STUDENT PERFORMANCE PREDICTION USING MACHINE LEARNING

Regression Method

TEAM-6
K.Navya Poojitha
P.Priyanka
S.Manohar
Saswat Deepak

Bhavan's Vivekananda College|Bsc Hons Data Science

Project Stages

- ➔ Data Pre-Processing
- ➔ Exploratory Data Analysis
- ➔ Training the model
- ➔ Determining model Performance
- ➔ Comparing results with other models
- ➔ Conclusion and Insights

About The Data

- The data is about the grades of the student and the variables that affect the **“TARGET_G3”.**
- This data approach student achievement in secondary education of two Portuguese schools. The data attribute includes student grades, demographic, social and school related features and it was collected by using school reports and questionnaires.

Objective

- Student performance in academics is influenced by many factors in student’s life. **Our goal is to predict the Final grade of the student based on the factors and to build a good machine learning model for this Purpose.**

Path

- Implementing multiple machine learning models to fit the best model for the dataset

Data Pre-Processing

Data and Data Quality Check

Description	Details
Total rows	355
Total Columns	33
Number of Missing Values	None
Dropped Columns	1

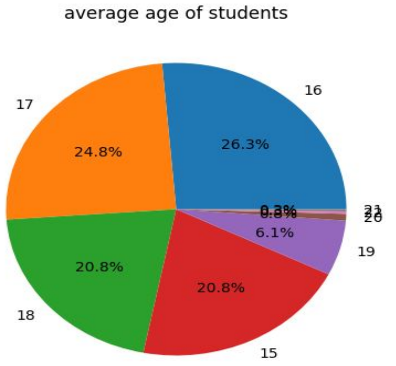
Data Variables

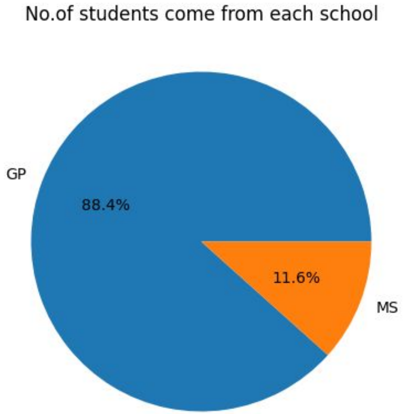
Variable Name	Description
school	student's school (binary: 'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)
Medu	mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 -“ 5th to 9th grade”, 3-“secondary education or 4 -“ higher education
Fedu	Father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 -“ 5th to 9th grade”, 3-“secondary education or 4 -“ higher education
Mjob	mother's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
Fjob	father's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')

Studytime	weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
absences	number of school absences (numeric: from 0 to 93)
famsup	family educational support (binary: yes or no)
Pstatus	parent's cohabitation status (binary: 'T' - living together or 'A' - apart)
health	current health status (numeric: from 1 - very bad to 5 - very good)
schoolsup	extra educational support (binary: yes or no)
romantic	with a romantic relationship (binary: yes or no)
goout	going out with friends (numeric: from 1 - very low to 5 - very high)
G1,G2,G3	Math grade,Portugese grade,Final grade

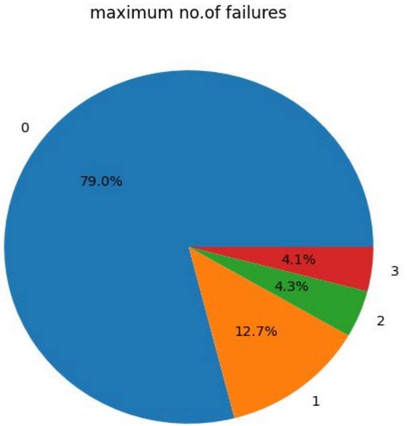
Exploratory Data Analysis (EDA)

