**COMSATS University Islamabad,   
Abbottabad Campus**

**Data Science**

**(subject)**

**Assignment # 2**

**Predicting the age of Abalone shell**

***By***

**Muhammad Talah Khan CIIT/ FA20-BSE-042/ATD**

***Lecturer*Dr Ahmed Saeed Khattak**

***Bachelor of Science in Software Engineering (2020-2024)***

**Table of Contents**

[**Problem Statement:** 3](#_Toc153731566)

[**Objectives:** 3](#_Toc153731567)

[**Methodology:** 3](#_Toc153731568)

[0.**Importing Libraries:** 3](#_Toc153731569)

[1. **Dataset Loading and Exploration:** 3](#_Toc153731570)

[2**. Data Analysis:** 4](#_Toc153731571)

[3. **Data Preprocessing:** 4](#_Toc153731572)

[4. **Model Building and Training:** 4](#_Toc153731573)

[5. **Model Evaluation:** 5](#_Toc153731574)

[6. **Compare Results:** 5](#_Toc153731575)

[**Results:** 7](#_Toc153731576)

[**Correlations of features with each other:** 8](#_Toc153731577)

[**Comparison for linear regression:** 9](#_Toc153731578)

[**Comparison for Decision Tree:** 9](#_Toc153731579)

[**Comparison for Random Forest:** 9](#_Toc153731580)

[**Comparison for Support Vector Machine:** 10](#_Toc153731581)

[**comparing results of different models:** 10](#_Toc153731582)

[**R^2 Score:** 10](#_Toc153731583)

[**Scattering plot graph:** 10](#_Toc153731584)

[**Residuals Distribution:** 11](#_Toc153731585)

[**Learning curves:** 11](#_Toc153731586)

[**Cross-validated RMSE comparison:** 12](#_Toc153731587)

[**Conclusion:** 12](#_Toc153731588)

[**Findings and Key Takeaways:** 12](#_Toc153731589)

[**Model Performance:** 12](#_Toc153731590)

[**Potential Improvements:** 13](#_Toc153731591)

# **Problem Statement:**

The goal of this project is to predict the age of Abalone Sea snails using various classifiers. Using the Abalone dataset containing measurements related to potential age factors, the objective is to build a model that accurately estimates the age of these creatures.

# **Objectives:**

* Utilize various classifiers to predict the age of Abalone Sea snails.
* Explore the dataset to understand features and their relevance to age prediction.
* Train and evaluate models to assess performance accurately.
* Identify features that significantly impact age prediction.
* Implement feature engineering techniques to enhance model performance.

# **Methodology:**

## 0.**Importing Libraries:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LinearRegression

from sklearn.tree import DecisionTreeRegressor

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error

from sklearn.svm import SVR

import matplotlib.pyplot as plt

from sklearn.model\_selection import learning\_curve

import numpy as np

from sklearn.model\_selection import cross\_val\_score

## 1. **Dataset Loading and Exploration:**

# Load the dataset

data = pd.read\_csv('abalone.csv')

# Explore Data Features and Distributions

print(data)  # Display all records in the dataset

print(data.describe())  # Descriptive statistics

# Pairplot - visualize distributions

sns.pairplot(data)

plt.show()

## 2**. Data Analysis:**

# Investigate Correlations

# One-hot encode 'Sex' column

data\_encoded = pd.get\_dummies(data, columns=[data.columns[0]], drop\_first=True)

correlation\_matrix = data\_encoded.corr()

plt.figure(figsize=(10, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt='.2f')

plt.title('Correlation Matrix')

plt.show()

correlation\_with\_target = correlation\_matrix['rings'].sort\_values(ascending=False)

print(correlation\_with\_target)

## 3. **Data Preprocessing:**

# Data preprocessing

X = data\_encoded.drop('rings', axis=1)

y = data\_encoded['rings']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Normalize Data

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

## 4. **Model Building and Training:**

# Model Development

models = {

    'Linear Regression': LinearRegression(),

    'Decision Tree': DecisionTreeRegressor(),

    'Random Forest': RandomForestRegressor()

