csv	import csv file as a dataframe
df.to_csv	export dataframe as a csv file

Lab Task 1 – Gaussian Distribution _____

Write a function in python that generates a Gaussian distribution given a mean value and standard deviation value. Provide the codes and the plot at least 3 distributions.

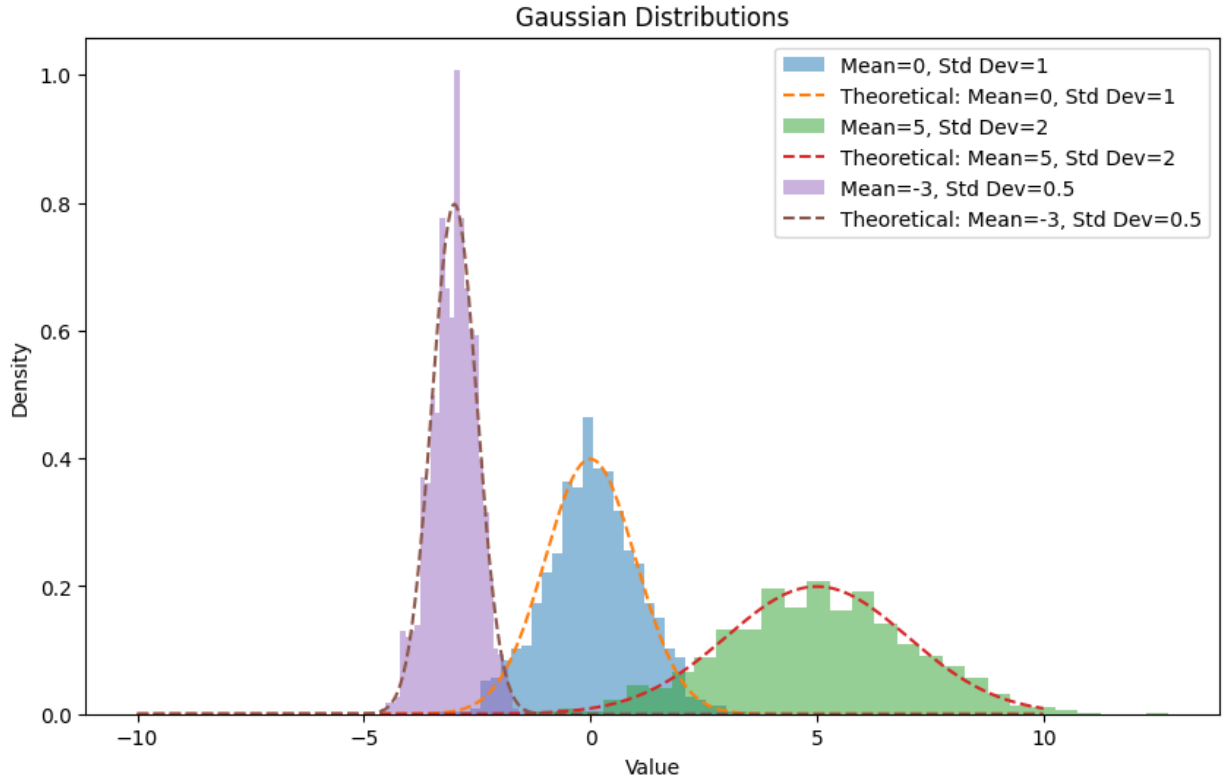
TASK 1 CODE STARTS HERE

```
import numpy as np
import matplotlib.pyplot as plt

def generate_gaussian_distribution(mean, std_dev, size=1000):
    data = np.random.normal(loc=mean, scale=std_dev, size=size)
    return data

params = [(0, 1), (5, 2), (-3, 0.5)]  # (mean, std_dev)

plt.figure(figsize=(10, 6))

x = np.linspace(-10, 10, 1000)  # x values for plotting
for mean, std_dev in params:
    # Generate the Gaussian distribution
    data = generate_gaussian_distribution(mean, std_dev)
    # Plot the histogram of the generated data
    plt.hist(data, bins=30, density=True, alpha=0.5, label=f'Mean={mean}, Std
Dev={std_dev}')

    # Plot the theoretical Gaussian distribution curve
    plt.plot(x, (1/(std_dev * np.sqrt(2 * np.pi))) * np.exp(-0.5 * ((x - mean) /
std_dev) ** 2), label=f'Theoretical: Mean={mean}, Std Dev={std_dev}',
linestyle='--')

# Set plot labels and title
plt.title('Gaussian Distributions')
plt.xlabel('Value')
plt.ylabel('Density')
plt.legend()

# Show the plot
plt.show()
```

