The Superior University

Project Title :

“ Cricket Score Predictior “

Project Details :

Course :	**Artificial Inteligence Lab**
Instructor :	**Sir Rasikh Ali**
Semester :	**03rd**
Section :	**3 - C**
Submission Date :	**02/12/2024**

Name	Roll No	Email
Muhammad Ahmad	SU92-BSAIM-F23-135	su92-bsaim-f23-135@superior.edu.pk

“ Project Overview “

The project aims to predict the final score of an IPL cricket match based on the current state of the match. We use various machine learning models to achieve this, including Decision Tree Regressor, Linear Regression, Random Forest Regressor, and Support Vector Regressor (**SVR**). The data is preprocessed, and the models are trained to predict the final score.

Libraries

The following libraries were used in this project:

pandas: For data manipulation and analysis.

numpy: For numerical computations.

sklearn.preprocessing.LabelEncoder: For encoding categorical labels with a value between 0 and n_classes-1.

sklearn.preprocessing.OneHotEncoder: For converting categorical variables into a form that could be provided to ML algorithms to do a better job in prediction.

sklearn.compose.ColumnTransformer: For applying different preprocessing steps to different subsets of features.

sklearn.model_selection.train_test_split: For splitting the dataset into training and testing sets.

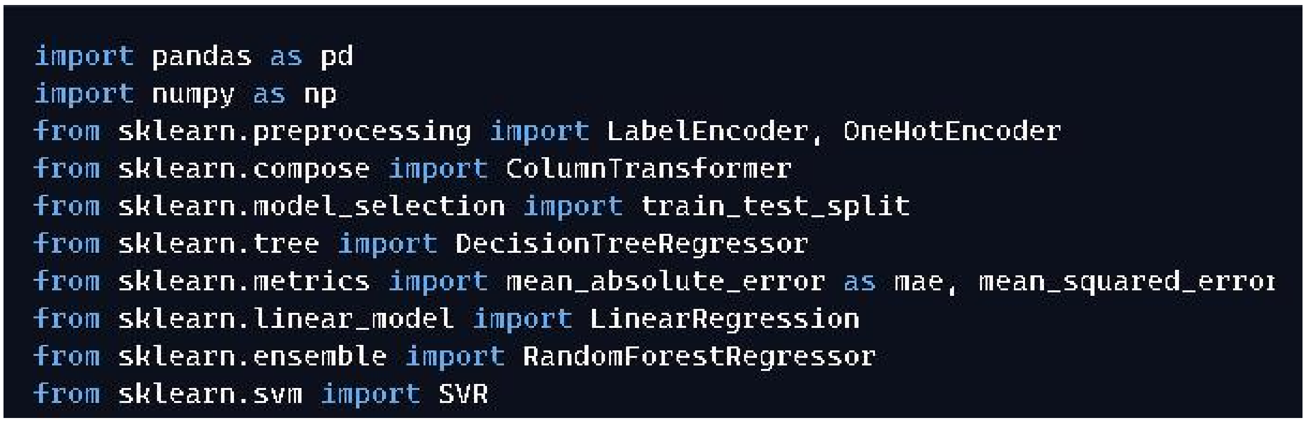
sklearn.tree.DecisionTreeRegressor: For building a decision tree regressor.

sklearn.metrics: For evaluating the performance of the models, including Mean Absolute Error (MAE) and Mean Squared Error (MSE).

sklearn.linear_model.LinearRegression: For building a linear regression model.

sklearn.ensemble.RandomForestRegressor: For building a random forest regressor.

sklearn.svm.SVR: For building a support vector regressor.

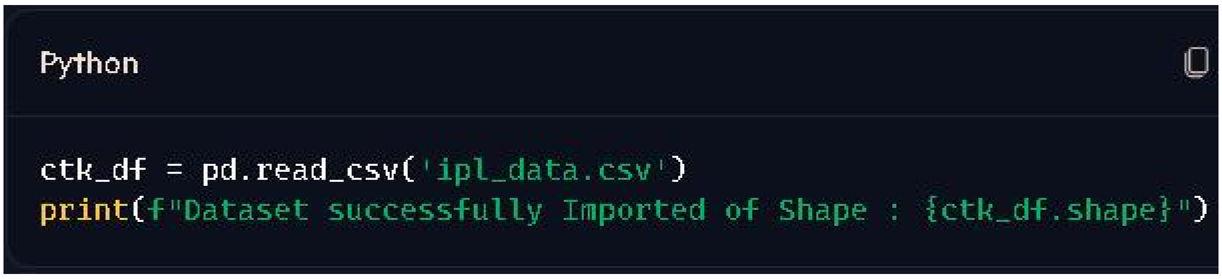


```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error as mae, mean_squared_erro
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
```

Dataset

The dataset used for this project is 'ipl_data.csv'. It contains information about Cricket matches, including team names, player names, and various match details.

