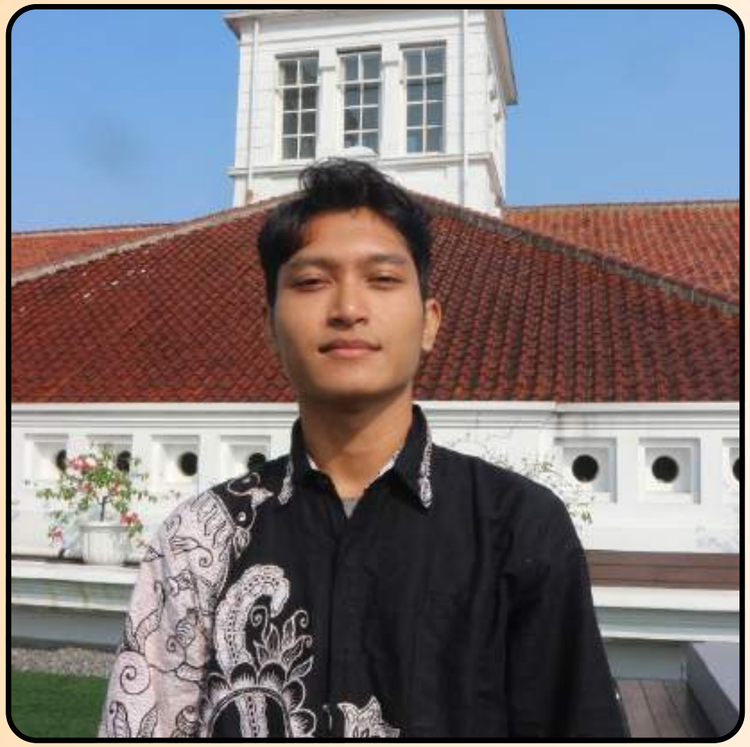
Student score Prediction model

NANA CASMANA ADE WIKARTA

About Me



Nana Casmana Ade W.

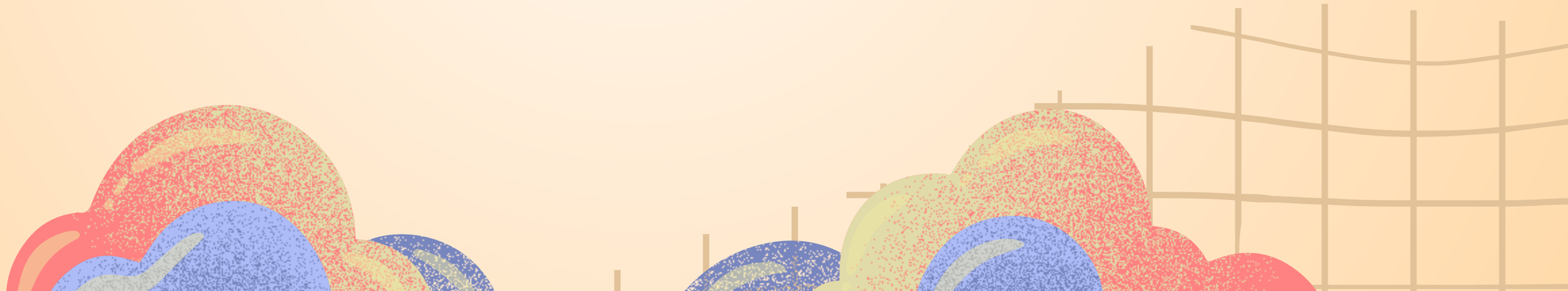
As an **informatics student**, I am **interested** in becoming a **dedicated data analyst** with a **strong interest** in **processing** and **analyzing data** to generate **meaningful insights**. With a solid background in data processing and visualization,
I am able to **identify trends** and **patterns** that support strategic **decision-making**.

Project Overview

This project **aimed to predict student scores based on the number of hours studied**. Machine learning techniques were employed to analyze a dataset containing student information and build predictive models. The project showcases skills in **data cleaning, data preparation, visualization, model building, and evaluation.**

Data Overview

The dataset contains **two columns** – '**Hours**' and '**Scores**'. The 'Hours' column represents **the number of hours studied** by a student, and the 'Scores' column represents the **score obtained** by the student.

