UNIVERSITY OF TOMORROW

Elements of AIML

Lab Assignment 1 on 7th November

Name:	Sanchit Panwar

Course: **B.Tech. C.S.E.**

Batch: 11

Sap.id.: 500123669

Enrollment no.: R2142230336

ZERO HUNGER DATASET:

Introduction:-

The Zero Hunger dataset aligns with the United Nations' Sustainable Development Goal (SDG) #2, which is to achieve "Zero Hunger." This goal aims to eradicate hunger, improve food security, enhance nutrition, and promote sustainable agriculture worldwide. The dataset provides vital indicators that relate to food availability, nutrition levels, agricultural productivity, and the prevalence of undernourishment across different regions and time periods.

Purpose:-

The dataset is intended to facilitate data-driven insights into global hunger trends, helping policymakers, researchers, and development organizations understand the current food security landscape and its challenges. By analyzing this dataset, stakeholders can identify factors contributing to hunger, assess the impact of various interventions, and design targeted solutions to reduce hunger and promote sustainable agricultural practices.

Dataset Used:-

The Zero Hunger dataset typically includes multiple indicators across different regions and time frames. Common variables may include:

- **Entity**: Country or region name.
- **Year**: Time reference for the data.
- **Prevalence of undernourishment (% of population)**: The proportion of the population that is undernourished.
- **Combined figures (kg/capita/year)**: A metric possibly indicating food production, consumption, or availability.
- **Region and Source**: Region classifications and data sources for better contextualization.

This dataset may contain other food security and agriculture-related indicators, making it suitable for time-series and cross-sectional analysis.

Libraries Used:-

For analyzing the Zero Hunger dataset in Python, the following libraries are commonly employed:

- **Pandas**: For data loading, cleaning, and manipulation.
- **NumPy**: For numerical operations and handling arrays.
- **Scikit-learn**: For modeling, including classification, regression, and metrics evaluation.
- **Imbalanced-learn (imblearn)**: Specifically, **SMOTE** (Synthetic Minority Oversampling Technique) is used to handle class imbalances.
- **Matplotlib / Seaborn**: For data visualization to illustrate trends and insights clearly.

CODE:-

```python
        results[name] = {
            'Accuracy': accuracy,
            'Precision': precision,
            'Recall': recall,
            'F1 Score': f1,
            'ROC AUC': roc_auc,
            'Confusion Matrix': conf_matrix
        }

    # Display results
    for model_name, metrics in results.items():
        print(f"Model: {model_name}")
        print(f"Accuracy: {metrics['Accuracy']:.4f}")
        print(f"Precision: {metrics['Precision']:.4f}")
        print(f"Recall: {metrics['Recall']:.4f}")
        print(f"F1 Score: {metrics['F1 Score']:.4f}")
        print(f"ROC AUC: {metrics['ROC AUC']:.4f}")
        print("Confusion Matrix:")
        print(metrics['Confusion Matrix'])
        print("\n")
```

[16] Python

```python
# Step 6: K-Fold Cross Validation and Hyperparameter Tuning
kf = KFold(n_splits=5, shuffle=True, random_state=42)
```

[15] Python

```python
# Define parameter grids for hyperparameter tuning
param_grid_rf = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10]
}
param_grid_lr = {'C': [0.01, 0.1, 1, 10, 100]}

# Dictionary of models with GridSearchCV for hyperparameter tuning
models = {
    'Logistic Regression': GridSearchCV(LogisticRegression(), param_grid_lr, cv=kf),
    'Random Forest': GridSearchCV(RandomForestClassifier(), param_grid_rf, cv=kf)
}

# Step 7: Cross-validation and metrics calculation
results = {}
for name, model in models.items():
    y_pred = cross_val_predict(model, X_resampled, y_resampled, cv=kf)
    accuracy = accuracy_score(y_resampled, y_pred)
    precision = precision_score(y_resampled, y_pred)
    recall = recall_score(y_resampled, y_pred)
    f1 = f1_score(y_resampled, y_pred)
    roc_auc = roc_auc_score(y_resampled, y_pred)
    conf_matrix = confusion_matrix(y_resampled, y_pred)
```

```python
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import KFold, cross_val_predict, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import LabelEncoder, StandardScaler
from google.colab import drive

# Mount Google Drive
drive.mount('/content/drive')

# Load the Excel file
file_path = '/content/drive/My Drive/7 nov as_1/Zero_Hunger.xlsx'
df = pd.read_excel(file_path)
```

```
Mounted at /content/drive
```

```python
# Step 1: Fill missing values in numeric columns
numeric_df = df.select_dtypes(include=['number'])
df[numeric_df.columns] = numeric_df.fillna(numeric_df.mean())
```

```python
# Step 2: Encode categorical variables
label_encoder = LabelEncoder()
if 'Entity' in df.columns:  # Check if 'Entity' column exists to avoid KeyError
    df['Entity'] = label_encoder.fit_transform(df['Entity'])
else:
    print("Error: 'Entity' column not found in data.")
```

```python
drop_columns = ['Entity', 'Code']  # Only keep necessary columns to drop
X = df.drop(columns=drop_columns, errors='ignore')
y = df['Prevalence of undernourishment (% of population)']
```

```python
# Step 4: Apply SMOTE to handle class imbalance
smote = SMOTE()
X_resampled, y_resampled = smote.fit_resample(X, y)  # Execute this cell first
```

```python
# Step 5: Feature Scaling
scaler = StandardScaler()
X_resampled = scaler.fit_transform(X_resampled)
```

OUTPUT:-

```
Model: Logistic Regression
Accuracy: 0.9813
Precision: 1.0000
Recall: 0.9626
F1 Score: 0.9810
ROC AUC: 0.9813
Confusion Matrix:
[[2541    0]
 [  95 2446]]


Model: Random Forest
Accuracy: 1.0000
Precision: 1.0000
Recall: 1.0000
F1 Score: 1.0000
ROC AUC: 1.0000
Confusion Matrix:
[[2541    0]
 [   0 2541]]
```

Summary:-

The Zero Hunger dataset offers a comprehensive view of global food security metrics, supporting in-depth analyses and visualizations to understand hunger patterns. By applying machine learning algorithms and statistical analysis to this dataset, researchers can explore the impact of factors influencing hunger and food security and assess the effectiveness of interventions across regions. Ultimately, this dataset serves as a valuable resource for supporting data-driven decisions in pursuit of the goal of zero hunger.