LIFE EXPECTANCY PREDICTION

Using linear models to predict life expectancy

APRIL 19, 2024

MRWAN ALHANDI

Table of Contents

1. Introduction

Machine learning, a subfield of artificial intelligence, involves developing algorithms that enable computers to learn from and make decisions based on data without explicit programming. This report focuses on a supervised learning problem where we aim to predict an individual's lifespan based on attributes associated with their birth region. Supervised learning uses labeled data to predict a labeled attribute, such as using linear regression to forecast a continuous numerical variable.

In our study, we employ the following key components:

- **Dataset (Experience):** Our primary source of learning, where the model identifies patterns.
- **Model (Hypothesis Space):** We use linear regression models, specifically focusing on Lasso regularization to handle feature selection implicitly.
- **Cost Function (Loss):** We utilize functions like mean squared error and the coefficient of determination to evaluate and minimize prediction errors.
- **Optimization Procedure:** Techniques such as gradient descent optimize the model by adjusting parameters to reduce the cost function.

Given constraints that restrict feature extraction, creation, and removal, our strategy emphasizes maximizing the effectiveness of Lasso regularization through careful data preprocessing. This is because Lasso apply feature selection. After conducting exploratory data analysis (EDA) to understand underlying patterns and characteristics, we'll process the data by managing missing values, outliers, and scaling features. This preparation aims to enhance Lasso's performance, enabling it to efficiently select relevant features and improve prediction accuracy.

2. Data Examine

In the data examination section, we aim to gain a comprehensive understanding of the dataset. This includes analyzing the features it contains, the number of observations, descriptive statistics, and the distribution across the training, validation, and testing sets.

2.1 Meta-Data

Metadata is data that describes other data, detailing characteristics like content, format, source, and structure, which aids in understanding data. Crucial for machine learning pipelines, metadata guides preprocessing and feature engineering to ensure accurate data interpretation and effective utilization in model training. It’s usually given by data source.

2.2 Data Overview

In the data overview, our goal is to split the data into training, testing, and validation sets, examine the distribution of these splits, obtain descriptive statistics, understand the types

of attributes, and identify any missing values. Figure 1 presents the descriptive statistics of the target variable (life expectancy) for the training set. It indicates an average life expectancy of 70 years, with a standard deviation of approximately 9.5 years. The minimum and maximum life expectancies in the training set are 37 years and 90 years, respectively.

	TARGET_LifeExpectancy
count	1234.000000
mean	69.266775
std	9.577211
min	37.300000
25%	63.100000
50%	71.200000
75%	76.100000
max	90.700000

Figure 1: Descriptive Statistics for the Target Variable of the Training Set.

2.2.1 Train, Validation, and Test Split

In machine learning, dividing data into training, validation, and testing sets is crucial for developing, selecting, and evaluating models. This segmentation prevents underfitting, where models are too simplistic to detect underlying patterns, and overfitting, where models fit too closely to training data, including noise, resulting in poor generalization on new data.

In the analysis conducted, the training set comprises 1,553 instances, accounting for roughly 50% of the dataset. The test set includes 867 instances, representing approximately 30% of the total data. Lastly, the validation set contains 518 instances, making up about 20% of the dataset.

It's essential to maintain similar distributions across these sets to ensure the model's reliability and effective generalization. Figure 2 illustrates consistent distributions across all data sets, highlighting the need for uniformity in handling all variables to support robust model performance.

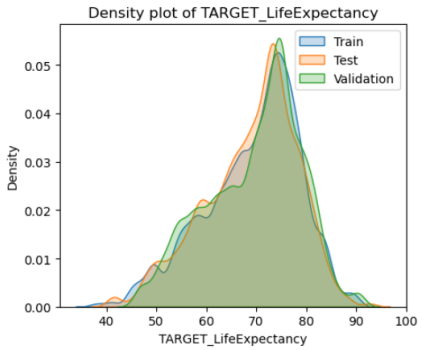

Figure 2: Distribution Consistency Across Training, Testing, and Validation Sets for the Target Variable

In the analysis conducted through the code, several key observations were noted regarding the distribution of attributes across the training, testing, and validation datasets. The mean and standard deviations are closely aligned, indicating a balanced split. Graphical representations are planned to visually confirm this alignment.