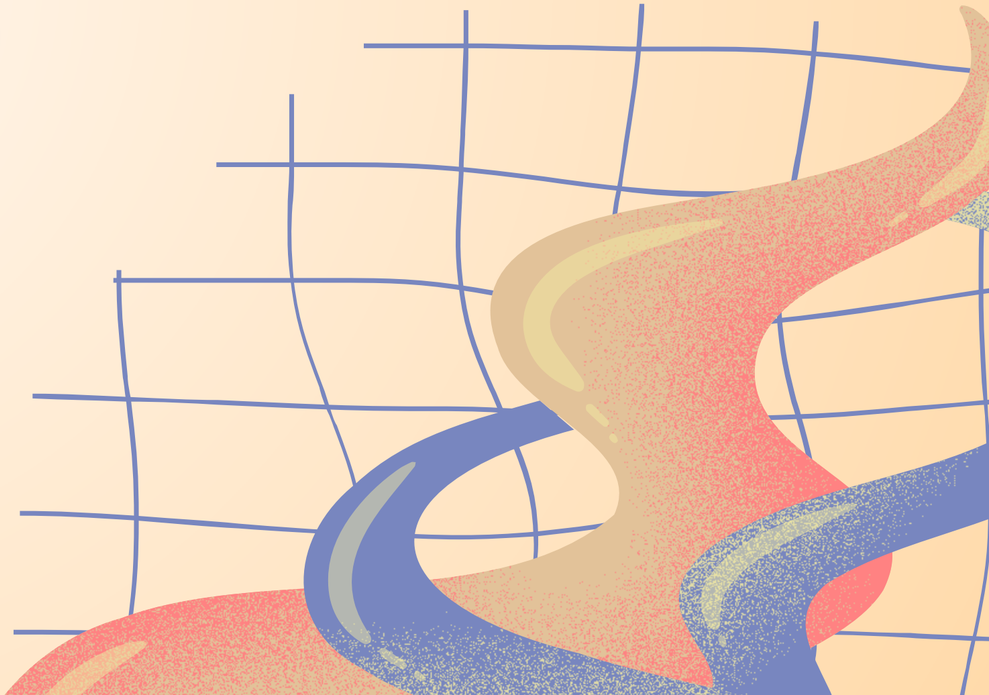


Tools

Microsoft Excel

Google Colab

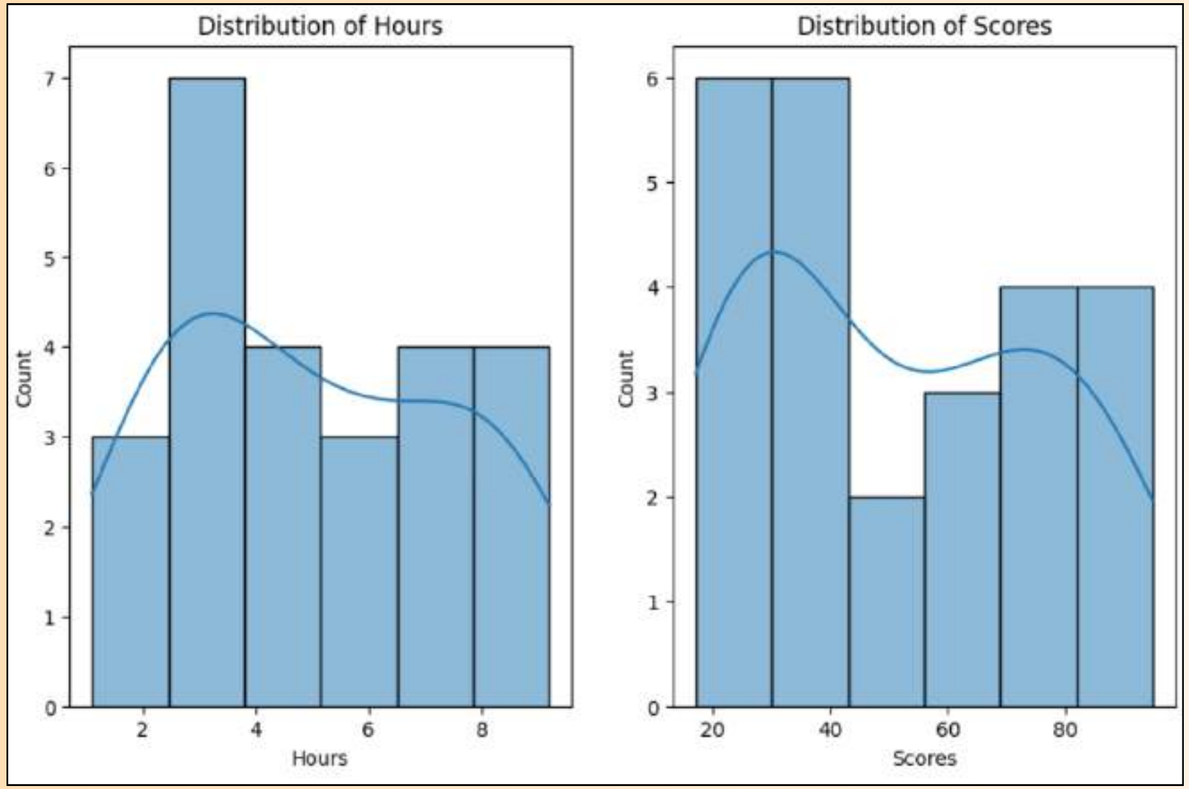
Insight



```
# Check missing values
print("")
print(data.isnull().sum())

# Check duplicate data
print("\nDuplicated data: ")
print(data.duplicated().sum())
data.shape

# Handling Missing Values (If there is)
# data = data.drop_duplicates()
```

