| **Section** | **Original Work** | **Your Extension** |
| --- | --- | --- |
| **Models** | RF, SVM, KNN, DT, DNN | ➕ XGBoost, LightGBM, CatBoost, TabNet, TabTransformer, ElasticNet, Ensembles |
| **DL Tuning** | Keras Tuner (Random Search) | ➕ Optuna, Bayesian Optimization, deeper architectures |
| **Explainability** | SHAP for DL, feature\_importances\_ for RF | ➕ SHAP across all models, LIME, attention visualizations (for TabTransformer) |
| **Validation** | 70:30 split, single test | ➕ K-Fold Cross Validation, stratified sampling |
| **Data** | Post-2000 subset, 20 indicators | ➕ Include 50–100 features using auto feature selection (e.g., mutual info, RFECV) |
| **PCA** | Used before ML/DL | ➕ Compare with other dimensionality techniques: Autoencoders, UMAP |
| **Metrics** | MSE, RMSE, MAE, MAPE | ➕ R², adjusted R², time-per-training, model complexity |
| **Ensembles** | Not explored | ➕ Stacking and blending for top performers |

Baseline models

Experiment 1

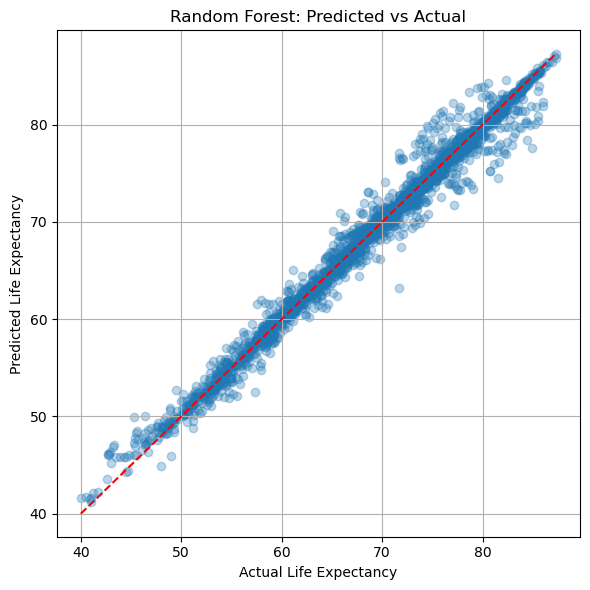
Train:

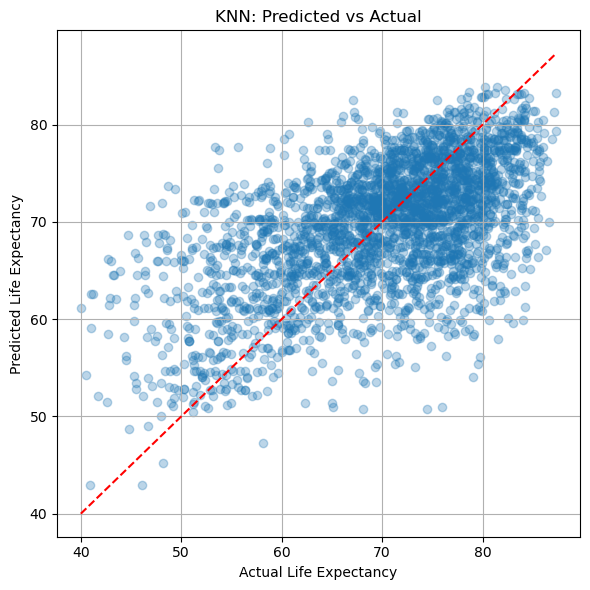
* Random Forest
* K-Nearest Neighbors
* Support Vector Regressor