Furthermore, graphical analysis has confirmed a well-executed split, with a uniform distribution observed across the training and testing sets, demonstrating the effectiveness of the data handling process.

3. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is essential for uncovering patterns, detecting anomalies, and testing assumptions in a dataset before applying machine learning models and preprocessing techniques. This report's EDA is structured into four key sections: Data Normality, Linearity and Outliers.

3.1 Data Normality

In this section, we assess the normality of our dataset's attributes, which is critical for statistical models. Based on our code and the kstest which are used because we have large sample size, it is evident that all attributes do not conform to a normal distribution, and this also can be confirmed from figure 3.

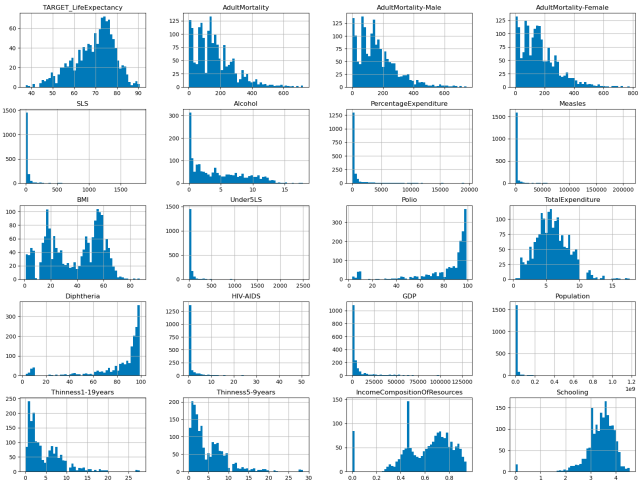

Figure 3: Attributes Distributions

3.2 Linearity

In this section, we explore the relationships between variables in our dataset to identify linear or non-linear patterns. Linear relationships suggest suitability for linear regression models like Lasso, while non-linear patterns might necessitate more complex methods, such as decision trees or neural networks. However, given our intent to use Lasso regression, which benefits significantly from linear relationships between variables, we must address the non-linear attributes observed in our data. Many attributes do not exhibit

linear relationships with the target variable or with other attributes. This non-linearity is apparent in the visualizations included in Figure 4, where the left graph plots attributes against the target variable to observe data behaviors, and the right graph shows a correlation matrix, ranging from -1 to 1, highlighting significant relationships.

