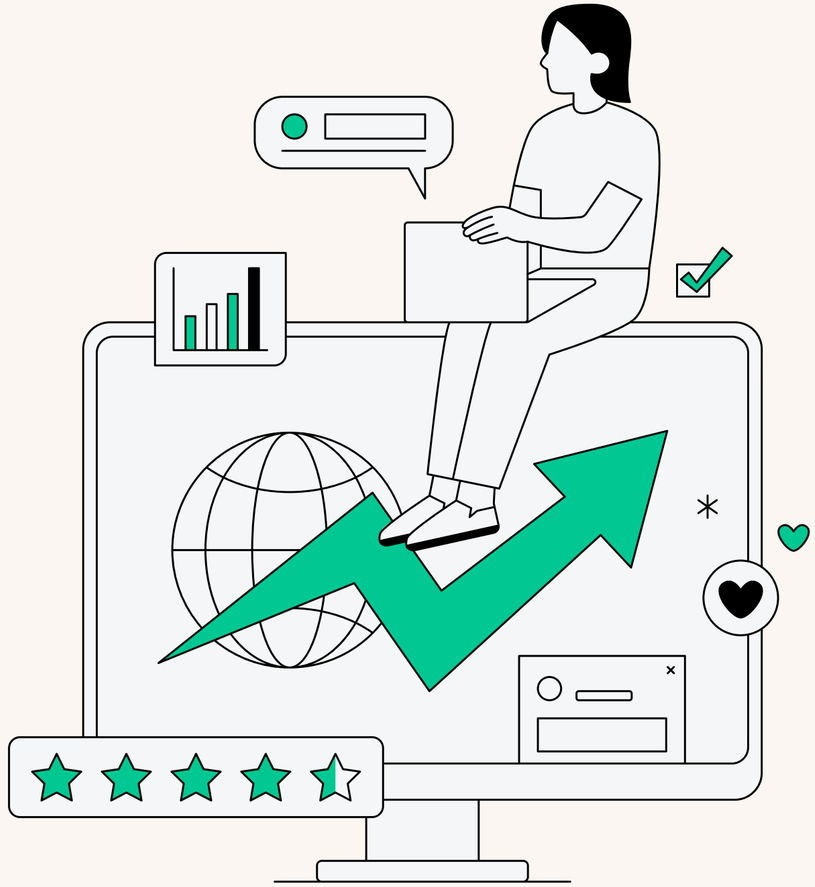
Presented by Syamsul Rizal Fany

Predicting Student Scores Based on Study Hours

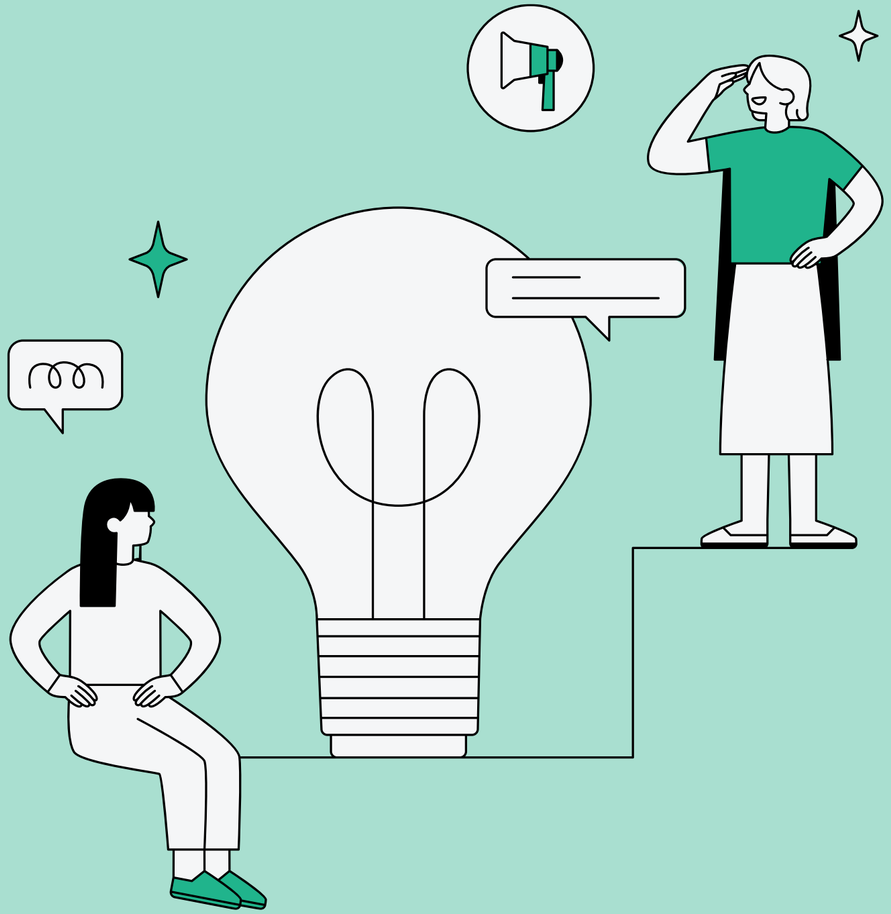
Using Machine Learning



saemfany@gmail.com LinkedIn GitHub

Introduction

This project aims to predict student scores based on the number of hours studied using machine learning models. The models explored in this project are Linear Regression, Decision Tree, and Random Forest. The dataset contains the number of hours students studied and their corresponding scores. We evaluate and compare the performance of these models to determine which provides the best predictions.



Tools

Python

Google Colab

