19CSE305 - Machine Learning

Project Phase II - Documentation

4th November 2021

Machine Learning Project Phase-II

Life Expectancy Predictor

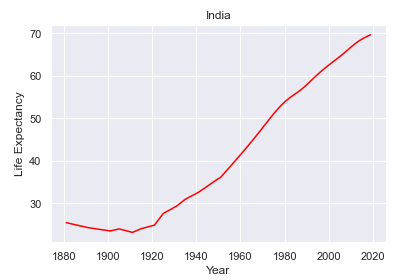
**By**

**Adithya Nair - AM.EN. U4CSE19103**

# Problem Description

Life expectancy has become one of the most frequently used health status indicators. It is a measure that **summarizes the mortality of a country**, allowing us to compare it by generation and analyse trends.

Our aim is to train different models that are capable of learning the important features which lead to life expectancy rates. The model chosen is the one that is able to predict the life expectancy rates with the highest accuracy.





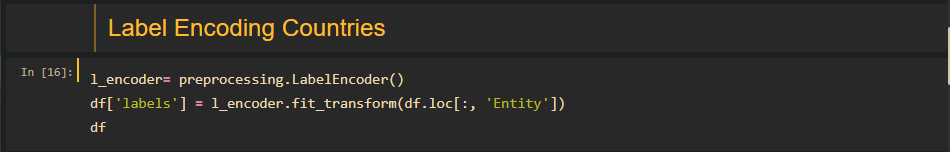
# Dataset

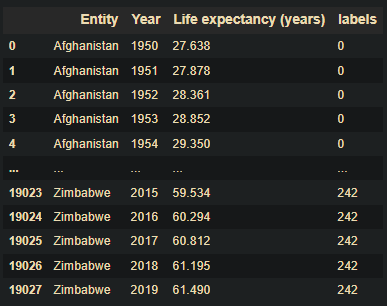
|  |  |
| --- | --- |
| **Name/Link** | **Description (Format)** |
| Adithya Nair ([Link](https://www.kaggle.com/prateekmaj21/lifeexpectancy)) | 1. Data of life expectancy across various countries of the world. Found to know about the lifestyle nature in countries. 2. Data-set consists of 4 columns, which includes: Country names, Years, Country code, Average life expectancy   Importance: Dataset gives us information about the average life expectancy of different countries over a couple of years.   1. Data-set was used to summarize life expectancy for 69 years of each country provided. |

# Data Preparation

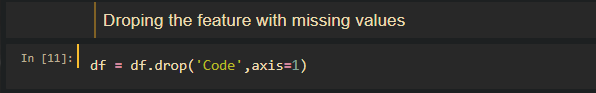
## Data Pre-processing

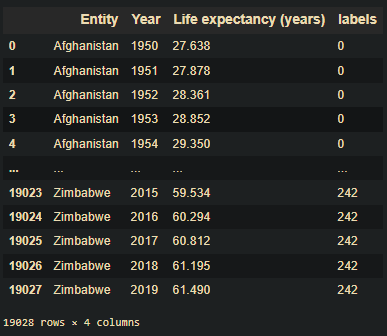
Dataset contains some features such as country names, country code, and life expectancy. These were replaced with integer values using the **Label Encoding** method.



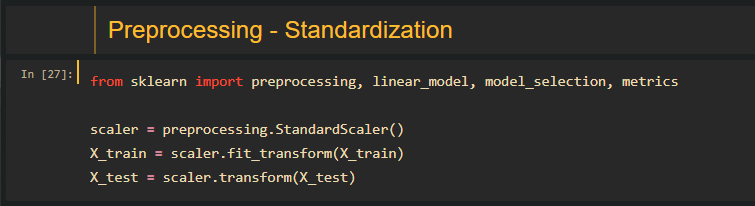


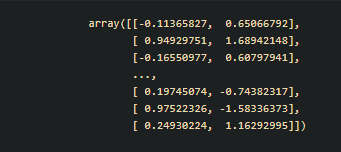
Feature ‘Code’ contains **`nan`** parameters, so they are using the ‘drop’ method. we remove the entire column as it contained with high counts of such values.





Since some features have vastly varying values, we standardize the values using the **StandardScale** method.