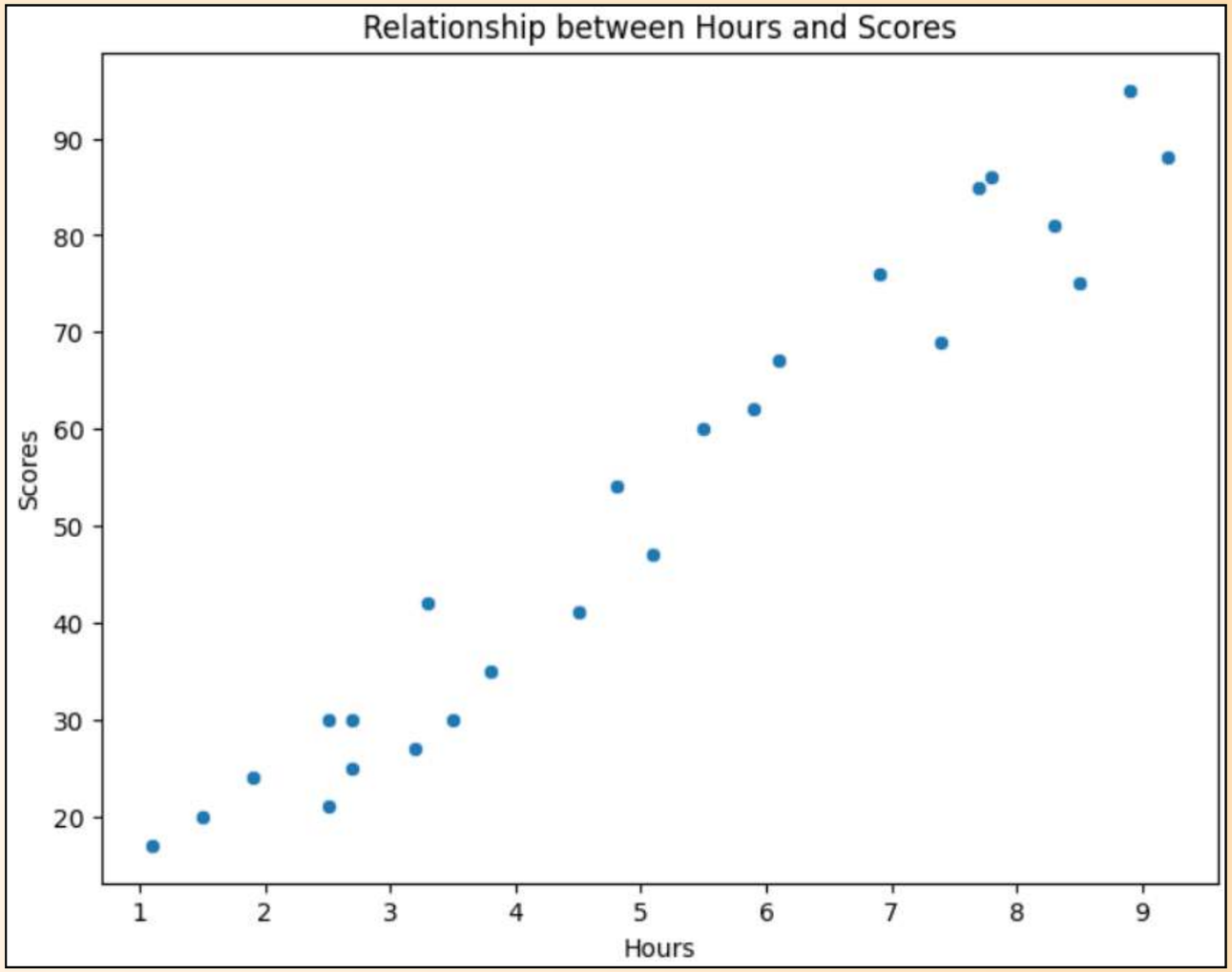
```
Hours     0
Scores    0
dtype: int64

Duplicated data:
0
(25, 2)
```