}

# Train models

for name, model in models.items():

    model.fit(X\_train\_scaled, y\_train)

## 5. **Model Evaluation:**

# Model Evaluation

for name, model in models.items():

    y\_train\_pred = model.predict(X\_train\_scaled)

    y\_test\_pred = model.predict(X\_test\_scaled)

    rmse\_train = mean\_squared\_error(y\_train, y\_train\_pred, squared=False)

    rmse\_test = mean\_squared\_error(y\_test, y\_test\_pred, squared=False)

    print(f"{name} RMSE on train dataset: {rmse\_train}")

    print(f"{name} RMSE on test dataset: {rmse\_test}")

## 6. **Compare Results:**

Result was compared using different techniques.

#compare results

plt.figure(figsize=(10, 6))

for name, model in models.items():

    model.fit(X\_train\_scaled, y\_train)

    y\_pred = model.predict(X\_test\_scaled)

    plt.scatter(y\_test, y\_pred, label=name, alpha=0.5)

plt.xlabel('Actual values')

plt.ylabel('Predicted values')

plt.title('Model Performance: Actual vs Predicted values')

plt.legend()

plt.show()

#another way

for name, model in models.items():

    model.fit(X\_train\_scaled, y\_train)

    y\_pred = model.predict(X\_test\_scaled)

    comparison = pd.DataFrame({'Actual': y\_test, 'Predicted': y\_pred})

    print(f"Comparison for {name}:")

    print(comparison.head(10))  # Displaying the first 10 records for brevity

    print("\n")

# Plotting residuals

plt.figure(figsize=(10, 6))

for name, model in models.items():

    model.fit(X\_train\_scaled, y\_train)

    y\_pred = model.predict(X\_test\_scaled)

    residuals = y\_test - y\_pred

    plt.hist(residuals, bins=30, label=name, alpha=0.7)

plt.xlabel('Residuals')

plt.ylabel('Frequency')

plt.title('Residuals Distribution')

plt.legend()

plt.show()

# Model Evaluation: R^2 score

from sklearn.metrics import r2\_score

for name, model in models.items():

    model.fit(X\_train\_scaled, y\_train)

    y\_pred = model.predict(X\_test\_scaled)

    r2 = r2\_score(y\_test, y\_pred)

    print(f"{name} R^2 Score: {r2}")

#learning curve

plt.figure(figsize=(10, 6))

for name, model in models.items():

    train\_sizes, train\_scores, test\_scores = learning\_curve(model, X, y, cv=5, scoring='neg\_mean\_squared\_error')

    train\_scores = np.sqrt(-train\_scores)

    test\_scores = np.sqrt(-test\_scores)

    train\_scores\_mean = np.mean(train\_scores, axis=1)

    test\_scores\_mean = np.mean(test\_scores, axis=1)

    plt.plot(train\_sizes, train\_scores\_mean, label=f'{name} Training score')

    plt.plot(train\_sizes, test\_scores\_mean, label=f'{name} Validation score')

plt.xlabel('Training examples')

plt.ylabel('RMSE')

plt.title('Learning Curves')

plt.legend()

plt.grid()

plt.show()

# Evaluate models using cross-validation

cv\_results = {}

for name, model in models.items():

    cv\_scores = cross\_val\_score(model, X, y, cv=5, scoring='neg\_mean\_squared\_error')

    cv\_rmse = np.sqrt(-cv\_scores)

    cv\_results[name] = cv\_rmse

# Plotting boxplots for comparison

plt.figure(figsize=(10, 6))

sns.boxplot(data=pd.DataFrame(cv\_results))

