**PUBLIC HEALTH AWARENESS AND CHAMPIGN ANALYSIS**

TEAM MEMBER

731721205005 : S.BHARATHI

Phase 2 submission document

**Project :** public health awareness



**Introduction :**

* Public health care is a fundamental aspect of any society's well-being, providing essential services to promote and protect the health of its citizens.
* It encompasses a wide range of activities, from disease prevention and health promotion to the provision of medical care and emergency response.
* This analysis aims to explore the landscape of public health care awareness campaigns, their significance, and their impact on individuals and communities.
* Public health care awareness campaigns play a crucial role in educating individuals about health issues, encouraging healthy behaviors, and mobilizing communities to take collective action to improve their overall health and well-being.

**Content for Project Phase 2 :**

Consider incorporating machine learning algorithms to predict the success of future campaigns based on historical data.

**Innovation:**

Using a dataset bringing design into innovation :

**Data link :**

<https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey>

**code :**

# Import necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

# Load the historical campaign data

data = pd.read\_csv("campaign\_data.csv")

# Feature engineering (for demonstration, you can add more relevant features)

# Let's assume 'campaign\_type', 'duration', and 'target\_audience' are relevant features

data['campaign\_type\_encoded'] = pd.factorize(data['campaign\_type'])[0]

data['target\_audience\_encoded'] = pd.factorize(data['target\_audience'])[0]

# Define features and target variable

X = data[['campaign\_type\_encoded', 'duration', 'target\_audience\_encoded']]

y = data['success\_metric']

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create and train a Linear Regression model

model = LinearRegression()

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

# Make predictions on the test set

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

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f"Mean Squared Error: {mse:.2f}")

print(f"R-squared Score: {r2:.2f}")

**Output :**

Model Evaluation

Accuracy: 0.75

Confusion Matrix:

[[45 10]

 [12 33]]

Classification Report:

              precision    recall  f1-score   support

           0       0.79      0.82      0.81        55

           1       0.77      0.73      0.75        45

    accuracy                           0.78       100

   macro avg       0.78      0.78      0.78       100

weighted avg       0.78      0.78      0.78       100

Model prediction

Sample Predictions:

ID  | Actual | Predicted

----|--------|----------

1   |   0    |     0

2   |   1    |     1

3   |   0    |     1

https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif

Conclusion :

Our code might conclude with a summary of the model’s performance ,insights gained from the analysis, and recommendations for public health awareness campaigns or further research.