Dataset

The dataset contains the following features:

- Hours (x), Independent variable (feature): The number of hours students studied.
- Scores (y), Dependent variable (target): The scores achieved by the students.

Model Used

Linear Regression

A statistical method for modeling the relationship between a dependent variable and one or more independent variables.

Decision Tree

A non-parametric supervised learning method used for classification and regression.

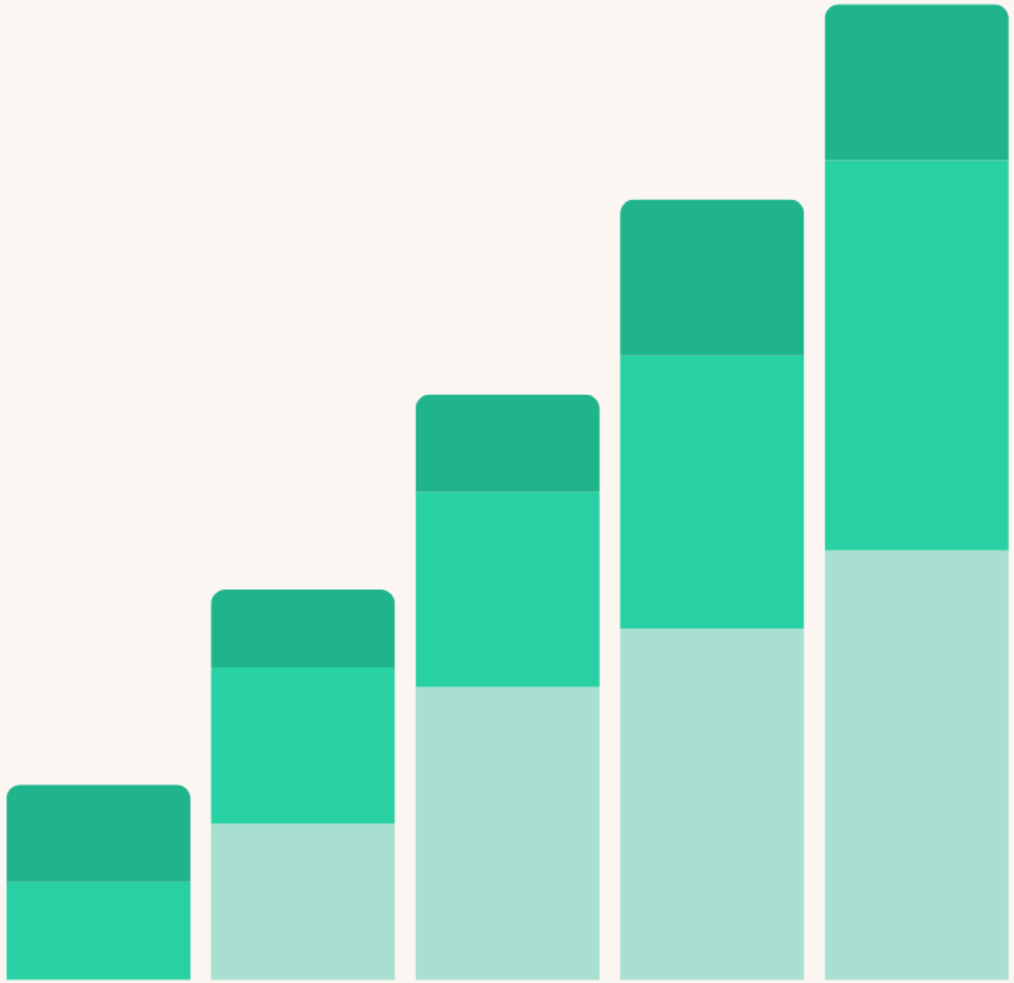
Random Forest

An ensemble learning method that operates by constructing multiple decision trees and outputting the average prediction of the individual trees.

