Random Forest

python Copy code

```python
# ============================================================
# Foundations of Data Science Project
# Dataset: Cleaned_Matches_Dataset.csv
# Topic: Random Forest Classifier
# ============================================================

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

Copy code

```python
# 1. Load Dataset
# ================================
df = pd.read_csv("Cleaned_Matches_Dataset.csv")

# Drop unwanted index column if present
if "Unnamed: 0" in df.columns:
    df = df.drop(columns=["Unnamed: 0"])

print("Shape of dataset:", df.shape)
print(df.head())

# ================================
# 2. Prepare Features & Target
# ================================
# Target: whether team1 wins
df_clf = df.dropna(subset=["team1","team2","toss_decision","winner"]).copy()
df_clf["is_team1_winner"] = (df_clf["winner"] == df_clf["team1"]).astype(int)

X = df_clf[["team1","team2","toss_decision"]].copy()
y = df_clf["is_team1_winner"]

# Encode categorical features
for col in X.columns:
    X[col] = LabelEncoder().fit_transform(X[co..astype(str))
```

```
# ================================
# 3. Train-Test Split
# ================================
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=42)

# ================================
# 4. Train Random Forest
# ================================
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)

# ================================
# 5. Results
# ================================
print("\n=== Random Forest Results ===")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```


```
# ================================
# 3. Train-Test Split
# ================================
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=42)

# ================================
# 4. Train Random Forest
# ================================
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)

# ================================
# 5. Results
# ================================
print("\n=== Random Forest Results ===")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Team2 Wins","Team1 Wins"], ytickl
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Random Forest Confusion Matrix")
plt.show()

# ==============================
# 6. Feature Importance
# ==============================
importances = rf.feature_importances_
feat_names = X.columns
plt.barh(feat_names, importances, color="green")
plt.title("Feature Importances (Random Forest)")
plt.show()
```

Guys this 5th and 6th SS is same just one line is extended

```
y_test, y_pred))
, classification_report(y_test, y_pred))


ed)

", cmap="Blues", xticklabels=["Team2 Wins","Team1 Wins"], yticklabels=["Team2 Wins","Team1 Wins"])


Matrix")




es_

color="green")
ndom Forest)")
```

Copy code

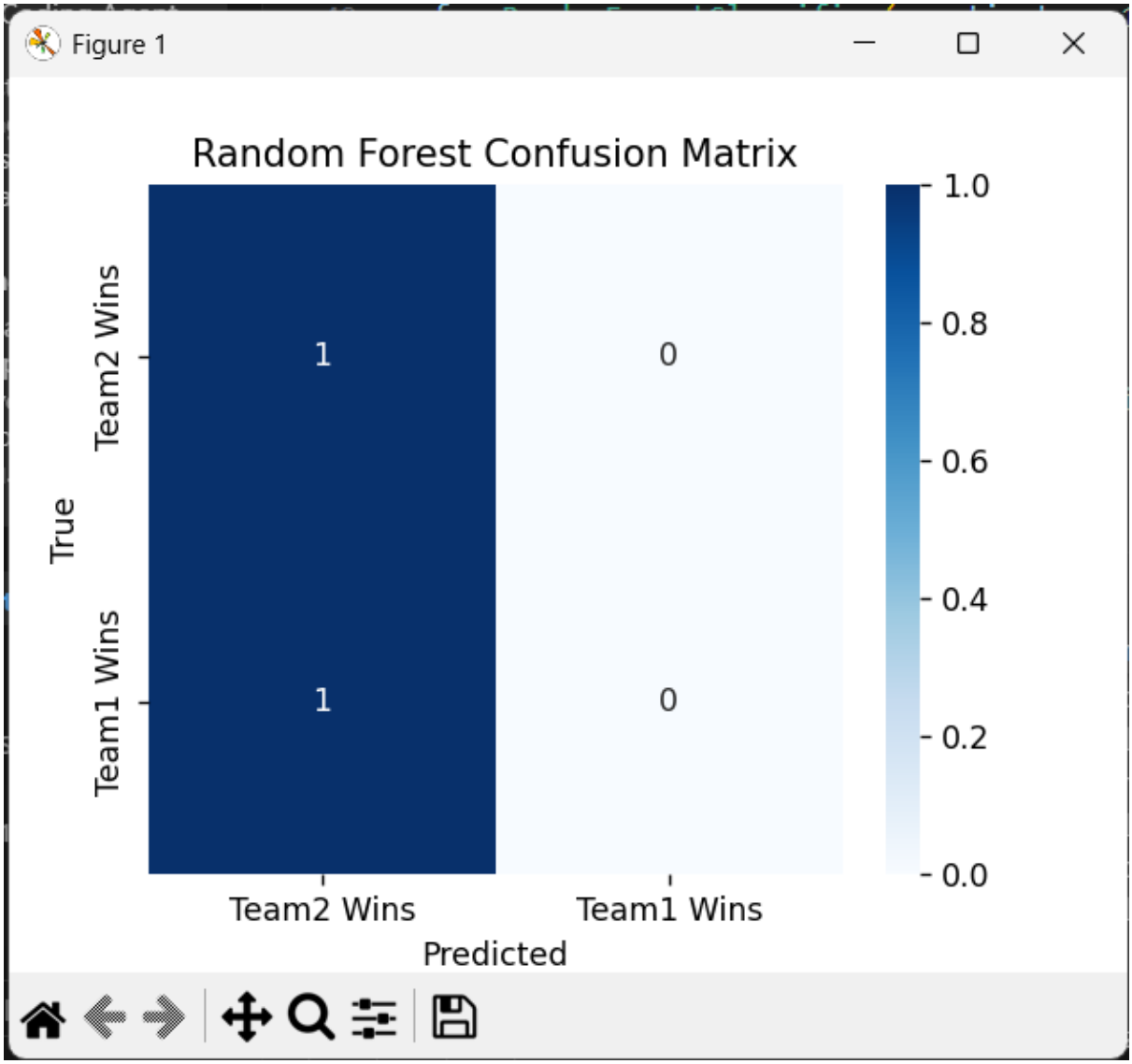
Figure 1
Random Forest Confusion Matrix
Team2 Wins
Team1 Wins
True
1
0
1
0
1.0
0.8
0.6
0.4
0.2
0.0
Team2 Wins
Team1 Wins
Predicted