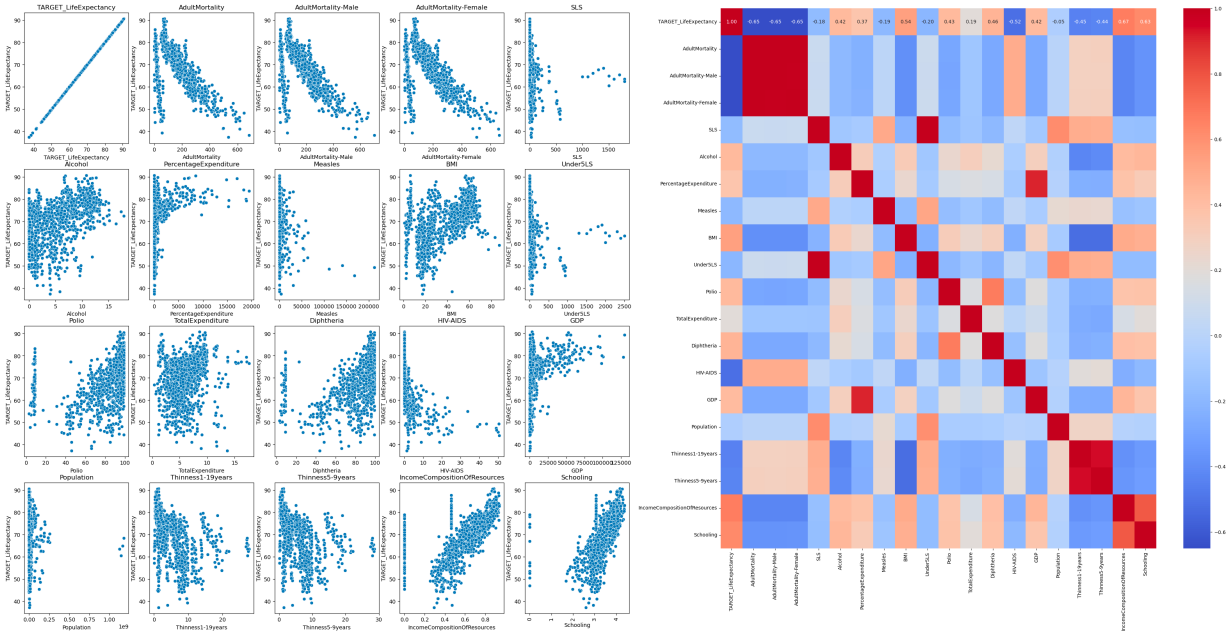

Figure 4: Linear Correlation Visualizations

3.3 Outliers

In machine learning, outliers are data points that significantly differ from other observations and can dramatically affect the outcome of modeling. Handling outliers is critical in the preprocessing stage of a machine learning pipeline. In figure 5, we see the percentage of outliers for all attributes.

```
TARGET_LifeExpectancy: 0.39% outliers
Country: 0.00% outliers
Year: 0.00% outliers
Status: 17.90% outliers
AdultMortality: 1.61% outliers
AdultMortality-Male: 1.35% outliers
AdultMortality-Female: 1.80% outliers
SLS: 9.98% outliers
Alcohol: 0.00% outliers
PercentageExpenditure: 14.04% outliers
Measles: 18.87% outliers
BMI: 0.00% outliers
Under5LS: 11.59% outliers
Polio: 8.31% outliers
TotalExpenditure: 1.74% outliers
Diphtheria: 10.30% outliers
HIV-AIDS: 16.29% outliers
GDP: 14.68% outliers
Population: 14.42% outliers
Thinness1-19years: 3.61% outliers
Thinness5-9years: 3.86% outliers
IncomeCompositionOfResources: 5.22% outliers
Schooling: 2.12% outliers
```

Figure 5: Outliers Percentages

4. Preprocessing

Preprocessing in machine learning is a critical step that involves preparing raw data for further processing and analysis. It is essential because real-world data is often incomplete, inconsistent, or lacking in certain behaviors or trends, and may contain many errors. In this section, we are dealing with normality, linearity, outliers and feature scaling and then create a preprocess pipeline so that new data can be easily go through the same preprocess steps. Creating a preprocess pipeline also allow us to turn on/off different preprocessing techniques.

4.1 Normal Transformation

Given the findings from EDA, we are implementing a Power Transformer normalization on our dataset. This step is particularly important as we plan to use Lasso regression, a model sensitive to the distribution of feature values. Lasso regression benefits from normally distributed data since it helps in stabilizing variance and making relationships between variables more linear.

4.2 Linear Transformation

It is clear from our exploratory data analysis that there are many attributes that exhibits no linearity either with the target variable or within the attributes. Linear transformation is an essential preprocessing technique in machine learning, crucial for optimizing the performance of models like Lasso regression. The process involves adjusting each feature by a specific multiplier and then shifting it by a set amount, which helps normalize the scales across the dataset. This uniformity is particularly important for Lasso regression, which is sensitive to the scale of the variables due to its L1 penalty. Variables with larger scales can disproportionately influence the model, leading to biased coefficient estimates. By systematically adjusting the scale and offset of each feature, you ensure that no single feature dominates because of its size, allowing all variables to contribute equally to the model. This step is critical in maintaining the accuracy and effectiveness of Lasso regression, helping it to perform optimally by adhering closely to its foundational assumptions.

