Machine Learning

Course Project Report

(Phase-I)

Title of the project: Abalone

Student : Bhavani Pediredla and bhavani.p-26@scds.saiuniversity.edu.in

ML Category: Regression

---

1. Introduction

In this project , we aim to solve a regression problem using the Abalone dataset. The main objective is to predict the age of Abalones based on various physical measurements. The age of abalones is determined by counting the number of rings in their shell , where each ring corresponds to approximately 1.5 years of age . This project involves exploring firrent regression algorithms to find the most accurate model for predicting the number of rings in an abalone. The prediction of abalone age is valuable for ecological and commercial purposes. Understanding the age distribution of abalone populations can help in conservation efforts and sustainable harvesting.

2. Dataset and Features

The abalone dataset consists of 4,177 samples with 8 numerical features and a target variable , each representing an individual abalone with multiple physical measurements. The dataset aims to predict the age of abalone by using the number of rings as the target variable.

<u>Features:</u>

1. Sex : Categorical features with three subcategories - male , female and infant
2. Length: Measured in millimetres , representing the longest shell measurement
3. Diameter : Measured in millimetres, representing the shell width perpendicular to length
4. Height : Measured in millimetres , representing the height with meat in the shell
5. Whole weight : Measured in grams , representing the weight of the whole abalone
6. Shucked weight : Measured in grams , representing the weight of the abalone meat
7. Viscera weight : Measured in grams , representing the gut weight after bleeding shell
8. Shell weight : Measured in grams , representing the weight of the shell after drying

<u>Target Variable :</u>

1. Rings

There are no missing values in this dataset; as such, the data is complete and reliable. By analysing the different features in the abalone dataset, we shall aim at developing a regression model that would provide an exact estimation of age.

3. Methods

The entire dataset is further split into **75% training set and 25 % testing set**:

- Training Dataset: Consisting of 75% of the original dataset, it includes :
  - **X_train**: Training data for the first 8 features.
  - **y_train:** Training data for the target variable (Rings).
- Testing Dataset: Comprising the remaining 25% of the original dataset, it includes :
  - **X_test:** Testing data for the first 8 features.
  - **y_test:** Testing data for the target variable (Rings**).**

3.1 Baseline - Linear Regression

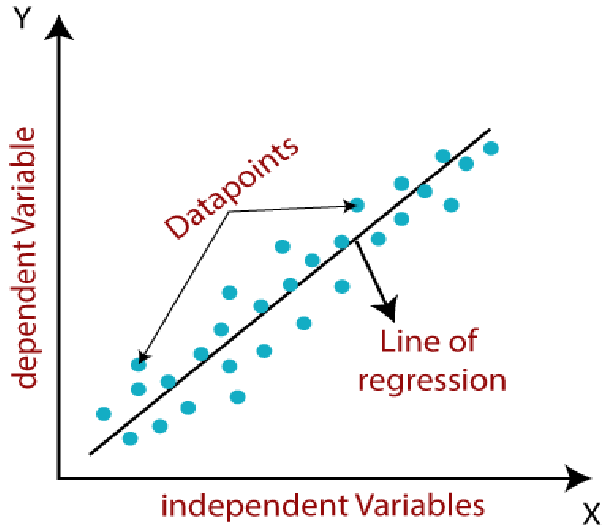
The simplest and most applicable algorithms in machine learning have to be linear regression. In nature, it's a statistical technique applied to the prediction of an analysis; hence, quite fitting for the prediction of continuous or numeric variables such as sales, salary, age, and price of the product.

The algorithm of linear regression models a linear relationship between a dependent variable (y) and one or more independent variables (x); hence, it is called linear regression. This model delineates how changes in the independent variable(s) affect the dependent variable.

It is a type of supervised machine learning algorithm whereby linearity regression computes a linear relationship through fitting a linear equation to the observed data.
Mathematically, we can represent a linear regression as:

$$\mathbf{Y} = a_0 + a_1\mathbf{X} + \varepsilon$$

Y = Dependent Variable (Target Variable)
X = Independent Variable (Predictor Variable)
a_0= Intercept of the line (Gives an additional Degree of freedom)
a_1= linear regression coefficient (scale factor to each input value)
ε = random error
The values for x and y variables are training datasets for Linear Regression model representation.



