Naan Mudhalvan

Data Analytics With Cognos

Phase-2

Team: Shivani Suresh, Shyamala R B, Daphny Jessica

Introduction

Public health awareness campaigns play a critical role in educating the public about health issues, changing behaviors, and improving overall community health. It is essential to measure the success of these campaigns and use insights to plan and execute future initiatives more effectively. In this document, we present an innovative solution that leverages machine learning algorithms to predict the success of future campaigns based on historical data.

Problem Statement

- Difficulty in assessing the potential impact of new campaigns.
- Limited understanding of which factors contribute to campaign success.
- Lack of data-driven insights to optimize campaign strategies.

Solution

Our solution involves implementing predictive analytics using machine learning to forecast the success of future public health awareness campaigns. This approach will enable campaign planners to make informed decisions, allocate resources efficiently, and maximize the impact of their initiatives.

Implementation Steps

1. Data Collection and Preparation

Collect historical campaign data, including engagement metrics, audience demographics, and awareness survey results.
Clean and preprocess the data to remove outliers and handle missing values.
Ensure data privacy compliance and obtain necessary consents when collecting personal data.

2. Feature Engineering

Identify relevant features that may influence campaign success (e.g., engagement metrics, demographics, campaign duration, messaging).
Engineer new features if necessary, such as sentiment analysis of user-generated content.

3. Model Selection

Choose appropriate machine learning algorithms for prediction (e.g., regression, classification, time series forecasting). Here we have chosen the Random Forest Algorithm.

4. Training and Validation

Split the historical data into training and validation sets.
Train the selected machine learning model on the training data and validate its performance on the validation set.
Evaluate model accuracy, precision, recall, and other relevant metrics.

5. Predictive Modeling

Once the model is trained and validated, use it to predict the success of future campaigns.
Input relevant campaign features into the model to obtain predictions.
Monitor model performance and update it regularly with new data.

6. Dashboard and Reporting

Create a user-friendly dashboard using IBM Cognos to visualize campaign predictions and historical data.
Generate reports that provide insights into predicted campaign success factors.

Benefits

Data-Driven Decision-Making: Campaign planners can use predictive analytics to make informed decisions about resource allocation and campaign strategies.

Optimized Campaigns: By understanding which factors contribute to success, campaigns can be tailored for maximum impact.

Resource Efficiency: Avoid wasting resources on campaigns that are unlikely to succeed and focus efforts where they are most needed.

Continuous Improvement: Regular model updates with new data ensure that predictions remain accurate and relevant.

Conclusion

Incorporating machine learning algorithms into public health awareness campaigns can revolutionize the way we plan, execute, and measure the success of these initiatives. By leveraging historical data and predictive analytics, we can create more impactful campaigns, allocate resources efficiently, and ultimately improve public health outcomes.

Random Forest Machine Learning Algorithm

```
# Import the necessary libraries
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report

# Load your public health awareness campaign dataset and split it into features (X) and target labels (y)
X = # Your feature data specific to public health campaigns (e.g., demographic data, campaign duration, messaging sentiment, etc.)
y = # Your target labels (e.g., campaign success labels, such as "successful" or "not successful")

# Split the data into a training set and a test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Instantiate the Random Forest classifier
model = RandomForestClassifier(n_estimators=100, max_depth=None, random_state=42)

# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model's performance
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

# Print the results
print(f"Accuracy: {accuracy}")
print("Classification Report:\n", report)
```