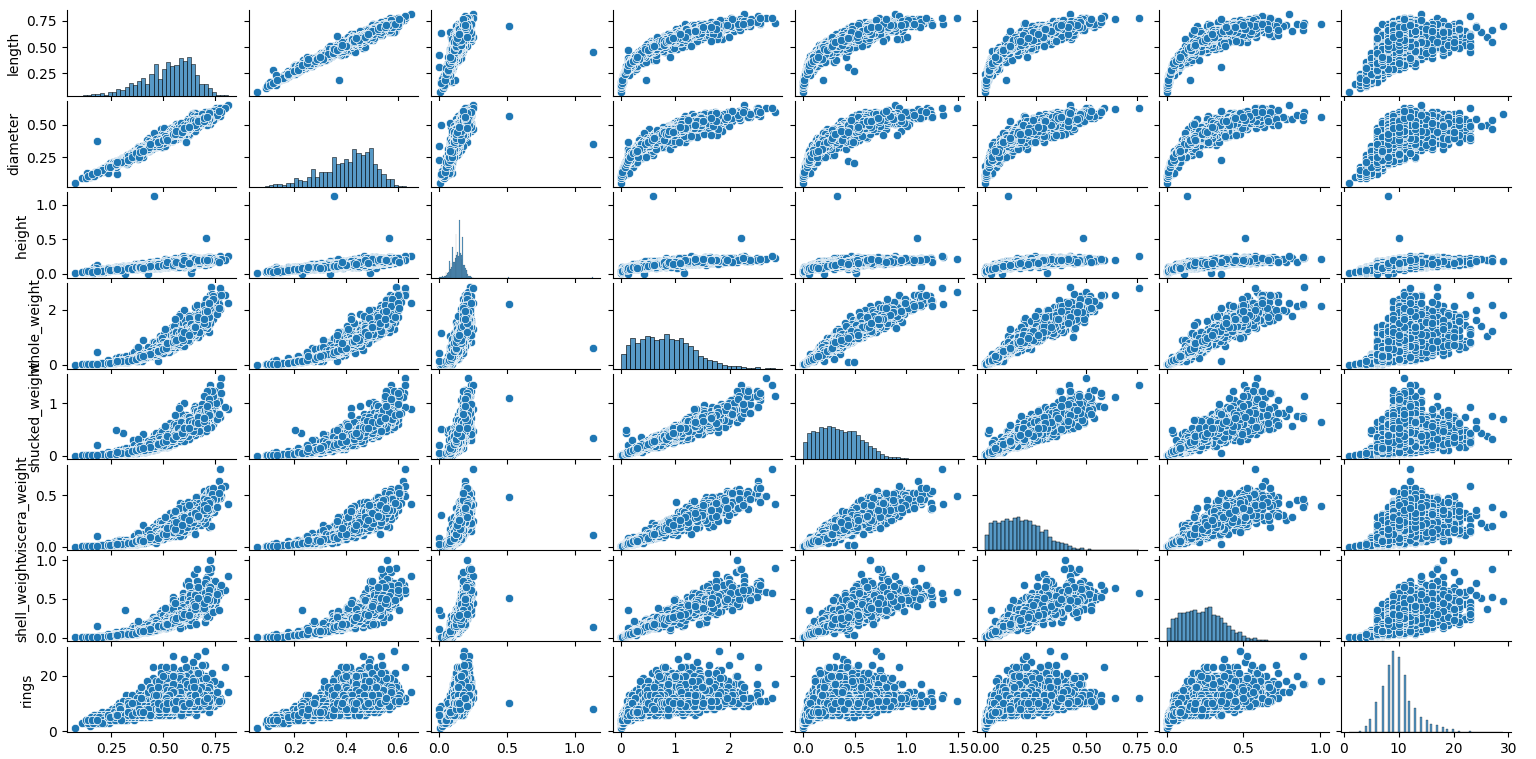