- Results : The steps involved in creating and training the linear regression model as follows :
    1. Data Preparation :
        - ➔ The data is split into 75% training and 25 % testing sets . the training set is used to train the model while the testing set is used to evaluate its performance
        - ➔ Features scaling is performed to standardise the input features, ensuring that they have mean of zero and standard deviation of one . this is crucial for algorithms like linear regression to perform optimally
    2. Model Training:
        - ➔ A linear regression model is created using the '**linear regression**' class front he **Scikit- learn library**
        - ➔ The model is trained on the scaled training data (**X_train_scale** and **y_train**)
    3. Prediction:
        - ➔ After training the model makes predictions on both the training data (**X_train_scale**) and the testing data (**X_test_scale**)
    4. Performance Evaluation :
        - ➔ Mean Squared Error (MSE): Measures the average squared difference between observed and predicted values

          MSE train : 4.8526

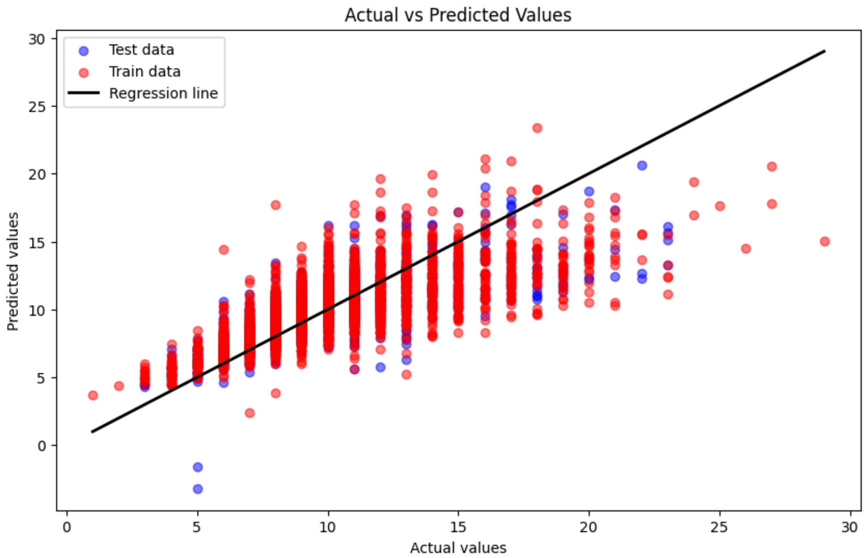
          MSE test : 4.6759
        - ➔ R - Squared (R^2) : Indicates the proportion of the variance in the the dependent variable that is predictable from the independent variable

          R^2 **Train : 0.5345**

          R^2 **Test : 0.5455**

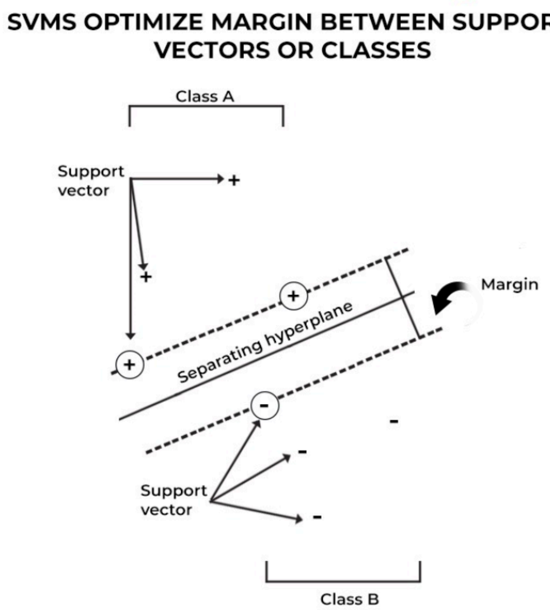
➔ Mean Absolute Error (MAE) : Measures the average magnitude of the errors in a set of predictions, without considering their direction

MAE Test : 1.5694



3.2 Support Vector Machines

SVM acts as one of the best learning algorithms that help in classification, regression, and outlier detection in machine learning. In this process, it finds the best decision boundaries that would allow the separation of different classes in the data. SVMs can be pretty effective in handling complex classification problems by carrying out a transformation of the input into higher-dimensional space, consequently making it significantly easier to draw clear boundaries between the different classes. Indeed, the SVM performs optimal data transformations, which in turn help to define boundaries between data points according to predefined classes, labels, or outputs. This makes them widespread in their applicability: from health to Natural Language Processing, Signal Processing, Speech Recognition, to Image Recognition. The main purpose of the SVM algorithm is to find a hyperplane that distinctly separates data points of different classes. This hyperplane is placed so as to maximise the margin between the classes and ensures maximum separation of points from classes. This maximised margin helps in improving the model's capacity for generalisation and, hence, in being more robust for new, unseen data.