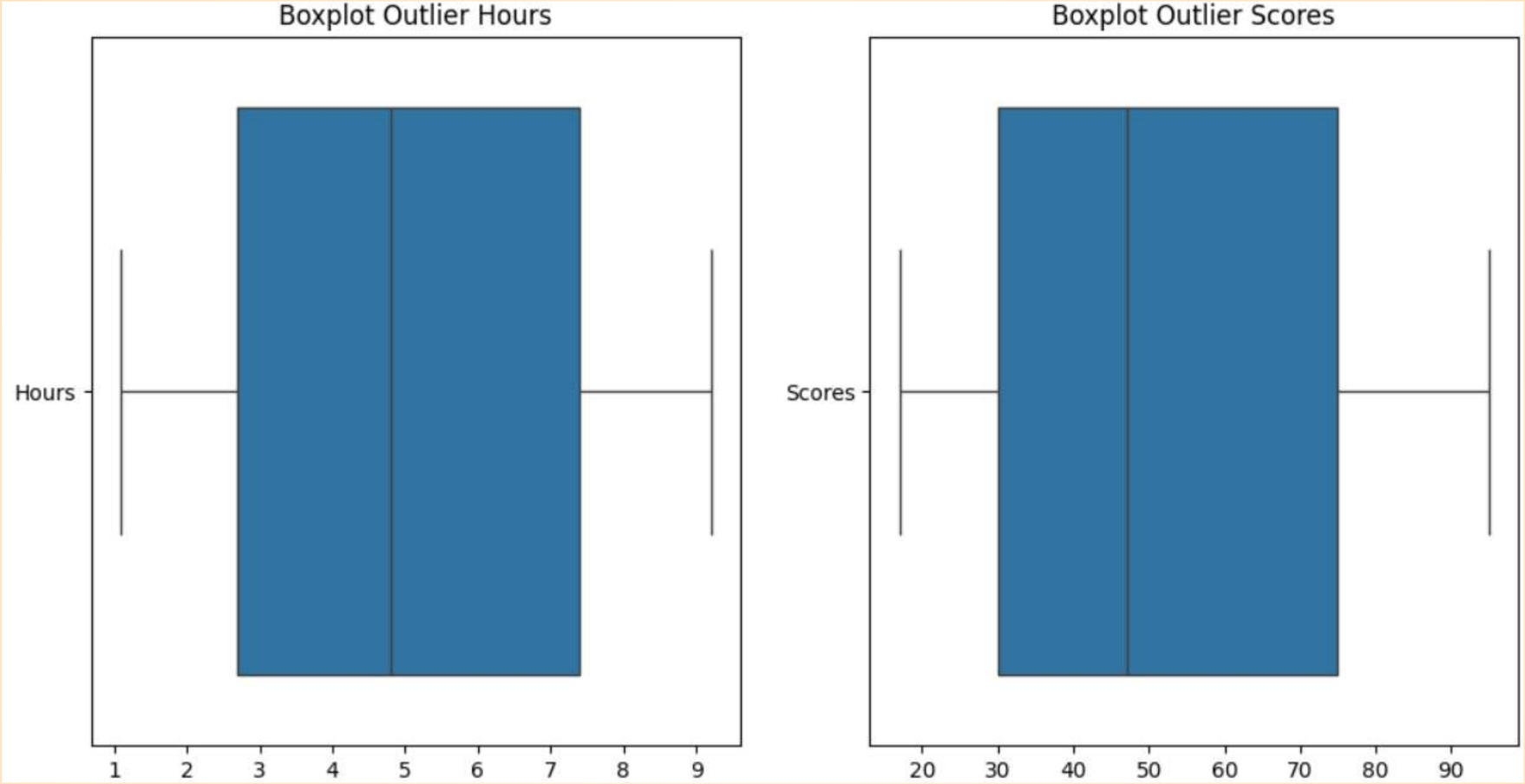
This dataset **has no missing values** or **data duplicated.**