- This Pie Chart shows percentage of average age of students



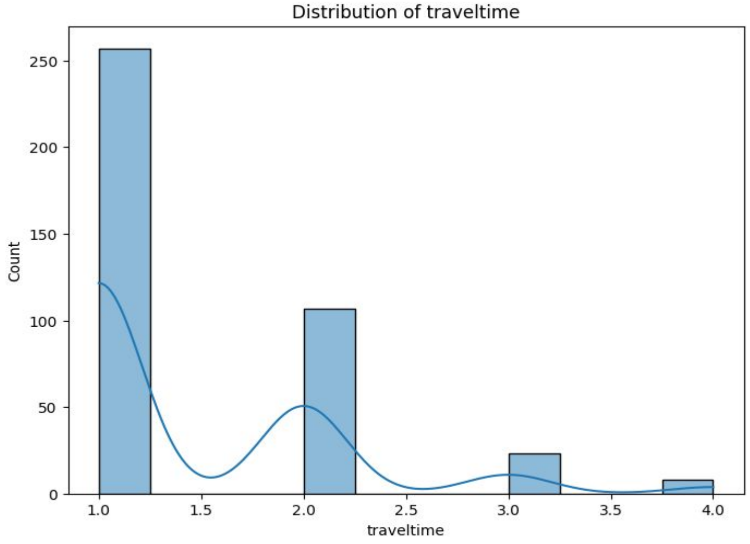


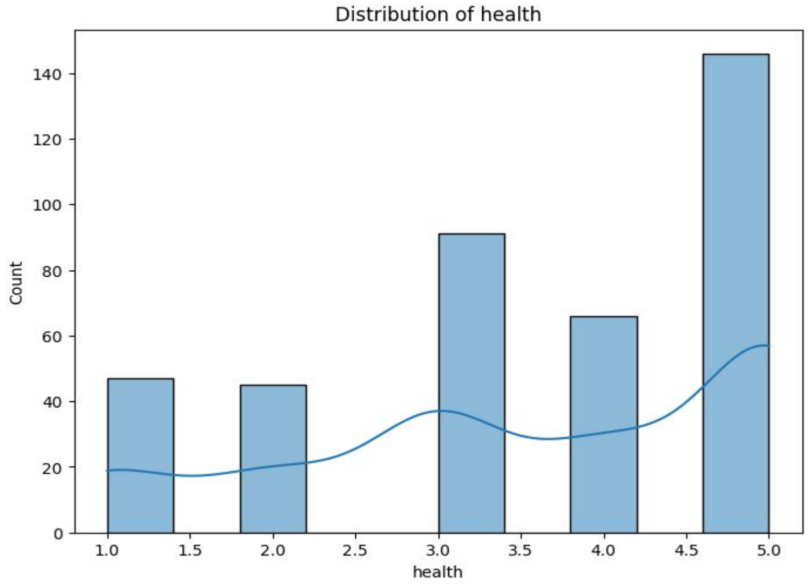
- This Pie Chart shows the Percentage of students from each School



- This Pie Chart shows Percentage of number of past class failures

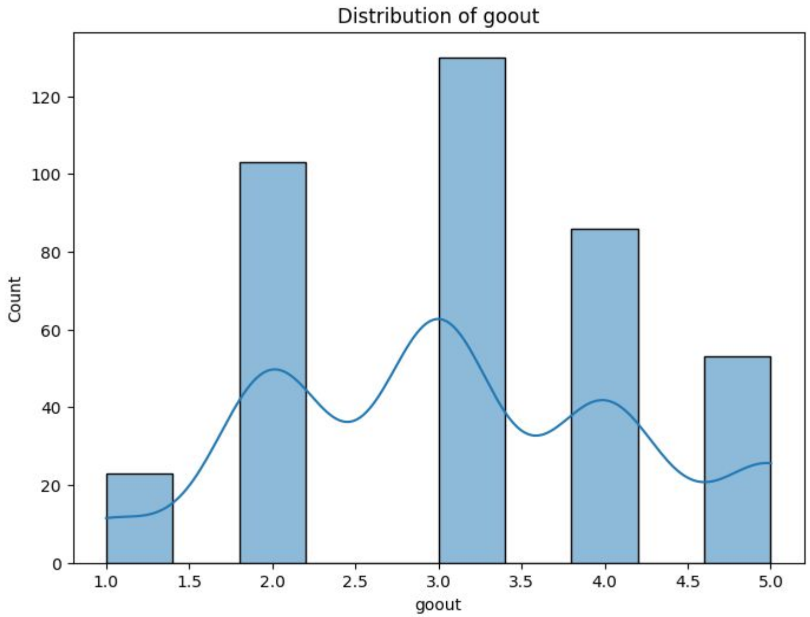
- The hist plot shows the distribution of home to school travel times of the students.

