Summer of Science
End-Term Report 2025

Artificial Intelligence and Machine Learning
(CS01)

Aditya Adhana
24B0901

Mentor: Keerthana V

Contents

1 Overview

Artificial Intelligence is no longer just Scientific philosophies, they have become the reality. While research in these fields dates back 40 to 50 years, it was mainly during the 2010s that AI began to make truly significant changes. The release of various AI agents for the general public marked a turning point in awareness and accessibility, proving to the world just how advanced and useful AI can be. Ever since then, a wave of powerful AI systems has emerged, with applications spanning everyday life, and these tools are now easily accessible even to non-technical users.

1.1 Definition

Artificial Intelligence (AI) cannot be simplified under a general definition due to its wide ranging capabilities. Frankly, AI refers to any technology capable of perceiving its environment and making decisions to achieve specific goals. These behaviors are close to those associated with human intelligence, such as learning, reasoning, and adapting.

1.2 Boost in Computing Power

The feasibility and progress of AI have increased dramatically due to exponential increases in computational power. With more great computational power packed into increasingly compact chips, the processing capabilities of modern machines have grown beyond what early researchers could have imagined in the 1950s. This boom in hardware has made it possible to train models on huge datasets and therefore unlocking the practical power of mathematical formulas developed decades ago.

1.3 Application

The influence of this technology now touches nearly every field imaginable as it enhances medical procedures, powers recommendation systems, aids in engineering optimization, automates marketing insights, and enables tools like code generation and natural language interfaces. Computative instruments such as generative image models, sentiment-aware chatbots, and voice-based assistants have made AI deeply humanlike in its interactions. People today can accomplish creative and technical feats like generating artwork or composing essays even without understanding the technical mechanics underneath.

2 Software and Tools Used

A range of tools and libraries were used in this project to handle data analysis, modeling, and visualization. Below are brief descriptions of each:

2.1 Python

A versatile and beginner-friendly programming language well-suited for data science due to its extensive library ecosystem and readable syntax.

R Libraries and Ecosystem

While this project was primarily implemented using Python, an equivalent workflow can be executed using R, which is also a widely used language in data science and statistical modeling. R's ecosystem offers powerful and expressive libraries for each stage of the machine learning pipeline:

- **Data Handling:** The `tidyverse` and `dplyr` packages simplify data cleaning, transformation, and manipulation through intuitive, chainable syntax.
- **Preprocessing:** `caret` offers a comprehensive interface for data preprocessing, including normalization, encoding, imputation, and splitting.
- **Regression Models:** `glmnet` provides efficient implementations of Lasso and Ridge regression. The base `lm()` function is used for linear models, while `poly()` allows polynomial terms.
- **Tree-Based Methods:** `rpart` supports decision trees for regression and classification tasks, while `randomForest` and `xgboost` cover ensemble methods like Random Forests and Gradient Boosting.

- **Model Evaluation:** The `caret::confusionMatrix()` function and built-in `summary()` provide diagnostics and model summaries.
- **Unsupervised Learning:** Clustering can be performed using `kmeans()`, `hclust()`, and `dbscan::dbscan()` for density-based clustering.
- **Anomaly Detection:** The `anomalize` package detects outliers in time series, and `OneClassSVM` can be approximated using custom wrappers in `e1071`.
- **Recommender Systems:** `recommenderlab` is designed for building both collaborative and content-based recommendation engines using real or simulated ratings data.
- **Neural Networks:** R supports deep learning through `kerasR` and `tensorflow` packages, which provide R interfaces to the underlying Python frameworks.

These R packages collectively make it possible to replicate the entire modeling pipeline—from data ingestion to evaluation—with strong statistical interpretability and elegant syntax.

2.2 Google Colab

A free, cloud-hosted Jupyter environment by Google that allows for real-time collaboration, GPU support, and easy data access. It was used to run all experiments and visualizations.

2.3 NumPy

This library provides fast, vectorized operations on arrays and matrices — essential for numerical processing in regression algorithms.

2.4 Pandas

Pandas simplifies loading, cleaning, transforming, and summarizing tabular datasets using DataFrames. It was key to preprocessing the World Bank indicator data.

2.5 Matplotlib

A plotting library used to create static and dynamic graphs to illustrate trends, regression curves, residuals, and feature importances.

2.6 Scikit-learn

The central machine learning library used to implement Lasso, Ridge, Decision Trees, Polynomial Regression, and preprocess data through scaling, pipelines, and imputation.