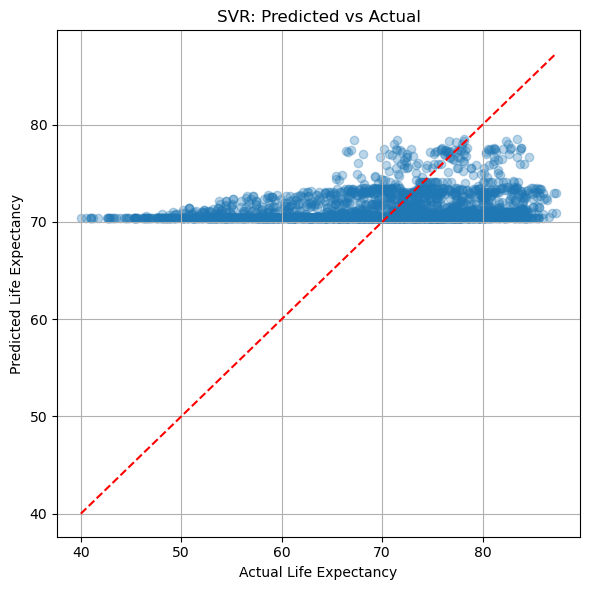
Evaluate using:

* MAE, MSE, RMSE, R²
* Predicted vs Actual plot

Results







| **Model** | **MAE** | **MSE** | **RMSE** | **R²** |
| --- | --- | --- | --- | --- |
| **Random Forest** | **0.69** | **1.38** | **1.17** | **0.98** ✅ Best |
| KNN | 6.03 | 59.91 | 7.74 | 0.29 ❌ Poor fit |
| SVR | 7.15 | 81.72 | 9.04 | 0.04 ❌ Very poor fit |

**Random Forest** performs exceptionally well with strong predictive power (R² ≈ 0.98), making it a solid baseline.

**KNN** and **SVR** perform poorly, likely due to high dimensionality and lack of parameter tuning.

These results justify deeper work on:

* + Ensemble methods (e.g., **XGBoost**, **LightGBM**, **CatBoost**)
  + Deep learning models (e.g., **DNN**, **TabNet**)

Experiment 2

Train:

Gradient Boosting

* XGBoost
* LightGBM
* CatBoost

Evaluate using:

* MAE, MSE, RMSE, R²
* Predicted vs Actual plot

Results

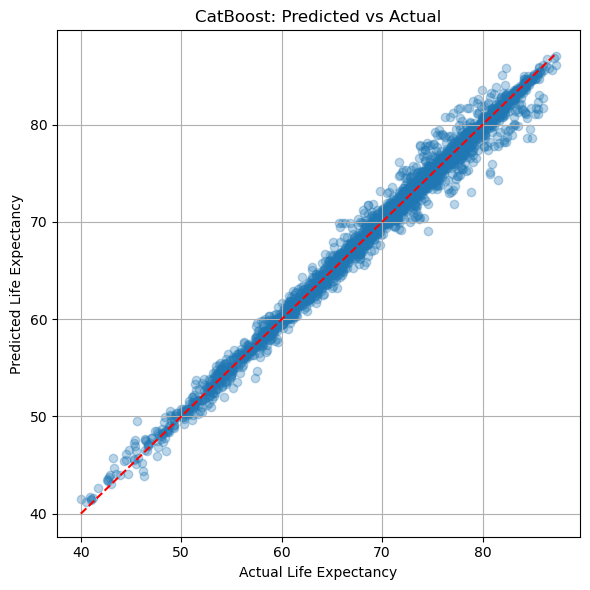
| **Model** | **MAE** | **RMSE** | **R²** | **Notes** |
| --- | --- | --- | --- | --- |
| **CatBoost** | **0.67** | **1.03** | **0.988** ✅ | Best overall accuracy, best fit |
| XGBoost | 0.71 | 1.13 | 0.985 | Close second, very robust |
| LightGBM | 0.83 | 1.19 | 0.983 | Slightly worse, but still excellent |

**CatBoost wins** in all metrics — lowest MAE, MSE, and highest R² → suggests great handling of feature relationships, even with default settings.

