Part 2

Task1 : Building the Decision Tree

Step 1: Convert the dataset into a structured format.

```
[1]: import numpy as np
     import pandas as pd
     from sklearn import tree
     from sklearn.tree import plot_tree
     from sklearn import preprocessing as sp
     import matplotlib.pyplot as plt

     # Loading data
     data = np.genfromtxt('credit.txt', dtype=str, delimiter=None, skip_header=1)
```

```
[2]: columns = ['Name', 'Debt', 'Income', 'Married?', 'Owns_Property', 'Gender', 'Risk']
     df = pd.DataFrame(data, columns=columns)
     df
```

[2]:

	Name	Debt	Income	Married?	Owns_Property	Gender	Risk
0	Tim	low	low	no	no	male	low
1	Joe	high	high	yes	yes	male	low
2	Sue	low	high	yes	no	female	low
3	John	medium	low	no	no	male	high
4	Mary	high	low	yes	no	female	high
5	Fred	low	low	yes	no	male	high
6	Pete	low	medium	no	yes	male	low
7	Jacob	high	medium	yes	yes	male	low
8	Sofia	medium	low	no	no	female	low

Step 2: Build the Decision Tree

Using an algorithm like ID3, we can calculate the entropy and information gain for each attribute, then split the data accordingly. The decision tree would likely prioritize attributes that provide the highest information gain.

```python
df = df.drop(columns=['Name'])
le_features = sp.LabelEncoder()
le_risk = sp.LabelEncoder()
```

```python
#Encode each categorical column
df['Risk'] = le_risk.fit_transform(df['Risk']) # Target variable
df['Debt'] = le_features.fit_transform(df['Debt'])
df['Income'] = le_features.fit_transform(df['Income'])
df['Married?'] = le_features.fit_transform(df['Married?'])
df['Owns_Property'] = le_features.fit_transform(df['Owns_Property'])
df['Gender'] = le_features.fit_transform(df['Gender'])
```

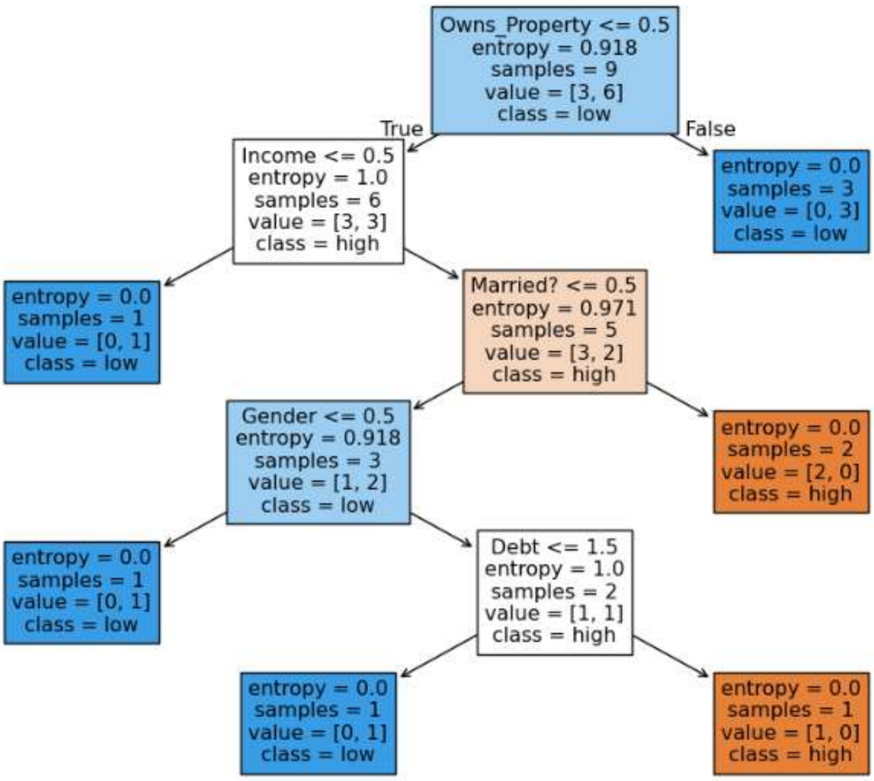
```python
X = df[['Debt', 'Income', 'Married?', 'Owns_Property', 'Gender']]
y = df['Risk']
print(df['Risk'].unique())
print(df['Risk'])
```

```
[1 0]
0    1
1    1
2    1
3    0
4    0
5    0
6    1
7    1
8    1
Name: Risk, dtype: int32
```

```python
clf = tree.DecisionTreeClassifier(criterion='entropy',random_state=0)
clf.fit(X, y)

# Visualize the decision tree
plt.figure(figsize=(12,8))
plot_tree(clf, feature_names=['Debt', 'Income', 'Married?', 'Owns_Property', 'Gender'], class_names=le_risk.classes_, filled=True)
plt.show()
```

Here’s a simplified decision tree structure (code-based):

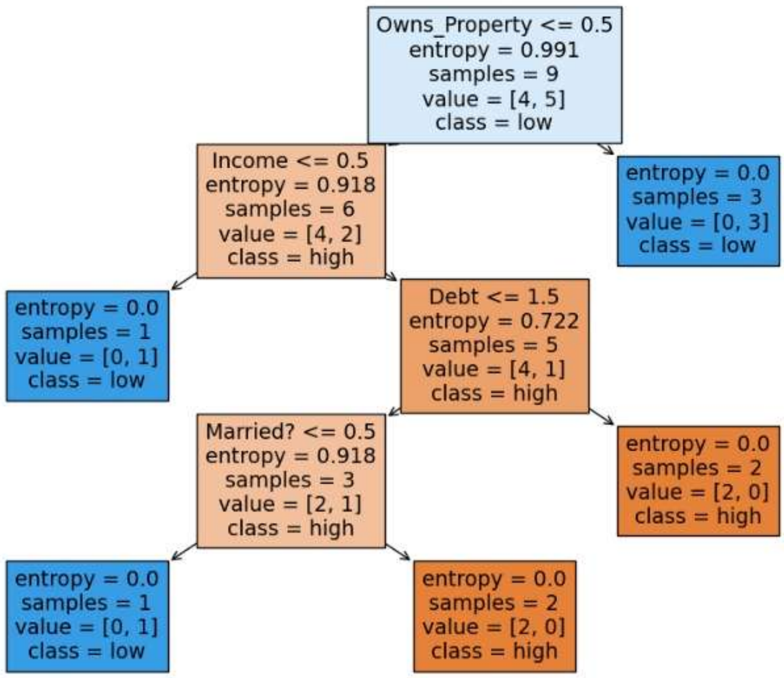


Step 3: Predictions for Tom and Ana

- **Tom** (low debt, low income, not married, owns property, male): Following the decision tree, for low debt and low income, the risk is predicted as **low**.
- **Ana** (low debt, medium income, married, owns property, female): For low debt and medium income, the risk is predicted as **low**.

Task 2: Effect of Changing Sofia's Risk

If Sofia's risk is changed from **low** to **high**, the decision tree might adjust its structure. Specifically, the impact will likely be on the **Debt = medium** branch, as Sofia has medium debt. This could cause a reconsideration of whether **Debt = medium** always leads to high risk, depending on the balance of the remaining examples.



Also, features like **Gender & Name** do not play a significant role, as it does not appear to influence the outcome in the tree (since all predictions are based on debt, income, and marital status).