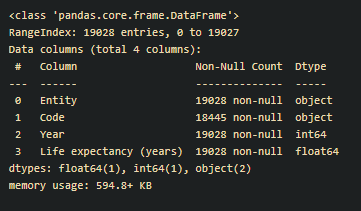
## Summarization

Dataset can be summarized into four columns containing the **Entity, Code, Year,** and the corresponding **Life Expectancy** rate. The entire data consists of about 19028 entries which are non-null. The feature ‘**Entity’** follows a string data type, ‘**Year’** follows integer format, and ‘**Life Expectancy’** follows the floating-point format.

Dataset dimensions consist of 19028 rows and 4 columns. (19028 x 4)

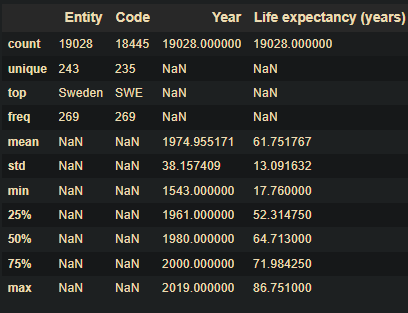






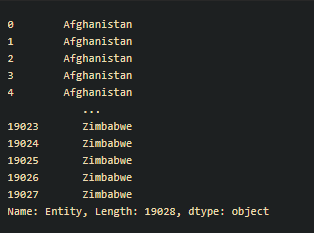


The **describe()** method is used for calculating some statistical data like **percentile, mean**, and **std** of the numerical values of the dataset. The statistical summary indicates that the dataset consists of 15 different unique countries.

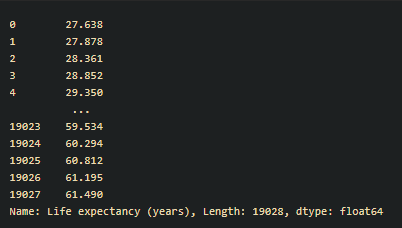


**Breaking down each of the column class variables:**



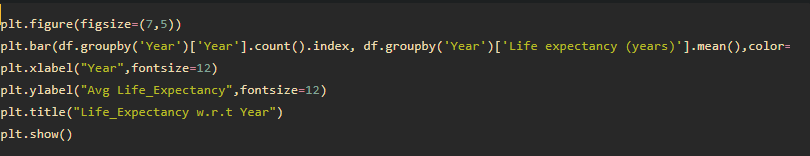


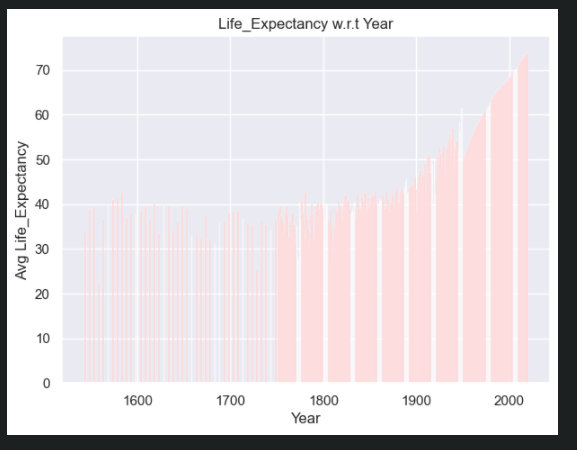


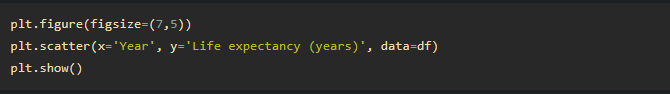


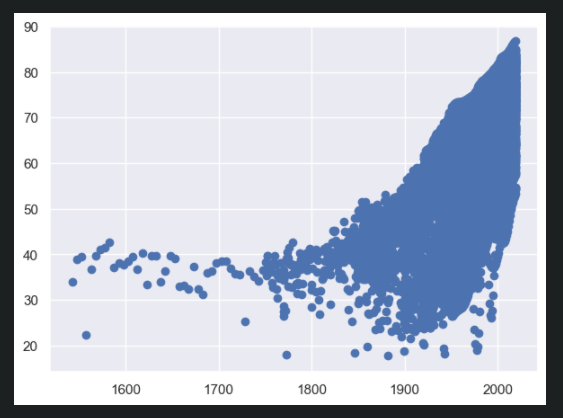
## Data Visualization

Different types of graph showing Life Expectancy w.r.t Years are given below:

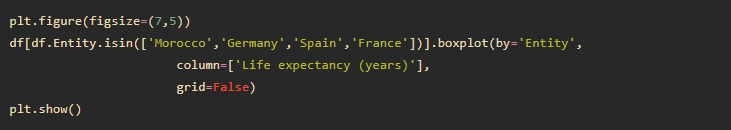


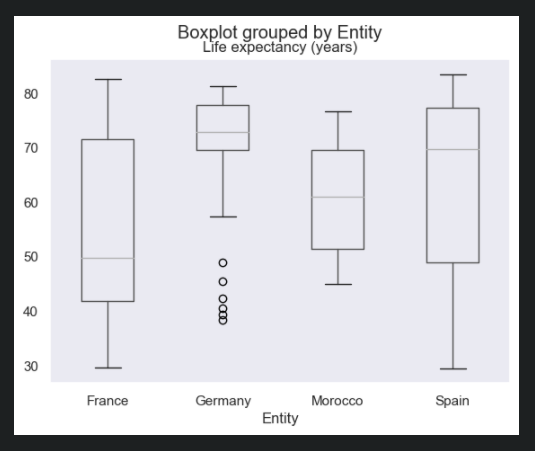




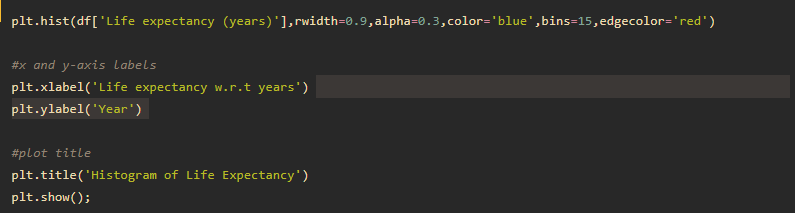


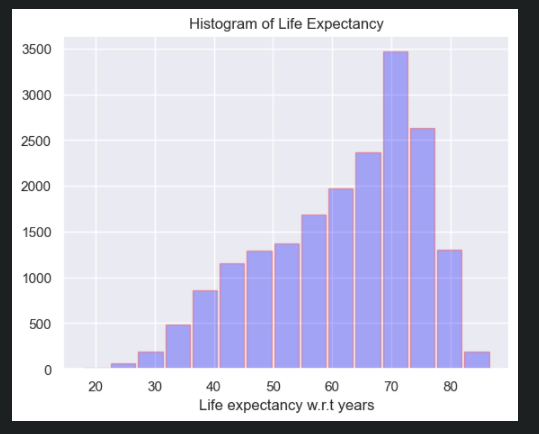
**Next is the boxplot of life expectancy of some countries like Morocco, Germany, Spain and France**



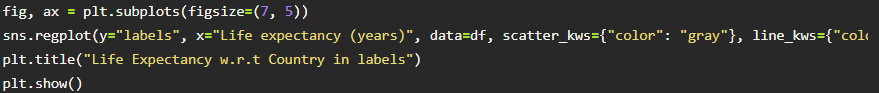


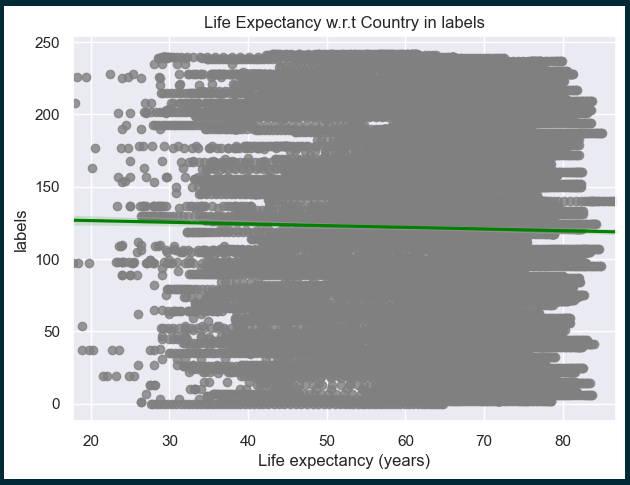
**Here is a histogram showing Life expectancy w.r.t years:**





Next is aregplot from **seaborn** library showing a scatter plot for life expectancy w.r.t respective countries.