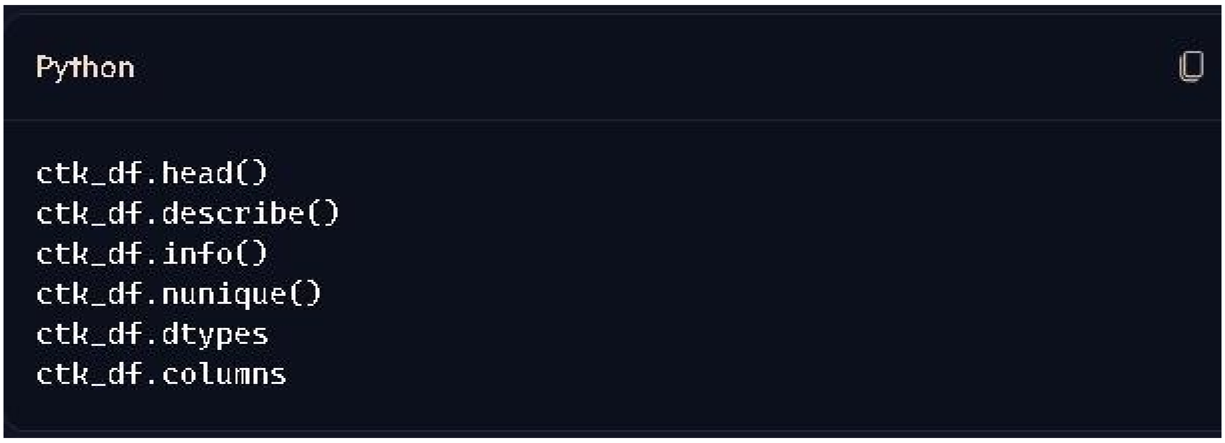
- **Importing the Dataset :**



Python

```python
ctk_df = pd.read_csv('ipl_data.csv')
print(f"Dataset successfully Imported of Shape : {ctk_df.shape}")
```

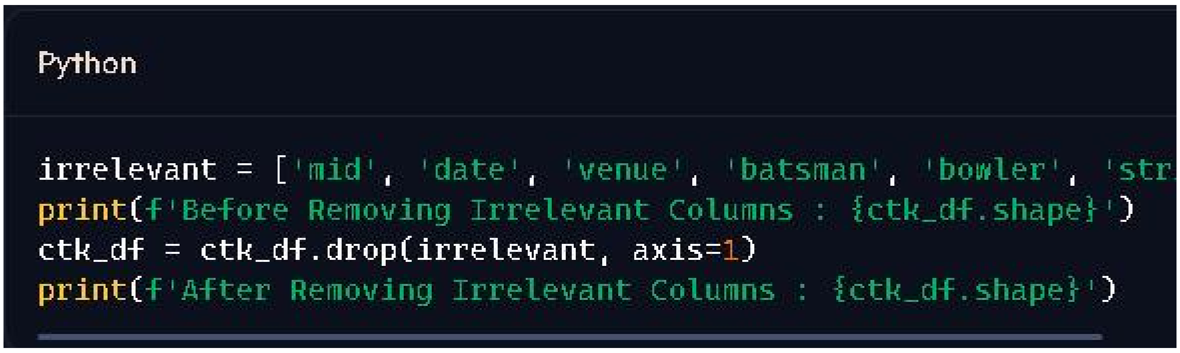
- **Exploratory Data Analysis :**



Python

```python
ctk_df.head()
ctk_df.describe()
ctk_df.info()
ctk_df.nunique()
ctk_df.dtypes
ctk_df.columns
```

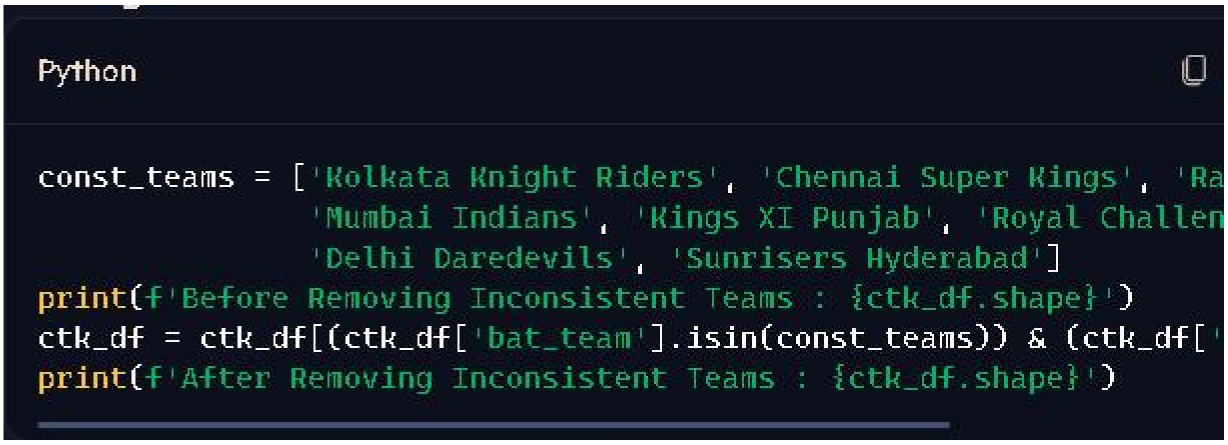
- **Dropping Irrelevant Columns :**



Python

```python
irrelevant = ['mid', 'date', 'venue', 'batsman', 'bowler', 'str
print(f'Before Removing Irrelevant Columns : {ctk_df.shape}')
ctk_df = ctk_df.drop(irrelevant, axis=1)
print(f'After Removing Irrelevant Columns : {ctk_df.shape}')
```