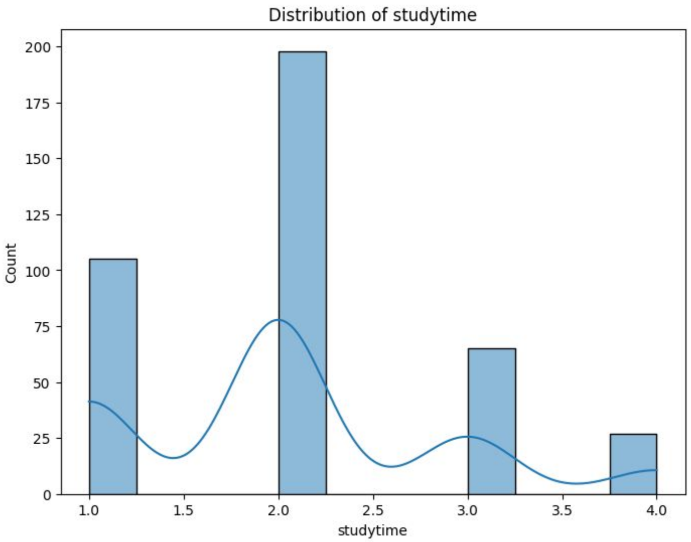




- The histogram shows the distribution of current health status of students (numeric: from 1 - very bad to 5 - very good).

- This hist plot shows the Count of students who are going out with friends from a scale 1 - very low to 5 - very high.