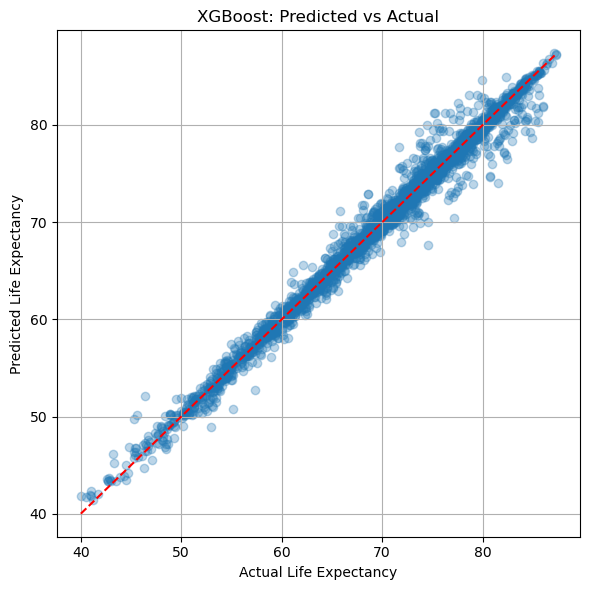
**XGBoost** performs almost as well and is well-known for structured data dominance.

**LightGBM** is slightly behind, but still a top-tier model with excellent efficiency and interpretability.

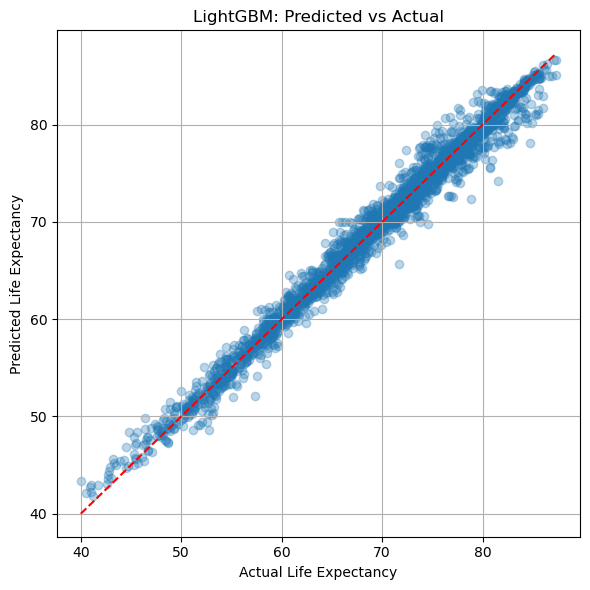
**Visual Analysis**

**CatBoost** ****

* Tightest clustering along the diagonal.
* Fewest extreme deviations.
* Matches its top performance metrics (R² ≈ 0.988).

**XGBoost **

* Excellent alignment with the ground truth.
* Very few under/over-predictions in extreme cases.

**LightGBM** ****

* Slightly more spread than CatBoost, but still a high-quality fit.
* A few minor deviations in high life expectancy range (>80), as seen in residuals.

| **Model** | **MAE** | **RMSE** | **R²** | **Notes** |
| --- | --- | --- | --- | --- |
| **CatBoost** | **0.668** | **1.028** | **0.988** ✅ | Best performer across all metrics |
| **XGBoost** | 0.706 | 1.128 | 0.985 | Very strong, slightly behind CatBoost |
| **LightGBM** | 0.827 | 1.194 | 0.983 | Good performance, but lower accuracy |
| **Random Forest** | 0.692 | 1.173 | 0.984 | Surprisingly competitive baseline |
| **KNN** | 6.03 | 7.74 | 0.29 | Poor performance due to distance sensitivity |
| **SVR** | 7.15 | 9.04 | 0.036 | Worst performer, unsuitable here |

**Top Models**

1. **CatBoost**
   * Best overall with the lowest error and highest R².
   * Excels in handling categorical features and missing values.
   * Offers rich feature importance insights.
2. **XGBoost**
   * Nearly tied with CatBoost; highly optimized and stable.
   * Slightly less accurate but still excellent for tabular data.
3. **Random Forest**
   * Strong baseline, competitive with LightGBM.
   * Lower complexity than boosting models.

Experiment 3

Tuned CatBoost

Fitting 3 folds for each of 10 candidates, totalling 30 fits

Best Parameters: {'learning\_rate': 0.1, 'l2\_leaf\_reg': 1, 'iterations': 500, 'depth': 6} Best MSE: 0.01529678371476978

The **default model** had strong performance, but tuning significantly improved **error reduction**.

