In [162…
```
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd


from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

import matplotlib.pyplot as plt
```

In [163…
```
df = pd.read_csv("hayes-roth.csv")
df.head()
```

Out[163…

	92	2	1	1.1	2.1	1.2
0	10	2	1	3	2	2
1	83	3	1	4	1	3
2	61	2	4	2	2	3
3	107	1	1	3	4	3
4	113	1	1	3	2	2

Preprocessing

In [164…
```
# Renaming Columns
col_names = ['name', 'hobby', 'age', 'educational_level', 'marital_status', 'classe']
df.columns = col_names

df.head()
```

Out[164…

	name	hobby	age	educational_level	marital_status	classe
0	10	2	1	3	2	2
1	83	3	1	4	1	3
2	61	2	4	2	2	3
3	107	1	1	3	4	3
4	113	1	1	3	2	2

In [165…
```
# Dropping name because it has no influence to the result
df = df.drop(['name'], axis=1)
df.head()
```

Out[165…

	hobby	age	educational_level	marital_status	classe
0	2	1	3	2	2
1	3	1	4	1	3
2	2	4	2	2	3
3	1	1	3	4	3
4	1	1	3	2	2

In [166…
```
df.dtypes
```

Out[166…
```
hobby               int64
age                 int64
educational_level   int64
marital_status      int64
classe              int64
dtype: object
```

In [167…
```
df.isnull().sum()
```

Out[167…
```
hobby               0
age                 0
educational_level   0
marital_status      0
classe              0
dtype: int64
```

Train-Test Split

In [168…
```
X = df.values[:, 0:5]
y = df.values[:, 4]
```

In [169…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

Model Training - ID3 Decision Tree i.e., entropy

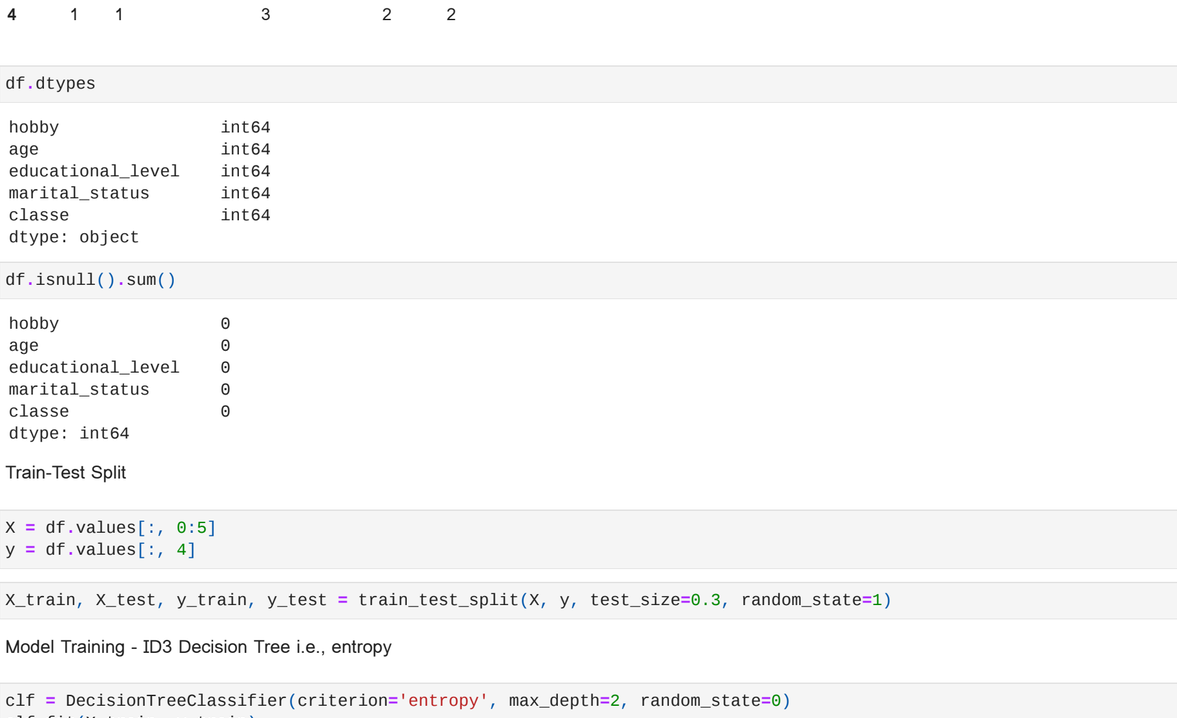
In [170…
```
clf = DecisionTreeClassifier(criterion='entropy', max_depth=2, random_state=0)
clf.fit(X_train, y_train)
```

Out[170…
```
DecisionTreeClassifier(criterion='entropy', max_depth=2, random_state=0)
```

Visualization

In [171…
```
plt.figure(figsize=(12,8))
tree.plot_tree(clf.fit(X_train, y_train))
```

Out[171…
```
[Text(0.4, 0.8333333333333334, 'x[4] <= 1.5\nentropy = 1.551\nsamples = 91\nvalue = [38, 31, 22]'),
 Text(0.2, 0.5, 'entropy = 0.0\nsamples = 38\nvalue = [38, 0, 0]'),
 Text(0.6, 0.5, 'x[4] <= 2.5\nentropy = 0.979\nsamples = 53\nvalue = [0, 31, 22]'),
 Text(0.4, 0.16666666666666666, 'entropy = 0.0\nsamples = 31\nvalue = [0, 31, 0]'),
 Text(0.8, 0.16666666666666666, 'entropy = 0.0\nsamples = 22\nvalue = [0, 0, 22]')]
```



Model Evaluation

In [172…
```
from sklearn.metrics import classification_report
y_pred=clf.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        12
           2       1.00      1.00      1.00        20
           3       1.00      1.00      1.00         8

    accuracy                           1.00        40
   macro avg       1.00      1.00      1.00        40
weighted avg       1.00      1.00      1.00        40
```

In [174…
```
print('Training set score: {:.4f}'.format(clf.score(X_train, y_train)))
print('Test set score: {:.4f}'.format(clf.score(X_test, y_test)))
```

```
Training set score: 1.0000
Test set score: 1.0000
```

1. Why Decision Tree for this dataset?
    - This dataset is small and the independent variables are catagorical and hence the best choice would be to use decision tree.

2. Model Interpretation

- Model although has an accuracy of 100%, it is not over-fitting since the training set and test set score are similar.
- Precision is 100 i.e., the fraction of instances correctly classified as belonging to class 1 out of all instances the model predicted to belong to that class.