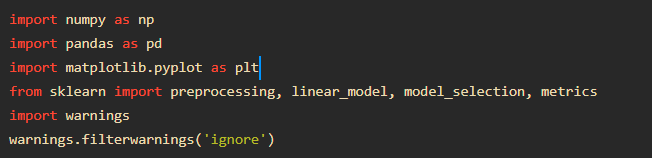




# Python Packages

Python packages used in the dataset are:

* **NumPy**: NumPy is a Python library used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices.
* **Pandas**: Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is used for cleaning, transforming, manipulating, and analysis of data.
* **Matplotlib**: Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot.
* **Sklearn**: Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.



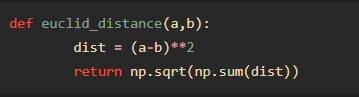
# Supervised/unsupervised Learning Algorithms

## K Nearest Neighbours (KNN) Algorithm - Supervised:

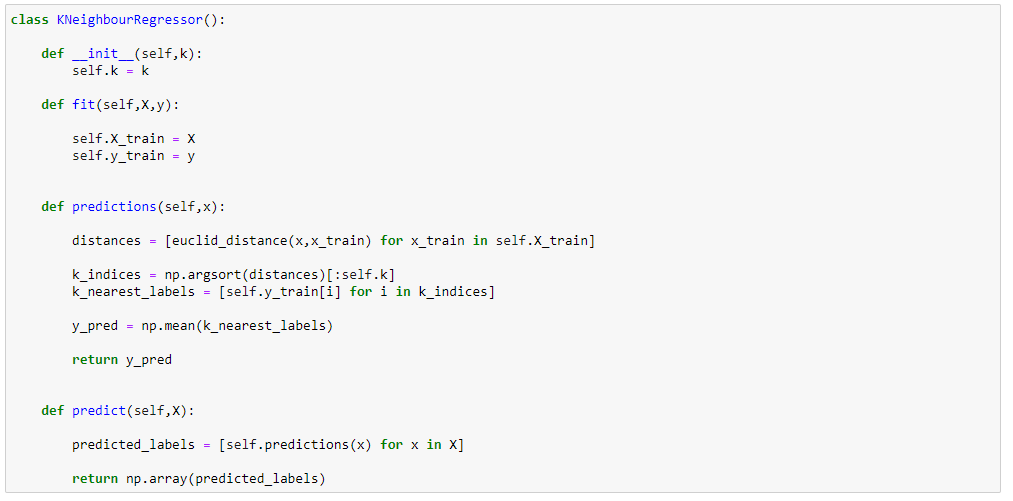
The KNN algorithm finds K closest values to an input and gives an approximate guess of the correct answer to the input. In this algorithm we are using Euclidean Distance Function to calculate the distance of each entry & mean to predict the solution.

The major down-side to the algorithm is that it will have to process the entire database for every out. Although the algorithm is inefficient & naive. The KNN algorithm sets a basic level of accuracy for the forthcoming algorithms.

Below is the scratch code for the implementation of KNN Regression Algorithm.



Above is the function to calculate Euclidean distance from the neighbours.





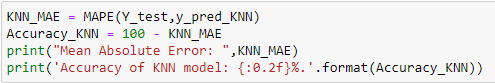














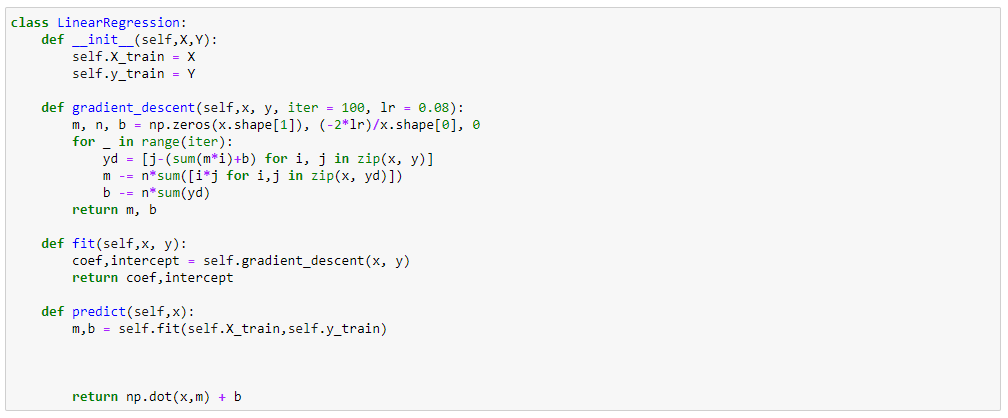
KNN Regressor gives 86% accuracy.

