Department of Computer Science
Ashoka University

CS-3410 Introduction to Machine Learning

Assignment 1

Submitted by: Anand Agarwal and Charchit Agarwal

---

1 Life Expectancy

Introduction

1. This dataset contains country-wise life expectancy data, along with various factors it can depend on. Below are the variable descriptions.

2. The goal is to develop a regression model that accurately predicts the value of life expectancy based on the different variables provided.

Table 1: Life Expectancy Dataset: Variables and Their Datatype

Variable	**dtype**
Country	object
Year	int64
Status	object
Life expectancy	float64
Adult Mortality	float64
infant deaths	int64
Alcohol	float64
percentage expenditure	float64
Hepatitis B	float64
Measles	int64
BMI	float64
under-five deaths	int64
Polio	float64
Total expenditure	float64
Diphtheria	float64
HIV/AIDS	float64
GDP	float64
Population	float64
thinness 1–19 years	float64
thinness 5–9 years	float64
Income composition of resources	float64
Schooling	float64

Methodology

1. **Data Preprocessing**

   - Since **Country** is like the primary key of the data set to differentiate rows, it does not contribute to the prediction of life expectancy, so it was removed.
   - **Status** It is a categorical variable with two values: Developed and Developing. Life Expectancy was found to be a of a higher range for Developed Countries (See dataexploration.ipynb file). Hence 0-1 encoding was used (0 for developing and 1 for developed countries.
   - **under-five deaths** and **infant deaths** were found to have high correlation (See dataexploration.ipynb). Hence **infant deaths** was removed and **under-five deaths** to ensure greater age coverage.
   - **thinness 1–19 years** and **thinness 5–9 years** were found to have high correlation (See dataexploration.ipynb). Hence **thinness 5–9 years** was removed and **thinness 1–19 years** to ensure greater age coverage.
   - For experimentation and exploration of different parameters, we grouped variables into three indices — **Health Index (under-five deaths, Adult Mortality, BMI, thinness 1–19 years)**, **Economic Index (Income composition of resources, Schooling, Status_binary, Alcohol, percentage expenditure, Total expenditure, GDP, Population)**, and **Disease Index (Hepatitis B, Measles, Polio, Diphtheria, HIV/AIDS)** — by normalizing features within each group and averaging them. These were added as new columns in the dataset.

2. **Algorithm**

   - The data was preprocessed according to the above mentioned steps.
   - **Feature standardization (global).** After selecting the model features (either original features or the constructed indices), we apply z-score standardization:

     $$\widetilde{X} = \frac{X - \bar{X}}{\mathrm{std}(X)}$$

     Note: in the current code the group indices are already z-scored per column in the previous step, and then `load_data(..., scale=True)` standardizes the selected features again — effectively a second standardization step on the indices. This is permissible but should be noted; to avoid repeated scaling either skip the group z-scoring or call `load_data(..., scale=False)` when using already standardized indices.
   - **Train–test split and reproducibility.** Shuffle with a fixed seed (42) and split 80% train / 20% test to ensure reproducible results.
   - **Model formulation.** Let $X \in \mathbb{R}^{n \times p}$ be the feature matrix and $y \in \mathbb{R}^n$ the target. We prepend a bias column (constant 1) producing $\tilde{X} = [\mathbf{1}, X]$ and parameter vector $w \in \mathbb{R}^{p+1}$. The optimization objective used in training is:

     $$L(w) \;=\; \frac{1}{2n}\|\tilde{X}w - y\|_2^2 \;+\; \lambda R(w),$$

     where $R(w)$ is the regularizer:

- OLS: $\lambda = 0$.
- Ridge (L2): $R(w) = \frac{1}{2}\sum_{j=1}^{p} w_j^2$ (bias w_0 is *not* regularized).
- Lasso (L1): $R(w) = \sum_{j=1}^{p} |w_j|$ (bias excluded).