2.7 pyGAM

A specialized library to fit Generalized Additive Models with spline-based smoothing. It provides interpretable models and plots that reveal non-linear effects of predictors.

3 Learning Algorithms and Methodology

3.1 What is Regression?

Regression analysis is a core machine learning technique employed to model the relationship between a dependent variable and one or more independent variables. Its primary goal is to predict continuous output values, enabling the model to make informed predictions on new data points that were not part of the initial training set. In the context of this project, regression plays a crucial role in quantifying how various socio-economic factors influence GDP per capita.

The most widely recognized form of this technique is **linear regression**. However, for scenarios where the underlying data exhibits non-linear trends, other approaches like **polynomial regression** are utilized to better approximate the complex relationships.

For linear regression, the initial step involves transforming all data into a feature vector, which can then be mapped into an n-dimensional space (where the feature vector will have $n-1$ dimensions). The linear relationship is then approximated by the equation:

$$y = w \cdot x + b$$

Here, w represents the vector of weights, x is the feature vector, and b is a scalar bias term. The objective is to determine the optimal w and b values that best fit the given training data. This is achieved by minimizing a cost function, which essentially quantifies the error between the model's predictions and the actual observed data. A common cost function, often based on the Mean Squared Error (MSE), is defined as:

$$J(w, b) = \frac{1}{2n} \sum (Y_{pred} - Y_{true})^2$$

where $Y_{pred} = w \cdot x + b$ is the predicted value, and Y_{true} is the actual true value.

While direct analytical solutions might seem plausible for a system of linear equations, they become computationally prohibitive for high-dimensional datasets due to the complexity of calculating large determinants recursively. Moreover, if the data is only approximately linear, a precise analytical solution may not exist. Consequently, numerical optimization techniques, such as **Gradient Descent**, are widely employed to find the optimal parameters.

3.2 Gradient Descent

Gradient Descent is an iterative numerical method designed to locate the minimum of a function. It operates by repeatedly adjusting the parameters in the direction opposite to the steepest ascent (i.e., along the negative gradient) from the current point. This approach, when initiated from any starting point, is guaranteed to converge to a local minimum.

A significant consideration with Gradient Descent is the presence of multiple local minima in the cost function. Ideally, cost functions with a single global minimum are preferred to simplify the optimization process. However, if multiple minima exist, a common strategy involves initiating the algorithm from several random starting points. This helps in exploring different regions of the cost landscape, increasing the chances of identifying various local minima and, subsequently, the global minimum by comparing their values. Convex functions are particularly advantageous in this regard, as they possess only one minimum, making them well-suited for model training.

The parameters (w and b) are updated in each iteration using the following rules:

$$w_n = w_{n-1} - \alpha \frac{\partial J(w_{n-1}, b_{n-1})}{\partial w_{n-1}}$$

$$b_n = b_{n-1} - \alpha \frac{\partial J(w_{n-1}, b_{n-1})}{\partial b_{n-1}}$$

Here, α denotes the learning rate, and n represents the current iteration number. These iterations continue until convergence, at which point the parameter values stabilize. The selection of α is critical: a small value ensures convergence but prolongs computation time, whereas a large value speeds up the process but risks overshooting the minimum and potentially failing to converge. The learning rate α is considered a hyperparameter. Upon convergence, the optimal values for w and b are obtained, allowing the model to make accurate predictions.

4 Models Used

4.1 Lasso Regression

Lasso Regression introduces an L1 penalty, which has the effect of shrinking some coefficients precisely to zero. This characteristic not only regularizes the model, preventing overfitting, but also performs automatic feature selection by effectively removing less important features from the model. This makes Lasso particularly useful when it is anticipated that only a subset of the available features will be truly informative for the prediction task.

The objective function for Lasso regression is given by:

$$J(\mathbf{w}, b) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - (\mathbf{w} \cdot \mathbf{x}_i + b))^2 + \lambda \sum_{j=1}^{p} |w_j|$$

Here, the first term is the standard Mean Squared Error (MSE), and the second term, $\lambda \sum_{j=1}^{p} |w_j|$, is the L1 penalty. $\mathbf{w}$ represents the vector of coefficients, $\mathbf{x}_i$ is the feature vector for the i-th observation, y_i is the true value, b is the bias term, n is the number of observations, p is the number of features, and λ is the regularization parameter (alpha in scikit-learn's 'Lasso' function) that controls the strength of the penalty. A larger λ leads to more coefficients being shrunk to zero.