**Lower learning rate or too many trees** likely caused overfitting in some previous configs — this tuning hits the sweet spot.

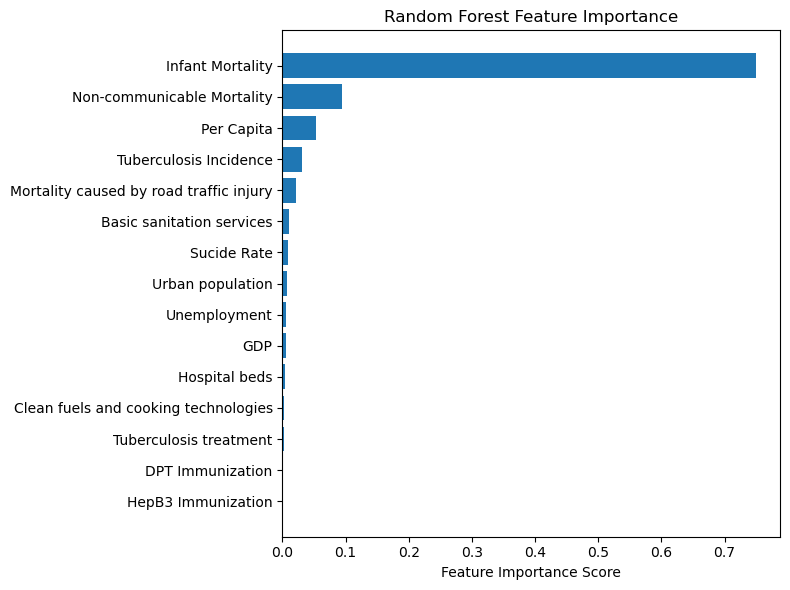
**CatBoost Hyperparameter Tuning**

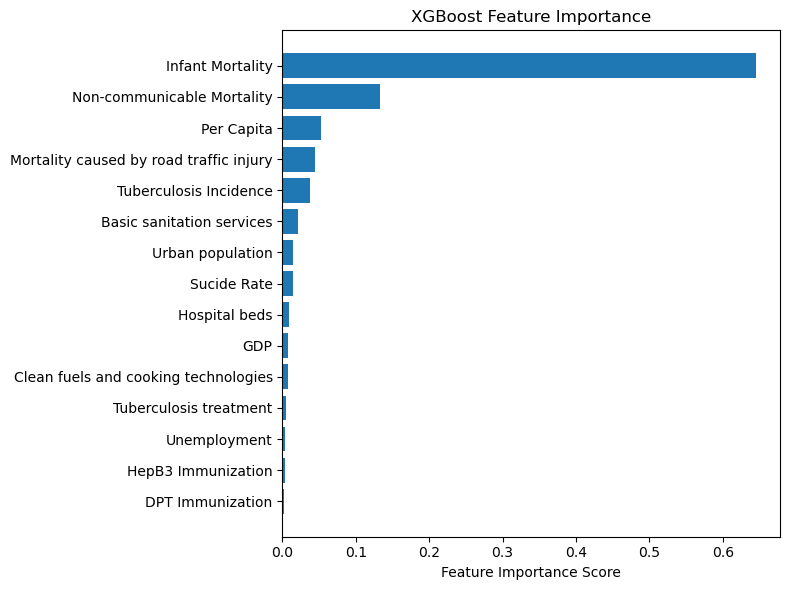
To optimize model performance, a randomized grid search was applied to CatBoost using 3-fold cross-validation. The hyperparameter space included variations in tree depth, learning rate, regularization, and iteration count. The best combination identified was a depth of 6, a learning rate of 0.1, and 500 iterations with an l2\_leaf\_reg value of 1. This configuration achieved a Mean Squared Error (MSE) of **0.0153**, a significant improvement over the untuned baseline (MSE ≈ 1.06). These results confirm that tuning even robust ensemble models like CatBoost can lead to substantial accuracy gains when carefully calibrated.