- **Gradient / subgradient.** Using the loss above and excluding the bias from regularization, the (sub)gradient used for updates is:

$$g(w) \;=\; \frac{1}{n}\tilde{X}^\top(\tilde{X}w - y) \;+\; \begin{cases} \lambda \begin{bmatrix} 0 \\ w_{1:p} \end{bmatrix} & \text{(Ridge)} \\ \lambda \begin{bmatrix} 0 \\ \operatorname{sign}(w_{1:p}) \end{bmatrix} & \text{(Lasso, subgradient)} \end{cases}$$

  (For Lasso the subgradient at exactly zero is any value in $[-1, 1]$; gradient descent with a simple sign correction is an approximation. In practice, coordinate descent or proximal methods are more robust for L1.)

- **Optimization (gradient descent).** Using a fixed learning rate η and T epochs, perform full-batch gradient descent:

$$w^{(t+1)} \;=\; w^{(t)} - \eta\, g(w^{(t)}), \quad t = 0, \ldots, T-1.$$

  Implementation details:
  - Initialize $w^{(0)} = \mathbf{0}$.
  - Use a small learning rate (e.g. $\eta = 0.001$ in experiments) and a large number of epochs (e.g. 5000).
  - Do not apply regularization to w_0 (the bias).
  - The implementation currently appends the bias column *before* polynomial expansion; this causes the bias column to also be exponentiated when degree > 1 (constant columns are duplicated). A better order is: expand original features to polynomial terms first, then prepend the bias.

- **Polynomial expansion (optional).** When using polynomial degree $d > 1$ we expand the original features with elementwise powers up to d. In practice the expansion should exclude the bias column to avoid duplicated constant columns.

- **Evaluation metrics.** To measure model quality we compute:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \qquad \text{RMSE} = \sqrt{\text{MSE}}, \qquad R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}.$$

  Metrics are reported on both train and test splits.

- **Model persistence and outputs.** Trained models are saved (pickled). For Ridge and Lasso we also export learned weights to CSV alongside feature names. Final metrics for each model (train/test) are written to a text file.

Experimentation

To investigate the effect of different feature groupings and the usefulness of aggregated indices, we conducted a series of controlled experiments. Each experiment trained regression models using the same optimization framework (gradient descent with OLS, Ridge,

and Lasso regularization), but with varying feature inputs. This design allowed us to assess whether compact, interpretable indices could capture sufficient predictive signal compared to using all raw features.

1.0.1 Experimental Design

The experiments are structured around four progressively complex feature sets:

1. **Model 1 (Economic & Health Indices):** Uses the aggregated *Economic Index* and *Health Index* as predictors. This experiment evaluates whether socioeconomic and health-related aggregates alone can explain life expectancy variation.
2. **Model 2 (Disease & Economic Indices):** Uses the *Disease Index* alongside the *Economic Index*. This experiment focuses on the joint role of disease burden and economic development factors.
3. **Model 3 (All Indices):** Combines *Health Index*, *Economic Index*, and *Disease Index*. This experiment assesses whether combining all three interpretable aggregates improves predictive power without resorting to the full feature set.
4. **Model 4 (Full Feature Set):** Uses all available numerical features individually, without aggregation into indices. This acts as the baseline to compare against the index-based models, providing insight into the trade-off between interpretability and predictive performance.

Results

1.0.2 Model 1: Health Index + Economic Index

Table 2: Regression Metrics for Model 1

Model	Dataset	MSE	RMSE	R^2
OLS	Train	43.03	6.56	0.52
	Test	41.46	6.44	0.56
Ridge	Train	43.32	6.58	0.51
	Test	41.67	6.46	0.56
Lasso	Train	43.03	6.56	0.52
	Test	41.45	6.44	0.56

1.0.3 Model 2: Health Index + Disease Index

Table 3: Regression Metrics for Model 2

Model	Dataset	MSE	RMSE	R^2
OLS	Train	68.11	8.25	0.24
	Test	73.45	8.57	0.22
Ridge	Train	68.30	8.26	0.23
	Test	73.01	8.54	0.22
Lasso	Train	68.11	8.25	0.24
	Test	73.43	8.57	0.22