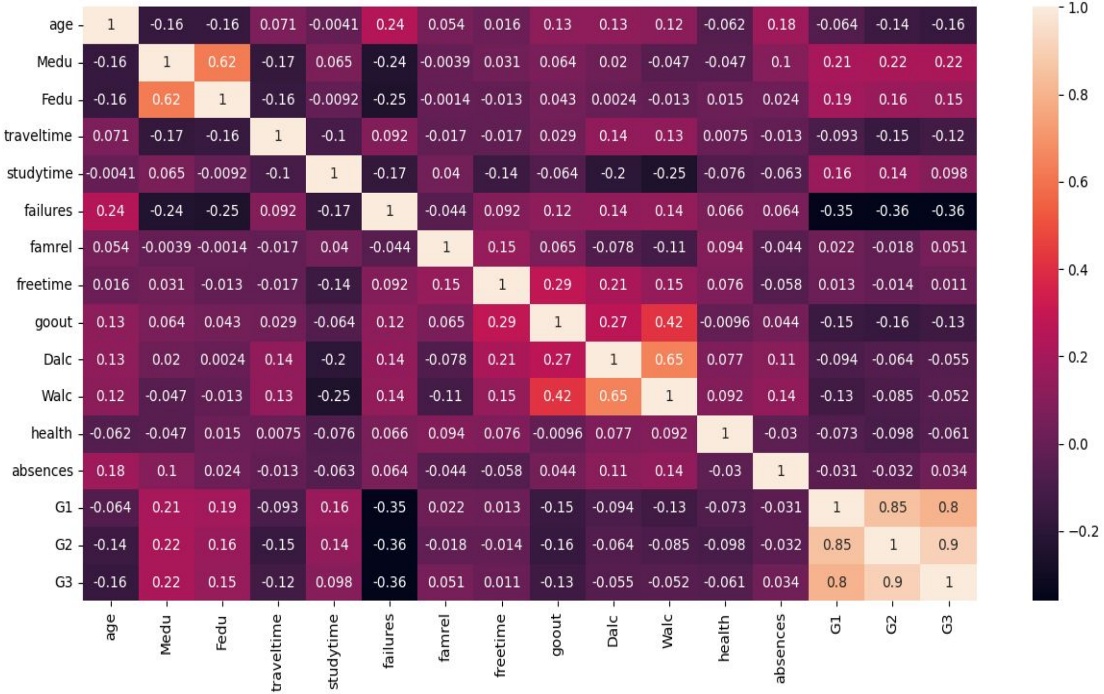




- This hist plot shows the count of students and their study time that is given by the scale 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours.

Predictor Correlation Heatmap

A heatmap is a data visualization technique that displays the magnitude of a phenomenon as color in two dimensions. It is commonly used in exploratory data analysis (EDA) to identify patterns, correlations, and trends within large datasets.



Multicollinearity :

Multicollinearity refers to a situation in statistical modeling where predictor variables in a regression model are highly correlated with each other. In simpler terms, it's when two or more independent variables in a regression analysis are so highly correlated that they convey similar information, and vif is our **"Variance Inflation Factor"**.

	variables	VIF
0	Medu	21.869064
1	Fedu	13.892826
2	traveltime	7.754369
3	studytime	9.595059
4	failures	1.765801
5	famrel	27.630291
6	freetime	14.289834
7	goout	13.299594
8	Dalc	9.152380
9	Walc	9.243292
10	health	8.742245
11	absences	2.163849
12	G1	56.545664
13	G2	43.378110

TRAINING THE MODEL AND DETERMINING THE MODEL PERFORMANCE

1.Linear Regression

Train_Test	R_squared value	Mean Absolute Error
65-35	0.76	1.31
70-30	0.72	1.31
75-25	**0.75**	**1.21**
80-20	0.71	1.31

2.Random Forest Regressor

Train_Test	R_squared value	Mean Absolute Error
65-35	0.77	1.10
70-30	0.76	1.11
75-25	0.77	0.99
80-20	**0.77**	**0.97**

3.Gradient Boosting

Train_Test	Learning rate	n_estimators	R_squared value	Mean Absolute Error
75-25	0.1	100	0.73	1.08
75-25	0.1	200	0.72	1.13
70-30	0.1	100	0.75	1.10
70-30	0.1	200	0.73	1.19
80-20	**0.1**	**100**	**0.77**	**1.04**
80-20	0.1	200	0.75	1.13

4.Support Vector Regressor

Train_Test	R_squared value	Mean Absolute Error
65-35	0.71	1.23
70-30	0.68	1.20
75-25	**0.71**	**1.10**
80-20	0.66	1.20

5.XG Boosting

Train_Test	Learning rate	n_estimators	R_squared value	Mean Absolute Error
75-25	0.1	100	0.79	1.04
75-25	0.1	200	0.79	1.06
70-30	0.1	100	0.75	1.17
70-30	0.1	200	0.76	1.16
80-20	**0.1**	**100**	**0.77**	**1.06**
80-20	0.1	200	0.77	1.16

6.K-Nearest Neighbor Regressor

Train_Test	R_squared value	Mean Absolute Error
65-35	0.72	1.35
70-30	0.67	1.41
75-25	0.75	1.27
80-20	**0.75**	**1.25**