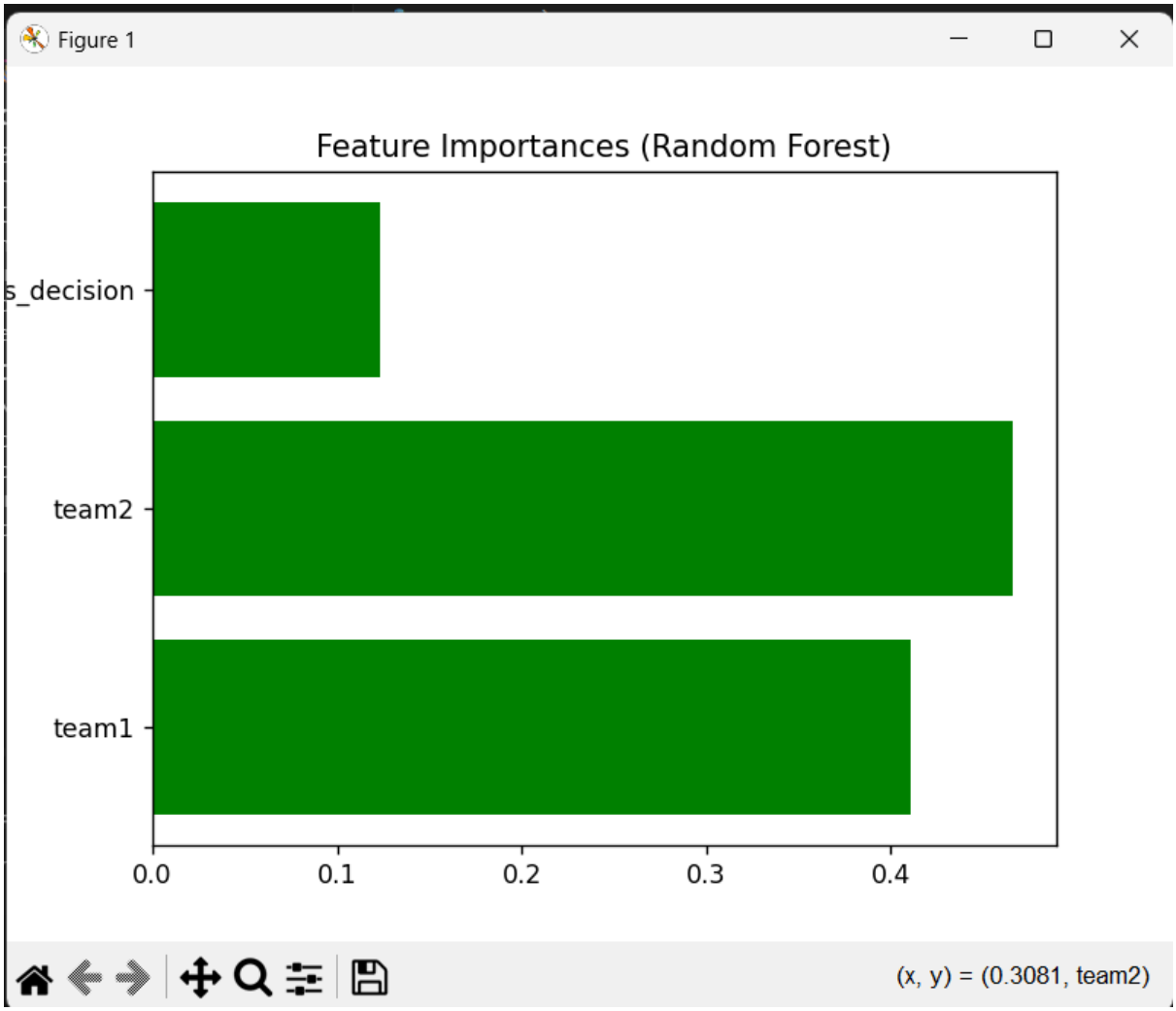
Figure 1
Feature Importances (Random Forest)
s_decision
team2
team1
0.0
0.1
0.2
0.3
0.4
(x, y) = (0.3081, team2)