1.0.4 Model 3: Health Index + Disease Index + Economic Index

Table 4: Regression Metrics for Model 3

Model	Dataset	MSE	RMSE	R^2
OLS	Train	42.70	6.53	0.52
	Test	41.51	6.44	0.56
Ridge	Train	42.99	6.56	0.52
	Test	41.73	6.46	0.56
Lasso	Train	42.70	6.53	0.52
	Test	41.49	6.44	0.56

1.0.5 Final Model: Features Taken Individually

Table 5: Regression Metrics for Final Model

Model	Dataset	MSE	RMSE	R^2
OLS	Train	17.13	4.14	0.81
	Test	16.75	4.09	0.82
Ridge	Train	17.25	4.15	0.81
	Test	16.19	4.02	0.83
Lasso	Train	17.13	4.14	0.81
	Test	16.71	4.09	0.82

Explanation

- The results indicate a gradual improvement in model performance as more grouped indices are included, with accuracy increasing from combinations of two indices to all three.
- However, when individual features are used instead of grouped indices, the model achieves its highest accuracy ($R^2 \approx 0.80$), suggesting that grouping features leads to information loss.
- This highlights a clear tradeoff: grouped indices simplify the model but reduce predictive accuracy, while using all individual features increases complexity yet provides the most reliable results.

2 Laptop Price

Introduction

1. This dataset contains company wise specifications for different Laptops.
2. The goal is to develop a regression model that accurately predicts the price of the laptop based on the different specifications provided.

Table 6: Laptop Price Dataset: Variables and Their Datatype

Variable	**dtype**
Company	object
TypeName	object
Inches	float64
ScreenResolution	object
Cpu	object
Ram	object
Memory	object
Gpu	object
OpSys	object
Weight	object
Price	float64

Methodology

1. **Data Preprocessing** The raw laptop dataset underwent a series of preprocessing steps to standardize and extract meaningful features. The following steps were performed:

   - **CPU Feature Encoding:**
     - Mapped each CPU model to a performance score obtained from the online Geekbench Processor Benchmark dataset. This converted categorical data to float.
     - CPU Models which weren't in the dataset were given average values collected from similar CPU series.
   - **Memory Parsing:**
     - Extracted storage capacities for SSD, HDD, and Flash storage types using regular expressions.
     - Converted all storage capacities to gigabytes (GB) for uniformity.
     - Represented each laptop with three numerical columns: SSD, HDD, and Flash, containing the number of GBs of each storage type
   - **Screen Feature Engineering:**
     - Derived binary indicators for touchscreen, IPS panel, 4K display, and HD resolution.
     - Extracted screen width and height in pixels from the resolution strings and represented them in their own columns.
   - **Other Numerical Features:**
     - Cleaned the RAM column by removing the "GB" suffix and converted values to integers.
     - Standardized the weight column by stripping the "kg" suffix and converting values to floating-point numbers.
   - **Categorical Feature Encoding:**

  - One-hot encoded the operating system and laptop type columns, dropping the first category to prevent multicollinearity. This will help identify if "Gaming" laptops, which usually contain better specs, are valued more than just because of labelling.
  - Transformed the GPU column into a binary indicator representing entry-level GPUs. For example, those GPUs which are default and subsequently have poor performance (Intel HD Graphics, etc.) were separated from those like Nvidia.
- **Final Cleanup:**
  - Dropped redundant columns such as Memory, ScreenResolution, Company, GPU, OpSys, and TypeName (since we cretaed other relevant columns from these).
  - Removed rows with missing values to ensure model robustness.
  - Standardized all numerical features to have zero mean and unit variance.

2. **Algorithm** - Same as Life Expectancy task algorithm

Experimentation

To understand the effect of preprocessing strategies and feature engineering on model performance, we designed a series of controlled experiments. Each experiment applied the same regression framework (OLS, Ridge, and Lasso with gradient descent), while varying the feature transformations and data preparation steps. This approach allowed us to compare the benefits of scaling, encoding, and engineered variables against using raw inputs.