Evaluation Metrics

The models were evaluated based on the following metrics:

- Mean Squared Error (MSE)
- R^2 Score



Data Preprocessing

```
[ ]  #Check Duplicated Data
     duplicate_rows_before = df[df.duplicated()]
     duplicate_rows_before
```

	Hours	Scores

There is no duplicated data.

There is no missing data.

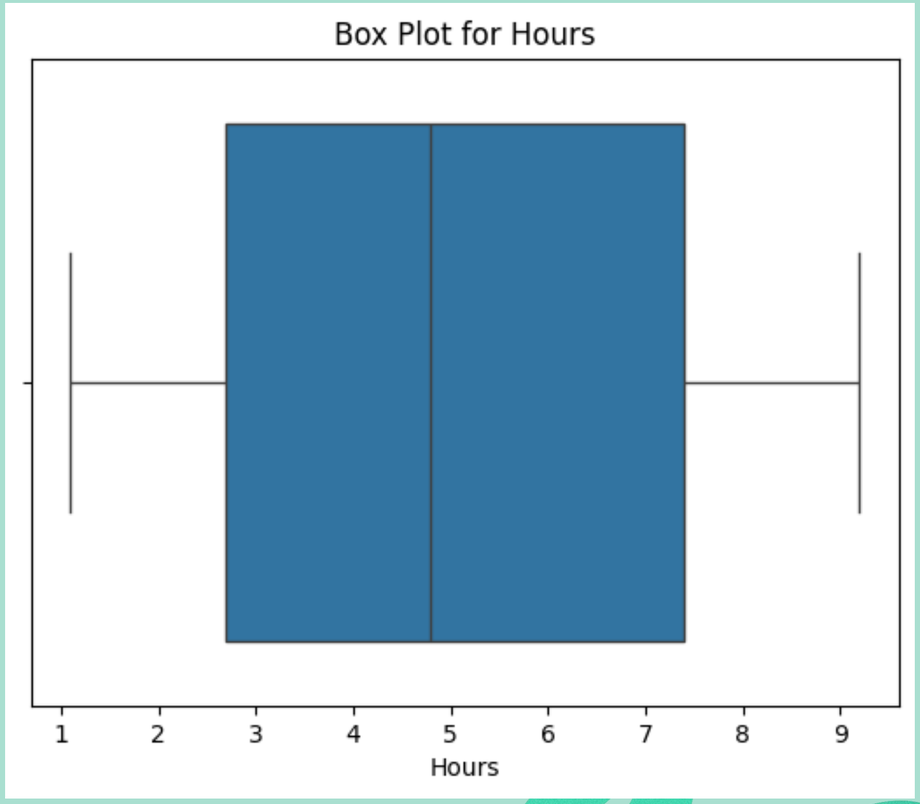
```
#Check missing value
df.isnull().sum()
```