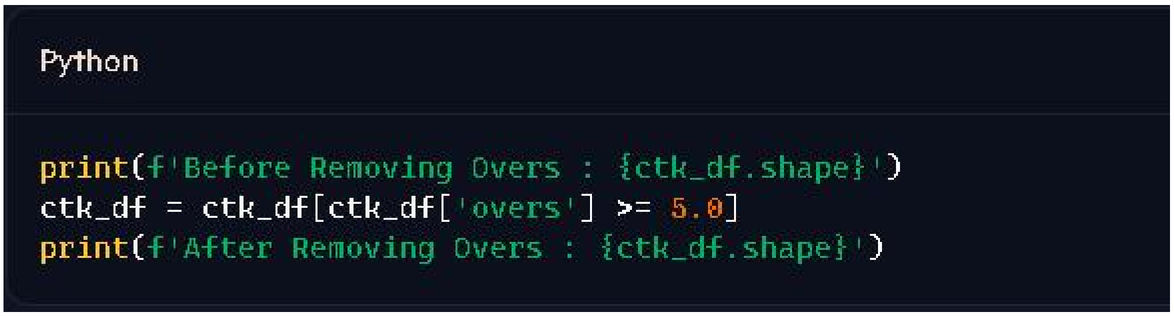
- **Filtering Inconsistent Teams:**



```
Python
```

```python
const_teams = ['Kolkata Knight Riders', 'Chennai Super Kings', 'Ra
               'Mumbai Indians', 'Kings XI Punjab', 'Royal Challen
               'Delhi Daredevils', 'Sunrisers Hyderabad']
print(f'Before Removing Inconsistent Teams : {ctk_df.shape}')
ctk_df = ctk_df[(ctk_df['bat_team'].isin(const_teams)) & (ctk_df['
print(f'After Removing Inconsistent Teams : {ctk_df.shape}')
```

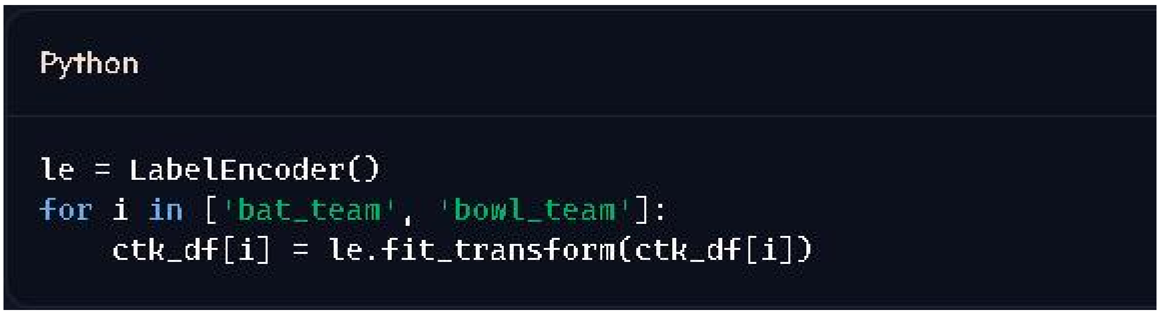
- **Filtering Overs:**



```
Python
```

```python
print(f'Before Removing Overs : {ctk_df.shape}')
ctk_df = ctk_df[ctk_df['overs'] >= 5.0]
print(f'After Removing Overs : {ctk_df.shape}')
```

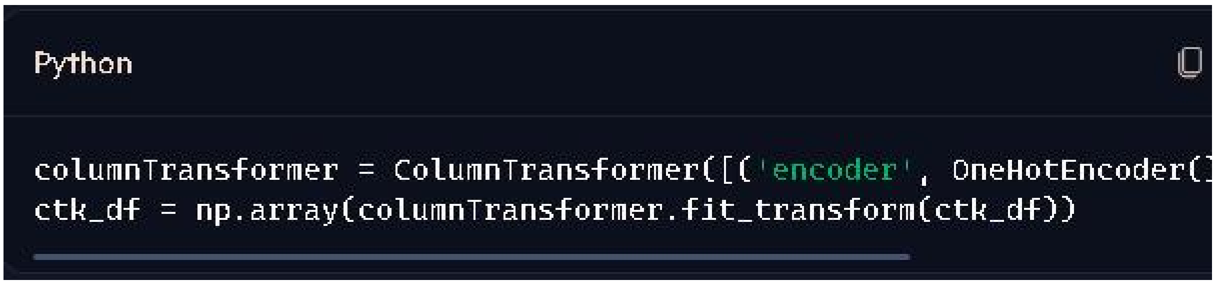
- **Encoding Categorical Variables:**



```
Python
```

```python
le = LabelEncoder()
for i in ['bat_team', 'bowl_team']:
    ctk_df[i] = le.fit_transform(ctk_df[i])
```

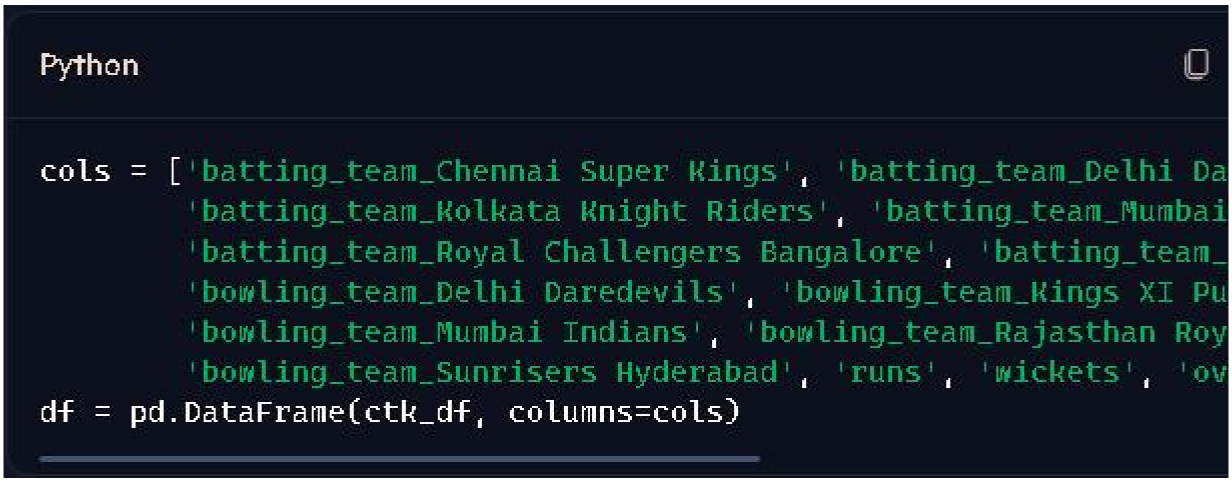
- **One-Hot Encoding:**



```
Python
```

```python
columnTransformer = ColumnTransformer([('encoder', OneHotEncoder()
ctk_df = np.array(columnTransformer.fit_transform(ctk_df))
```

