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
Objectives: 3
Methodology: 3
0.**Importing Libraries:** 3
1. **Dataset Loading and Exploration:** 3
2**. Data Analysis:** 4
3. **Data Preprocessing:** 4
4. **Model Building and Training:** 4
5. **Model Evaluation:** 5
6. **Compare Results:** 5
Results: 7
Correlations of features with each other: 8
Comparison for linear regression: 9
Comparison for Decision Tree: 9
Comparison for Random Forest: 9
Comparison for Support Vector Machine: 10
comparing results of different models: 10
R^2 Score: 10
Scattering plot graph: 10
Residuals Distribution: 11
Learning curves: 11
Cross-validated RMSE comparison: 12
Conclusion: 12
Findings and Key Takeaways: 12
Model Performance: 12
Potential Improvements: 13

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}")
```

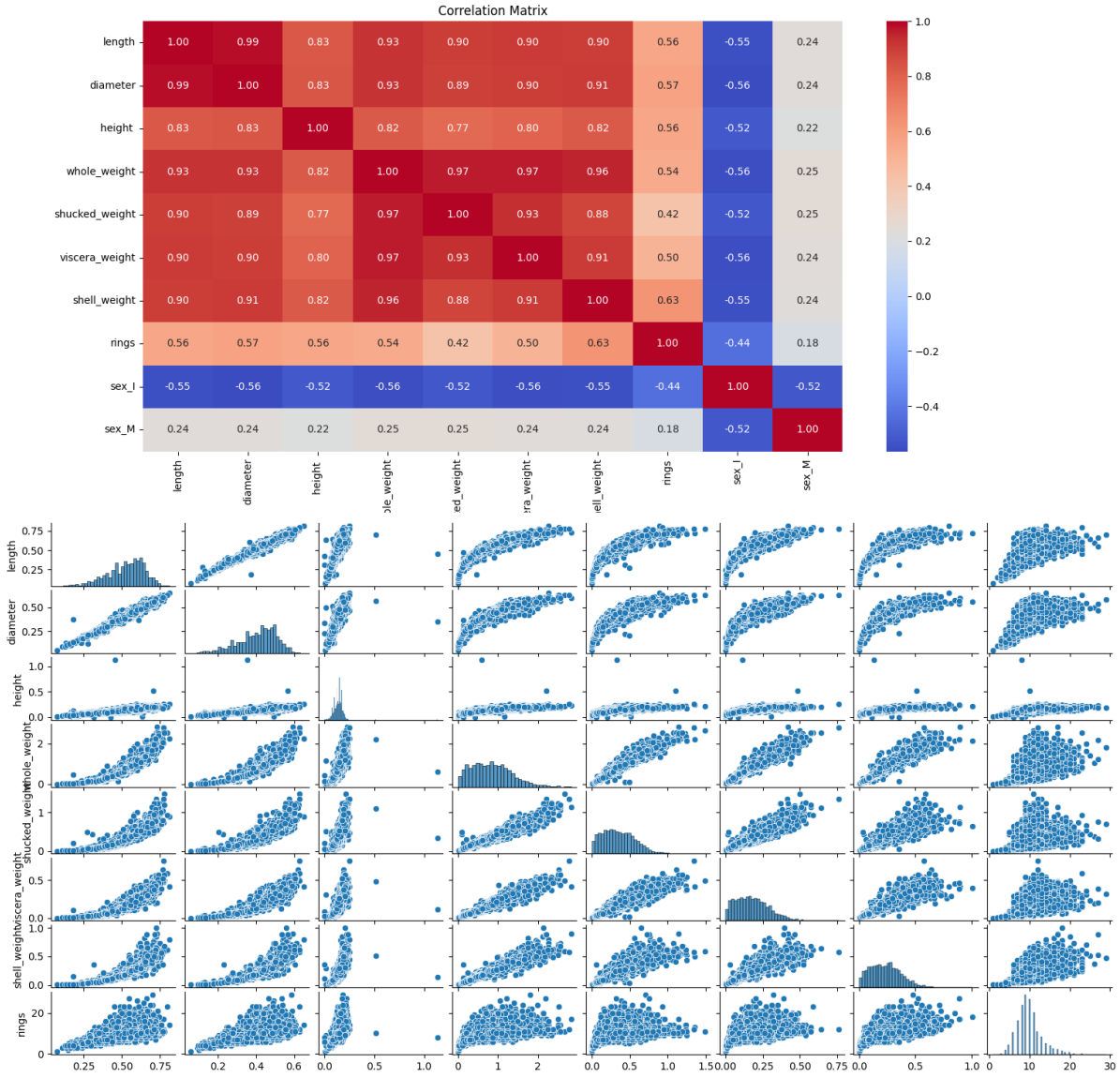
6. **Compare Results:**

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	B	C	D	E	F	G	H	I	J
1		sex	length	diameter	height	whole_weight	shucked_weight	viscera_weight	shell_weight	rings
2	0	M	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
3	1	M	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
4	2	F	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
5	3	M	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
6	4	I	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7
7	...	..	...	...	...	...	...	...	...	...