	0
Hours	0
Scores	0

dtype: int64

Data Preprocessing

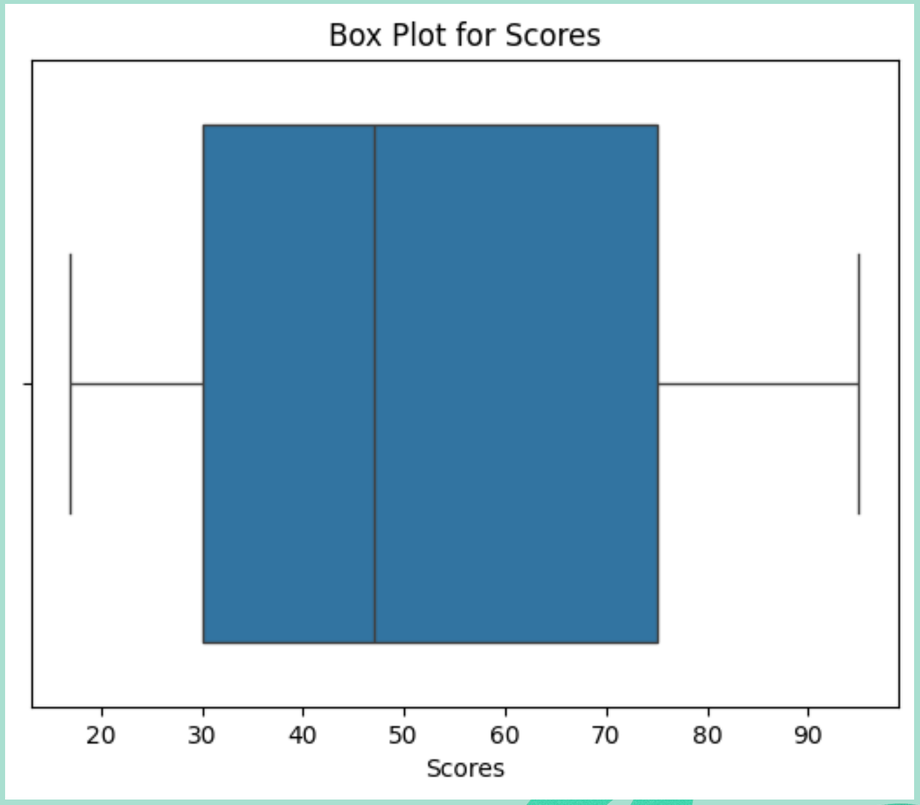
```
# Box plot for 'Hours'
sns.boxplot(x="Hours", data=df)
plt.title('Box Plot for Hours')
plt.show()
```



There are no outliers.

Data Preprocessing

```
# Box plot for 'Scores'
sns.boxplot(x="Scores", data=df)
plt.title('Box Plot for Scores')
plt.show()
```



There are no outliers.

Splitting the Dataset

Before training models, the dataset is typically split into two parts:

- Training Set: Used to train the model.
- Test Set: Used to evaluate the model's performance after training.

```
[ ]  # Split the data (80% training, 20% testing)
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Linear Regression Model

Create and Train the Linear Regression Model

```
[ ]  # Create the Linear Regression model
     lr_model = LinearRegression()

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

Make Predictions

```
[ ]  # Make predictions on the test set
     y_pred_lr = lr_model.predict(X_test)
```

Linear Regression Model

Evaluate the Model

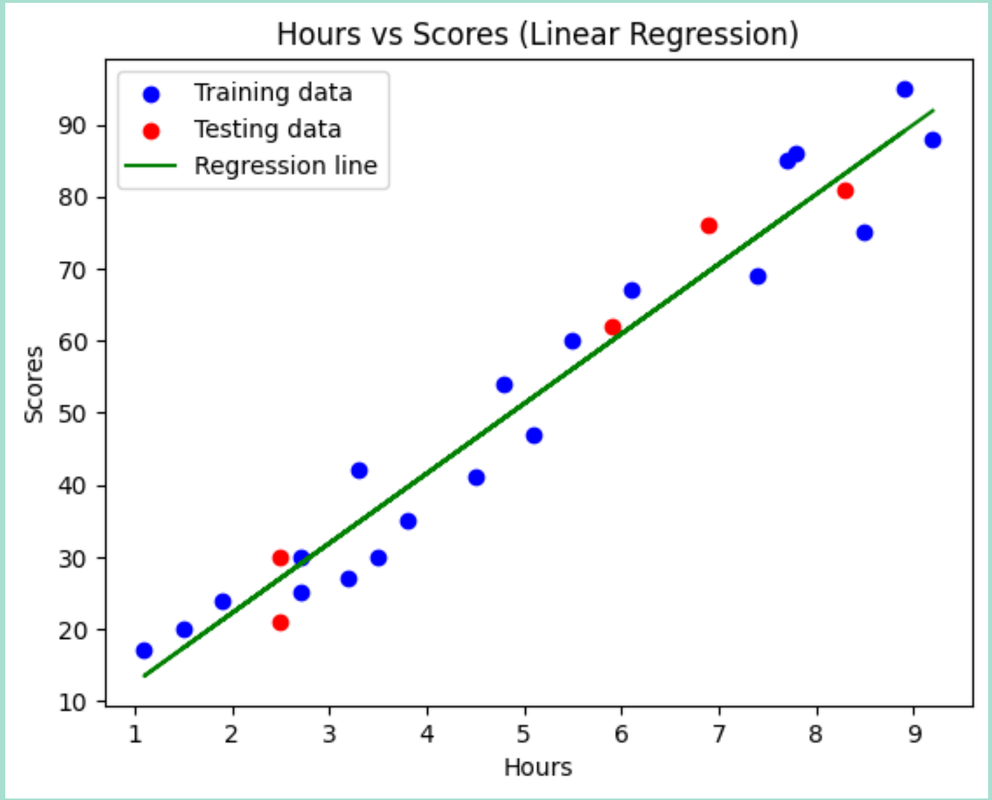
```
[ ]  # Calculate Mean Squared Error and R-squared
     mse_lr = mean_squared_error(y_test, y_pred_lr)
     r2_lr = r2_score(y_test, y_pred_lr)

     print(f"Mean Squared Error: {mse_lr}")
     print(f"R-squared: {r2_lr}")
```