TASK 1 CODE ENDS HERE

TASK 1 SCREENSHOT STARTS HERE

All three distributions in one plot.



TASK 1 SCREENSHOT ENDS HERE

Lab Task 2 – Dataset with Outliers

Download a dataset containing at least 3 features and 500 examples. Create a scatter plot of the dataset. Calculate the mean and standard deviations of each feature. Next, add a few anomalies into the dataset and plot it again. Provide the code and all relevant screenshots.

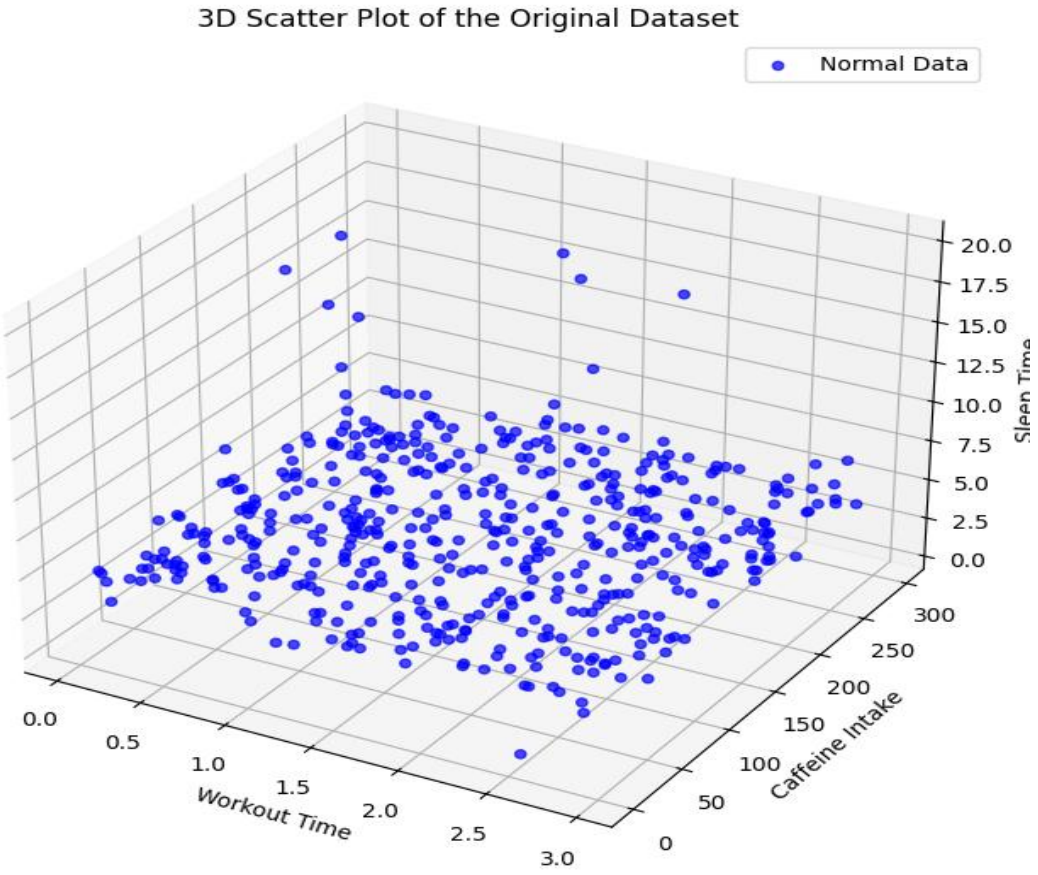
TASK 2 CODE STARTS HERE

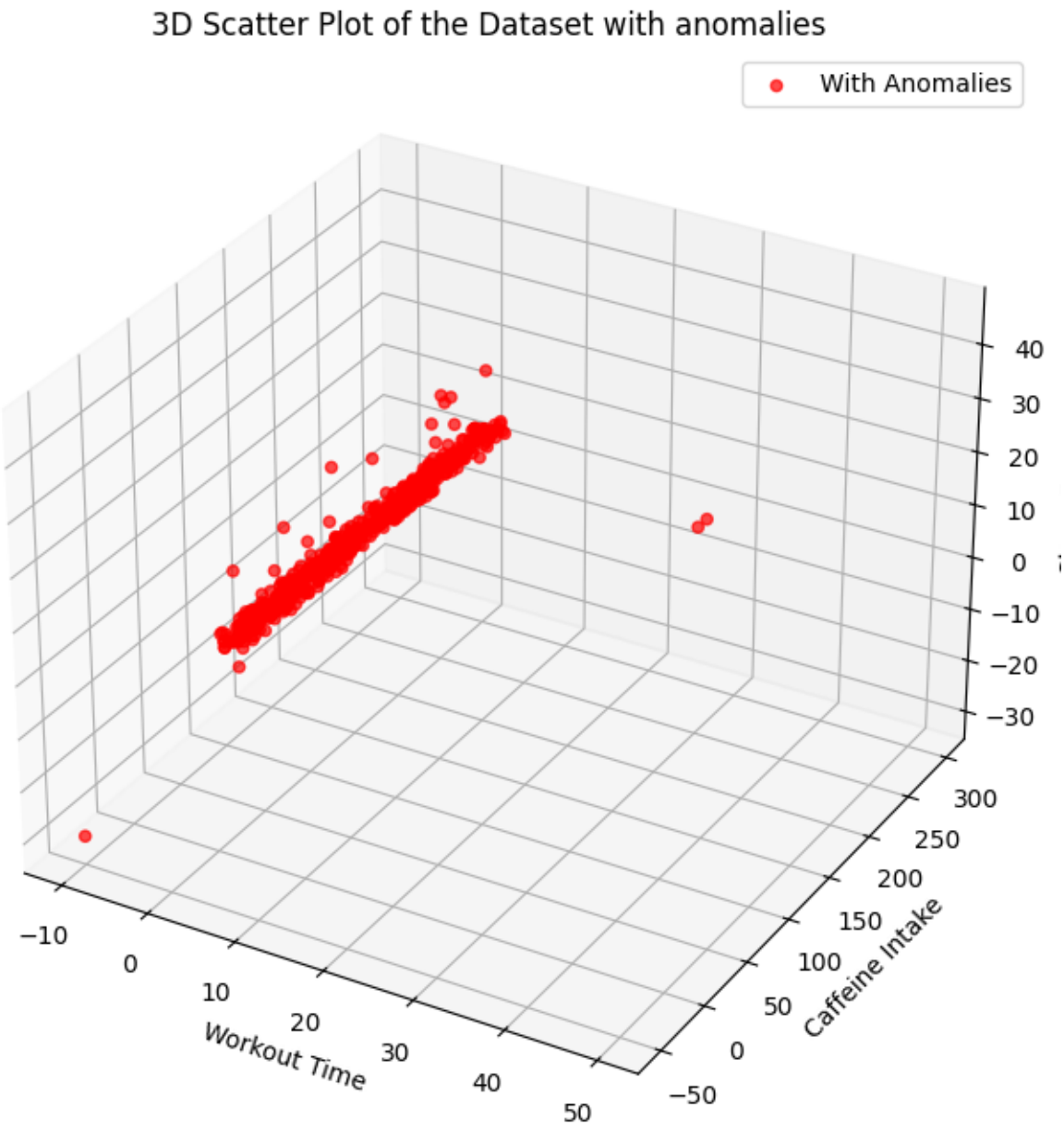
```
import pandas as pd
import numpy as np
df = pd.read_csv('sleeptime_prediction_dataset.csv')
dataset = df[["WorkoutTime", "ReadingTime", "CaffeineIntake",
"SleepTime"]][:500]
print(dataset.shape)
means = np.mean(dataset, axis=0)
std_dev = np.std(dataset, axis=0)
print("###### Mean of each feature is ######\n ", means)
print("###### Standard Deviation of each feature is ######\n ", std_dev)
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(dataset["WorkoutTime"], dataset["CaffeineIntake"],
dataset["SleepTime"], alpha=0.7, label='Normal Data', color='blue')

ax.set_xlabel('Workout Time')
ax.set_ylabel('Caffeine Intake')
ax.set_zlabel('Sleep Time')
ax.set_title('3D Scatter Plot of the Original Dataset')
ax.legend()
plt.show()

dataset = dataset.drop(columns=['ReadingTime'])


##adding anomalies

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
anomalies = np.array([[50, 40, 45], [50, 30, 45], [-10, -50, -30]])
dataset_with_anomalies = np.vstack([dataset, anomalies])
ax.scatter(dataset_with_anomalies[:, 0], dataset_with_anomalies[:, 1],
dataset_with_anomalies[:, 2], alpha=0.7, label='With Anomalies', color='red')
ax.set_xlabel('Workout Time')
ax.set_ylabel('Caffeine Intake')
ax.set_zlabel('Sleep Time')
ax.set_title('3D Scatter Plot of the Original Dataset')
ax.legend()
plt.show()
```

