**SMART PUBLIC RESTROOM**

**TEAM MEMBER**

**622621121026: KIRUBA.P**

**Phase 3 Submission Document**

**Project Title: Smart Public Restroom**

**Phase 3:Development Part 1**

**Topic: *Start building the smart public restroom model by loading and pre-processing the dataset.***



**Smart Public Restroom**

**Introduction:**

* All the public restrooms should be clean and hygiene. In our country, our government has introduced the scheme called “SWATCHH BHARAT” (Clean India). Keeping the public restrooms uncontaminated is the one of the objective of Clean India scheme. This paper can be helpful to encourage the clean India project. In an Existing system, they are focused only on identifying the dirt in the public restrooms.
* In our proposed system, we have determined on keeping clean public restroom, observing the sweeper’s working activities. Building a smart public restroom model is a data-driven process that involves harnessing the power of Internet of Things to analyse occupancy, temperature and hygiene. This journey begins with the fundamental steps of data-loading and pre-processing.
* This dataset includes data from a variety of sensors in a smart public restroom, such as occupancy sensors, door sensors, temperature sensors, humidity sensors, soap dispenser sensors, toilet paper dispenser sensors, and flush sensors. The data is collected at a regular interval, such as every second, and can be used to monitor the status of the restroom and to identify potential problems, such as low supplies. Data processing is the crucial as it helps clean, format and prepare the data for further analysis. We’ll explore how to import essential libraries, load the dataset, and perform critical pre-processing steps. Data preprocessing is crucial as it helps clean, format, and prepare the data for further analysis.

**Given Dataset:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Date and time | Toilet ID | Occupancy | Temperature | Humidity | Air quality | Water flow | Paper level | Soap level | Trash level | Other sensors |
| Date and time | 1 | Occupied | 25°C | 50% | Good | 1 L/min | 75% | 50% | 75% | None |
| Date and time | 2 | Unoccupied | 26°C | 55% | Good | 0 L/min | 100% | 100% | 50% | Motion sensor |
| Date and time | 3 | Occupied | 27°C | 60% | Fair | 2 L/min | 50% | 25% | 100% | None |
| Date and time | 4 | Unoccupied | 28°C | 65% | Poor | 0 L/min | 25% | 0% | 100% | CO2 sensor |

**Necessary step to follow:**

1. **Import Libraries:**

Start by importing the necessary libraries:

**Program:**

import numpy as np

import pandas as pd

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import classification\_report

1. **Load the Dataset:**

Once you have imported the necessary libraries, you can load the dataset that you created in the previous step.

**Program:**

# Load the dataset

data = pd.read\_csv('smart\_public\_restroom\_dataset.csv')

1. **Exploratory Data Analysis(EDA):**

Perform EDA to understand your data better. This includes exploring and analysing data to discover useful insights and patterns.

**Program:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Load the data

data = pd.read\_csv('smart\_public\_restroom\_dataset.csv')

# Clean the data

# TODO: Remove outliers, correct errors, and convert the data into a consistent format.

# Explore the data

# Calculate summary statistics

data.describe()

# Create histograms and scatter plots

plt.hist(data['Occupancy sensor'])

plt.xlabel('Occupancy sensor value')

plt.ylabel('Frequency')

plt.title('Occupancy sensor distribution')

plt.show()

plt.scatter(data['Carbon dioxide level'], data['Temperature'])

plt.xlabel('Carbon dioxide level (ppm)')

plt.ylabel('Temperature (degrees Fahrenheit)')

plt.title('Carbon dioxide level vs. temperature')

plt.show()

# Perform correlation analysis

data.corr()

# Identify patterns and trends in the data

# TODO: Look for unusual values, outliers, and correlations between different variables.

# Draw conclusions from the data

# TODO: Identify potential problems or areas for improvement, or generate hypotheses for further testing.

1. **Feature Engineering:**

Depending on your dataset, you may need to create new features or transform existing ones. This can involve handling date/time data, or scaling numerical features.