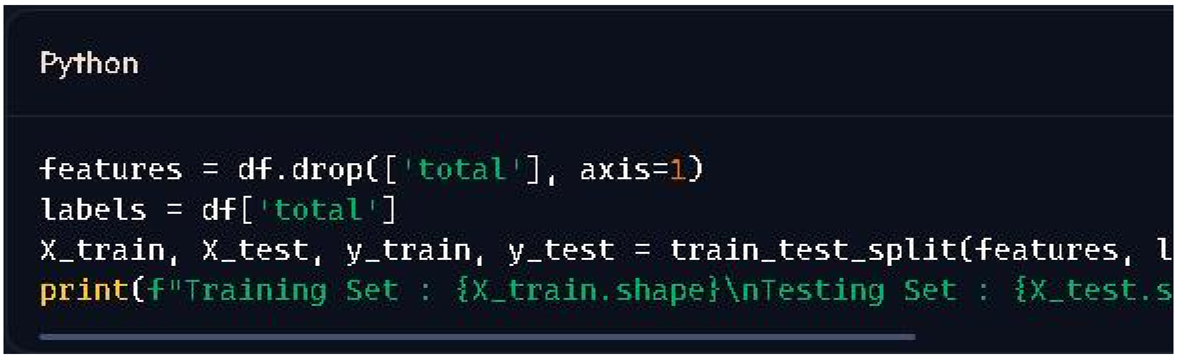
- **Defining Column Names:**



```
Python

cols = ['batting_team_Chennai Super Kings', 'batting_team_Delhi Da
        'batting_team_Kolkata Knight Riders', 'batting_team_Mumbai
        'batting_team_Royal Challengers Bangalore', 'batting_team_
        'bowling_team_Delhi Daredevils', 'bowling_team_Kings XI Pu
        'bowling_team_Mumbai Indians', 'bowling_team_Rajasthan Roy
        'bowling_team_Sunrisers Hyderabad', 'runs', 'wickets', 'ov
df = pd.DataFrame(ctk_df, columns=cols)
```

Model Training and Evaluation

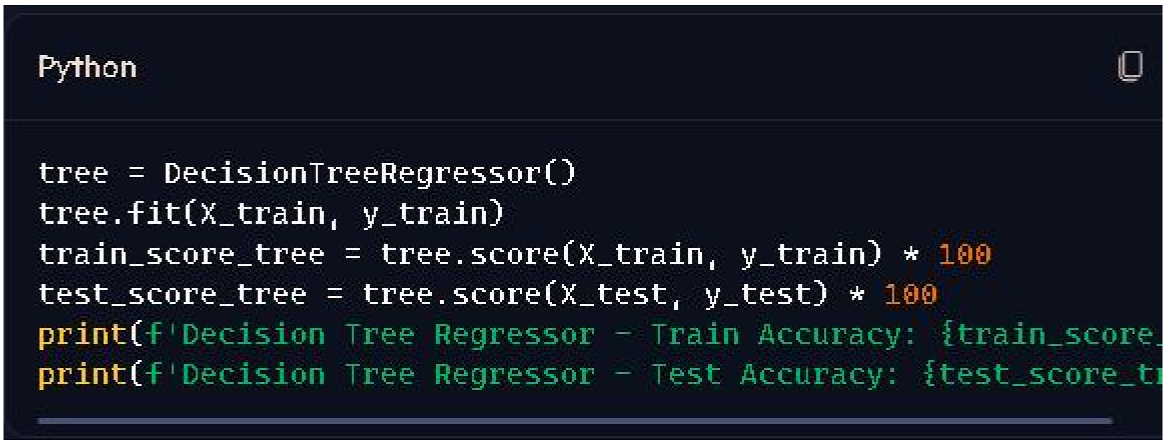
- **Splitting the Data :**



```
Python

features = df.drop(['total'], axis=1)
labels = df['total']
X_train, X_test, y_train, y_test = train_test_split(features, l
print(f"Training Set : {X_train.shape}\nTesting Set : {X_test.s
```