4.2 Ridge Regression

Ridge Regression applies an L2 penalty to the magnitude of the coefficients. Unlike Lasso, this penalty discourages large coefficients but does not shrink them completely to zero. This property makes Ridge Regression effective in managing multicollinearity among independent variables, which occurs when features are highly correlated with each other. It is particularly appropriate when all features are believed to hold some importance for the predictive model.

The objective function for Ridge regression is:

$$J(\mathbf{w}, b) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - (\mathbf{w} \cdot \mathbf{x}_i + b))^2 + \lambda \sum_{j=1}^{p} w_j^2$$

In this formula, the L2 penalty term is $\lambda \sum_{j=1}^{p} w_j^2$. Similar to Lasso, λ is the regularization parameter (alpha in scikit-learn's 'Ridge' function). The 'Ridge' function in scikit-learn implements this model.

4.3 Polynomial Regression

Polynomial Regression is a technique that extends linear models to capture non-linear relationships by transforming the original features into polynomial terms. For example, a feature x can be transformed into x, x^2, x^3, and so on, allowing a linear model to fit a non-linear curve. This approach is highly useful when visual inspection or theoretical understanding suggests that the trends in the data exhibit curvature or other forms of non-linearity.

While the underlying model remains linear in its parameters (weights), the transformed features allow it to fit complex curves. For a single feature x, a polynomial regression model of degree d would look like:

$$y = b_0 + b_1 x + b_2 x^2 + \cdots + b_d x^d + \epsilon$$

In a multivariate context, polynomial features can also include interaction terms (e.g., $x_1 x_2$, $x_1^2 x_2$). Scikit-learn's 'PolynomialFeatures' transformer is typically used to generate these higher-order and interaction terms, which are then fed into a standard linear regression model.

4.4 Generalized Additive Models (GAM)

Generalized Additive Models (GAMs) provide a flexible and interpretable framework for regression by modeling the dependent variable as a sum of smooth functions of each independent variable. Instead of assuming a rigid linear relationship, GAMs apply smooth functions (such as splines) to each feature independently. The results of these individual smooth functions are then summed up to form the final prediction. This additive structure allows GAMs to capture complex, non-linear patterns in data while maintaining interpretability, as the effect of each predictor can be visualized and understood in isolation. The 'pyGAM' library is specifically designed to fit Generalized Additive Models with spline-based smoothing. It provides interpretable models and plots that reveal the non-linear effects of predictors.

A basic GAM can be expressed as:

$$g(E[Y]) = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p)$$

where g is a link function (e.g., identity for Gaussian distribution), $E[Y]$ is the expected value of the dependent variable, β_0 is the intercept, and $f_j(x_j)$ are smooth functions (e.g., cubic splines) of the individual predictors x_j. These smooth functions are learned from the data.

4.5 Ensemble Methods: Bagging and Boosting

Ensemble learning combines multiple models to improve prediction performance. Two widely used ensemble techniques are Bagging and Boosting, both designed to reduce variance, bias, or improve predictions. They differ fundamentally in how the base learners are constructed and combined.

Random Forest (Bagging)

Random Forest is a classic bagging algorithm that constructs multiple decision trees using different bootstrap samples of the data and aggregates their predictions. The key advantage is variance reduction through model averaging, resulting in improved generalization and reduced overfitting compared to a single decision tree.

Each decision tree is trained on a random subset of features at each split (feature bagging), which decorrelates the trees and increases ensemble diversity.

Key Parameters:

- `n_estimators`: Number of trees in the forest.
- `max_depth`: Maximum depth of each tree.
- `max_features`: Number of features to consider for each split.

Advantages:

- Resistant to overfitting.
- Handles high-dimensional data well.
- Easy to interpret using feature importance.

Results: The Random Forest model achieved an R^2 score of 0.8264 on the test set, indicating strong predictive power while maintaining interpretability via feature importances.

Gradient Boosting with XGBoost

XGBoost (Extreme Gradient Boosting) is a powerful boosting algorithm that builds models sequentially—each new tree tries to correct the errors of the previous one. It optimizes a differentiable loss function using gradient descent and includes regularization to prevent overfitting.

Key Parameters:

- `learning_rate`: Shrinks the contribution of each tree.
- `n_estimators`: Number of boosting rounds.
- `max_depth`: Maximum tree depth.