**Program:**

import pandas as pd

import numpy as np

from datetime import datetime

# Load the data

df = pd.read\_csv('smart\_public\_restroom\_data.csv')

# Clean the data

# ... (e.g., remove outliers, fill in missing values)

# Create new features

# Feature 1: Time of day

df['time\_of\_day'] = df['timestamp'].apply(lambda x: datetime.fromtimestamp(x).time())

# Feature 2: Day of the week

df['day\_of\_week'] = df['timestamp'].apply(lambda x: datetime.fromtimestamp(x).weekday())

# Feature 3: Stall occupancy

df['stall\_occupancy'] = df['stall\_sensors'].sum(axis=1)

# Feature 4: Wait time

df['wait\_time'] = df['stall\_occupancy'] \* df['average\_stall\_occupancy\_time']

# Feature 5: User satisfaction

df['user\_satisfaction'] = df['user\_feedback'].apply(lambda x: 1 if x == 'satisfied' else 0)

# Save the engineered features

df.to\_csv('smart\_public\_restroom\_engineered\_features.csv', index=False)

1. **Split the Data:**

Next, you need to split the dataset into training and test sets. This helps you to evaluate your model’s performance later.

**Program:**

# Split the dataset into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, data['Occupancy sensor'], test\_size=0.25, random\_state=42)

1. **Feature Scaling**:

Apply feature scaling to normalize your data, ensuring that all features have similar scales. Standardization is a common choice.

Now, you can train a logistic regression model to predict whether a stall is occupied or not.

**Program**:

# Train a logistic regression model

model = LogisticRegression()

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

Once the model is trained, you can evaluate its performance on the test set.

**Program:**

# Evaluate the model on the test set

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

print(classification\_report(y\_test, y\_pred))

This will give you a report of the model's accuracy, precision, recall, and F1 score.

Once you are satisfied with the model's performance, you can deploy it to production. This could involve integrating it with a smart restroom system or creating a mobile app that users can use to check the status of the restrooms in their area.

Importance of loading and processing dataset:

Loading and processing datasets is an essential part of developing and deploying smart public restrooms using IOT. The data collected from these sensors can be used to improve the efficiency, cleanliness, and sustainability of public restrooms.

By loading and preprocessing the dataset, we can ensure that the machine learning algorithm is able to learn from data effectively and accurately.

Here are some of the specific benefits of loading and processing datasets for smart public restrooms:

* Improved efficiency: By tracking restroom usage patterns, facility managers can identify peak hours and areas that need to be cleaned and restocked more frequently. This can lead to reduced wait times for users and improved cleanliness overall.
* Reduced costs: By optimizing cleaning and maintenance schedules, facility managers can save money on labor and supplies. Additionally, smart restrooms can help to reduce water and energy consumption.
* Enhanced sustainability: Smart restrooms can help to reduce waste and promote sustainability by tracking things like toilet paper usage and water consumption. This data can then be used to identify areas where improvements can be made.

**Challenges involved in loading and preprocessing a smart public restroom dataset**:

Loading and preprocessing a smart public restroom dataset can be challenging for a number of reasons. Here are some of the most common challenges:

* ***Data quality***: Smart public restroom sensors can collect a lot of data, but it's important to make sure that the data is high quality. This means cleaning the data for errors and inconsistencies.
* ***Data volume***: Smart public restroom datasets can be very large, especially if they are collected from multiple sensors over a long period of time. This can make it challenging to load and process the data using traditional methods.
* ***Data variety***: Smart public restroom datasets can contain a variety of data types, including sensor data, occupancy data, and usage data. This can make it challenging to develop a single preprocessing pipeline that can handle all of the different data types.
* ***Data security***: Smart public restroom datasets can contain sensitive data about restroom users, such as occupancy data and usage data. It's important to take steps to protect the privacy of this data when loading and processing it.