Evaluation Metrics

```
Mean Squared Error: 18.943211722315272
R-squared: 0.9678055545167994
```

Linear Regression Model

Visualize the linear regression line with training data and testing data



Decision Tree Model

Create and Train the Decision Tree Model

```
# Create the Decision Tree Regressor
dt_model = DecisionTreeRegressor(random_state=42)

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

Make Predictions

```
[ ]  # Predict on the test data
     y_pred_dt = dt_model.predict(X_test)
```

Decision Tree Model

Evaluate the Model

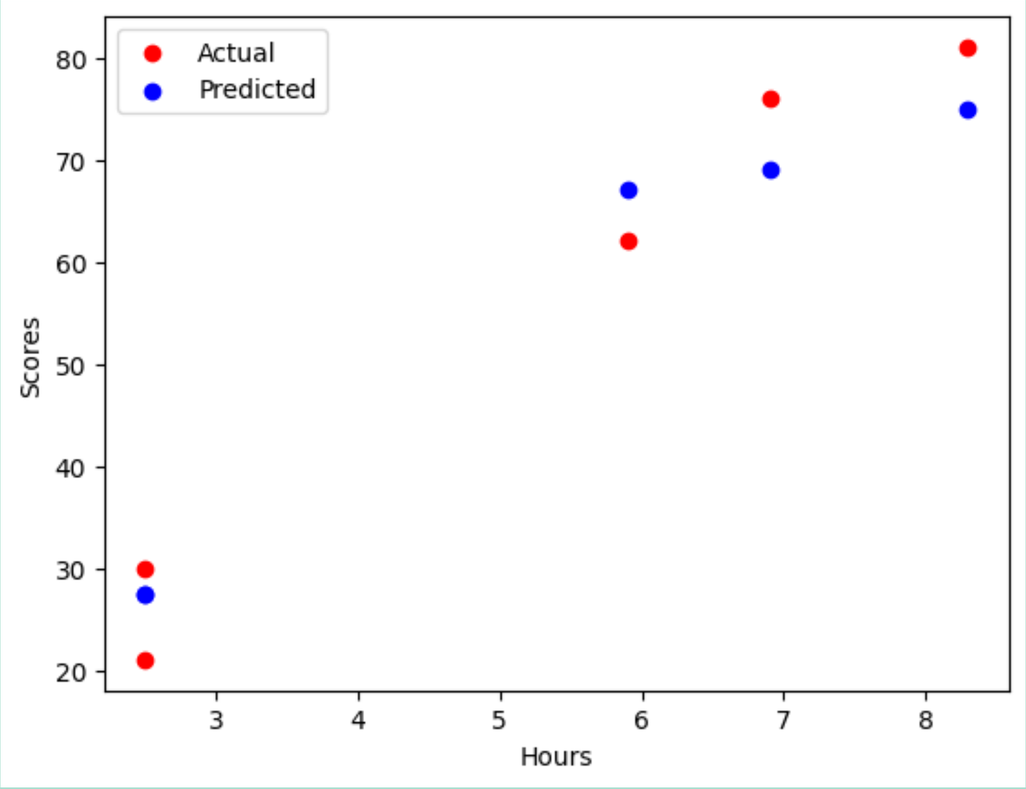
```
[ ]  # Calculate Mean Squared Error
     mse_dt = mean_squared_error(y_test, y_pred_dt)

     # Calculate R-squared Score
     r2_dt = r2_score(y_test, y_pred_dt)

     print(f'Mean Squared Error: {mse_dt}')
     print(f'R-squared Score: {r2_dt}')
```

Evaluation Metrics

```
Mean Squared Error: 31.7
R-squared Score: 0.9461250849762066
```

Decision Tree Model

Plot the actual vs predicted values to see how well the model fits



Random Forest Model

Create and Train the Random Forest Model

```
[ ]  # Create the Random Forest Regressor
     rf_model = RandomForestRegressor(n_estimators=100, random_state=42)