- First image suggests a **skewed distribution of study hours,** with more students **concentrated** towards studying fewer hours **(2–4 hours)** and a smaller proportion studying for significantly longer durations.
- Second image suggests a **bell-shaped**, it means have **normal distribution.**

The scatter plot **shows** a **positive correlation** between scores and hours studied. This means there's a **general upward trend**, where students who tend to study for more hours tend to achieve higher scores.





The dataset is **consistent** with **no outliers**, indicating that the data collection process was **robust** and **did not capture** any **extreme** or anomalous **values**.

```
# Separating features and targets
x = data_clean[['Hours']]
print(x.head())

y = data_clean[['Scores']]
print(y.head())

# Divide the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state = 42)
```

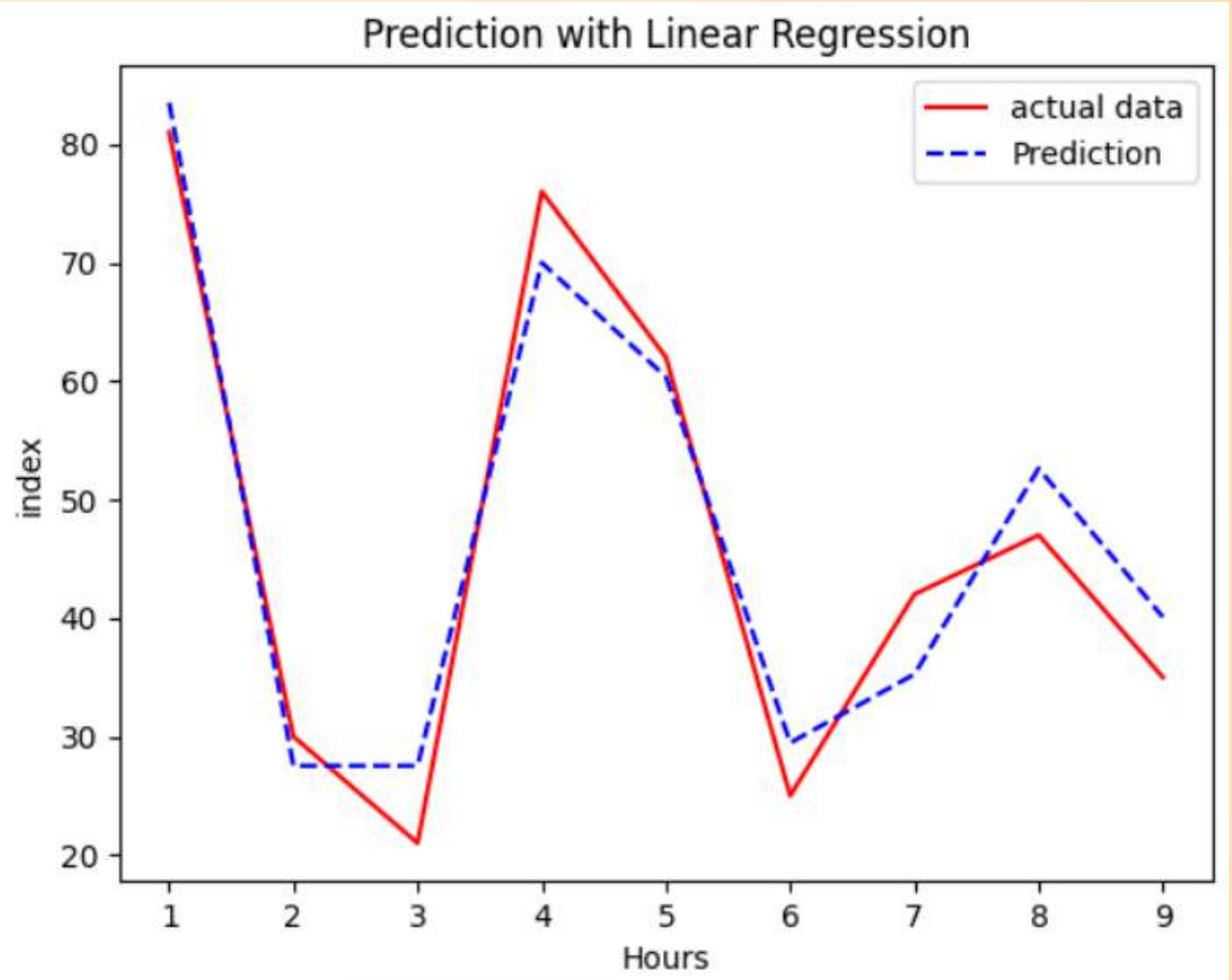
Here, we use **67%** data for **training** and **33%** data for **testing**.

```
# Train model linear regression
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Predict with regresi linear
y_pred_linear = linear_model.predict(X_test)

# Evaluation metrics
mse_linear = mean_squared_error(y_test, y_pred_linear)
r2_linear = r2_score(y_test, y_pred_linear)

# Display
print(f'Linear Regression Mean Squared Error: {mse_linear}')
print(f'Linear Regression R-squared: {r2_linear}')
```

```
Linear Regression Mean Squared Error: 24.074686364260987
Linear Regression R-squared: 0.9435488190277577
```