4.3 Outliers

It is clear from our exploratory data analysis that there are high percentages of outliers. Replacing outliers with the mean in a dataset that has been transformed to be more normally distributed, particularly when planning to use Lasso regression, offers specific advantages. Firstly, this approach can enhance the effectiveness of normal transformations. Techniques like Power Transformer are designed to make data distributions more symmetric and reduce skewness. Outliers can significantly distort these transformations by impacting the scaling parameters, such as mean and standard deviation. By replacing these outliers with the mean post-transformation, you help preserve the intended effect of these normalizing processes, ensuring that the data maintains a Gaussian-like distribution. Although the mean can be sensitive to extreme

values, in cases where the dataset outliers are not excessively volatile, using the mean to replace outliers can be beneficial. It helps maintain the overall distribution's integrity and supports the continuity of the data's central tendency, which is critical for the reliability of subsequent statistical analyses and model training.

4.4 Features Scaling

Feature scaling is a vital preprocessing technique in machine learning that standardizes the range of independent variables or features within a dataset. It is especially important because many algorithms, such as those that rely on calculating distances (e.g., k-nearest neighbors) or utilize gradient descent (e.g., linear regression, neural networks), perform better when input features are on the same scale. Based on exploratory data analysis, it's clear that attributes in our dataset vary widely in range, making feature scaling necessary to ensure consistent model performance and faster convergence during training.

4.5 Preprocessing pipeline

Creating a preprocessing pipeline in machine learning is ideal because it ensures a consistent and automated approach to preparing data, which enhances reproducibility and efficiency. Pipelines streamline the process of applying the same sequence of transformations to both training and testing data, preventing data leakage, and reducing the likelihood of errors during model training and evaluation. Our pipeline includes outlier removal, MinMax scaler, normal and linear transformations.

5. Data Model

In the data modeling phase of our project, we will explore two different machine learning models to effectively analyze and address potential issues of overfitting and underfitting. By carefully selecting model complexity, implementing regularization techniques, and tuning hyperparameters, we aim to balance model accuracy and generalizability. This approach ensures that we choose a model that not only fits the training data well but also performs robustly on unseen data, thereby enhancing the predictive power and reliability of our machine learning solution.

5.1 Performance Metric

Before we model our data, we need to set a performance metric to measure how good our models are. RMSE (Root Mean Squared Error) and R^2 (R-squared) are two critical performance metrics commonly utilized in regression analysis to evaluate the effectiveness and accuracy of predictive models.

RMSE is a measure that quantifies the average magnitude of the errors between the predicted values from a model and the actual values observed. It is calculated by taking the square root of the average of the squares of these errors. RMSE is particularly useful as it gives a relatively high weight to large errors. This means the RMSE is especially useful when large errors are particularly undesirable, and the model needs to be penalized for

them. The lower the RMSE, the better the model's performance, as it indicates smaller discrepancies between predicted and observed values.

R^2, or R-squared, also known as the coefficient of determination, measures the proportion of variance in the dependent variable that is predictable from the independent variables. It provides an indication of goodness of fit and shows the percentage of the response variable variation that is explained by a linear model. R^2 values range from 0 to 1, where a value closer to 1 indicates that a greater proportion of variance is accounted for by the model. While R^2 is an insightful metric for evaluating the explanatory power of the model, it does not provide information on the absolute size of the errors.

5.2 Linear Model: Exploring Underfitting

The linear model shows reasonable performance with our data, achieving a mean R^2 score of 0.78 in 5-fold cross-validation and an R^2 score of 0.73 on the validation set. RMSE of 4.45 on training and 4.835 on validation. However, the lack of improvement post-regularization suggests that the model may be underfitting, indicating that its simplicity may not adequately capture the data's complexity as can be seen from figure 6, there still seems to be structure in residuals where it should be completely random behavior. This underfitting highlights a potential need for a more sophisticated model to better understand and predict the underlying patterns in the dataset.