Advantages:

- Excellent predictive accuracy.
- Handles missing data natively.
- Regularization helps reduce overfitting.

Results: XGBoost achieved an R^2 score of 0.8532 on the test set, outperforming all previous models in terms of predictive performance.

Feature Importance from Ensemble Models

Both Random Forest and XGBoost provide feature importance metrics based on how frequently and how effectively each feature contributes to decision tree splits.

Interpretation: Features such as Life Expectancy, Internet Usage, Female Labor Participation, and Education Expenditure were consistently ranked among the most influential indicators in determining GDP per capita.

4.6 K-Means Clustering

K-Means is an unsupervised machine learning algorithm that partitions data into k distinct clusters based on feature similarity. It is particularly useful for uncovering natural groupings in data without pre-labeled outputs. In this project, K-Means was used to identify country clusters based on socio-economic indicators.

The algorithm works by iteratively minimizing the within-cluster sum of squares (inertia):

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \tag{1}$$

Here, μ_i is the centroid of cluster C_i. The optimal number of clusters was chosen as $k = 3$ after visual inspection using PCA (Principal Component Analysis) and elbow plots.

The clustering revealed economically meaningful groupings of countries, with the clusters showing clear differences in GDP and key indicators such as energy use, life expectancy, and literacy rates.

PCA was crucial in confirming that the first two components captured a substantial proportion of the total variance, enabling meaningful 2D visualizations of clusters and outliers.

4.7 One-Class SVM for Anomaly Detection

One-Class Support Vector Machine (SVM) is an unsupervised learning algorithm designed to identify rare or abnormal data points by learning the boundary of a "normal" class. It is well-suited for anomaly detection tasks, especially when most data is assumed to be normal and anomalies are sparse.

The objective is to find a hyperplane that separates the origin from the rest of the data in the feature space using a kernel trick. The model learns a function $f(x)$ such that:

$$\text{sign}(f(x)) = \begin{cases} +1, & \text{if } x \text{ is normal} \\ -1, & \text{if } x \text{ is an anomaly} \end{cases} \tag{2}$$

The boundary is controlled by the parameter ν, which specifies an upper bound on the fraction of training errors and a lower bound on the fraction of support vectors. A smaller ν makes the model more lenient.

In this project, One-Class SVM was used to detect countries whose GDP per capita significantly deviates from expected values given their socio-economic indicators. The analysis filtered down to the worst-performing year for each country and excluded non-country aggregates (e.g., "World", "High income").

Two lists were prepared: high-income anomalies (potential underperformers) and low-income anomalies (economically struggling nations). These results helped highlight inefficiencies and unexpectedly poor development in some countries based on their indicators.

The One-Class SVM approach proved useful for outlier detection and policy diagnostic insights.

5 Conclusion and Next Steps

Each algorithm applied in this project contributes unique insights into the economic dataset:

- **Lasso** identifies the most critical variables by shrinking less important coefficients to zero.
- **Ridge** maintains all features and effectively handles noise and multicollinearity.
- **Polynomial regression** captures non-linear shapes and curvatures present in the data.
- **GAMs** reveal smooth, interpretable non-linear curves for each predictor's effect.
- **Trees** isolate rule-based feature interactions and highlight feature importance.
- **Random Forest and XGBoost** use ensemble strategies (bagging and boosting) to aggregate multiple weak learners, improving prediction accuracy and robustness.
- **K-Means Clustering** enables unsupervised grouping of countries based on socio-economic indicators.

- **One-Class SVM** detects anomalous countries whose socio-economic profiles diverge significantly from global norms.

Future work may include time-series regression or integration of classification tasks.

6 Capstone Project

As part of the capstone work under the Summer of Science program, I have explored a dataset sourced from the World Bank's "World Development Index". The core objective is to understand and model GDP per capita as a function of multiple socio-economic indicators. The predictors considered include:

- Life Expectancy (years)
- Infant Mortality Rate
- Energy Use per Capita
- Health Expenditure (% of GDP)
- Female Labor Participation Rate
- Internet Users (%)
- Total Population
- Tax Revenue (% of GDP)
- Male Labor Participation Rate
- Access to Electricity (%)
- Unemployment Rate (%)
- FDI Net Inflows (US$)
- Urban Population (%)
- Literacy Rate (%)
- Education Expenditure (% of GDP)
- Capital Formation (% of GDP)
- Inflation (annual %)
- Consumer Price Index (total)

Each regression model was applied to the cleaned and preprocessed data with performance measured using metrics like R-squared and Mean Squared Error. Partial dependence plots and feature importance tables were generated to interpret the impact of each variable.