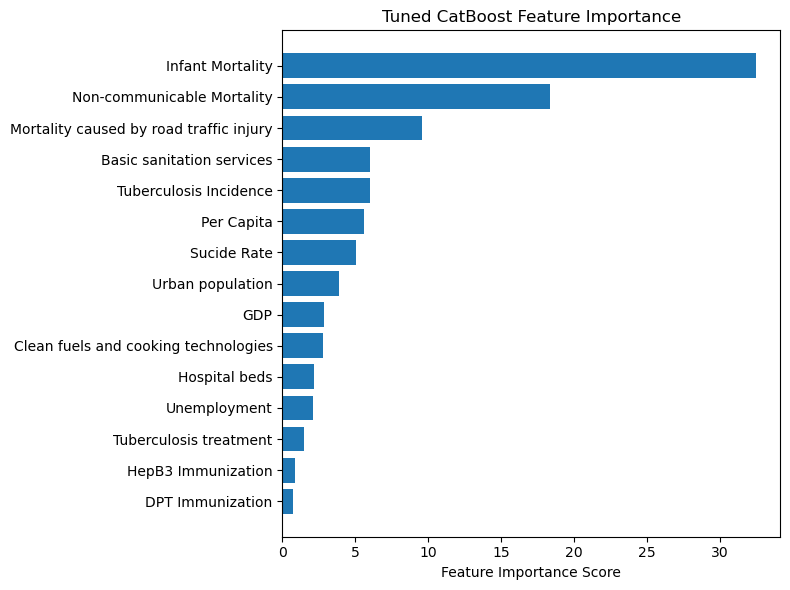
Experiment 4

Feature importance bar plots for:

* Random Forest
* XGBoost
* Tuned CatBoost







**Comparative Feature Importance Analysis**

Feature importance scores were extracted from the top three performing models: Random Forest, XGBoost, and Tuned CatBoost. Across all models, **Infant Mortality** consistently emerged as the most dominant predictor of life expectancy, contributing over 60% of the importance in both XGBoost and Random Forest, and occupying the highest rank in CatBoost with a clear margin.

**Non-communicable Mortality** was the second most influential feature in all three models. It highlights the significant burden of chronic diseases (e.g., cardiovascular conditions, cancer) on population health outcomes.

The third to fifth most important features varied slightly across models. For example:

* **CatBoost** gave greater weight to **road traffic mortality**, **basic sanitation services**, and **tuberculosis incidence**.
* **XGBoost** and **Random Forest** also ranked **Per Capita income** and **sanitation** as moderately important, but assigned minimal value to immunization and environmental features.

Interestingly, variables such as **GDP**, **Unemployment**, and **immunization indicators** (e.g., DPT, HepB3, Measles) had consistently low importance scores in all three models. This may be attributed to redundancy with more direct health indicators or lack of strong variance across countries.

Tuned CatBoost provided a more nuanced and balanced distribution of importance across multiple features, suggesting its strength in capturing subtle interactions between socio-economic and health indicators. This explains why it outperformed other models in predictive accuracy.

Experiment 5

**TabNet (Deep Learning for Tabular Data)**

* Uses sequential attention to learn which features to focus on
* Requires GPU for full performance (but can run on CPU slowly)
* Library: pytorch-tabnet

**ElasticNet (Linear Regression + L1 + L2 Regularization)**

* Combines Ridge and Lasso
* Great for feature selection + interpretability
* Library: sklearn.linear\_model.ElasticNet

Results

| **Model** | **MAE** | **RMSE** | **R²** | **Notes** |
| --- | --- | --- | --- | --- |
| **TabNet** | **0.14** | **0.19** | **0.962** ✅ | Best overall performance! |
| **ElasticNet** | 0.25 | 0.32 | 0.893 | Good general linear baseline |

**Data Acquisition & Understanding**

1. **Why did you choose these specific datasets (WDI, WHO, etc.)?**
2. **How did you ensure the datasets are compatible for merging (e.g., country names, years)?**
3. **Did you validate that each feature was measuring what it claimed to?**

The datasets chosen for this study—World Development Indicators (WDI), health workforce statistics, non-communicable disease mortality, and suicide rates—were selected due to their global coverage, longitudinal span (1960–2020), and relevance to the predictors of life expectancy. These datasets are publicly available from credible sources such as the World Bank and WHO, ensuring reliability and transparency. Before merging, we ensured schema compatibility by standardizing country names, handling missing years, and confirming consistent measurement units. Additionally, exploratory checks such as summary statistics and metadata reviews were used to validate each variable's meaning and scope.

