Name: **Swarup Tripathy** Assignment Number: **4**
Course: **ECE3502: IoT Domain Analyst** Date: March 24, 2022

**Reg No: 19BEE0167**

# 1 Problem

**A. Write a python script to generate data set like salary data. Perform anomaly detection using k-means algorithm. Use necessary plots to illustrate the results.**

**B. Summarize COAP and MQTT protocols and list its practical applications**

# 2 Python Code

1. Installing faker on google colab with the following code

```
pip install Faker
```

2. Importing the necessary libraries

```
##############################################################################
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot') # ggplot - a data visualisation package
from faker import Faker # Faker - library which generates fake data for us
##############################################################################
```

3. Now we need to generate a random sample data for salary with the corresponding fake names assigned to it with the help of Faker.

```
##############################################################################
fake = Faker()
# To make the results are reproducible

Faker.seed(4321)
name_list = []
fake = Faker()

for _ in range(100):
  name_list.append(fake.name())
  # len(name_list)
# print(name_list)
np.random.seed(7)

salary_list = []
```

```
for _ in range(100):
  sal = np.random.randint(1000,3000)
  salary_list.append(sal)
  # len(salary)

# CREATE A DATAFRAME
salary_df = pd.DataFrame(
    {'person':name_list,
     'salary':salary_list})

# DISPLAY THE DATAFRAME
print(salary_df.head())
################################################################################
```

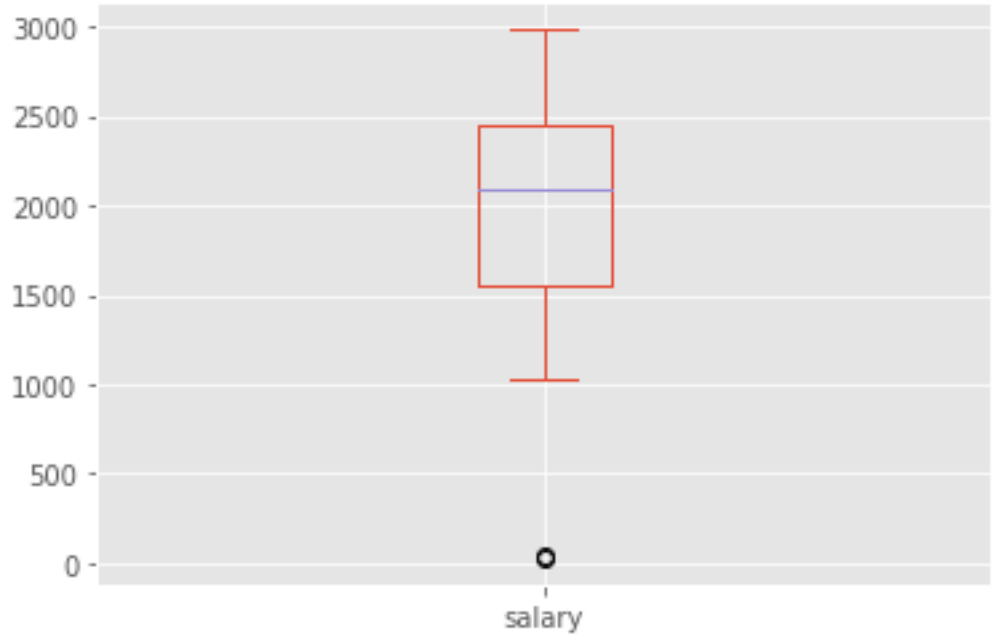
### Code output

```
              person  salary
0        Jason Brown    1175
1        Jacob Stein    2220
2         Cody Brown    1537
3      Larry Morales    1502
4  Jessica Hendricks    2603
```

## 2.1 Code continued...

1. Creation of Outlier

```
################################################################################
# Next is creating the outlier of our own
# Purposefully generating two anomalies
salary_df.at[16,'salary'] = 30
salary_df.at[70,'salary'] = 50
print(salary_df['salary'].plot(kind='box'))

# the black circle represents the anomaly
################################################################################
Output is
```




```
# THE OUTLIER IS ALSO PROVEN USING THE HISTOGRAM

ax = salary_df['salary'].plot(kind='hist')
ax.set_xlabel('Salary')
ax
```


```
print('min salary '+str(salary_df['salary'].min()))
print('max salary '+str(salary_df['salary'].max()))
################################################################################
```

```
OUTPUT IS
min salary 30
max salary 2984
```

