Mini Research Paper on "Predicting Well-being with Machine Learning and Explainable AI"

Author: Pulkit Chauhan
Roll No: R2142230354
SAP ID: 500121424

Abstract

This research paper presents the development of a machine learning model to predict well-being based on socioeconomic factors, with interpretability enhanced through Explainable AI (XAI) techniques. The goal of this project is aligned with the United Nations' Sustainable Development Goal (SDG) for **Good Health and Well-being**. By combining a Random Forest model with SHAP (SHapley Additive exPlanations), this work seeks to make well-being predictions interpretable and actionable, emphasizing model transparency.

Introduction

The need for interpretable machine learning models is increasingly recognized in applications involving social and economic data. Black-box models, while accurate, often lack transparency, limiting their usability in fields where understanding drivers behind predictions is crucial. Explainable AI (XAI) tools, like SHAP, can bridge this gap by providing insights into model predictions. This project employs multiple machine learning algorithms with XAI to interpret well-being predictions based on socio-economic indicators, contributing to SDG 3: Good Health and Well-being.

Data and Methodology

Data Source and Preprocessing

The dataset used for training the model is the **World Happiness Report 2024** dataset from Kaggle. This data includes various factors that influence well-being across different countries, such as GDP per capita, social support, healthy life expectancy, and perceptions of corruption.

To begin, the data was inspected for missing values, which were subsequently removed to ensure consistency. The features selected for training include:

- GDP per capita
- Social support
- Healthy life expectancy
- Freedom to make life choices
- Generosity

- Perceptions of corruption

The target variable, Ladder score, represents the well-being score, which the model aims to predict.

Code Snippet: Data Loading and Preprocessing

```
# Load the dataset
url = 'WHR2024.csv'
df = pd.read_csv(url)

# Drop rows with missing values and select features
df = df.dropna()
X = df.drop(columns=['Country name', 'Ladder score', 'upperwhisker', 'lowerwhisker'])
y = df['Ladder score']

# Standardize numerical features
scaler = StandardScaler()
X = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

Model Training and SMOTE Resampling

To enhance model performance, **Synthetic Minority Over-sampling Technique (SMOTE)** was attempted to address any class imbalance. If successful, SMOTE resampled the data to improve representativeness. The resampled data was then split into training and test sets for evaluation.

Code Snippet: SMOTE Application and Data Splitting

```
# Apply SMOTE if applicable
try:
    smote = SMOTE(sampling_strategy='auto', k_neighbors=2)
    X_resampled, y_resampled = smote.fit_resample(X, y)
except ValueError as e:
    print(f"SMOTE could not be applied: {e}")
    X_resampled, y_resampled = X, y

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2,
```

Model Selection and Evaluation

Three regression models were evaluated: Random Forest, Gradient Boosting, and Support Vector Regression (SVR). A **K-Fold cross-validation** technique (with 5 splits) was employed to ensure generalizable model performance. Key evaluation metrics include Mean Squared Error (MSE), Mean Absolute Error (MAE), and R^2 score.

Code Snippet: Model Training and Evaluation

```
regressors = {
    "Random Forest": RandomForestRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(random_state=42),
    "SVM": SVR()
}
kf = KFold(n_splits=5, shuffle=True, random_state=42)

for name, model in regressors.items():
    cv_scores = cross_val_score(model, X_train, y_train, cv=kf, scoring='neg_mean_squared_err
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    print(f"{name} Regressor:")
    print(f"  Cross-Validation MSE: {-cv_scores.mean():.4f}")
    print(f"  Test MSE: {mse:.4f}")
    print(f"  Test MAE: {mae:.4f}")
    print(f"  Test R^2: {r2:.4f}\n")
```

The Random Forest model was selected as the final model due to its robust performance across metrics. This model was then saved using joblib for integration with a Flask web application.

Code Snippet: Model Saving

```
import joblib
joblib.dump(model, 'random_forest_model.pkl')
joblib.dump(scaler, 'scaler.pkl')
```

Explainable AI (XAI) with SHAP

SHAP values provide interpretable insights into how each feature contributes to the well-being score predictions. By employing SHAP's KernelExplainer, the study visualizes feature impacts for individual predictions, making the model's behavior transparent.

Code Snippet: SHAP Analysis and Visualization

```
import shap
import matplotlib.pyplot as plt
from io import BytesIO
import base64

# Generate SHAP values for visualization
explainer = shap.KernelExplainer(model.predict, shap.sample(X_test, 100))
shap_values = explainer.shap_values(X_test)

plt.figure()
shap.force_plot(explainer.expected_value, shap_values[0], X_test.iloc[0], matplotlib=True, show=False)
buf = BytesIO()
plt.savefig(buf, format="png", bbox_inches="tight")
buf.seek(0)
shap_img = base64.b64encode(buf.getvalue()).decode("utf-8")
```

Conclusion

This research demonstrates the development of a well-being prediction model that is both accurate and interpretable. By integrating SHAP with a Random Forest model in a Flask application, this project provides transparent predictions for well-being scores, supporting SDG 3: Good Health and Well-being. This framework shows potential as a tool for researchers and policymakers to gain insights into socio-economic factors affecting well-being.