The performance evaluation results of the linear regression model **show good performance** because the **MSE** value is **close to low** and the **R-square** is **close to 1**, indicating that the machine learning model **can predict** most patterns **well** (close to **95%**).

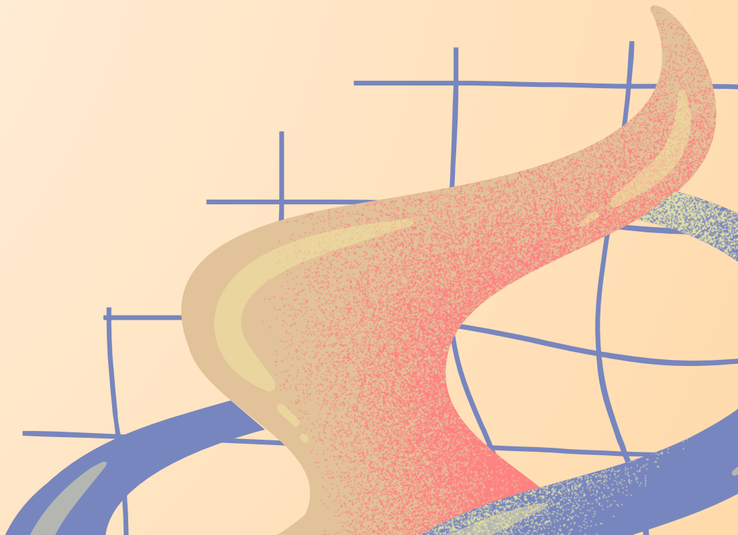
In this graph, it can be seen the **prediction follows the actual data well** and **are not too stretched out**. There are only **small differences** that are consistent throughout the graph.