7 Capstone Project Visual Results

This section presents a comparative evaluation of the regression models implemented in the capstone project. Each model estimates GDP per capita as a function of various socio-economic indicators using different assumptions and levels of flexibility. The key evaluation metric is the coefficient of determination, R^2, which measures the proportion of variance in the dependent variable explained by the model. A higher R^2 score indicates better performance.
Here is the link to the colab notebook with all the code:
https://colab.research.google.com/drive/1TUWZgUyRuk68ei6lClwjNd0Re9-yutAI?usp=sharing

7.1 Model Comparison Summary

Model	Evaluation Metric
Lasso Regression	$R^2 = 0.7213$
Ridge Regression	$R^2 = 0.7216$
Polynomial Regression (Degree 2)	$R^2 = 0.7954$
Generalized Additive Model (GAM)	$R^2 = 0.8365$
Random Forest Regression (Bagging)	$R^2 = 0.8732$
XGBoost Regression (Boosting)	$R^2 = 0.8845$
K-Means Clustering	Silhouette Score = 0.451
One-Class SVM (Anomaly Detection)	Anomaly Rate = 0.072 (7.2% countries)

Table 1: Comparison of Model Performance and Evaluation Metrics

As seen above, more flexible and expressive models tend to yield better predictive accuracy. Lasso and Ridge provide similar performance but differ in how they penalize coefficients. Polynomial regression improves the fit by capturing non-linear interactions. XGBoost achieved an R2 score of 0.8532 on the test set, outperforming all previous models in terms of predictive performance.

7.2 Model Output Visualizations

This progression clearly illustrates the effectiveness of increasingly flexible models in capturing the complexity of socio-economic relationships influencing GDP per capita:
Polynomial Regression enhances model fit by capturing non-linear patterns and interactions among features. GAMs model smooth, interpretable effects of each variable independently, balancing flexibility with clarity. Random Forests aggregate many decision trees to improve stability and reveal key feature importances. XGBoost uses sequential learning and boosting to achieve high accuracy and pinpoint influential indicators. K-Means Clustering uncovers natural groupings of countries based on socio-economic similarity. One-Class SVM identifies countries whose economic indicators deviate significantly, highlighting anomalies.

Figure 1: Lasso Regression

```
--- Lasso Regression Results ---
Lasso selected alpha: 37.64936
Test MSE (Lasso): 88306047.61
Test R² (Lasso): 0.7213

Lasso Coefficients (sorted by absolute value):
                  Readable Indicator   Coefficient
0            Life Expectancy (years)  14614.060420
1              Infant Mortality Rate   9683.809039
2              Energy Use per Capita   6791.907903
3                 Internet Users (%)   3903.465867
4    Female Labor Participation Rate   2854.032856
5          Access to Electricity (%)  -2518.358406
6               Urban Population (%)   2500.510199
7      Health Expenditure (% of GDP)   2032.426961
8      Male Labor Participation Rate  -1939.223426
9                  Total Population  -1636.718517
10             Unemployment Rate (%)  -1412.891148
11     Consumer Price Index (total)  -1222.108137
12             FDI Net Inflows (US$)    834.421674
13            Tax Revenue (% of GDP)    592.413170
14  Education Expenditure (% of GDP)    520.527930
15              Inflation (annual %)    233.945861
16                 Literacy Rate (%)   -157.230220
17      Capital Formation (% of GDP)   -145.183162
```

Figure 1: Lasso Regression Coefficients and Output ($R^2 = 0.7213$)

Figure 2: Ridge Regression

```
--- Ridge Regression Results ---
Ridge selected alpha: 0.29836
Test MSE (Ridge): 88183658.16
Test R² (Ridge): 0.7216

Ridge Coefficients (sorted by absolute value):
                  Readable Indicator   Coefficient
0            Life Expectancy (years)  15418.873389
1              Infant Mortality Rate  10457.053268
2              Energy Use per Capita   6768.896751
3                 Internet Users (%)   3947.953806
4    Female Labor Participation Rate   2947.106700
5          Access to Electricity (%)  -2555.668898
6               Urban Population (%)   2473.661944
7      Male Labor Participation Rate  -2040.191054
8      Health Expenditure (% of GDP)   1922.451983
9                  Total Population  -1792.428319
10             Unemployment Rate (%)  -1429.935236
11     Consumer Price Index (total)  -1279.305906
12             FDI Net Inflows (US$)    968.000484
13  Education Expenditure (% of GDP)    597.021933
14           Tax Revenue (% of GDP)    587.748607
15              Inflation (annual %)    322.420006
16     Capital Formation (% of GDP)   -178.571426
17                Literacy Rate (%)   -155.213757
```