Characteristics of SVM :

1. Supervised Learning: SVMs use labelled training data to learn the decision boundaries.
2. Hyperplane: It is the decision boundary separating data points of different classes as optimally as possible. In two dimensions, it's a line; in higher dimensions, it becomes a plane or hyperplane.
3. Support Vectors: Data points lying closest to the decision boundary, which are relevant in defining the position and orientation of the Hyper-Plane.
4. Kernels: There are a lot of different kernel functions SVM can use, such as linear, polynomial, radial basis function (RBF), among others, which transform the data

into a higher dimensional space in which it becomes easy to separate classes linearly.

- Steps involved in SVM :

1. Initialise an empty dictionary 'result_dict' to store evaluation metrics for each SVM kernel type
2. Iterate over each kernel type ('linear','poly','rbf')
    - For each kernel instantiate an SVR model ('model') with the current kernel type ('kernel')
    - Train the SVR model ('model') using scaled training data ('X_train_scle' , 'y_tain')
    - Generate predictions ('pred_svm') on the scaled test data ('X_test_scale')
    - Compute and store the following metrics in 'result_dict' for the current kernel type :
        I. MSE
        II. R2
        III. MAE
3. Convert 'result_dict' to a dataframe as ['MSE','R2 ', ' MAE']
4. Print the resulting dataframe ('result') showing the metrics for each SVM kernel type

- Results :

These are the commonly used kernels in SVM, along with the results obtained from applying these kernels to my dataset:

➢ Linear Kernel:

The linear kernel is the simplest form, ideal for situations where the data is linearly separable. It calculates the dot product between the features vectors

Values :

MSE : 4.788017

R^2 **: 0.534574**

MAE : 1.526880

➢ Polynomial Kernel:

The polynomial Kernel is useful for non - linear data. It measures the similarity between two Vectors based on a Polynomial function of the original features .

Values:

MSE : 5.423970

R^2 **: 0.472755**

MAE : 1.547286

- Radial Basis Function (RBF) Kernel:

  The RBF Kernel is widely used in SVM for dealing with non - linear decision boundaries . It transforms the data into an infinite Dimensional Space , allowing the algorithm to handle complex data patterns

  Values:

  MSE : 4.627016

  R^2 **: 0.550224**

  MAE : 1.480766

kernel	MSE	R2	MAE
linear	4.788017	0.534574	1.526880
poly	5.423970	0.472755	1.547286
rbf	4.627016	0.550224	1.480766

3.3 Decision Tree

Decision Trees are a versatile supervised learning technique used in both classification and regression tasks . They operate by constructing a hierarchical structure resembling a tree composed of nodes. Each node represents a decision based on a feature of the data , directed to subsequent nodes or leaf nodes containing the final predicted outcomes . The method aims to predict the values of a target variable by learning straightforward decision rules derived from the data features . The trees' depth determines the complexity of these rules : deeper trees capture more intricate relationships in the data , potentially leading to a more precise model. Decision trees provide a flexible approach to modelling data capable of approximating complex functions through a series of logical decisions.

This can be written mathematically :

$$G_i = 1 - \sum_{k=1}^{n} P_{i,}k^2$$

pi,k is the ratio of class k instances among training instances in i th node

<u>Steps involved in Decision tree:</u>

1. Instantiate the decision tree regressor :
   - ➔ Create an instance of the '**DecisionTreeRegressor**' model with a fixed random state for reproducibility
2. Train the model :
   - ➔ Fit the decision tree regressor model using the scaled training data (**'X_train_scale' for features and y_train** for target)
3. Predict on training and testing data :
   - ➔ Use the trained model to predict the target variable (**'y'**) for both the training and testing
4. Calculate the performance metrics
5. Print results :

- <u>Results:</u>

  Mean Square Error (MSE train): 0.0

  Mean Square Error (MSE test): 8.807655502392345

  R^2 **Error (R2 train): 1.0**

  R^2 **Error (R2 test): 0.14383879902539387**

  Mean absolute error (MAE): 2.04

3.4 Random Forest

The Random Forest algorithm is a robust machine learning technique that enhances prediction accuracy through ensemble learning. It works by constructing multiple Decision Trees during the training phase. Each tree is built using a randomly selected subset of the dataset and evaluates a random subset of features at each split. This randomness introduces diversity among the individual trees, which helps reduce the risk of overfitting and enhances the model's overall performance.During the prediction phase, the algorithm combines the results of all the trees: it uses majority voting for classification tasks and averaging for regression tasks. This ensemble approach, where multiple trees collaborate to make a decision, leads to more stable and accurate predictions.Random Forests are widely used for both classification and regression due to their ability to manage complex data, mitigate overfitting, and deliver reliable predictions across various scenarios.