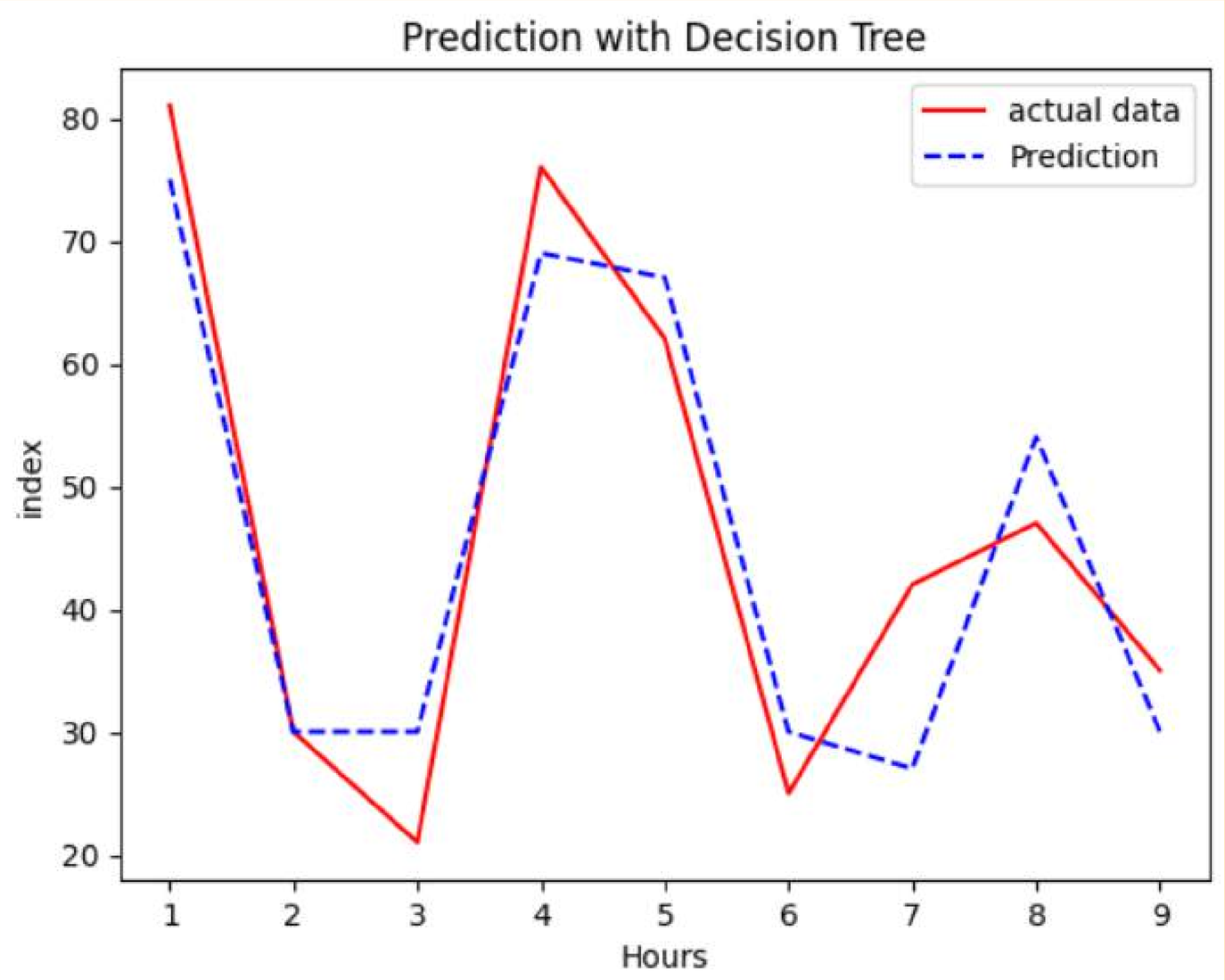
```
# Train model
tree_model =  DecisionTreeRegressor(random_state=42)
tree_model.fit(X_train, y_train)

# Predict with DT
y_pred_tree = tree_model.predict(X_test)

# Evalution model DT
mse_tree = mean_squared_error(y_test, y_pred_tree)
r2_tree = r2_score(y_test, y_pred_tree)

# Display
print(f'Decision Tree Regression Mean Squared Error: {mse_tree}')
print(f'Decision Tree Regression R-squared: {r2_tree}')
```