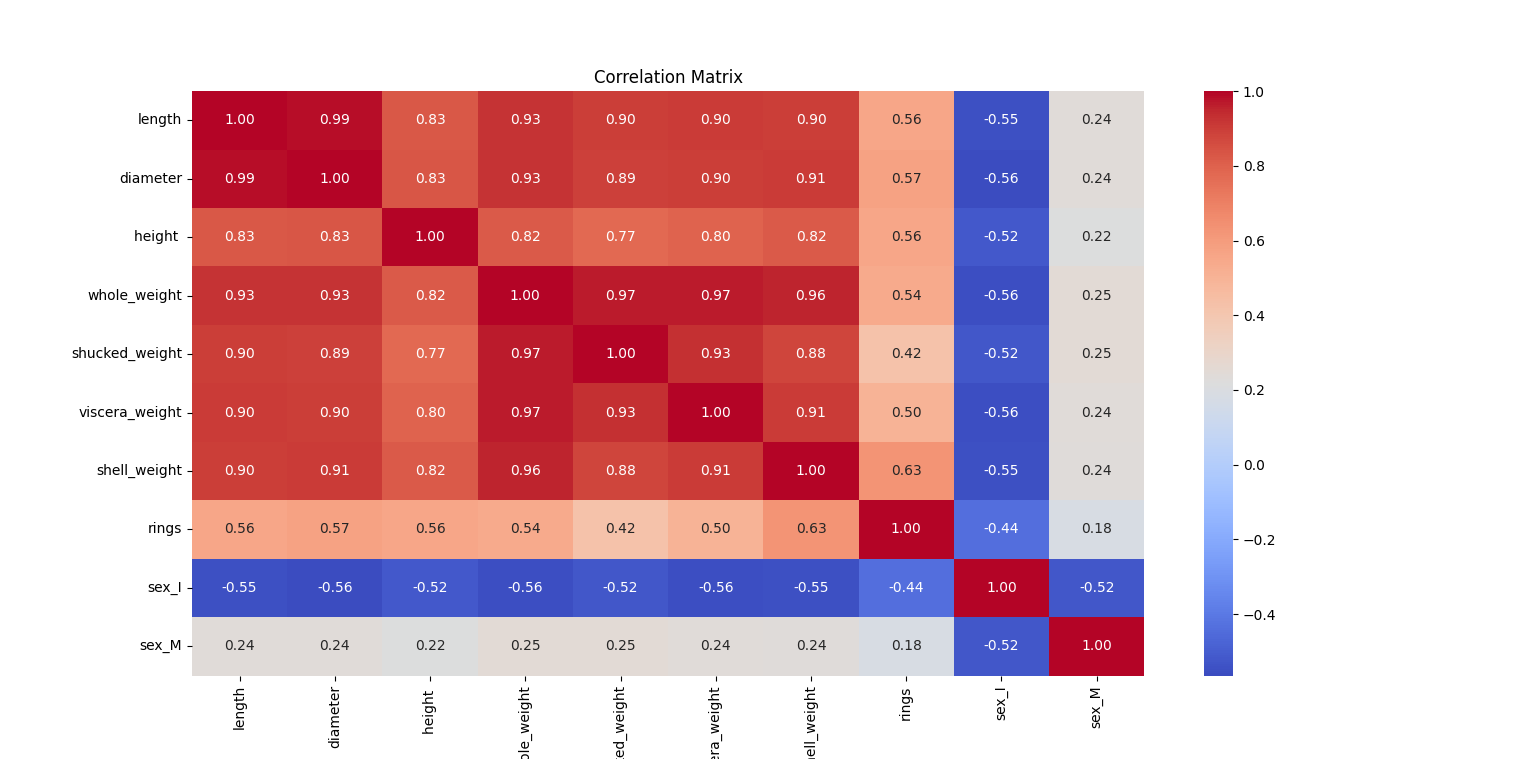
plt.xlabel('Models')