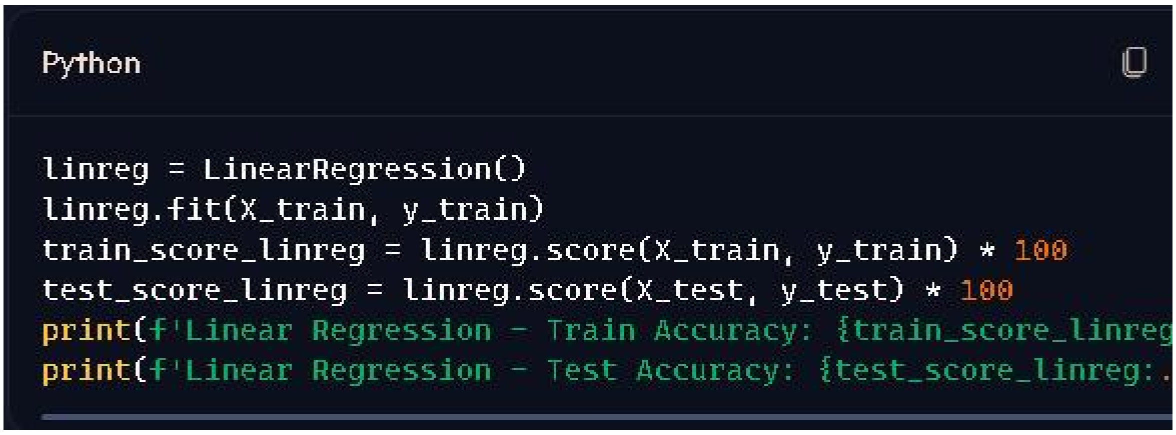
Training and Evaluating Models

- **Decision Tree Regressor :**



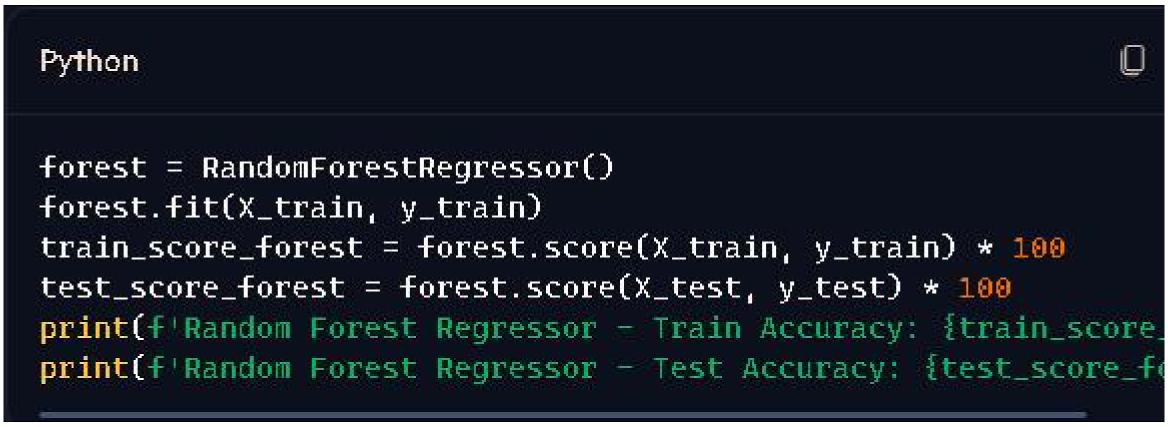
```
Python

tree = DecisionTreeRegressor()
tree.fit(X_train, y_train)
train_score_tree = tree.score(X_train, y_train) * 100
test_score_tree = tree.score(X_test, y_test) * 100
print(f'Decision Tree Regressor - Train Accuracy: {train_score_
print(f'Decision Tree Regressor - Test Accuracy: {test_score_tr
```

- **Linear Regression :**



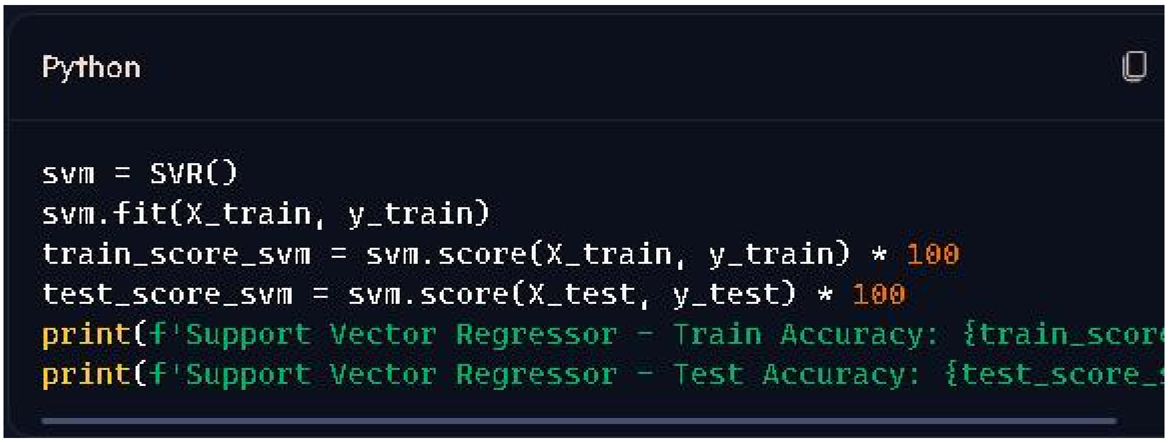
```python
linreg = LinearRegression()
linreg.fit(X_train, y_train)
train_score_linreg = linreg.score(X_train, y_train) * 100
test_score_linreg = linreg.score(X_test, y_test) * 100
print(f'Linear Regression - Train Accuracy: {train_score_linreg
print(f'Linear Regression - Test Accuracy: {test_score_linreg:.
```

- **Random Forest Regressor :**



```python
forest = RandomForestRegressor()
forest.fit(X_train, y_train)
train_score_forest = forest.score(X_train, y_train) * 100
test_score_forest = forest.score(X_test, y_test) * 100
print(f'Random Forest Regressor - Train Accuracy: {train_score_
print(f'Random Forest Regressor - Test Accuracy: {test_score_fo
```