8	4172	F	0.565	0.45	0.165	0.887	0.37	0.239	0.249	11
9	4173	M	0.59	0.44	0.135	0.966	0.439	0.2145	0.2605	10
10	4174	M	0.6	0.475	0.205	1.176	0.5255	0.2875	0.308	9
11	4175	F	0.625	0.485	0.15	1.0945	0.531	0.261	0.296	10
12	4176	M	0.71	0.555	0.195	1.9485	0.9455	0.3765	0.495	12
13										

[4177 rows x 9 columns]

	A	B	C	D	E	F	G	H	I
1		length	diameter	height	whole_weight	shucked_weight	viscera_weight	shell_weight	rings
2	count	4177	4177	4177	4177	4177	4177	4177	4177
3	mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
4	std	0.120093	0.09924	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
5	min	0.075	0.055	0	0.002	0.001	0.0005	0.0015	1
6	25%	0.45	0.35	0.115	0.4415	0.186	0.0935	0.13	8
7	50%	0.545	0.425	0.14	0.7995	0.336	0.171	0.234	9
8	75%	0.615	0.48	0.165	1.153	0.502	0.253	0.329	11
9	max	0.815	0.65	1.13	2.8255	1.488	0.76	1.005	29
10									

Correlations of features with each other:

	A	B
1	rings	1
2	shell_weight	0.627574
3	diameter	0.57466
4	height	0.557467
5	length	0.55672
6	whole_weight	0.54039
7	viscera_weight	0.503819
8	shucked_weight	0.420884
9	sex_M	0.181831
10		

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	B	C
1	minimum values	Actual	Predicted
2	866	9	11.76136
3	1483	8	10.24193
4	599	16	14.00104
5	1702	9	11.99509
6	670	14	11.16142
7	2430	11	10.23009
8	1590	7	9.417632
9	949	6	9.149558
10	4026	7	7.191891
11	3668	10	10.80526
12			

Comparison for Decision Tree:

	A	B	C
1	minimum values	Actual	Predicted
2	866	9	13
3	1483	8	12
4	599	16	18
5	1702	9	8
6	670	14	12
7	2430	11	11
8	1590	7	8
9	949	6	8
10	4026	7	4
11	3668	10	10

Comparison for Random Forest:

	A	B	C
1	minimum values	Actual	Predicted
2	866	9	12.13
3	1483	8	9.53
4	599	16	14.55
5	1702	9	10.52
6	670	14	12.49
7	2430	11	10.11
8	1590	7	8.93
