REPORT

Group Members and Contributions:

Naureen Firdous: Code for decision tree from scratch(30%) , report part 1
Nikitha Krishna Vemulapalli: Code for decision tree from scratch(30%) , part 2 using library, report part 3
Vijayantika Inkulla: Code for decision tree from scratch(40%) , report part 2

PART1:

Libraries used:

Numpy :We have used numpy arrays and the mathematical functions in numpy.

- finfo( ) : Used this to obtain an infinitely small number to prevent log from being 0.
- log2( ) : to find logarithmic base 2
- argmax( ) : to find the attribute with maximum gain
- unique( ) : to get the unique values of an attribute
- numpy array : used to create a data frame

Pandas:

- dataframe : to read the input training data and separate it as header and the values

Challenges :

1. If there are no attribute values while predicting the label of test data using the built decision tree, we are giving it a class label based on the majority class label of the entire training data.
2. On using up all the attributes while building the decision tress, if there is no single class label for that subset of data, we are assigning the majority class label for that branch.
3. If there is a contradiction between the class labels, the tie-breaker is a random class label.

Prediction:

The code predicts the label Yes.

Format of decision tree :

If the attributes are A1, A2, …An and considering that the values as V11, V12…..for attribute A1 and similarly for other attributes, the format is : {A1 : {V11 : {sub-tree-1, V12 : {sub-tree-2}, …} }, where sub-tree-k is the decision tree of subset of data having V1k value for attribute A1. A1 is the root with maximum Information Gain.

PART2:

Decision tree can be implemented using the Scikit learn library. The sklearn.tree.DecisionTreeClassifier( ) method can be used to create a decision tree based on entropy or gini-index. The fit( ) method is used to generate the tree.

Drawback: This cant handle categorical attributes. This can be overcome using encoding. Encoding can be done using LabelEncoder or One-hot encoding. We have used One-hot encoding by creating dummy variables as follows:

```
import pandas as pd
dum=pd.get_dummies(dat[head]).rename(columns=lambda x: head+'_' + str(x))
dat=pd.concat([dat, dum], axis=1)
dat=dat.drop([head], axis=1)
```

Improvements:

The termination criteria can be added to reduce the number of splits when there are larger number of values for an attribute. To avoid overfitting, prepruning can be done which is usually implemented in scikit DecisionTreeClassifier.

PART3:

Some real life applications of decision tree are :

1.**Selecting a flight to travel :** We check first if the flight is available on that day or not. If it is not available, we will look for some other date. But if it is available then we look for some other attribute , say the duration of the flight, and so on continue our search.

2. **Business applications :** Decision trees help organisations to view alternatives to different events that can happen.
3. **Sentiment Analyser, Investment Solutions etc.** are some other examples where decision trees can be used.