## Linear Regression Algorithm - Supervised:

The Linear Regression Algorithm finds the ideal line that represents the given instances and predicts the output of an unknown instance based on where the unknown value falls on the line. This is done by minimising the distance from all instances to an arbitrary line. The distance is calculated by the Mean Squared Error Function (MSE). And the minimisation is done by Gradient Descent Algorithm.

A major downside to Linear Regression is that an extreme instance could affect the regression algorithm. This downside is rectified early on by standardizing the data, thus removing all extreme data.

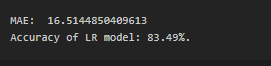
Below is the scratch code for the implementation of Linear Regression Algorithm.







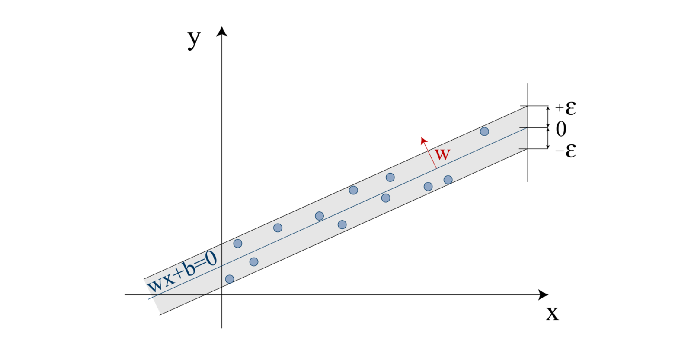




## Support Vector Regression Algorithm - Supervised:

In machine learning, Support Vector Machines are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. In Support Vector Regression, the straight line that is required to fit the data is referred to as hyperplane.

The objective of a support vector machine algorithm is to find a hyperplane in an n-dimensional space that distinctly classifies the data points.



Functionality of the Support Vector Regression —Image by the author (inspired by [Smo04])

The data points on either side of the hyperplane that are closest to the hyperplane are called Support Vectors. These influence the position and orientation of the hyperplane and thus help build the SVM.

Below is the scratch code for the implementation of SVR Algorithm using inbuilt functions:

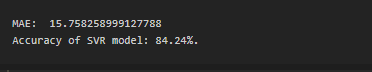
Here I imported the SVR inbuilt function from Sklearn library of python.