9	949	6	8.73
10	4026	7	6.58
11	3668	10	10.78

Comparison for Support Vector Machine:

	minimum values	Actual	Predicted
2	866	9	11.32806
3	1483	8	9.714155
4	599	16	13.57415
5	1702	9	11.03729
6	670	14	11.43626
7	2430	11	9.637512
8	1590	7	8.951026
9	949	6	8.714812
10	4026	7	6.785089
11	3668	10	9.924822

comparing results of different models:

R^2 Score:

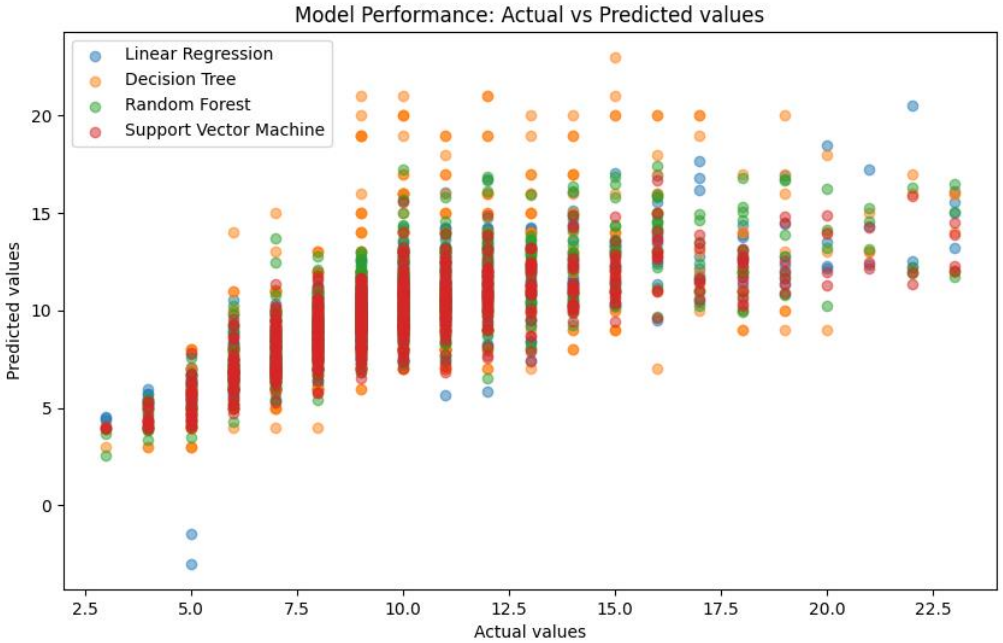
Linear Regression R^2 Score: 0.5481628137889263

Decision Tree R^2 Score: 0.12540234236215675

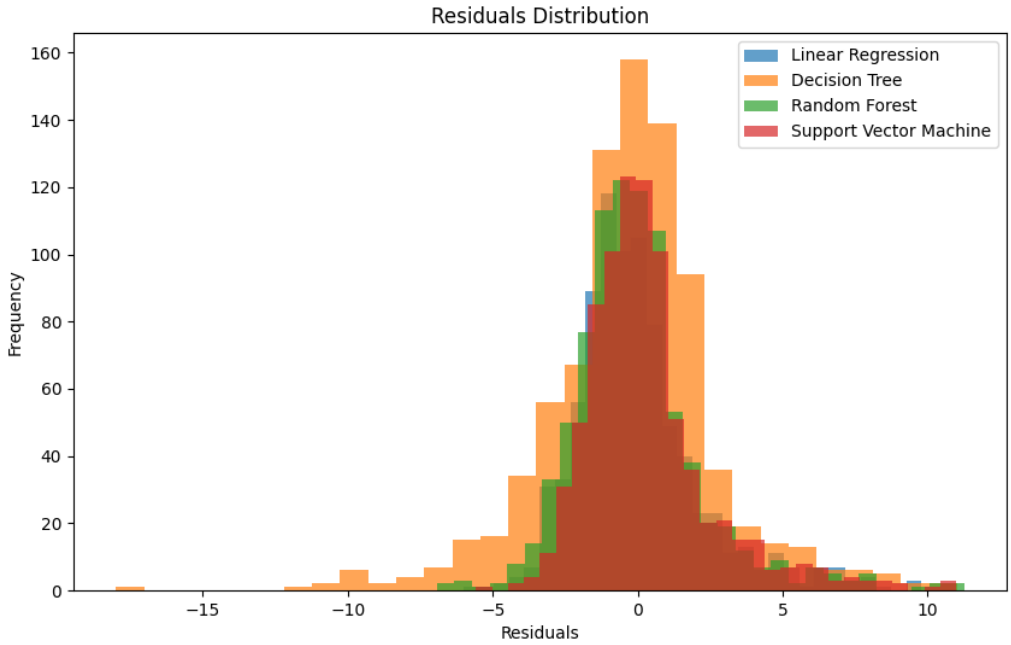
Random Forest R^2 Score: 0.5397797290295565

Support Vector Machine R^2 Score: 0.541095271664148

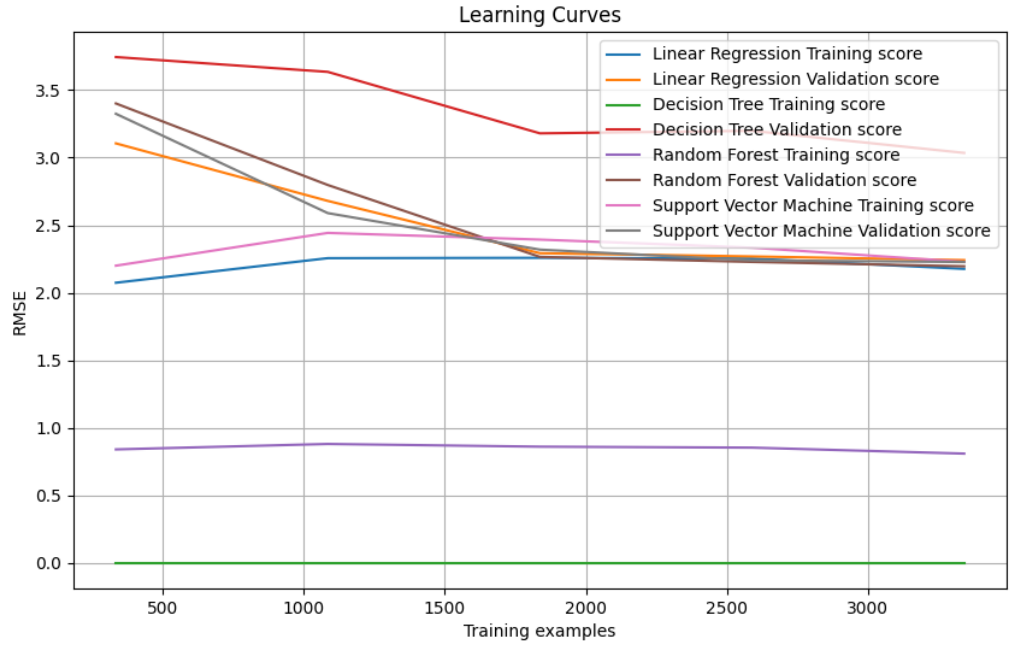
Scattering plot graph:



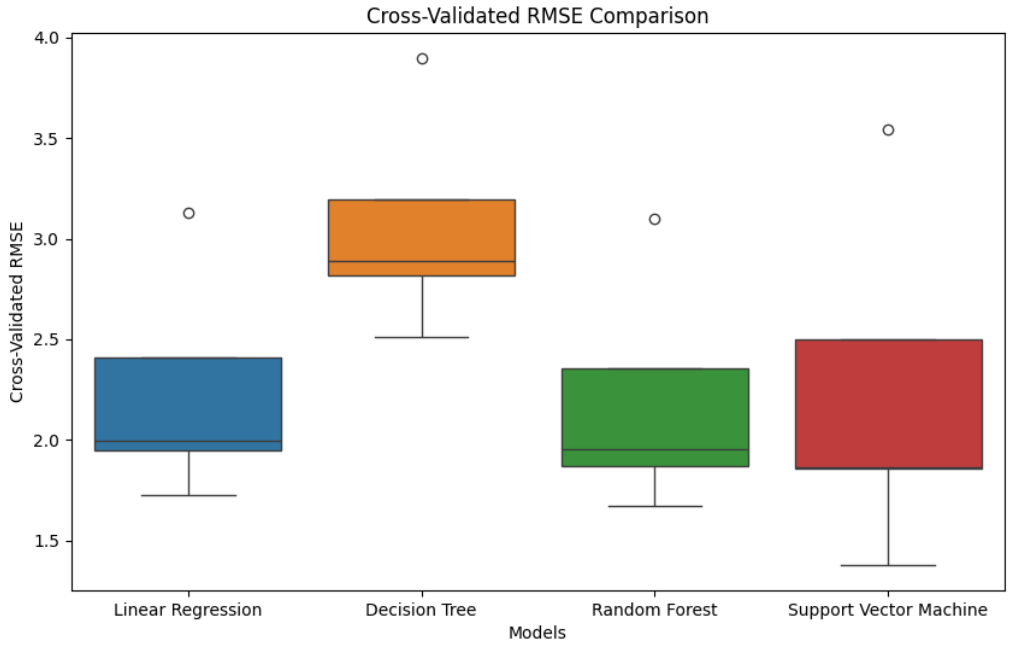
Residuals Distribution:



Learning curves:



Cross-validated RMSE comparison:



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.