Here are some specific examples of challenges that may be encountered when loading and preprocessing a smart public restroom dataset:

* ***Duplicate data***: Data may be duplicated if it is collected from multiple sensors or if it is stored in multiple databases.
* ***Missing values***: Sensor data may be missing if a sensor is malfunctioning or if there is a network connection issue.
* ***Outliers***: Sensor data may contain outliers, which are values that are significantly different from the other data points in the dataset.
* ***Incorrect data types***: Sensor data may be stored in the wrong data type, such as a numeric value being stored as a string.
* ***Inconsistent formatting***: Sensor data may be formatted differently in different databases or files.
* To address these challenges, it is important to develop a robust data loading and preprocessing pipeline. This pipeline should be able to clean the data for errors and inconsistencies, missing values and outliers, convert the data to the correct data types, and format the data consistently.
* In addition, it is important to implement security measures to protect the privacy of the data. This may include encrypting the data, restricting access to the data, and auditing access to the data.

**How to overcome the challenges of loading and pre-processing a smart public restroom dataset:**

Thereare a number of things that can be done to overcome the challenges of loading and pre-processing a smart restroom dataset, including:

**Data Quality:**

* *Identify and remove outliers*: Outliers are data points that are significantly different from the rest of the data. They can be caused by sensor malfunctions, user errors, or other factors. Outliers can be identified using statistical methods, such as interquartile range (IQR) and standard deviation. Once outliers have been identified, they can be removed from the dataset or corrected if possible.
* *Handle missing values:* Missing values are data points that are missing altogether. They can be caused by sensor failures, communication problems, or other factors. Missing values can be handled in a variety of ways, such as imputing them with the mean, median, or mode of the data. However, it is important to choose a method that is appropriate for the specific dataset and analysis.

**Data volume:**

* *Sample the data:* If the dataset is too large to process in its entirety, you can sample the data to create a smaller dataset that is more manageable. However, it is important to make sure that the sample is representative of the entire dataset.
* *Use a distributed computing framework:* A distributed computing framework, such as Spark, can be used to process large datasets on multiple servers. This can significantly improve the performance of the data processing pipeline.

**Data integration:**

* *Create a data schema:* A data schema is a set of rules that define the structure and format of a dataset. Creating a data schema can help to ensure that the data is integrated in a consistent way.
* *Use a data warehouse:* A data warehouse is a specialized database that is designed to store and analyze large volumes of data from multiple sources. Data warehouses can be used to integrate data from multiple sources and to create complex reports and dashboards.

Data security:

* *Encrypt the data:* Encrypting the data can help to protect it from unauthorized access.
* *Use access control lists:* Access control lists (ACLs) can be used to restrict access to the data to authorized users.
* *Use a Cloud-based data storage and processing platform*: Cloud-based data storage and processing platforms can provide a high level of security for your data.
  1. **Loading the Dataset:**
* *Identify the data sources:* Your dataset may come from a variety of sources, such as occupancy sensors, air quality sensors, water flow sensors, and user surveys.
* *Clean the data:* Once you have identified the data sources, you need to clean the data to remove any errors or inconsistencies. This can be done using a variety of tools and techniques, such as data cleansing software and machine learning algorithms.
* *Transform the data:* Once the data is clean, you may need to transform it into a format that is compatible with your data warehouse or analytics platform. This may involve converting data types, splitting fields, or merging tables.
* *Load the dataset:* Once the data is transformed, you can load it into your data warehouse or analytics platform. This can be done using a variety of methods, such as bulk loading or streaming.
* *Preprocess the Dataset:* Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training and test sets.

**Program:**

import json

import paho.mqtt.client as mqtt

# Define the MQTT client

client = mqtt.Client()

# Connect to the MQTT broker

client.connect("localhost", 1883)

# Subscribe to the topic where the dataset is published

client.subscribe("smart\_public\_restroom\_dataset")

# Create a list to store the dataset

dataset = []

# Define a callback function to receive the dataset

def on\_message(client, userdata, msg):

global dataset

# Parse the JSON dataset

data = json.loads(msg.payload)

# Add the dataset to the list

dataset.append(data)

# Set the callback function

client.on\_message = on\_message

# Start the MQTT client loop

client.loop\_forever()

# Once the dataset has been loaded, it can be used to train a machine learning model or for other purposes

Here is an example of a dataset that could be published to the MQTT broker:

JSON

[

{

"timestamp": "2023-10-17T11:33:45Z",

"temperature": 25.5,

"humidity": 60,

"occupancy": 1

},

{

"timestamp": "2023-10-17T11:33:46Z",

"temperature": 25.6,

"humidity": 61,

"occupancy": 0

},

...