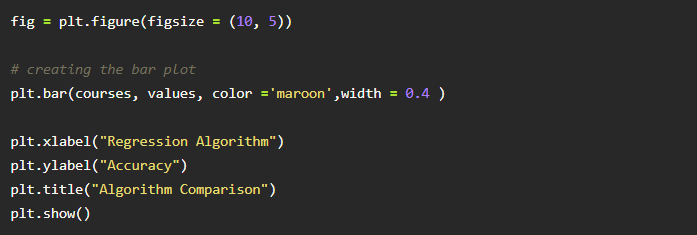


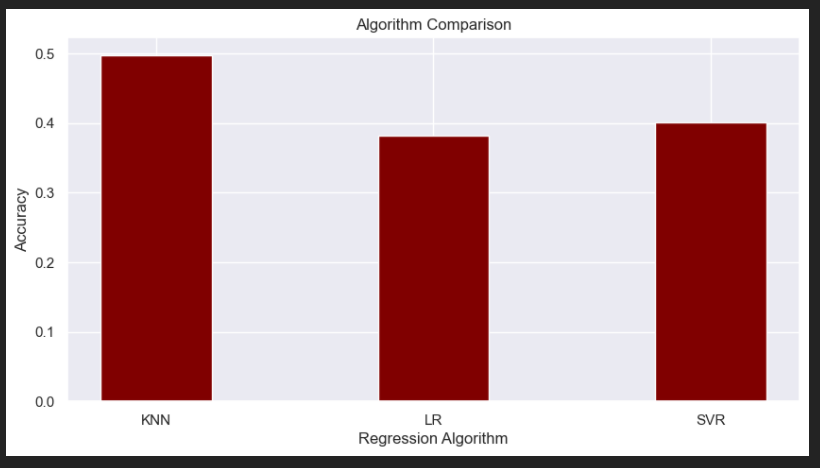


**Conclusion:**

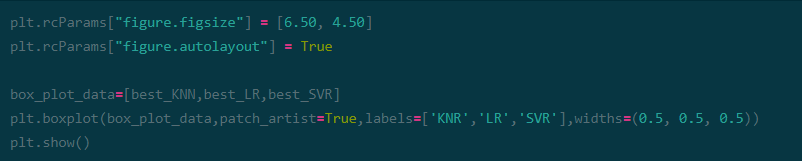
|  |
| --- |
| **From the three different models, it can be inferred that the KNN model gives the highest accuracy and performs much better when compared to the other models.** |

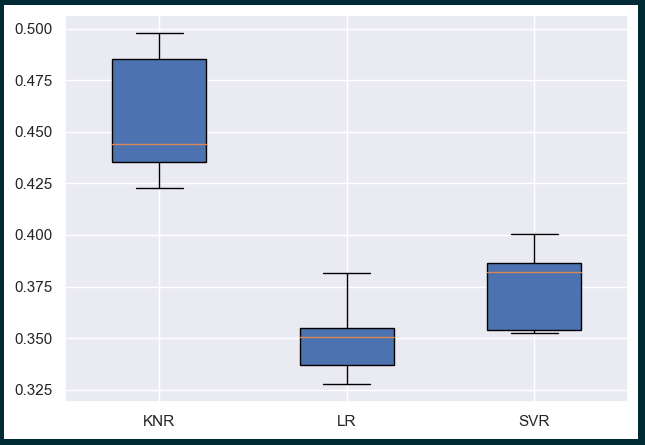
# Graph showing the accuracy comparison of various algorithms





Using boxplot for comparison:





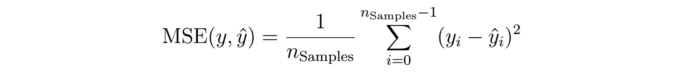
**On comparing different regression algorithms, we see that KNN regressor produces more accuracy than rest of the algorithms.**

# Evaluate the analysed dataset with proper metrics

There are various methods and procedures to evaluate the accuracy of a model.

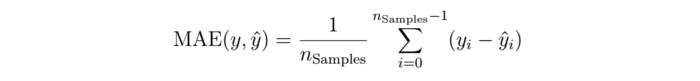
#### Metric Functions

The sklearn. metrics module includes several loss and evaluation functions to measure the quality of the regression models. *Mean Squared Error (MSE)* is a key criterion for assessing the quality of a regression model. If yˆ\_i describes the value predicted by the model at the i-th data sample, and y\_i describes the corresponding true value, then the *Mean Squared Error (MSE)*of the model over samples.



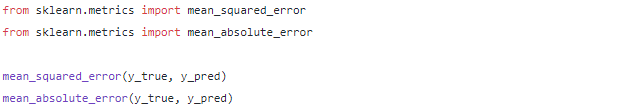
Another parameter for determining the accuracy of regression models is the *Mean Absolute Error*

*(MAE).*



Both metrics can be found in the module Sklearn.metrics. They compare the predicted and actual values for the test dataset.

**Code**:

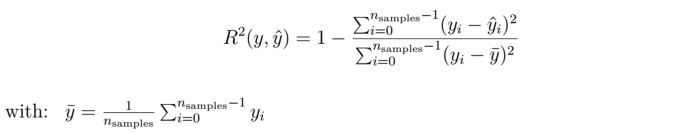


#### Coefficient of determination (R²)

The so-called coefficient of determination (R²) can be understood as a standardized version of the MSE. This allows an easier interpretation of the performance of the model. The best possible performance is described with the value 1.0.

The R² -score can also become negative if the model shows arbitrary deviations from the truth value. A constant model, which makes the prediction of the values without the consideration of the input characteristics, would receive a R² -score of 0.0.