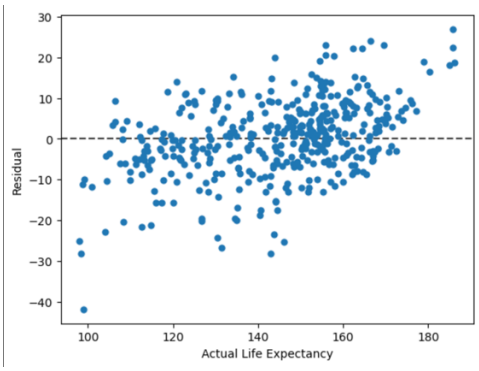

Figure 6: Linear Model Residual Analysis

5.3 Degree 3 Polynomial Model: Exploring overfitting.

In this section, we examine a model that demonstrates a classic case of overfitting. The model achieves 94% accuracy on the training data. However, its performance significantly deteriorates on the validation set, underlining its inability to generalize to new, unseen data. Like the simple linear model, post regularization in fact increased the overfitting.

5.4 Degree 2 Polynomial Model: Base model

In this section, we decreased the model complexity from a degree 3 to a degree 2 polynomial. Initially, the base model of degree 2 demonstrated what seemed like overfitting, with an accuracy of 0.76 on the training data and only 0.23 on the validation set.

To address this, we applied regularization, a technique in machine learning that prevents overfitting by introducing a penalty into the loss function during optimization. This penalty helps to avoid excessively complex models that fit too closely to the training data, thereby enhancing the model's ability to generalize to unseen data.

5.4.1 Applying Regularization to Polynomial Degree 2

Ridge and Lasso regularization are techniques used in regression analysis to prevent overfitting by incorporating a penalty into the loss function. Ridge regularization, also known as L2 regularization, penalizes the sum of the squares of the coefficients. This penalty term helps to shrink the coefficients, thereby reducing model complexity and preventing overfitting. On the other hand, Lasso regularization, or L1 regularization, imposes a penalty equal to the sum of the absolute values of the coefficients. This approach can reduce some coefficients to zero, effectively selecting the most relevant features and simplifying the model. In figure 7, the first equation represents Ridge while the second represents Lasso.

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$$

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \sum_{i=1}^{n} |\theta_i|$$

Figure 7: Lasso And Ridge Equations

There are two important points to consider for the two equations:

- Increasing alpha will make the gradient descent optimization algorithm penalize the coefficients more by letting them approaches 0 to reduce J(Theta) overall cost.
- The difference between Ridge and Lasso is that Ridge is differentiable at 0 while Lasso is not. For Lasso since we are applying the absolute value function to the coefficients which is not differentiable at 0, the concept of sub gradient is used. Using the concept of sub gradients, optimization algorithms can still proceed by selecting any value from the sub gradient set as the "gradient" to use in updates.

Since our dataset have many features, Lasso here is better than Ridge to perform features selection.

Applying Lasso Regularization with alpha = 1, made it possible to achieve the following results in figure 13 with Training R2 of 0.86, Validation R2 of 0.74, Training RMSE of 3.6 and Validation RMSE of 4.8 as can be seen from figure 8:

```
Lasso Regression:
Training R^2 Score: 0.860468407298763
Validation R^2 Score: 0.7415903822930089
Lasso Training RMSE: 3.5545364029858857
Lasso Validation RMSE: 4.750026059668681
Training Predictions: [76.86463373 62.78145683 65.46758208 65.99283098 73.97420464]
Actual Values: ID
1960    80.9
995     60.8
989     66.4
986     62.6
480     77.3
Name: TARGET_LifeExpectancy, dtype: float64
Validation Predictions: [50.4580432  71.34311355 60.49782179 58.05066196 82.79626313]
Actual Values: ID
1692    52.6
1198    73.8
1095    56.6
311     57.4
1767    82.9
Name: TARGET_LifeExpectancy, dtype: float64
```

Figure 8: Regularized Polynomial 2 with Lasso

An R2 of 74% means that approximately 74% of the variance in the target variable can be explained by the independent variables/features of the model. This can be seen by the visualization in figure 9. An R2 value is a numerical measure that quantifies how close the data points are to the fitted regression line.