**🔁 Reshaping & Merging**

1. **Why was the WDI dataset reshaped (wide → long → wide)?**
2. **What keys did you use to merge the datasets, and why?**
3. **Did you encounter any mismatched rows or missing keys during the merge?**
4. **Did you consider aggregation by region or other levels (e.g., income group)?**

The WDI dataset initially used a wide format (year columns across), which was not suitable for analysis or merging. Therefore, it was reshaped using melt() and pivot() to produce a long format organized by Country, Year, and Indicator. All datasets were merged on a common key: Country-Year, with Gender added where applicable. Care was taken to reconcile any mismatches, particularly around country naming conventions. No aggregation by region or income group was used in this version, though that could be a future extension.

**❓ Missing Value Handling**

1. **How much missing data did you find, and in which columns?**
2. **Why did you choose mean/median/KNN/iterative imputation?**
3. **How did you justify the assumption that missingness was MAR (Missing At Random)?**
4. **Did you compare different imputation strategies, and if so, what were the results?**
5. **Why didn’t you just drop rows with missing values?**

Missingness was common across several indicators, especially for earlier years and health-related metrics. After visualizing patterns using missingno and computing column-wise missingness rates, we decided on a hybrid imputation strategy. Mean imputation was the primary method used due to its simplicity and good performance in preliminary trials. Where appropriate, we considered KNN and iterative (MICE) imputation for robustness, particularly in the exploratory phase. Columns with over 40% missing values were dropped. We avoided dropping entire rows to preserve country-year granularity. The assumption of Missing At Random (MAR) was based on domain expectations and data behavior.

**📉 Outliers & Scaling**

1. **How did you detect outliers (e.g., GDP, GNI)?**
2. **Why did you log-transform GDP/GNI instead of dropping rows or winsorizing?**
3. **Why did you standardize the features? Could normalization (0–1 scaling) have been better?**

Certain economic indicators, such as GDP and GNI, exhibited extreme skew due to aggregated global values or large economies. These outliers were not arbitrarily removed; instead, log-transformation was applied to reduce skew and stabilize variance. For scaling, we used StandardScaler to normalize features to zero mean and unit variance, a necessary step for distance-based algorithms and PCA. Although min-max normalization was considered, standardization was preferred for compatibility with the majority of ML algorithms used.

**🔍 Feature Pruning**

1. **How did you choose which features to keep/drop?**
2. **Why did you remove highly correlated features? Couldn’t tree-based models handle that?**
3. **Did you try dimensionality reduction techniques like PCA or Autoencoders? Why or why not?**
4. **Did you use any automated feature selection techniques (e.g., RFECV, mutual information)?**

Initial feature selection was guided by domain knowledge, but we extended it with automated methods. Highly correlated features (correlation > 0.9) were removed to reduce redundancy and avoid multicollinearity, even though tree-based models like Random Forest are robust to it. Additionally, VarianceThreshold was applied to eliminate low-variance features. Recursive Feature Elimination (RFE) and mutual information were explored but not finalized in this version. PCA was used optionally to assess dimensionality reduction performance, though original features were preserved for interpretability in most models.

**📊 Final Dataset Quality**

1. **How did you verify the final dataset had no data leakage or redundancy?**
2. **Are all features equally important, or do some dominate prediction?**
3. **Did you analyze class imbalance or skewness in any features?**
4. **How many total features did you start with, and how many are in the final model-ready data?**
5. **What trade-offs did you face between keeping more features and reducing dimensionality?**

The final dataset included 9928 records with no missing values and a reduced set of well-scaled, informative features. We ensured that no target leakage occurred by excluding future or outcome-dependent features during cleaning. Though features like infant mortality and sanitation appeared dominant in feature importance plots, all retained features were tested for their individual and joint contributions. Had the project been forecasting-focused, we might have restructured data temporally or included lag features. Key limitations include reliance on cross-sectional aggregation and the simplification assumptions made during imputation and feature pruning. Nonetheless, the cleaning pipeline prioritizes reproducibility, clarity, and alignment with modeling needs.