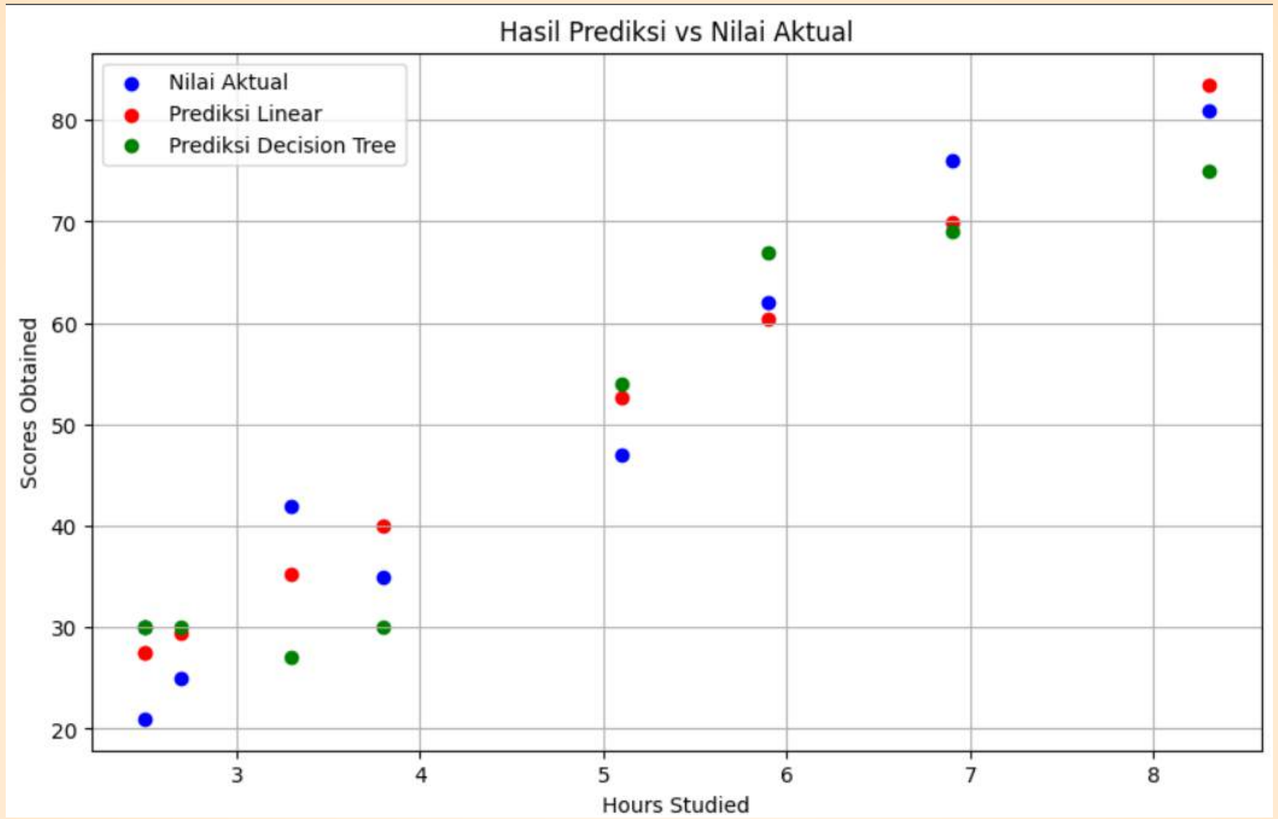
```
Decision Tree Regression Mean Squared Error: 57.22222222222222
Decision Tree Regression R-squared: 0.8658232978230662
```

It can be seen that the **performance result** of the model with the **decision tree obtained** is only **0.865 smaller** than **using Linear Regression**.



In the graph, the model **can** capture the overall trend of the actual data but **still** has **sharp fluctuations** at some points and **more tenuous gaps** than model Linear Regression.





- The **red dots** (linear regression predictions) mostly **follow** the pattern of the **blue dots** (actual values), but there is **some deviation** especially at **higher values**.
- The **green dots** (decision tree predictions) are **sometimes very close** to the **blue dots**, but also show **some larger deviations** in some areas.

Conclusion

Linear Regression is **better** suited for a more linear relationship between study hours and score obtained, which is **indicated** by the **lower MSE** performance and **higher R^2**. This **model** tends **to provide more stable** and **consistent predictions**. Meanwhile, Decision Tree can capture more complex relationships but maybe **less efficient in this case**, as indicated by the **higher MSE** and **lower R^2**. This model seems to be better at capturing variance at higher values but may suffer from **overfitting**.

Recommendation

- **Use Linear Regression** because it is an **efficient model** to use in this case.
- **Optimize** the Decision Tree with **Hyperparameter tunning** to **improve** performance model.
- Use **Cross - Validation** to **ensure** stable model performance and **avoid overfitting**.

Appendix

CLICK HERE

CLICK HERE

Ready to work with me?

: awnana123@gmail.com

: +62 812 1845 1055

: /in/nana-caw/

Thank you