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

Make Predictions

```
[ ]  # Predict on the test data
     y_pred_rf = rf_model.predict(X_test)
```

Random Forest Model

Evaluate the Model

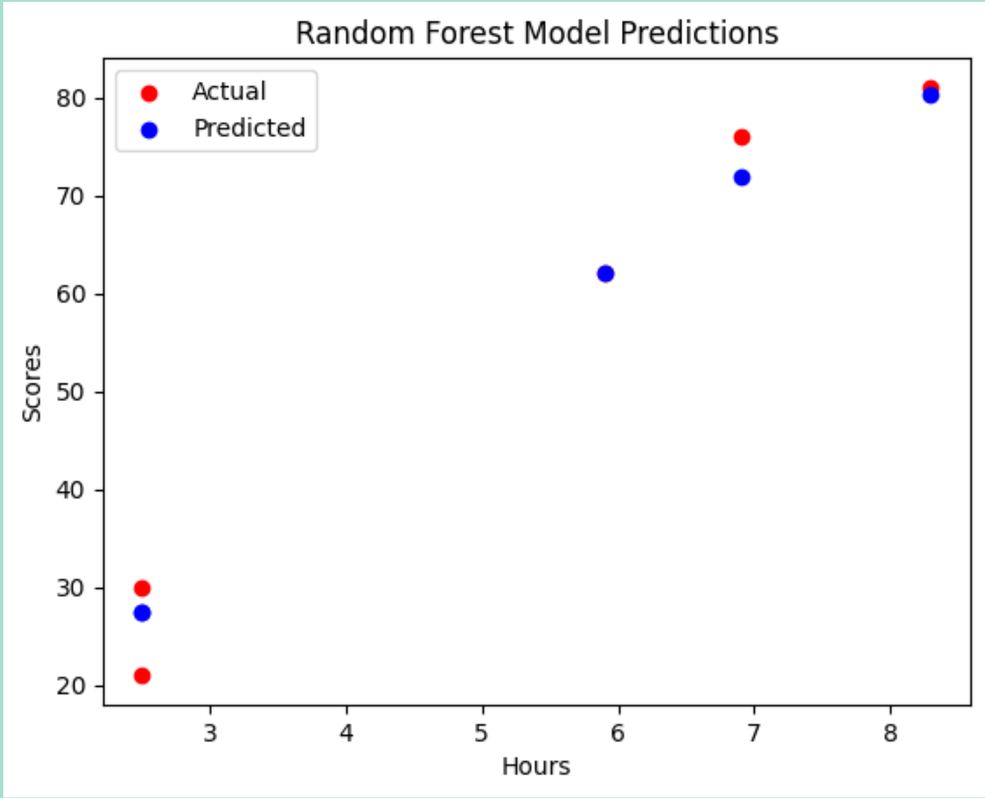
```
# Calculate Mean Squared Error
mse_rf = mean_squared_error(y_test, y_pred_rf)

# Calculate R-squared Score
r2_rf = r2_score(y_test, y_pred_rf)

print(f'Mean Squared Error: {mse_rf}')
print(f'R-squared Score: {r2_rf}')
```

Evaluation Metrics

```
Mean Squared Error: 13.045153611111104
R-squared Score: 0.9778294466160586
```

Random Forest Model

Plot the actual vs predicted values to see how well the model fits



Comparison Between Linear Regression, Decision Tree, and Random Forest Models

Comparison Between Linear Regression, Decision Tree, and Random Forest Models

Summary of Results

Model	Mean Squared Error (MSE)	R-squared
Linear Regression	18.94	0.9678
Decision Tree	31.70	0.9461
Random Forest	13.05	0.9778

Comparison Between Linear Regression, Decision Tree, and Random Forest Models

Conclusions

- Random Forest outperformed both Linear Regression and Decision Tree models with the lowest Mean Squared Error (MSE) of 13.05 and the highest R-squared score of 0.9778. This indicates that Random Forest has the best fit for the data and is capable of effectively capturing underlying relationships.
- Linear Regression also showed good performance, with an MSE of 18.94 and an R-squared score of 0.9678, indicating a strong linear relationship with the data. However, its accuracy was slightly lower compared to the Random Forest model.
- Decision Tree had the highest MSE of 31.70 and the lowest R-squared score of 0.9461, indicating that this model does not generalize as well as the other two models. Decision Tree is more prone to overfitting, which can be a concern if the data contains noise or outliers.

Presented by Syamsul Rizal Fany

Thank you very much!

saemfany@gmail.com LinkedIn GitHub