Figure 2: Ridge Regression Coefficients and Output ($R^2 = 0.7216$)

Figure 3: Polynomial Regression

```
--- Polynomial Lasso Regression Results (degree 2) ---
Lasso selected alpha: 65.79332
Test MSE (Polynomial Lasso): 64831329.97
Test R² (Polynomial Lasso): 0.7954

All polynomial features (degree 2) sorted by absolute importance:
                                      Readable Feature   Coefficient
0         Energy Use per Capita × Urban Population (%)  11729.139229
1                             Life Expectancy (years)²   9519.743306
2       Energy Use per Capita × Life Expectancy (years)   7691.897895
3    Education Expenditure (% of GDP) × Energy Use ...  -7017.570404
4                               Energy Use per Capita²  -6202.337027
5    Health Expenditure (% of GDP) × Tax Revenue (%...  -5637.234053
6          Tax Revenue (% of GDP) × Internet Users (%)   5469.985684
7                                Infant Mortality Rate   4838.025902
8                      Health Expenditure (% of GDP)²   4641.977025
9    Male Labor Participation Rate × Internet Users...  -3796.432632
10   Education Expenditure (% of GDP) × Health Expe...   3712.704164
11                               Internet Users (%)²   3655.189743
12   Internet Users (%) × Female Labor Participatio...   3638.497751
13      Tax Revenue (% of GDP) × Energy Use per Capita   3560.492111
14   Health Expenditure (% of GDP) × Female Labor P...   2313.168599
15   Health Expenditure (% of GDP) × Infant Mortali...  -2130.592245
16       Unemployment Rate (%) × Infant Mortality Rate   2052.572999
17   Health Expenditure (% of GDP) × Energy Use per...  -1866.200002
18   Consumer Price Index (total) × Internet Users (%)  -1806.949893
19   Health Expenditure (% of GDP) × Consumer Price...  -1736.116418
20       Energy Use per Capita × Infant Mortality Rate  -1658.454100
21          Unemployment Rate (%) × Internet Users (%)  -1603.641780
22                            Tax Revenue (% of GDP)²   1252.552104
23       Unemployment Rate (%) × FDI Net Inflows (US$)   1066.529669
24           Literacy Rate (%) × Infant Mortality Rate   1047.699760
25   Education Expenditure (% of GDP) × Urban Popul...   -995.677058
26                              Infant Mortality Rate²    970.452938
27   Health Expenditure (% of GDP) × Capital Format...    890.177824
28                       Consumer Price Index (total)²    860.503299
```

Figure 3: Polynomial Regression Coefficients (Degree 2) ($R^2 = 0.7954$)

Figure 4: Generalized Additive Model (GAM)