- <u>Steps involved:</u>
  - ➔ Create an instance of the 'RandomForestRegressor' model with a fixed random state for reproducibility
  - ➔ Train the model :
    1. Fit the random forest regressor model using the scaled training data ((x_train_scale for features and y_train for target).
  - ➔ Prediction on testing data :
    1. Use the trained model to predict the target variable (y) for the testing dataset.
  - ➔ Calculate Performance Metrics

- <u>Results :</u>

  Mean Square Error (MSE test): 4.805265550239234

  R^2 **(R2 test): 0.5328970435574765**

  Mean absolute error (MAE): 1.56

3.5 AdaBoost

- Adaptive Boosting, shortly called AdaBoost, is among the most influential ensemble learning algorithms used to improve weak classifiers. Through a focus on misclassified examples in turns and appropriateness in weighting examples, it creates a string of weak learners that lead to a strong predictor, hence improving accuracy and better decision-making within different data types.

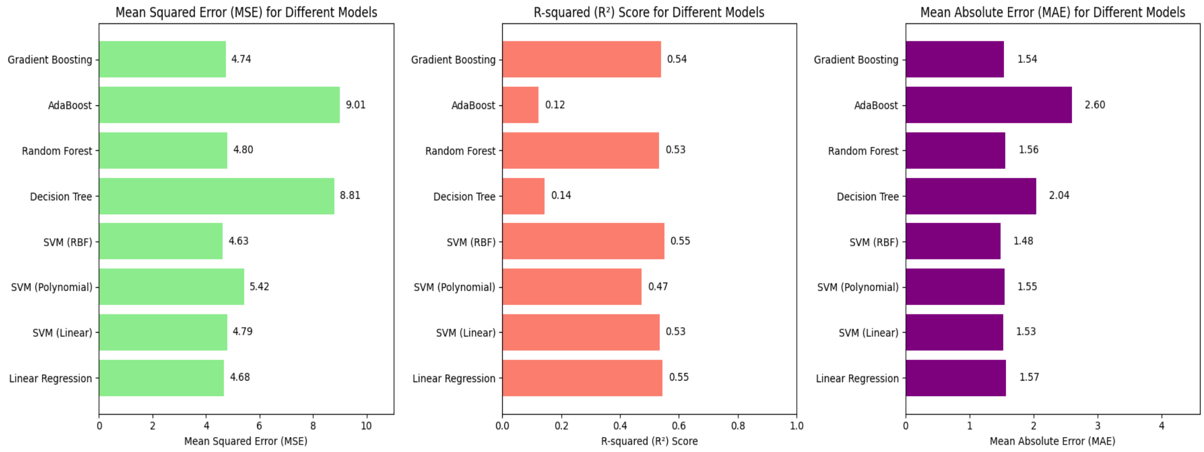
- Result:
  Mean Square Error (MSE test): 9.014977580629 26
  R-Squared Error (R2 test): 0.12368574927865306
  Mean absolute error (MAE) :2.60

3.6 Gradient Boosting

- Gradient Boosting is one of the most popular boosting techniques in machine learning, versed both for classification and regression. It is an ensemble learning process where models are built in a sequence, with each model attempting to correct mistakes involved in the previous ones. Through the combination of a number of weak learners, it makes a strong predictor. It involves the minimization of a loss function using gradient descent. In every cycle of this iteration, the algorithm calculates the gradient of the loss function according to the predictions made by the current model and trains a new weak model to minimise this gradient. This is repeated until some stopping criterion is achieved.

- Result:
  Mean Square Error (MSE test): 4.7367260255976165
  R-Squared Error (R2 test): 0.5395595295863018
  Mean absolute error (MAE) :1.54

4. Results:



.

	Model	MSE	R2	MAE
0	Linear Regression	4.676	0.545	1.5693
1	SVM (Linear)	4.788	0.534	1.5268
2	SVM (Polynomial)	5.424	0.473	1.5472
3	SVM (RBF)	4.627	0.550	1.4807
4	Decision Tree	8.808	0.144	2.0400
5	Random Forest	4.805	0.533	1.5600
6	AdaBoost	9.010	0.124	2.6000
7	Gradient Boosting	4.737	0.539	1.5400

SVM with RBF kernel shows the best performance for this dataset, indicating that a non -linear approach is beneficial. Linear models and ensemble methods like random forest and gradient boosting also perform well.