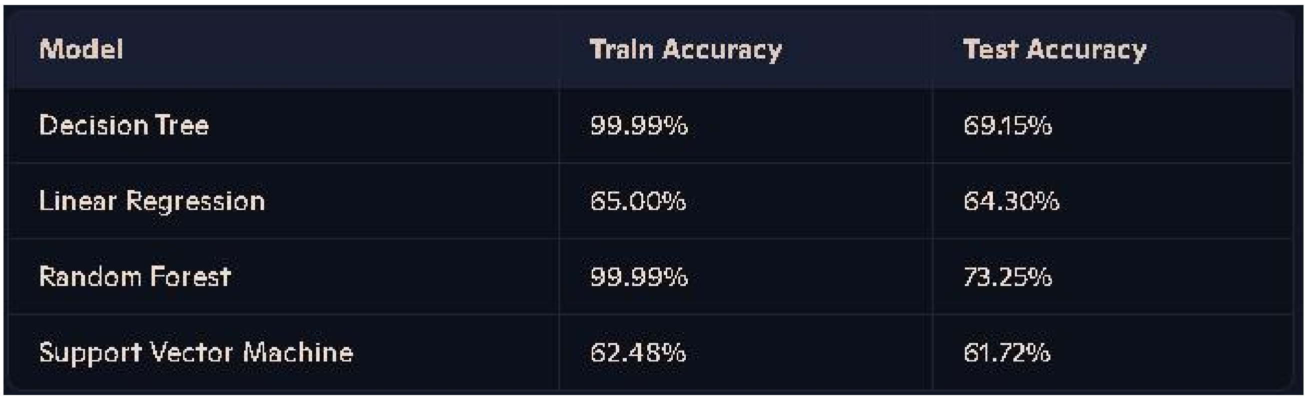
- **Support Vector Regressor (SVR) :**



```python
svm = SVR()
svm.fit(X_train, y_train)
train_score_svm = svm.score(X_train, y_train) * 100
test_score_svm = svm.score(X_test, y_test) * 100
print(f'Support Vector Regressor - Train Accuracy: {train_score
print(f'Support Vector Regressor - Test Accuracy: {test_score_
```

Results

The following table summarizes the accuracy of each model :



Model	Train Accuracy	Test Accuracy
Decision Tree	99.99%	69.15%
Linear Regression	65.00%	64.30%
Random Forest	99.99%	73.25%
Support Vector Machine	62.48%	61.72%

Future Work

Hyperparameter Tuning :

Fine-tune the hyperparameters of the models to improve performance.

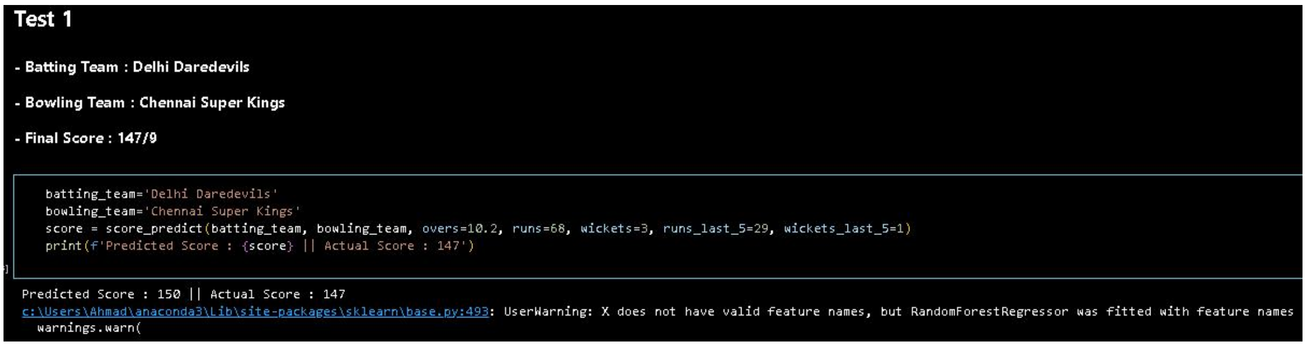
Feature Engineering :

Create additional features that might be relevant to the prediction.

Cross-Validation :

Use cross-validation techniques to better assess model performance.