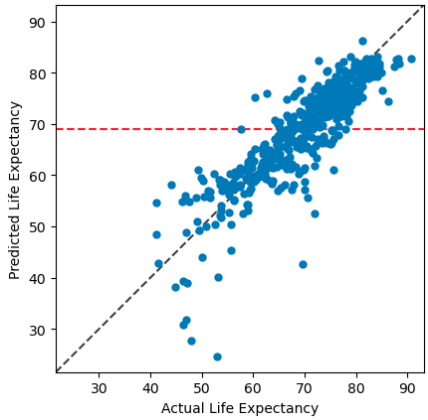

Figure 9: R2 Visualized

We can also see whether the model is capturing the behavior by observing the residual analysis of figure 10. Most of the points are around 0 and the data points seems to be scattered randomly with no specific behavior which means that our model is doing well.

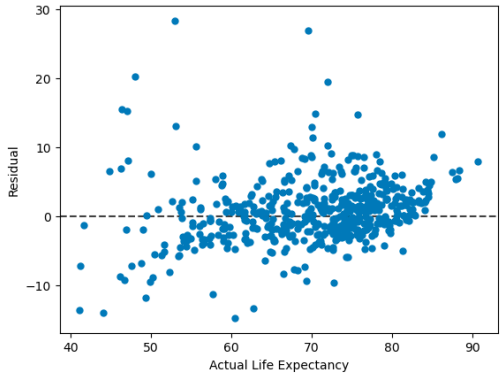

Figure 10: Residual Analysis

RMSE of 4.3 on validation means that the model on average would be off by 4.3 on the validation set. In the dataset context, the model will be off by 4 years on average of all predictions. Are we able to achieve better score with tuning alpha hyperparameter?

5.4.2 Better Performance: Cross Validation

How accurate are the metrics we have used before? We have used a single validation set; can we use more? Cross-validation is a robust method for evaluating the generalizability of machine learning models, offering a more comprehensive assessment than using a single hold-out validation set. This technique divides the dataset into multiple segments or folds, typically in a k-fold cross-validation scheme, where the model is trained on 'k-1' folds and tested on the remaining fold. This process repeats 'k' times, with each fold used once as the test set, ensuring every data point is used for both training and testing.

5.4.3 Tuning Hyperparameter Using Grid Search

Grid search is a method for finding the best hyperparameters for a machine learning model. It exhaustively tests all predefined parameter combinations, evaluating each one's performance. This approach is thorough but can be computationally intensive. It's often paired with cross-validation to ensure robust model evaluation and to avoid overfitting. In our preprocessing pipeline and during the application of regularization, we have identified the optimal model hyperparameters, which are presented in Figure 11:

```
param_grid = {
    'outlier_removal__method': ['mean', 'median'],
    'scaling__method': ['standard', 'minmax'],
    'normal_transformation__method': ['log', 'box-cox'],
    'linear_transformation__multiplier': [1, 2, 3],
    'linear_transformation__addend': [0, 5, 10],
    'lasso__alpha': np.logspace(-4, 4, 10)
}
```

Figure 11: Hyperparameter Tuning

In our analysis, we determined the optimal hyperparameters for our model, which are as follows:

- Lasso regularization strength (alpha): 0.046415888336127774
- Linear transformation parameters: Multiplier = 3, Addend = 0
- Normalization method: Logarithmic transformation
- Outlier removal method: Mean replacement
- Scaling method: Standard scaling

The model achieved a mean R-squared of 0.753 with a mean RMSE of 4.724 on the training data, and it demonstrated comparable performance on the validation data with an R-squared of 0.743 and an RMSE of 4.733.

6. Summary And Future Improvements

In our evaluation of polynomial models, we found that models with degrees higher than 2 tended to overfit, capturing noise along with underlying data patterns due to their high variance. Conversely, models with degrees less than 2 underfit, unable to delineate the essential patterns of the dataset due to their high bias. The degree 2 polynomial model struck an optimal balance, effectively capturing significant patterns without succumbing to the noise, indicating a suitable compromise between bias and variance.

To further refine and enhance our model's accuracy, exploring a broader range of hyperparameters through grid or random search could be beneficial. Considering non-linear models such as decision trees, random forests, or support vector machines might address the complexities of the data more effectively. Additionally, employing neural networks or ensemble methods could potentially capture more sophisticated data patterns. Regularization techniques and advanced feature engineering may also help mitigate overfitting and improve the model's ability to generalize to unseen data, ensuring robust performance across diverse datasets.