2.0.1 Experimental Design

The experiments are structured around four models with increasing refinement of features:

(a) **Model 1 (One-Hot Encoded Features):** Uses the dataset with categorical variables one-hot encoded but without standardisation of numerical features. This experiment isolates the effect of encoding while allowing us to test how the model behaves without feature scaling.

(b) **Model 2 (Standardised Features):** Applies standardisation to numerical features (e.g., Inches, Weight, Price). This experiment tests whether scaling improves model convergence and predictive accuracy.

(c) **Model 3 (Encoded + Standardised Features):** In previous model, we erroneously standardized one-hot encoded variables. This error was corrected in this model.

(d) **Model 4 (Engineered Features):** Builds on Model 3 by introducing improving the avg CPU score given to CPU models which aren't present in the dataset.

Results

2.0.2 Model 1: No Standardisation Applied

Table 7: Regression Metrics for Model 1

Model	Dataset	MSE	RMSE	R^2
OLS	Train	0.30	0.55	0.69
	Test	0.30	0.54	0.74
Ridge	Train	0.30	0.55	0.69
	Test	0.30	0.54	0.74
Lasso	Train	0.30	0.55	0.69
	Test	0.30	0.54	0.74

2.0.3 Model 2: Standardisation Applied to All Variables

Table 8: Regression Metrics for Model 2

Model	Dataset	MSE	RMSE	R^2
OLS	Train	0.24	0.49	0.75
	Test	0.24	0.49	0.79
Ridge	Train	0.24	0.49	0.75
	Test	0.24	0.49	0.79
Lasso	Train	0.46	0.68	0.52
	Test	0.49	0.70	0.57

2.0.4 Model 3: Standardisation Removed from One-Hot Variables and Hyperparameters Tweaked

Table 9: Regression Metrics for Model 3

Model	Dataset	MSE	RMSE	R^2
OLS	Train	0.24	0.49	0.75
	Test	0.24	0.49	0.79
Ridge	Train	0.24	0.49	0.75
	Test	0.24	0.49	0.79
Lasso	Train	0.25	0.50	0.74
	Test	0.24	0.49	0.79

2.0.5 Final Model: Tweaked Average CPU Values

Table 10: Regression Metrics for Final Model

Model	Dataset	MSE	RMSE	R^2
OLS	Train	0.19	0.43	0.82
	Test	0.12	0.34	0.86
Ridge	Train	0.19	0.43	0.82
	Test	0.12	0.34	0.86
Lasso	Train	0.19	0.44	0.82
	Test	0.12	0.35	0.85

2.1

Explanation of Results

- Model performance improves consistently when standardisation and feature preprocessing are applied, as seen in Model 2 and Model 3.
- Ridge and OLS regressions show very similar metrics across all models, indicating low multicollinearity and stable predictions.
- Lasso regression is more sensitive to feature scaling and hyperparameters, leading to slightly lower performance in some models.
- The Final Model with tweaked average CPU values achieves the best test performance, demonstrating the impact of careful feature engineering.
- Overall, applying standardisation to numeric variables and appropriate feature adjustments improves model generalisation and R^2 scores.

3 Retail

Introduction

(a) This dataset contains consumer purchase capacity and product type data along with various factors it can depend on. Below are the variable descriptions.

(b) The goal is to develop a regression model that accurately predicts the value of average purchase value based on the different variables provided. The variable types have been excluded due to their numerous size. However you can find it in dataexploration.ipynb

Methodology

(a) **Data Preprocessing** The raw retail dataset underwent multiple preprocessing steps to prepare it for modeling. The following steps were performed:

- **Feature Dropping:**
  - Removed transaction-related identifiers and temporal variables such as `transaction_id`, `transaction_date`, and `transaction_hour`, which do not contribute predictive value.
  - Dropped redundant geographic details such as `customer_zip_code`, `customer_city`, `store_zip_code`, and `store_city`.
  - Excluded seasonal variables like `season` and other unused attributes.
  - Removed all columns beginning with `product` and `promotion`, as they were deemed irrelevant for the chosen feature set.
- **Categorical Feature Encoding:**
  - Applied one-hot encoding to categorical attributes such as `gender`, `income_bracket`, `marital_status`, `education_level`, `occupation`, `payment_method`, `store_location`, and others.
  - This expanded categorical variables into multiple binary indicators, ensuring compatibility with regression models.
- **Binary Feature Encoding:**
  - Converted boolean-style features such as `email_subscriptions`, `weekend`, `holiday_season`, `churned`, and `loyalty_program` into numerical 0/1 representations.
  - This allowed the model to interpret customer behaviors and program participation as quantitative signals.
- **Numerical Standardization:**
  - Standardized the target variable `avg_purchase_value` to have zero mean and unit variance using Z-score normalization.
  - This ensures that numerical features are on a comparable scale, preventing dominance of variables with larger magnitude.
- **Final Cleanup:**
  - Ensured that the dataset contains only the relevant encoded features and standardized numerical values.
  - Removed redundant or unused variables to improve model efficiency and interpretability.

(b) **Algorithm** Same as Life Expectancy task

Experimentation

Firstly, because the dataset was large, it was time consuming to train the model for a high number of epochs. We reduced the number of epochs to 1000. However, the features we chose were not good indicators of our target variables and we got very low accuracy.

Hence, for our second model, we used polynomial regression with degree 2, but it failed to offer any meaningful improvement either.

Results

3.0.1 Model 1

Table 11: Regression Metrics for Model 1

Model	Dataset	MSE	RMSE	R^2
OLS	Train	20151.05	141.95	-0.01
	Test	20219.81	142.20	-0.01
Ridge	Train	20151.08	141.95	-0.01
	Test	20219.71	142.20	-0.01
Lasso	Train	20151.52	141.96	-0.01
	Test	20219.63	142.20	-0.01

3.0.2 Model 2

Table 12: Regression Metrics for Model 2

Model	Dataset	MSE	RMSE	R^2
OLS	Train	20151.05	141.95	-0.01
	Test	20219.81	142.20	-0.01

Challenges

During the course of this study, several challenges were encountered in preparing and modeling the dataset.

- **Data Standardization**
  Challenge: The dataset contained numerical variables with varying ranges (e.g., RAM in GB, weight in kg, pixel resolution in thousands). This scale disparity could bias model training.
  Solution: We applied **Z-score standardization** to rescale all numerical features to a common scale with zero mean and unit variance, ensuring fair contribution of each variable during model optimization.
- **Categorical Variables**
  Challenge: Many features, such as operating system type and laptop form factor, were categorical and could not be directly used in regression-based models.
  Solution: We employed **one-hot encoding** to convert categorical variables into binary dummy variables, preserving their information while making them compatible with numerical modeling.
- **Regularization**
  Challenge: The risk of overfitting necessitated the use of regularization. However, selecting the appropriate method—**Lasso** (L1) for feature selection or **Ridge** (L2) for coefficient shrinkage—was non-trivial.
  Solution: We systematically tested both approaches across multiple model

runs and compared performance metrics. This allowed us to identify the most effective regularization strategy for our dataset.

- **Model Complexity**
  Challenge: The features that we chose to model our situation were apparently not appropriate. We tried to choose a range of different features to improve prediction score. But we were not able to make improvements. Part of the problem also stemmed from not understanding the nature of the column "avg_price_value" as this value could have been calculated simply by the data provided in the table itself, not something to predict. Furthermore, we also noted inconsistencies within the data itself, such as customers having buying frequency labelled as "Daily", but having their last purchase over a 300 days ago. This weren't uncommon and convulted our understanding of the dataset

References

- https://browser.geekbench.com/processor-benchmarks.json
- Lecture Slides For Introduction To Machine Learning - Prof Lipika Dey - Monsoon 2025