Outputs

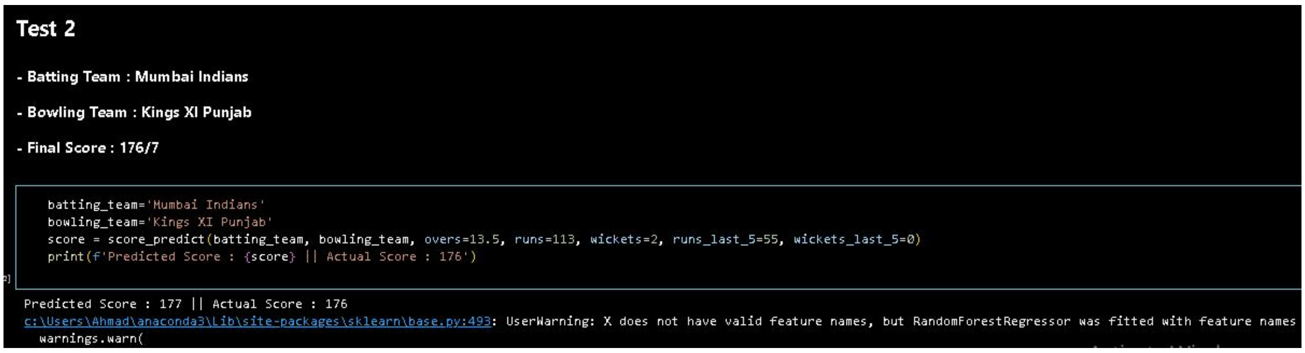


Test 1

- **Batting Team : Delhi Daredevils**
- **Bowling Team : Chennai Super Kings**
- **Final Score : 147/9**

```
batting_team='Delhi Daredevils'
bowling_team='Chennai Super Kings'
score = score_predict(batting_team, bowling_team, overs=10.2, runs=68, wickets=3, runs_last_5=29, wickets_last_5=1)
print(f'Predicted Score : {score} || Actual Score : 147')
```

```
Predicted Score : 150 || Actual Score : 147
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

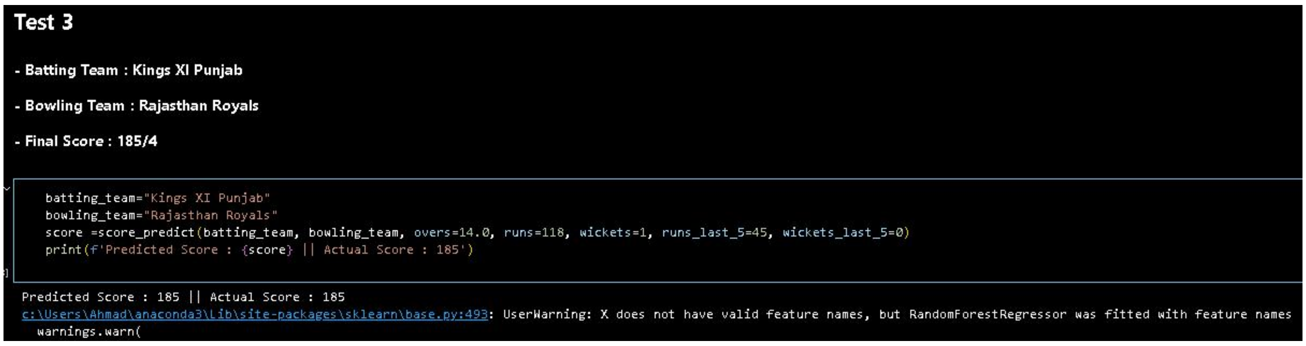


Test 2

- Batting Team : Mumbai Indians
- Bowling Team : Kings XI Punjab
- Final Score : 176/7

```
batting_team='Mumbai Indians'
bowling_team='Kings XI Punjab'
score = score_predict(batting_team, bowling_team, overs=13.5, runs=113, wickets=2, runs_last_5=55, wickets_last_5=0)
print(f'Predicted Score : {score} || Actual Score : 176')
```

```
Predicted Score : 177 || Actual Score : 176
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

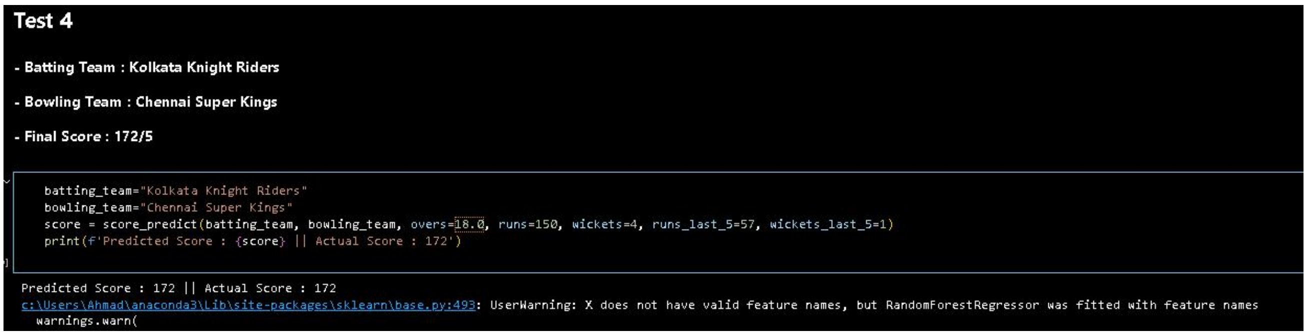


Test 3

- Batting Team : Kings XI Punjab
- Bowling Team : Rajasthan Royals
- Final Score : 185/4

```
batting_team="Kings XI Punjab"
bowling_team="Rajasthan Royals"
score =score_predict(batting_team, bowling_team, overs=14.0, runs=118, wickets=1, runs_last_5=45, wickets_last_5=0)
print(f'Predicted Score : {score} || Actual Score : 185')
```

```
Predicted Score : 185 || Actual Score : 185
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

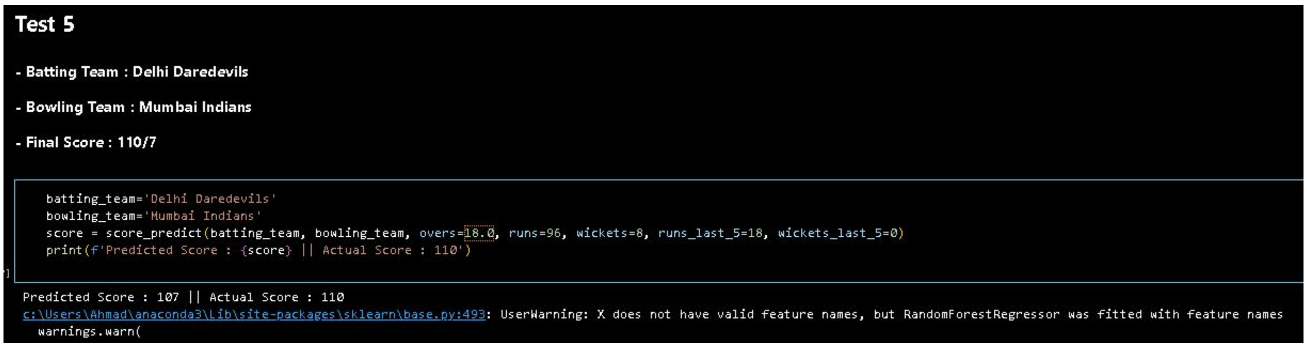


Test 4

- Batting Team : Kolkata Knight Riders
- Bowling Team : Chennai Super Kings
- Final Score : 172/5