TASK 2 CODE ENDS HERE

TASK 2 SCREENSHOT STARTS HERE

```
###### Mean of each feature is ######
  WorkoutTime         1.49572
ReadingTime           0.99952
CaffeineIntake      145.89392
SleepTime             4.91100
dtype: float64
###### Standard Deviation of each feature is ######
  WorkoutTime         0.895363
ReadingTime           0.571144
CaffeineIntake       86.112376
SleepTime             2.069704
```





TASK 2 SCREENSHOT ENDS HERE

Lab Task 3 – Anomaly Detection _____

Write a program in python from scratch to detect anomalies from a dataset. Use the Gaussian distribution function to calculate the probability of each example. Using the calculated probabilities, make a scatter plot in which the outliers are highlighted.

TASK 3 CODE STARTS HERE

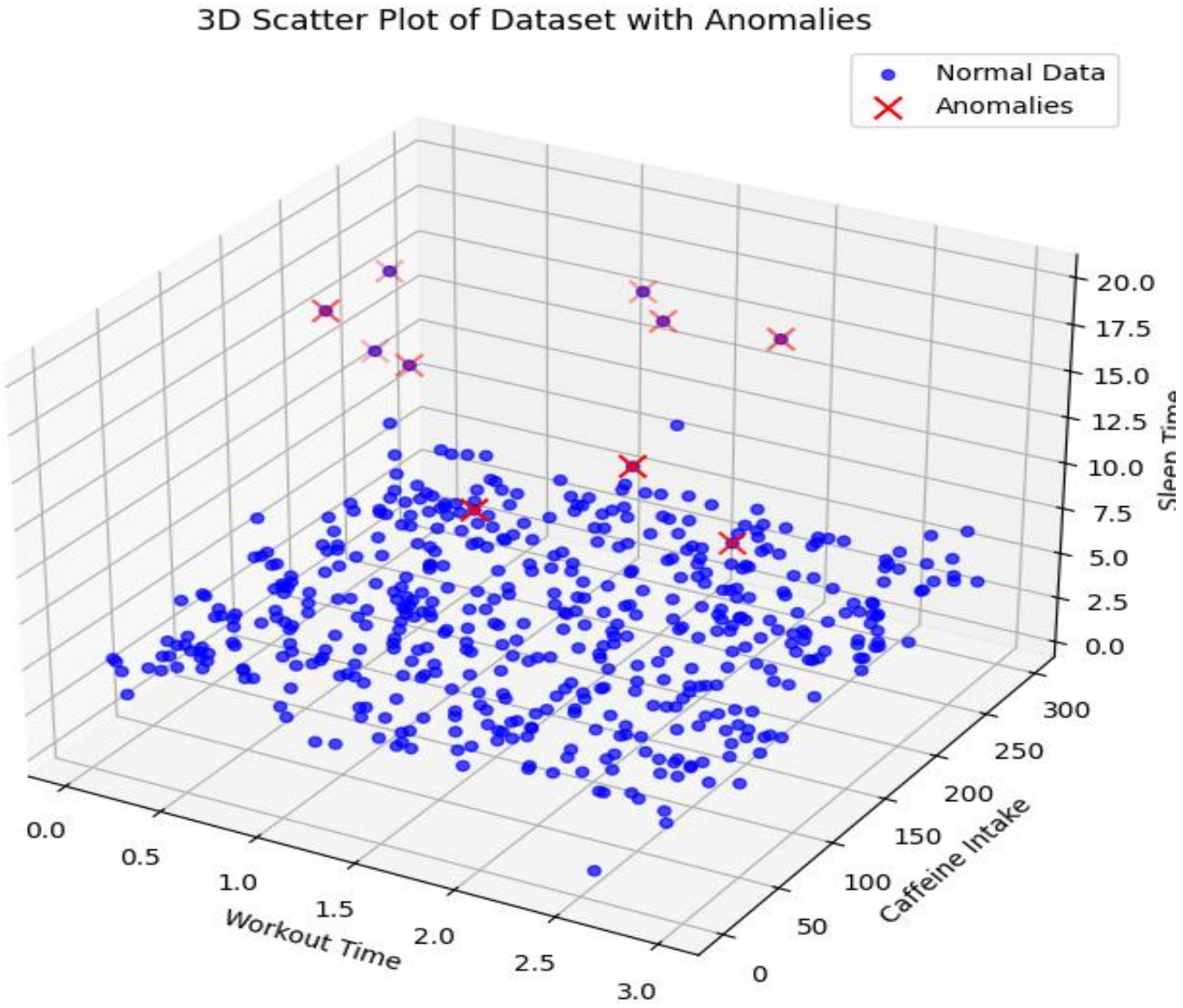
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

# Function to calculate the probability of each data point
def calculate_probability(data, means, std_devs):
    prob = []
    for i in range(len(data)):
        probability = 1
        for j in range(data.shape[1]):  # For each feature (column)
            # Using the Gaussian distribution formula to calculate probability
            prob_j = norm.pdf(data[i, j], means[j], std_devs[j])