]

**Loading the Dataset:**

data = pd.read \_csv('smart\_public\_restroom\_dataset.csv')

**Data Exploration:**

***Dataset Output:***

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Date and time | Toilet ID | Occupancy | Temperature | Humidity | Air quality | Water flow | Paper level | Soap level | Trash level | Other sensors |
| Date and time | 1 | Occupied | 25°C | 50% | Good | 1 L/min | 75% | 50% | 75% | None |
| Date and time | 2 | Unoccupied | 26°C | 55% | Good | 0 L/min | 100% | 100% | 50% | Motion sensor |
| Date and time | 3 | Occupied | 27°C | 60% | Fair | 2 L/min | 50% | 25% | 100% | None |
| Date and time | 4 | Unoccupied | 28°C | 65% | Poor | 0 L/min | 25% | 0% | 100% | CO2 sensor |

* 1. **Pre-processing the dataset:**

Data pre-processing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.

This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.

*Preprocessing the dataset for a smart public restroom using IOT involves the following steps:*

1. *Collect the data:* The data can be collected from various IOT sensors, such as temperature sensors, humidity sensors, occupancy sensors, water level sensors, soap level sensors, and toilet paper level sensors.
2. *Clean the data:* The data may contain errors or outliers. It is important to clean the data before preprocessing it. This can be done by removing any errors or outliers, and by normalizing the data.
3. *Feature engineering:* Feature engineering is the process of creating new features from the existing data. This can be done by combining or transforming the existing features.
4. *Split the data into training and test sets:* The data should be split into two sets: a training set and a test set. The training set is used to train the machine learning model, and the test set is used to evaluate the performance of the model.

**Here is a more detailed explanation of each step:**

* *Collect the data:* The data can be collected from various IOT sensors, such as temperature sensors, humidity sensors, occupancy sensors, water level sensors, soap level sensors, and toilet paper level sensors. The sensors can be placed inside the restroom or outside the restroom.
* *Clean the data:* The data may contain errors or outliers. It is important to clean the data before preprocessing it. This can be done by removing any errors or outliers, and by normalizing the data.
* *Removing errors:* Errors can be removed by manually reviewing the data or by using a statistical method to identify errors.
* *Removing outliers:* Outliers are data points that are significantly different from the rest of the data. Outliers can be removed by using a statistical method to identify outliers.
* *Normalizing the data:* Normalizing the data involves scaling the data to a common range. This can be done by dividing each data point by the standard deviation of the data.

**Feature engineering:**

Feature engineering is the process of creating new features from the existing data. This can be done by combining or transforming the existing features.

* *Combining features:* Combining features involves creating a new feature by combining two or more existing features. For example, a new feature could be created by combining the temperature and humidity data to create a new feature called "comfort index".
* *Transforming features:* Transforming features involves creating a new feature by transforming an existing feature. For example, a new feature could be created by transforming the occupancy data to create a new feature called "occupancy rate".

***Split the data into training and test sets:***

* The data should be split into two sets: a training set and a test set. The training set is used to train the machine learning model, and the test set is used to evaluate the performance of the model.
* The training set should be at least 70% of the data. The test set should be at least 30% of the data.
* Once the data is preprocessed, it can be used to train a machine learning model to predict the status of the restroom. The model can then be used to develop a smart restroom app that can provide users with real-time information about the restrooms in their area.