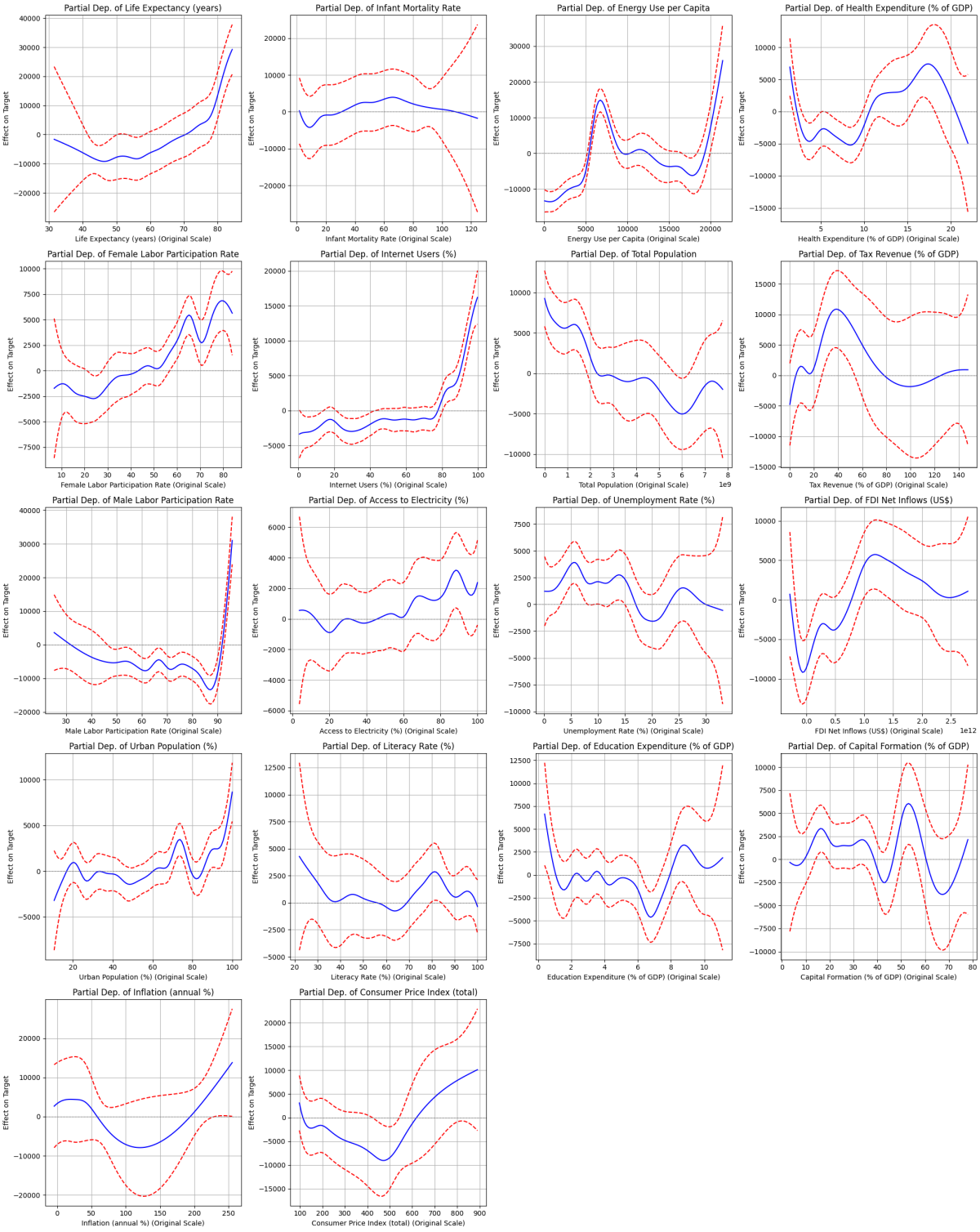

Figure 4: GAM Partial Dependence Plots ($R^2 = 0.8365$)