plt.ylabel('Cross-Validated RMSE')

plt.title('Cross-Validated RMSE Comparison')

plt.show()

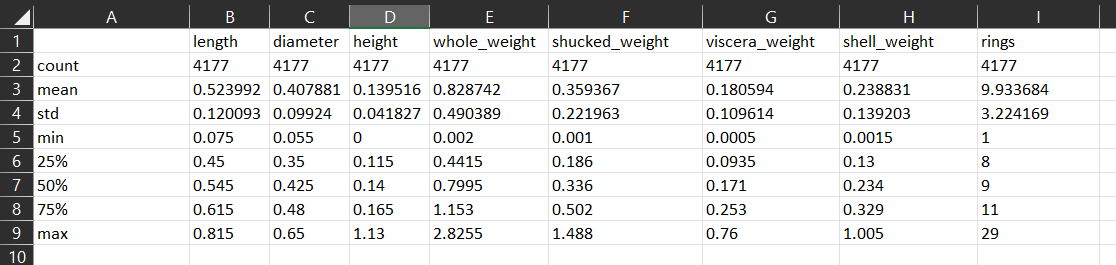
# **Results:**



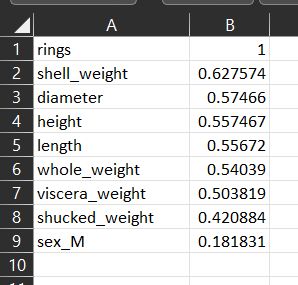
A screenshot of a computer

Description automatically generated

**[4177 rows x 9 columns]**



### **Correlations of features with each other:**



Name: rings, dtype: float64

Linear Regression RMSE on train dataset: 2.187202344131259

Linear Regression RMSE on test dataset: 2.2116130871218367

Decision Tree RMSE on train dataset: 0.0

Decision Tree RMSE on test dataset: 3.020068442457751

Random Forest RMSE on train dataset: 0.7994350556868267

Random Forest RMSE on test dataset: 2.2441084481977978

Support Vector Machine RMSE on train dataset: 2.1377035227956362

Support Vector Machine RMSE on test dataset: 2.2288427659068124

### **Comparison for linear regression:**

A screenshot of a table

Description automatically generated

### **Comparison for Decision Tree:**

A table with numbers and a black background

Description automatically generated

### **Comparison for Random Forest:**

A table with numbers and a black background

Description automatically generated

### **Comparison for Support Vector Machine:**

A table with numbers and a few black text

Description automatically generated

## **comparing results of different models:**

### **R^2 Score:**

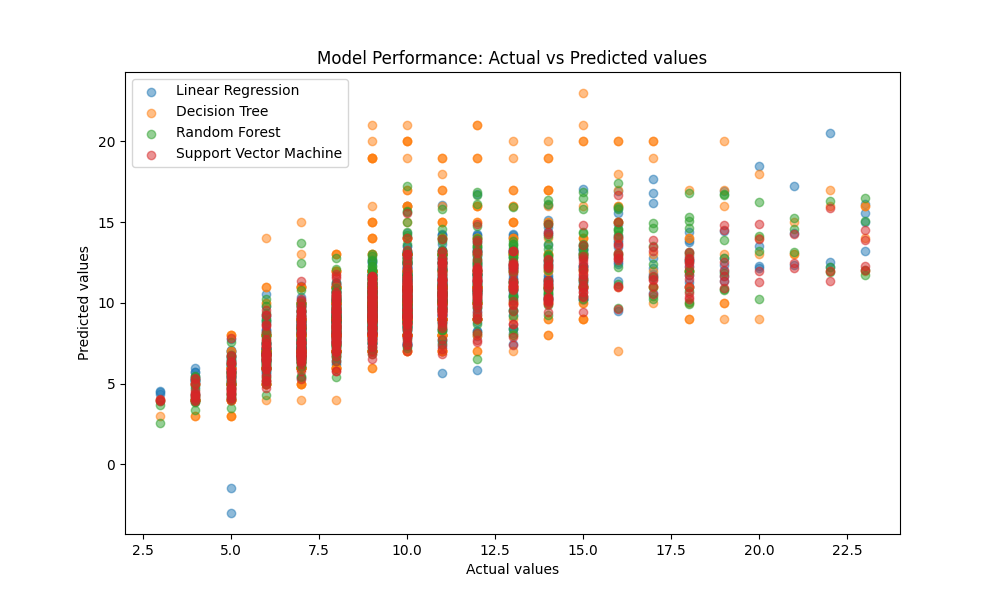
Linear Regression R^2 Score: 0.5481628137889263