*Here are some common data preprocessing tasks for smart public restroom using IOT:*

* Cleaning the data: This involves removing any errors or inconsistencies in the data. For example, you may need to remove rows with missing values or convert values to a consistent format.
* Handling missing values: This involves removing rows with missing values or filling in the missing values with a suitable value. For example, you could fill in missing temperature values with the average temperature of the restroom over the past day.
* Handling outliers: This involves identifying and removing outliers from the data. Outliers are values that are significantly different from the rest of the data. For example, you could remove an outlier temperature value that is caused by a faulty sensor.
* Scaling the data: This involves transforming the data so that all of the features are on the same scale. This is necessary for some machine learning algorithms to work properly. For example, you could scale the temperature and humidity features to a range of 0 to 1.
* Encoding categorical data: This involves converting categorical features, such as the time of day or the day of the week, into numerical features. This is necessary for most machine learning algorithms to work properly. For example, you could encode the time of day feature using a one-hot encoding scheme.

Once the data has been pre-processed, it can be used to train a machine learning model or to perform other data analysis tasks.

Here are some additional tips for pre-processing data for smart public restroom using IOT:

* Document all of the pre-processing steps that you perform. This will help you to reproduce the results of your analysis in the future.
* Avoid over fitting the data: Over fitting occurs when the machine learning model learns the training data too well and is unable to generalize to new data. To avoid over fitting, you can use a variety of techniques, such as cross-validation and regularization.
* Use a variety of pre-processing techniques to improve the performance of your machine learning model. There is no one-size-fits-all approach to data pre-processing. The best approach will vary depending on the specific dataset and the machine learning algorithm that you are using.

By following these tips, you can pre-process your data effectively to improve the performance of your machine learning model and gain valuable insights from your smart public restroom IOT system.

**Output Program:**

import numpy as np

import pandas as pd

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import classification\_report

# Load the dataset

data = pd.read\_csv('smart\_public\_restroom\_dataset.csv')

# Split the dataset into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, data['Occupancy sensor'], test\_size=0.25, random\_state=42)

# Train a logistic regression model

model = LogisticRegression()

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

# Evaluate the model on the test set

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

print(classification\_report(y\_test, y\_pred))

# Deploy the model to production

# For example, you could integrate it with a smart restroom system to automatically flush the toilet and turn

Here are some additional tips for loading and preprocessing the dataset for smart public restroom using IOT:

* Make sure that the dataset is secure and that only authorized users have access to it.
* Avoid collecting or storing any sensitive information, such as personally identifiable information (PII).
* If you are collecting data from users, make sure to obtain their consent first.
* Be transparent about how you are collecting and using the data.
* Regularly review and update your data collection and processing policies.

By following these tips, you can load and preprocess your data effectively to gain valuable insights from your smart public restroom IOT system.

Preprocessing the dataset for a smart public restroom is an essential step in preparing the data for machine learning or other data analysis tasks. By cleaning, handling missing values, handling outliers, scaling the data, and encoding categorical data, you can improve the quality of the data and make it more informative.

**Here are some of the benefits of preprocessing the dataset for a smart public restroom:**

* **Improved performance of machine learning models:**Preprocessed data is more likely to produce accurate and reliable results from machine learning models.
* **Reduced over fitting:** Preprocessing can help to reduce over fitting, which is a problem that occurs when machine learning models learn the training data too well and are unable to generalize to new data.
* **Increased interpretability of data:** Preprocessing can make the data more interpretable, which can help you to identify trends and patterns in the data more easily.

***Preprocessing Complete***

**Conclusion:**

* In the quest to build a smart public restroom model, we have embarked on a critical journey that begins with loading and preprocessing the dataset. We have traversed through essential steps, starting with importing necessary libraries to facilitate data manipulation and analysis.
* Understanding the data’s structure, characteristics, and any potential issues through exploratory data analysis (EDA)is essential for informed decision – making.
* Data processing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.
* Overall, preprocessing the dataset for a smart public restroom is an important step in gaining valuable insights from your IOT system. By following the tips provided in this article, you can preprocess your data effectively and improve the quality of your analysis.