Decision Trees

Entropy

Information Gain

ID3 Algorithm

I'll give you Python code that:

1. Calculates **Entropy**.
2. Calculates **Information Gain**.
3. Implements a **Decision Tree Classifier** using sklearn (which by default uses Gini, but we'll force criterion="entropy" to mimic ID3).
4. Small demonstration of **ID3** style working

python Copy code

```python
# ============================================================
# Foundations of Data Science Project
# Dataset: Cleaned_Matches_Dataset.csv
# Topic: Decision Trees (Entropy, Information Gain, ID3 Algorithm)
# ============================================================

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from math import log2
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report
```

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())
```

Copy code

```
# ================================
# 2. Helper Functions: Entropy & Information Gain
# ================================

def entropy(labels):
    """Calculate entropy of a list of labels"""
    n = len(labels)
    counts = pd.Series(labels).value_counts()
    ent = -sum((count/n) * log2(count/n) for count in counts)
    return ent

def information_gain(data, feature, target):
    """Calculate Information Gain of splitting 'data' on 'feature'"""
    total_entropy = entropy(data[target])
    values = data[feature].unique()
    weighted_entropy = 0
    for v in values:
        subset = data[data[feature] == v]
        weighted_entropy += (len(subset)/len(data)) * entropy(subset[target])
    return total_entropy - weighted_entropy
```

```
# Example: Compute entropy and info gain on Winner vs Toss Decision
print("\n=== Entropy & Information Gain ===")
print("Entropy of Winner:", entropy(df["winner"].dropna()))
if "toss_decision" in df.columns:
    ig = information_gain(df.dropna(subset=["winner","toss_decision"]), "toss_decision", "winner")
    print("Information Gain (Winner | Toss Decision):", ig)
```

```
# ================================
# 3. Decision Tree Classifier (ID3-like using Entropy)
# ================================
print("\n=== Decision Tree Classifier (ID3 Algorithm) ===")

# Prepare features
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[col].astype(str))

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=42)
```

```python
# Train Decision Tree using Entropy (ID3 style)
clf = DecisionTreeClassifier(criterion="entropy", random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# ================================
# 4. Visualize Decision Tree
# ================================
plt.figure(figsize=(10,6))
plot_tree(clf, feature_names=X.columns, class_names=["Team2 Wins","Team1 Wins"], filled=True)
plt.title("Decision Tree using Entropy (ID3 Algorithm)")
plt.show()
```