Decision Tree R^2 Score: 0.12540234236215675

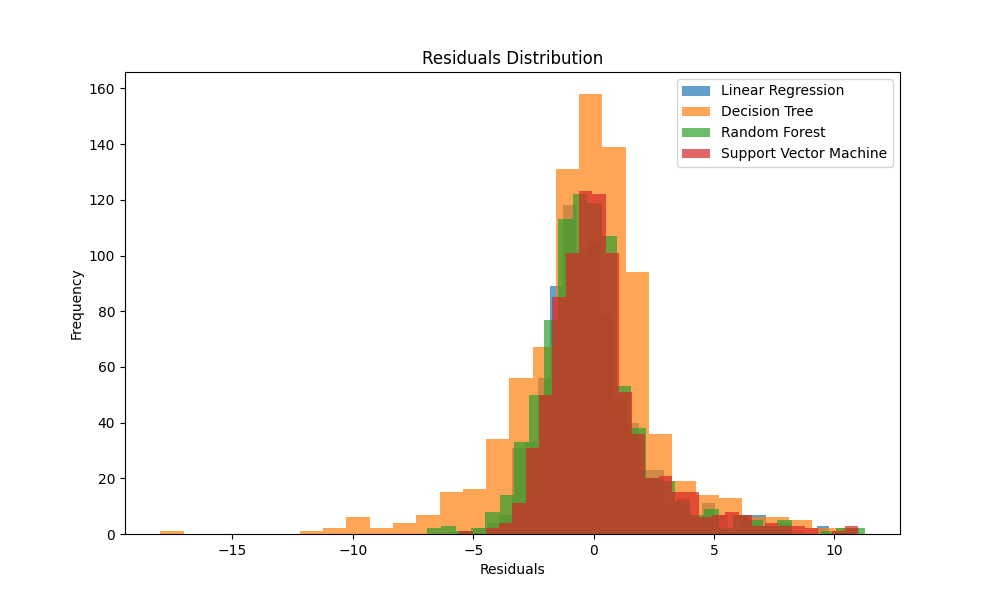
Random Forest R^2 Score: 0.5397797290295565

Support Vector Machine R^2 Score: 0.541095271664148

### **Scattering plot graph:**



### **Residuals Distribution:**



### **Learning curves:**

A graph with different colored lines

Description automatically generated

### **Cross-validated RMSE comparison:**

A diagram of a graph

Description automatically generated with medium confidence

# **Conclusion:**

## **Findings and Key Takeaways:**

**Feature Importance:**

Rings appears to be the target variable, indicating the age of the abalone.

Features like shell weight, diameter, and whole weight seem to have a significant impact on predicting the number of rings, potentially indicating strong correlations with the age of the abalone.

## **Model Performance:**

Among the models used (Linear Regression, Decision Tree, Random Forest, SVM), Linear Regression performed slightly better in terms of RMSE on the test dataset.

Decision Tree showed signs of overfitting as it achieved a perfect RMSE on the training dataset but higher error on the test dataset.

**Categorical Feature Impact:**

Categorical features like sex might also contribute to predicting the age of abalones, with the correlation coefficient indicating a mild impact on the number of rings.

**Model Performance Reflection:**

Linear Regression, Random Forest, and SVM provided moderate performance in predicting the number of rings, with Linear Regression having a slightly better performance based on the given metrics.

**Good Performance:**

Linear Regression displayed a relatively better fit to the test dataset compared to other models.

Random Forest also showcased decent performance in predicting the number of rings.

## **Potential Improvements:**

**Feature Engineering:**

Further exploration into feature combinations or transformations might enhance the predictive power of models. For instance, creating interaction terms or polynomial features could capture more complex relationships.

**Hyperparameter Tuning:**

Adjusting the hyperparameters of the models, especially for Decision Trees and Random Forests, could mitigate overfitting and potentially improve predictive accuracy.

**Cross-validation:**

Employing more robust cross-validation techniques could provide a better estimate of model performance and generalizability.

Linear Regression stands out marginally in predicting the age of abalones based on the given dataset, showcasing better performance compared to other models.

Feature engineering, hyperparameter tuning, and more comprehensive cross-validation could potentially enhance the models' predictive capabilities.

Further refinement of the models through these methods might yield improved accuracy and robustness in predicting the age of abalones based on their physical characteristics.