If yˆ\_i describes the value predicted by the model at the i-th data sample, and y\_i describes the associated true value, then the coefficient of determination R² over n\_Samples is defined as:



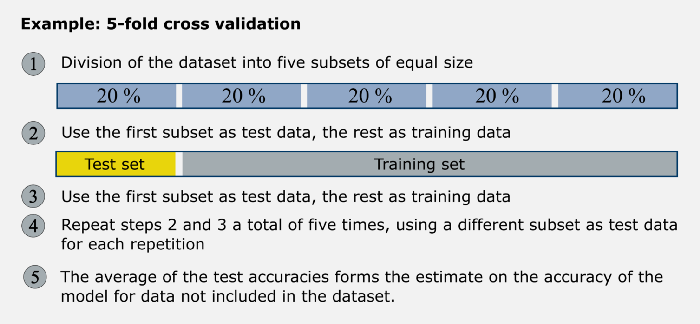
The output in Python is the function r2\_score , where y\_true is the true value of the dependent variable and y\_pred is the value predicted by the model.

### **Cross validation in regression**

Cross-validation is a statistical method for model selection. To evaluate a method, the entire dataset is divided into a training and a test dataset, whereby the training dataset usually comprises 80 to 90 % of the entire dataset. In order to achieve the best possible evaluation of the model, the aim is to have as large a test dataset as possible. Good model building is achieved by having as large a training dataset as possible.

Cross-validation is used to circumvent this dilemma. This method allows the entire dataset to be used for both training and testing. Compared to a fixed division into train and test data, cross-validation thus allows a more accurate estimate of model accuracy for future data or data not included in the dataset.

The k-fold cross validation divides the entire dataset X into k equal sized blocks (X\_1, …, X\_k). Then the algorithm is trained k times on k-1 blocks and tested with the remaining block.



Many learning methods allow an adjustment of the model complexity via one or more hyperparameters. This often leads to the problem of over- or underfitting. Cross-validation is used to find the optimal model complexity. The optimal complexity is achieved by minimizing the approximation error on a test dataset that is unknown during learning.

### Test Train Split

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

The train-test split procedure is appropriate when you have a very large dataset, a costly model to train, or require a good estimate of model performance quickly.

**Train Dataset:** Used to fit the machine learning model.

**Test Dataset:** Used to evaluate the fit machine learning model.

**Code:**

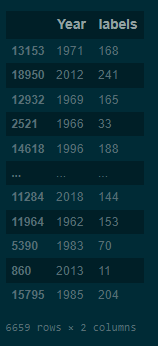
**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, y, test\_size=0.65, random\_state=42)**

The train\_test\_split function is imported from Sklearn library in python.

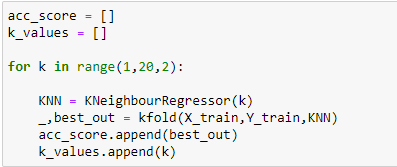
In my code, I split the train and test size into 0.65 of test size and rest for training from the dataset.

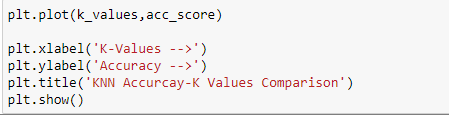


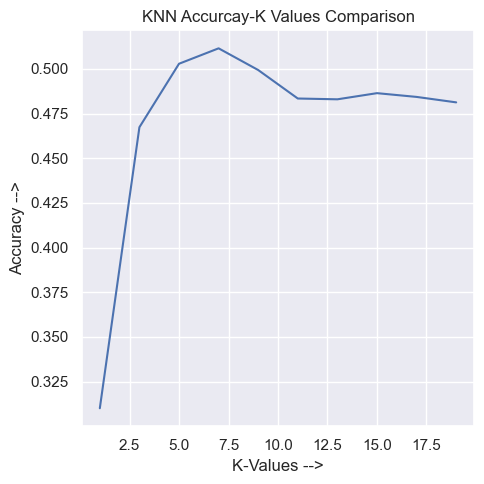


Make Predictions on validation dataset. Plot accuracy and time for varying parameters.

Plot accuracy for different values of k.







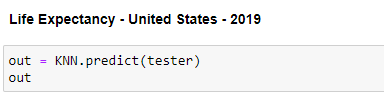
From the above model, we can see that the accuracy increases for KNN regressor till 7 neighbours and then it decreases as the number of neighbours increases.

## Predicting new data with the best model:





Here we are passing the year and country label for predicting new data.



Predicted the life expectancy as 77 for United States.