ML mini project one page writeup

Bhuvan hebbar - PES2UG23CS129

Bikram dutta – PES2UG23CS132

Problem Statement

PovertyEmbedding aims to analyze and model poverty and GDP data using machine learning techniques. The project leverages multiple datasets to understand the relationship between economic indicators and poverty levels, with the goal of extracting actionable insights for policy and research.

End-to-End Pipeline & Project Flow

This project builds a comprehensive pipeline to analyze and predict country-level poverty rates by combining:

- Wikipedia country summaries (textual context)

- Semantic text embeddings

- GDP per capita data

- Multidimensional Poverty Index (MPI) metrics

Pipeline Steps

1. Wikipedia Country Summary Collection

- Fetches short Wikipedia summaries for all countries using Python (wikipedia, pycountry, pandas).

- Output: wikipedia_countries.csv (country, summary)

2. Embedding Generation

- Loads wikipedia_countries.csv and uses SentenceTransformers (all-MiniLM-L6-v2) to generate semantic embeddings for each summary.

- Output: wikipedia_country_embeddings.csv (country, summary, embedding dims)

3. Poverty Data Integration

- Loads World Bank poverty data from API_SI.POV.NAHC_DS2_en_csv_v2_171.csv.

- Extracts poverty rates for a target year (e.g., 2018), aligns country names, and merges with Wikipedia embeddings.

4. GDP per Capita Integration

- Loads GDP data from API_NY.GDP.PCAP.CD_DS2_en_csv_v2_24794.csv.

- Merges GDP per capita with the combined Wikipedia-poverty dataset.

5. MPI Data Integration

- Loads MPI data from MPI_national.csv.

- Calculates average poverty rate from urban/rural MPI, merges with Wikipedia embeddings and other features.

6. Exploratory Data Analysis (EDA)

- Visualizes distributions, correlations, and relationships between GDP, poverty, and embedding features.

- Plots global poverty rates on a world map using Plotly.

7. Feature Engineering & Selection

- Scales features, selects top features based on correlation with poverty rate.

- Prepares data for machine learning.

8. Model Training & Evaluation

- Trains regression models (Linear, Ridge, Lasso, ElasticNet, SVR, RandomForest, LightGBM) to predict poverty rates.

- Uses Randomized and Grid Search for hyperparameter tuning.

- Evaluates models on test set and interprets results.

9. Error Analysis

- Analyzes model residuals to identify patterns and outliers in predictions.

Datasets & Their Roles

- API_SI.POV.NAHC_DS2_en_csv_v2_171.csv: World Bank poverty rates by country and year. Used as the main target variable for prediction and analysis.

- API_NY.GDP.PCAP.CD_DS2_en_csv_v2_24794.csv: GDP per capita by country and year. Used as a key economic feature to improve model accuracy and interpretability.

- MPI_national.csv: Multidimensional Poverty Index data, including urban/rural headcounts and deprivation intensity. Used for alternative poverty metrics and deeper analysis.

- wikipedia_countries.csv: Raw Wikipedia summaries for each country. Used for generating semantic embeddings.

- wikipedia_country_embeddings.csv: Final dataset with country, summary, and embedding vectors. Used as input features for modeling and merging with other datasets.

Implementation Overview

- Preprocessing: Data cleaning performed in fix.py and within the Jupyter notebook. Missing values handled, columns standardized, and relevant features selected.

- Analysis & Modeling: The Jupyter notebook (ml_mini_project_129_132.ipynb) contains EDA, visualization, and machine learning model development. Models include regression and classification techniques to predict poverty levels and analyze correlations.

- Tools: Python, pandas, scikit-learn, matplotlib, seaborn, plotly, sentence-transformers, and Jupyter Notebook.

Conclusions & Challenges

Conclusions:

- Strong correlation observed between GDP per capita and poverty indicators.

- Machine learning models can provide reasonable predictions but are limited by data quality and feature selection.

Challenges:

- Wikipedia dataset is on the whole way too large so scraping was needed to narrow down the articles.

- Data inconsistencies and missing values required extensive cleaning.

- Limited dataset size and scope affected model generalizability.

- Feature engineering was critical for improving model performance.