**PHASE- 3 ASSIGNMENT**

**PROJECT TITLE: MEASURED ENERGY CONSUMPTION**

**PROBLEM DEFINITION**

                      The problem of measuring energy consumption involves quantifying and monitoring the amount of energy consumed by various devices, systems, or entities in a given context. This task is essential for various reasons, including energy management, cost control, sustainability, and environmental impact assessment. Here is a more detailed problem definition for measuring energy consumption.

**GITHUB LINK**

<https://github.com/skyash2003/AI_phase_measure_energy_consumption.git>

**DATASET LINK**

<https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption>

**INTRODUCTION**

Energy consumption is a critical aspect of modern societies, influencing economic development, environmental sustainability, and overall quality of life. As the global population continues to grow, and technological advancements become more prevalent, the demand for energy has risen significantly. Consequently, understanding and measuring energy consumption has become imperative for governments, businesses, and individuals alike.

**Load the Dataset**

* Use a library like pandas to load your dataset into a DataFrame.

import pandas as pd   
   
# Load dataset   
df = pd.read\_csv('your\_dataset.csv')

**Explore and Understand the Data**

* Display basic information about the dataset, such as the number of rows, columns, and a few sample records.

# Display basic information about the dataset   
print(df.info())   
   
# Display the first few rows of the dataset   
print(df.head()) 

**Handle Missing Values**

* Check for missing values and decide how to handle them. You might choose to drop rows with missing values, fill them with a placeholder, or use more advanced imputation methods.

# Check for missing values   
print(df.isnull().sum())   
   
# Handle missing values (example: drop rows with missing values)   
df = df.dropna() 

**Text Cleaning**

* Perform text cleaning to remove unnecessary characters, HTML tags, punctuation, and special characters.

import re   
   
# Function for text cleaning   
def clean\_text(text):   
    # Example: remove non-alphanumeric characters   
    text = re.sub(r'[^a-zA-Z0-9]', ' ', text)   
    return text   
   
# Apply text cleaning to the 'text' column   
df['text'] = df['text'].apply(clean\_text) 

**Tokenization**

* Tokenize the text into individual words or tokens.

from nltk.tokenize import word\_tokenize   
   
# Tokenize the 'text' column   
df['tokens'] = df['text'].apply(word\_tokenize) 

**Remove Stopwords**

* Remove common stopwords (e.g., 'and', 'the') that do not contribute much to the meaning of the text.

from nltk.corpus import stopwords   
   
stop\_words = set(stopwords.words('english'))   
   
# Remove stopwords   
df['tokens'] = df['tokens'].apply(lambda x: [word for word in x if word.lower() not in stop\_words]) 

**Lemmatization or Stemming**

* Reduce words to their base or root form using lemmatization or stemming.

from nltk.stem import WordNetLemmatizer   
   
lemmatizer = WordNetLemmatizer()   
   
# Lemmatize the 'tokens' column

df['tokens'] = df['tokens'].apply(lambda x: [lemmatizer.lemmatize(word) for word in x])

**Join Tokens Back to Text**

* Join the tokenized words back into a cleaned text for further processing.

df['cleaned\_text'] = df['tokens'].apply(lambda x: ' '.join(x))

**Feature Extraction (e.g., TF-IDF)**

* Convert the cleaned text into numerical features using techniques like TF-IDF (Term Frequency-Inverse Document Frequency).

from sklearn.feature\_extraction.text import TfidfVectorizer   
   
vectorizer = TfidfVectorizer(max\_features=5000)   
X = vectorizer.fit\_transform(df['cleaned\_text'])

https://res-1.cdn.office.net/officeonline/we/s/hA3596C17DAD9A003_resources/1033/progress.gif 

**Encode Labels**

* If your dataset has categorical labels, encode them into numerical format using Label Encoding.

from sklearn.preprocessing import LabelEncoder   
   
label\_encoder = LabelEncoder()   
df['label'] = label\_encoder.fit\_transform(df['label'])

**SUMBITTED BY**

711221104062

au711221104062