```
batting_team="Kolkata Knight Riders"
bowling_team="Chennai Super Kings"
score = score_predict(batting_team, bowling_team, overs=18.0, runs=150, wickets=4, runs_last_5=57, wickets_last_5=1)
print(f'Predicted Score : {score} || Actual Score : 172')
```

```
Predicted Score : 172 || Actual Score : 172
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

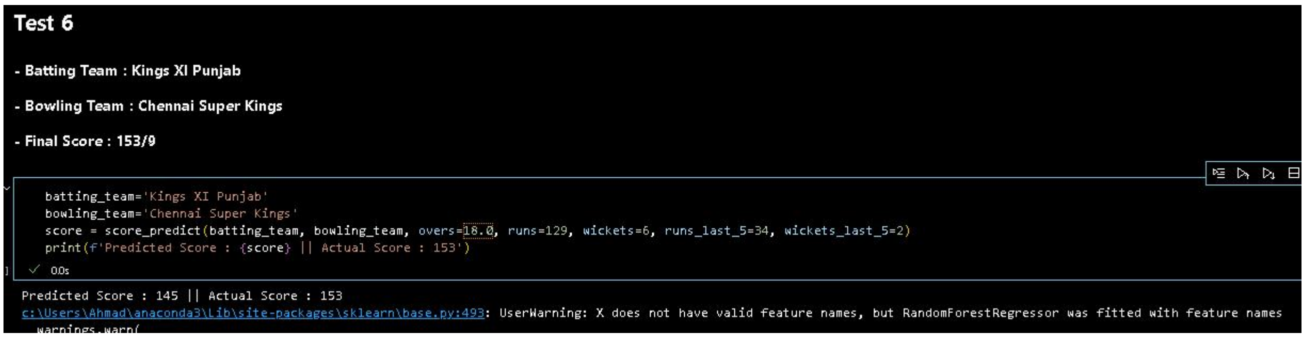


Test 5

- Batting Team : Delhi Daredevils
- Bowling Team : Mumbai Indians
- Final Score : 110/7

```
batting_team='Delhi Daredevils'
bowling_team='Mumbai Indians'
score = score_predict(batting_team, bowling_team, overs=18.0, runs=96, wickets=8, runs_last_5=18, wickets_last_5=0)
print(f'Predicted Score : {score} || Actual Score : 110')
```

```
Predicted Score : 107 || Actual Score : 110
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```



Test 6

- **Batting Team : Kings XI Punjab**

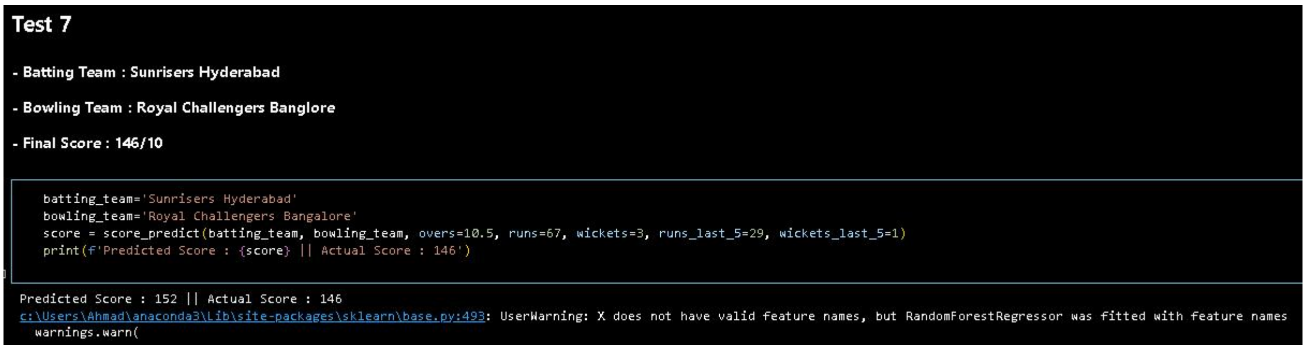
- **Bowling Team : Chennai Super Kings**

- **Final Score : 153/9**

```
batting_team='Kings XI Punjab'
bowling_team='Chennai Super Kings'
score = score_predict(batting_team, bowling_team, overs=18.0, runs=129, wickets=6, runs_last_5=34, wickets_last_5=2)
print(f'Predicted Score : {score} || Actual Score : 153')
```

✓ 0.0s

```
Predicted Score : 145 || Actual Score : 153
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```



Test 7

- **Batting Team : Sunrisers Hyderabad**

- **Bowling Team : Royal Challengers Banglore**

- **Final Score : 146/10**

```
batting_team='Sunrisers Hyderabad'
bowling_team='Royal Challengers Bangalore'
score = score_predict(batting_team, bowling_team, overs=10.5, runs=67, wickets=3, runs_last_5=29, wickets_last_5=1)
print(f'Predicted Score : {score} || Actual Score : 146')
```

```
Predicted Score : 152 || Actual Score : 146
c:\Users\Ahmad\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```