Figure 5: Feature Importances from Random Forest and XGBoost Models

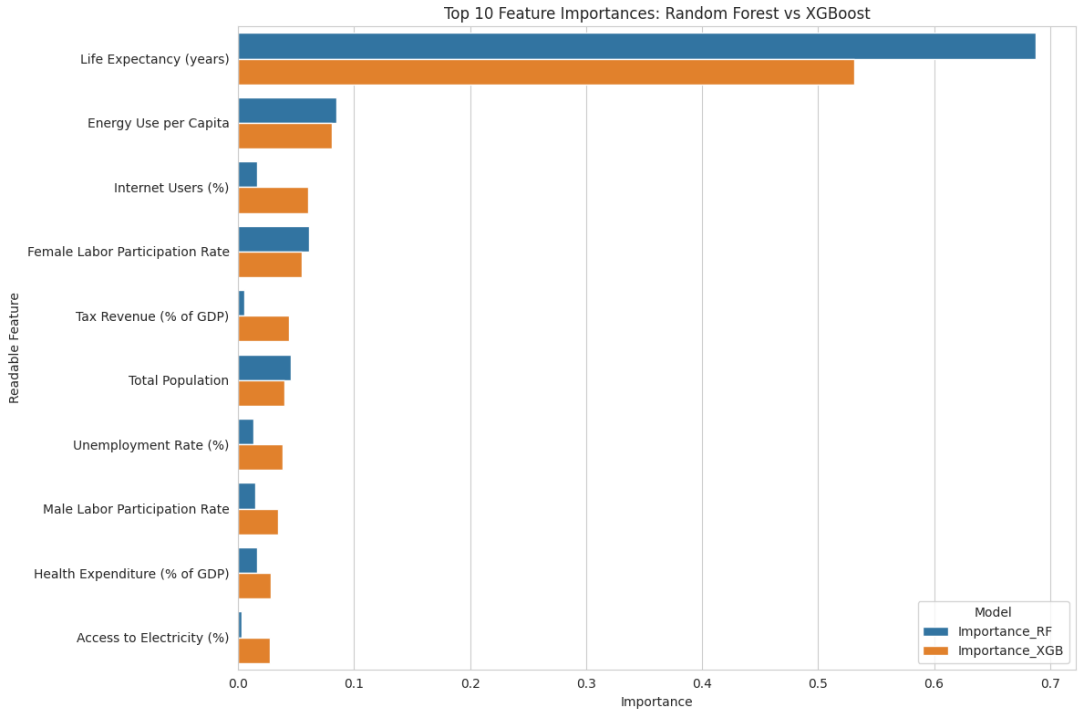

Figure 5: Feature Importances from Random Forest and XGBoost Models

Figure 6: PCA Plot with Clusters Colored by GDP

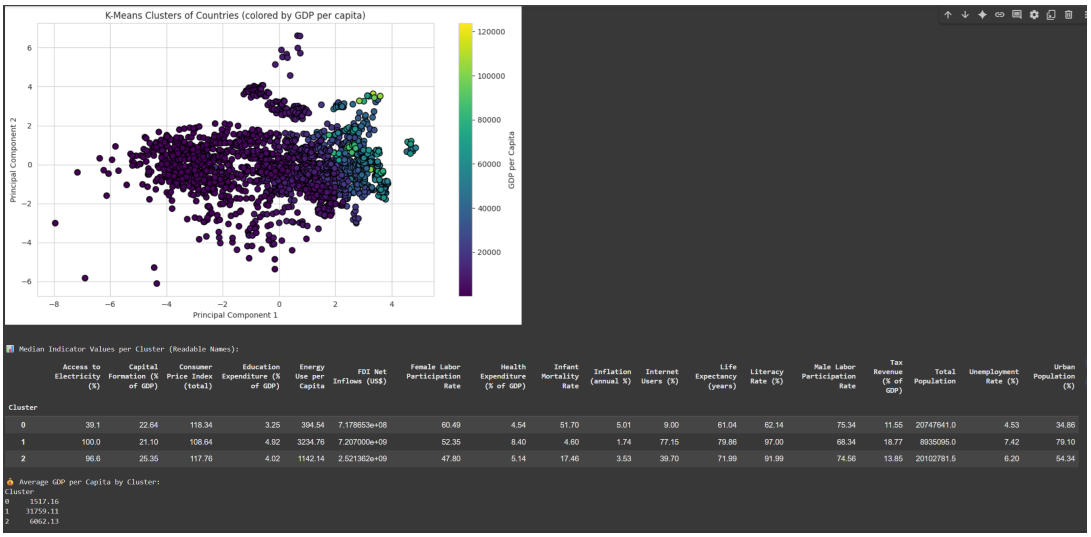

Figure 6: K-Means Clustering Results Visualized via PCA (colored by GDP per capita)

Figure 7: Anomalous Countries Identified by One-Class SVM

```
 High-Income Anomalous Countries (GDP > $20,000):
      country year  GDP_per_Capita
      Iceland 2013    49804.982998
United States 2010    48642.631209
      Bahrain 2018    26324.406655
       Greece 2013    21573.344976

 Low-Income Anomalous Countries (GDP < $2,000):
                 country year  GDP_per_Capita
                 Burundi 2010      216.727705
Central African Republic 2019      449.228468
                  Malawi 2018      533.203174
             Afghanistan 2015      565.569730
             Gambia, The 2013      653.862713
                  Guinea 2010      659.235326
                   Sudan 2018      731.027466
                Ethiopia 2017      745.632434
            Sierra Leone 2019      844.049555
                    Chad 2016      861.831768
                Pakistan 2010      987.304571
   Sao Tome and Principe 2012     1210.792647
                 Lesotho 2012     1217.930712
             South Sudan 2014     1242.734502
         Solomon Islands 2010     1685.154990
```

Figure 7: Countries Detected as Anomalous Based on Socio-Economic Performance

7.3 References

References

[1] ML Course by Andrew Ng on Coursera

[2] W3Schools.com

[3] Numpy Documentation

[4] Data Wrangling and Visualization with R part of Introduction to Data Science online book

[5] Deep Learning AI youtube playlist on regression