            probability *= prob_j
        prob.append(probability)
    return np.array(prob)

# Calculate probabilities for each data point
probabilities = calculate_probability(dataset.values, means, std_devs)

# Calculate a threshold for anomaly detection
threshold = np.percentile(probabilities, 2)  # Consider 2% of the lowest
probabilities as anomalies
print(f"Anomaly detection threshold: {threshold}")
```

```
# Identify anomalies (where probability is below the threshold)
anomalies_idx = np.where(probabilities < threshold)

# Plot the original dataset
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(dataset["WorkoutTime"], dataset["CaffeineIntake"],
dataset["SleepTime"], alpha=0.7, label='Normal Data', color='blue')

# Highlight anomalies in red
anomalies = dataset.iloc[anomalies_idx]
ax.scatter(anomalies["WorkoutTime"], anomalies["CaffeineIntake"],
anomalies["SleepTime"], color='red', label='Anomalies', marker='x', s=100)

ax.set_xlabel('Workout Time')
ax.set_ylabel('Caffeine Intake')
ax.set_zlabel('Sleep Time')
ax.set_title('3D Scatter Plot of Dataset with Anomalies')
ax.legend()
plt.show()
```

TASK 3 CODE ENDS HERE

TASK 3 SCREENSHOT STARTS HERE

SCREENSHOT ON NEXT PAGE.



TASK 3 SCREENSHOT ENDS HERE