7.Decision Tree Regressor

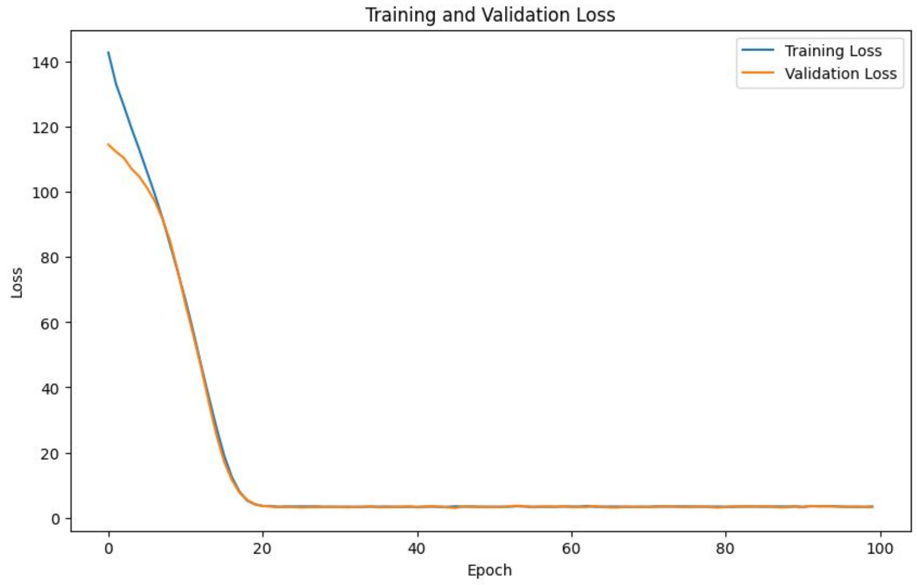
Train_Test	R_squared value	Mean Absolute Error
65-35	0.55	1.38
70-30	0.55	1.39
75-25	0.57	1.35
80-20	**0.65**	**1.11**

8.Artificial Neural Network

Train_Test	Architecture	Optimizer	Epochs	Mean Absolute Error
75-25	**64-32-1**	**Adam**	**100**	**1.21**
75-25	64-32-1	RMSprop	100	1.39
70-30	64-32-1	Adam	100	1.29
70-30	64-32-1	RMSprop	100	1.38
80-20	64-32-1	Adam	100	1.34
80-20	64-32-1	RMSprop	100	1.44

Plot showing Training loss and Validation loss in ANN

Train_Test	75-25
Architecture	64-32-1
Optimizer	Adam
MAE	1.21
Epochs	100



- Both training and validation losses stabilize in the plot.
- This implies that training can be stopped early to prevent overfitting.

Models Comparison

Model	Mean Absolute Error
Linear Regression	1.21
Random Forest Regressor	**0.97**
Gradient Boosting	1.04
Support Vector Regressor	1.10
XG Boosting	1.06
KNN	1.25
Decision Tree Regressor	1.11
Artificial Neural Network	1.21

Conclusion and Insights

- For Student Performance Prediction dataset,We applied 8 machine learning algorithms:Linear Regression,Random Forest,Gradient Boost,Support Vector Regressor,XGBoost,KNN,Decision Tree,Artificial Neural Network.

- After Performing all above algorithms,We got least MAE that is **0.97** by using Random Forest regressor at 80-20 train and test size.

- So,We here Conclude that Random Forest regressor can be the best model while predicting student’s Performance.

Presented By:

1. K.Navya Poojitha
2. P.Priyanka
3. S.Manohar
4. Saswat Deepak

APPENDIX

TRAINING AND TESTING

```
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
#Splitting data into Training and Testing
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2, random_state = 42)
X_train.shape , X_test.shape
```

```
((284, 42), (71, 42))
```

LINEAR REGRESSION

```
# Create a Linear Regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)
predictions
```

```
# Calculate Mean Squared Error and R-squared
mae = mean_absolute_error(y_test, predictions)
r2 = r2_score(y_test, predictions)

# Print the evaluation metrics
print('Mean absolute Error:', mae)
print('R-squared:', r2)
```

```
Mean absolute Error: 1.3102689375317635
R-squared: 0.7158170558140711
```