2. Raw Salary Data

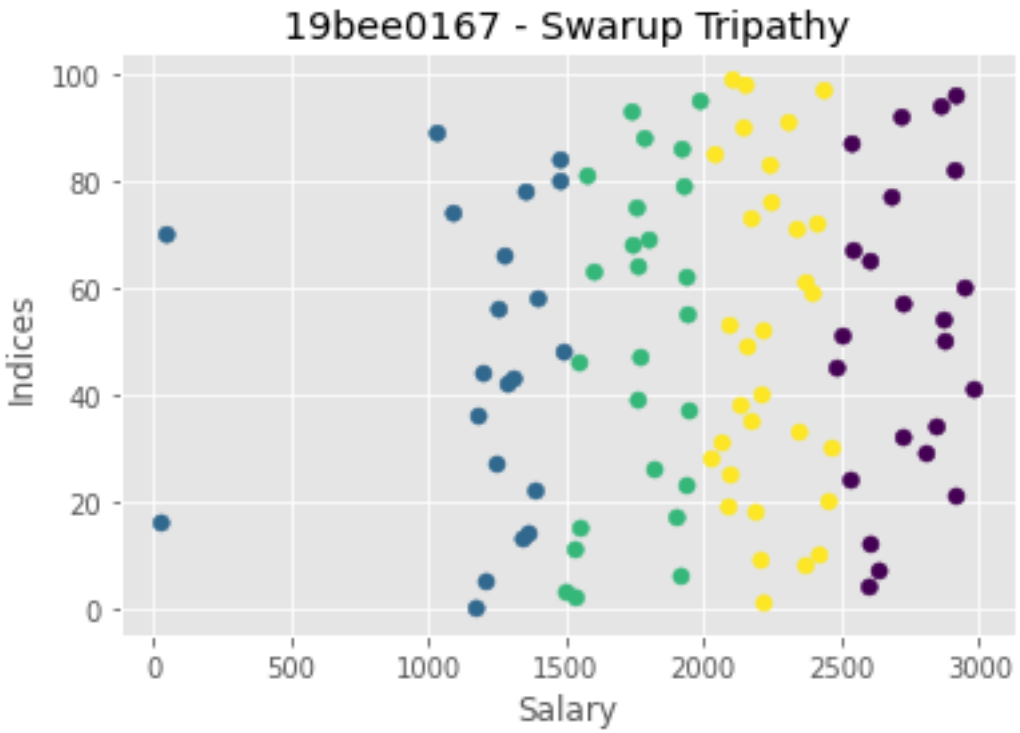
```
# CREATING A RAW SALARY DATA

salary_raw = salary_df['salary'].values
# print(salary_raw)
salary_raw = salary_raw.reshape(-1,1)
# print(salary_raw)
salary_raw = salary_raw.astype('float64')
```

3. K-Means Algorithm

```
# TO DO OUTLIER REDUCTION, K-MEANS IS CALLED

from scipy.cluster.vq import vq, kmeans # vq - vection quantisation

codebook, distortion = kmeans(salary_raw,4)
print('codebook=',codebook,',distortion=',distortion)

###############################################################################

OUTPUT is
codebook= [[2723.625     ]
 [1204.95652174]
 [1792.17391304]
 [2230.06666667]] ,distortion= 145.0509065934066
```

4. Final Outlier representation

```
groups,cdist = vq(salary_raw,codebook)
plt.scatter(salary_raw, np.arange(100), c=groups)
#Return evenly spaced values within a given interval
plt.xlabel('Salary')
plt.ylabel('Indices')
plt.title('19bee0167 - Swarup Tripathy')
plt.show()
```




---

# Problem

**B. Summarize COAP and MQTT protocols and list its practical applications**

1. **COAP**

- CoAP is for "use with constrained nodes and constrained (e.g., low-power, lossy) networks.
- CoAP is a client/server protocol and provides a one-to-one "request/report" interaction model with accommodations for multi-cast,
- CoAP is designed to interoperate with HTTP and the RESTful Web through simple proxies, making it natively compatible to the Internet.
- CoAP runs over UDP, which is inherently and intentionally less reliable than TCP, depending on repetitive messaging for reliability instead of consistent connections.
- CoAP uses DTLS on top of its UDP transport protocol. Like TCP, UDP is unencrypted, but can be—and should be—augmented with DTLS.
- Practical Applications
  - a temperature sensor may send an update every few seconds, even though nothing has changed from one transmission to the next. If a receiving node misses one update, the next will arrive in a few seconds and will likely be not much different than the first.

2. **MQTT**

- MQTT uses a "publish/subscribe" model and requires a central MQTT broker to manage and route messages among an MQTT network's nodes. Eclipse describes MQTT as "a many-to-many communication protocol for passing messages between multiple clients through a central broker." It uses TCP for its transport layer, which is characterized as "reliable, ordered and error-checked."
- MQTT's "pub/sub" model scales well and can be power efficient. Brokers and nodes publish information and others subscribe according to the message content, type, or subject.
- While the node and the broker need to have each other's IP address, nodes can publish information and subscribe to other nodes' published information without any knowledge of each other, since everything goes through the central broker
- MQTT uses unencrypted TCP and is not "out-of-the-box" secure. However, because it uses TCP, it can—and should—use TLS/SSL Internet security.
- Practical Applications
  - In order to solve problems of limited battery life and internet bandwidth associated with smartphone use, Facebook innovatively deployed MQTT for its messenger and Instagram chats that would enable it to function effectively even with the varying internet connections available across the world. Chats are associated with an MQTT topic, and all members of a chat group subscribe and publish to that topic. A "Topic Director" steers the MQTT chat packets to the different brokers that forward them to the appropriate destination-–subscribers.

---