IoT Use Case Design

Date:

Aim

To model a smart home automation using supervised learning method.

Device

Computer system, Python, IoT dataset (CSV), Jupyter Notebook

Dataset Description
Source: UCI Machine Learning Repository – Occupancy Detection Dataset https://archive.ics.uci.edu/dataset/357/occupancy+detection
Features:

- Temperature (°C)
- Humidity (%)
- Light (lux)
- CO_2 (ppm)
- Humidity Ratio

Target (for evaluation only):

- Occupancy (0 = Empty, 1 = Occupied)

Theory

Supervised Machine Learning is a type of machine learning in which a model is trained using **labelled data**. Each training example consists of:

- **Input features (X)** – measurable attributes or sensor readings
- **Output label (Y)** – known correct answer

a) Classification
b) Regression

Algorithms:

- Logistic Regression
- Decision Tree
- Random Forest
- Support Vector Machine
- K-Nearest Neighbors
- Naive Bayes

Automation Rules

Condition	Action
Occupied & Light < 200 lux	Turn ON lights
Occupied & Temp > 28°C	Turn ON fan

Condition	Action
Else	Turn OFF devices

Procedure

1. Understand the problem statement
2. Load and preprocess dataset
3. Apply relevant analytical/modeling steps
4. Analyze results

Example CODE

```
import numpy as np
import time

TEMP_THRESHOLD = 28
LIGHT_THRESHOLD = 200

def read_sensors():
    temp = np.random.normal(26, 3)
    light = np.random.normal(250, 100)
    motion = np.random.choice([0, 1], p=[0.7, 0.3])
    return temp, light, motion

def automation_logic(temp, light, motion):
    lights = "ON" if motion == 1 and light < LIGHT_THRESHOLD else "OFF"
    fan = "ON" if motion == 1 and temp > TEMP_THRESHOLD else "OFF"
    return lights, fan

print("Rule-Based Smart Home Automation\n")

for _ in range(10):
    temp, light, motion = read_sensors()
    lights, fan = automation_logic(temp, light, motion)

    print(f"Temp={temp:.2f} | Light={light:.1f} | Motion={motion}")
    print(f"Action → Lights={lights}, Fan={fan}")
    print("-"*40)

    time.sleep(1)

import pandas as pd

# --------------- Load Dataset ---------------
# Provide correct file paths
train = pd.read_csv("/content/datatraining.txt")
test1 = pd.read_csv("/content/datatest.txt")
test2 = pd.read_csv("/content/datatest2.txt")
```

```
# Combine all data
df = pd.concat([train, test1, test2], ignore_index=True)

# Inspect dataset
print("Columns:", df.columns)
print("Sample data:\n", df.head())

# --------------- Define Automation Rules ---------------
TEMP_THRESHOLD = 28        # Fan ON if temp > 28°C
LIGHT_THRESHOLD = 200      # Light ON if light < 200 (dark)

def automation_logic(temp, light, occupancy):
    """
    Even though the dataset has 'Occupancy' label,
    this rule uses sensor conditions directly.
    """
    # Turn lights ON if light is low AND room occupied
    lights = "ON" if occupancy == 1 and light < LIGHT_THRESHOLD else
"OFF"

    # Turn fan ON if temperature high AND room occupied
    fan = "ON" if occupancy == 1 and temp > TEMP_THRESHOLD else "OFF"

    return lights, fan

# --------------- Apply Logic on Data ---------------
print("\nSmart Home Automation Decisions:\n")

for idx, row in df.sample(10).iterrows():   # sample 10 rows
    temp = row["Temperature"]
    light = row["Light"]
    occ = row["Occupancy"]

    lights, fan = automation_logic(temp, light, occ)

    print(f"Temp={temp:.2f}°C | Light={light:.1f} Lux |
Occupied={occ}")
    print(f"Decision → Lights: {lights} | Fan: {fan}")
    print("-"*50)
```

Expected Output

Successful execution with meaningful insights and metrics.

Result & Conclusion

The experiment validates the use of AIoT techniques for real-world problems.

Learning Outcomes

1. Understanding of Sensor data.
2. Able to develop automation logic for IoT based cases.
3. Analysis of publicly available dataset and based on those experimental results were studied for further AI based modelling.