RANDOM FOREST REGRESSOR

```
# Initialize the Random Forest Regressor
rf_regressor = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model
rf_regressor.fit(X_train_scaled, y_train)
```

```
# Make predictions on the test set
predictions = rf_regressor.predict(X_test_scaled)
print(predictions)

# Calculate Mean Squared Error and R-squared
mae = mean_absolute_error(y_test, predictions)
r2 = r2_score(y_test, predictions)

# Print the evaluation metrics
print('Mean absolute Error:', mae)
print('R-squared:', r2)
```

```
[ 9.47 11.29 18.04  8.69 10.76 12.04 14.05  9.22 15.09  9.1   1.81 11.32
  9.4   2.25  0.06  7.21  7.2   9.88  9.15  2.41 12.14 12.04 10.43 16.04
  7.69 13.64  4.11 13.32 11.25 10.12 12.18  2.73 14.09 10.01 12.61  3.87
  9.03  7.71  6.23 13.57  5.32 13.41 13.25 10.86  5.74 13.39 15.8  11.75
  7.41 10.41 12.    7.55  9.95  7.56 10.   10.62  8.92 11.19 10.03  9.85
 15.97 12.68  7.04  8.39 11.31  5.72  5.85  5.93 10.81 12.71 10.67]
Mean absolute Error: 0.9718309859154928
R-squared: 0.7758665603977771
```

GRADIENT BOOSTING REGRESSOR

```
]] reg = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)
   reg.fit(X_train, y_train)
```

```
▾          GradientBoostingRegressor
GradientBoostingRegressor(random_state=42)
```

```
y_pred = reg.predict(X_test)
# Calculate Mean Squared Error and R-squared
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print('Mean absolute Error:', mae)
print('R-squared:', r2)
```

```
Mean absolute Error: 1.0435175864989616
R-squared: 0.774258657071633
```

SUPPORT VECTOR REGRESSOR

```
# Create an SVR model
model = SVR(kernel='poly')  # You can also use 'poly' or 'rbf' kernels
```

```
# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Calculate the Mean Squared Error (MSE) and r2_score to evaluate the model
mae = mean_absolute_error(y_test, y_pred)
print(f"Mean absolute Error: {mae}")
r2 = r2_score(y_test,y_pred)
print(f"R-squared:{r2}")
```

```
Mean absolute Error: 1.20965936644691
R-squared:0.6639816415681596
```

XGBOOST REGRESSOR

```
xg_reg = xgb.XGBRegressor(objective='reg:absoluteerror',
                          colsample_bytree=0.3,
                          learning_rate=0.1,
                          max_depth=5,
                          alpha=10,
                          n_estimators=100)
xg_reg.fit(X_train, y_train)
y_pred = xg_reg.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
print("Mean absolute Error:", mae)
r2 = r2_score(y_test, y_pred)
print('R-squared:', r2)
```

```
Mean absolute Error: 1.0669373727180589
R-squared: 0.7757343457161885
```

KNN REGRESSOR

```
# Initialize the k-NN regressor
knn_regressor = KNeighborsRegressor(n_neighbors=5)

# Fit the model on the training data
knn_regressor.fit(X_train, y_train)

# Make predictions on the test data
y_pred = knn_regressor.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
print(f'Mean absolute Error: {mae}')
r2 = r2_score(y_test, y_pred)
print('R-squared:', r2)
```

```
Mean absolute Error: 1.2591549295774649
R-squared: 0.7529224919567123
```

DECISION TREE REGRESSOR

```
data = DecisionTreeRegressor()
data.fit(X_train, y_train)

y_pred = data.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
print(f'Mean absolute Error: {mae}')
r2 = r2_score(y_test, y_pred)
print('R-squared:', r2)
```

```
Mean absolute Error: 1.1126760563380282
R-squared: 0.5607926294238081
```

ARTIFICIAL NEURAL NETWORK

```
# Build the neural network model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='linear',input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dense(32, activation='linear'),
    tf.keras.layers.Dense(1, activation='linear')
])

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
history = model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test,y_test))
```

```
# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error (MAE): {mae}")
print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared (R2):{r2}")
```

```
3/3 [==============================] - 0s 4ms/step
Mean Absolute Error (MAE): 1.356456652493544
Mean Squared Error (MSE): 4.006219287849888
R-squared (R2):0.7046599673873752
```