Algorithm 1: Decision Trees for classification (DT-C)

1.

The whole algorithms are divided into the following parts.

(1) Importing required libraries and loading the dataset;

(2) Understanding the structure and content of the dataset;

(3) Selecting three independent variables (“pclass”,”age”,”sex”) to build the model;

(4) Finding out the missing data, and cleaning the data by filling in the missing part;

(5) To understand model performance, dividing the dataset into a training set and a test set;

(6) Importing DecisionTreeClassifier Model to perform prediction

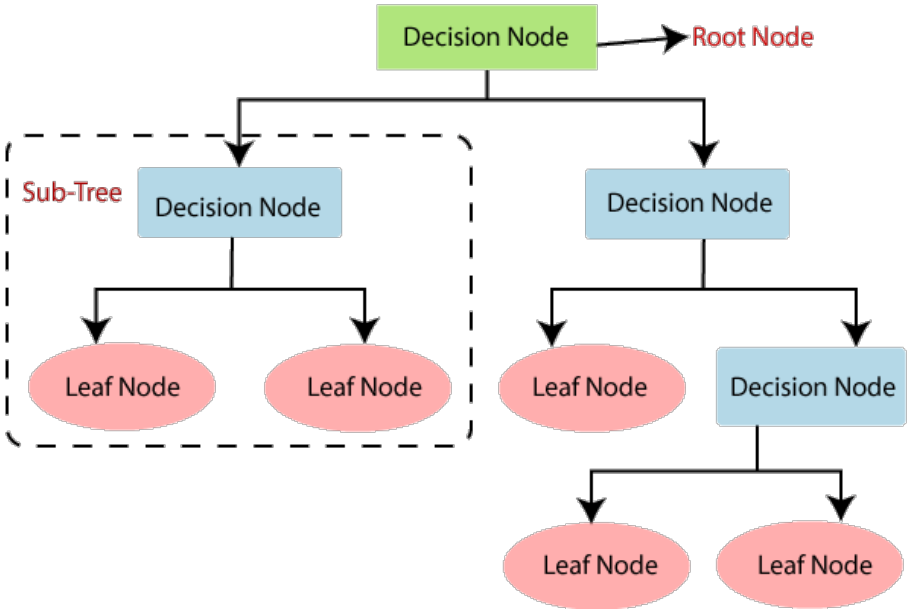
(7) Evaluating the result

Decision Tree Model can help to create a model which is able to predict the value of a target variable based on a series of input variables. It can divide the total dataset into several sub-dataset and is of great significance to future prediction.

2.

(1) Decision Tree has three types of nodes:

- A root node that has no incoming edges and zero or more outgoing edges
- Internal nodes, each of which has exactly one incoming edge and two or more outgoing edges
- Leaf or terminal nodes, each of which has exactly one incoming edge and no outgoing edges

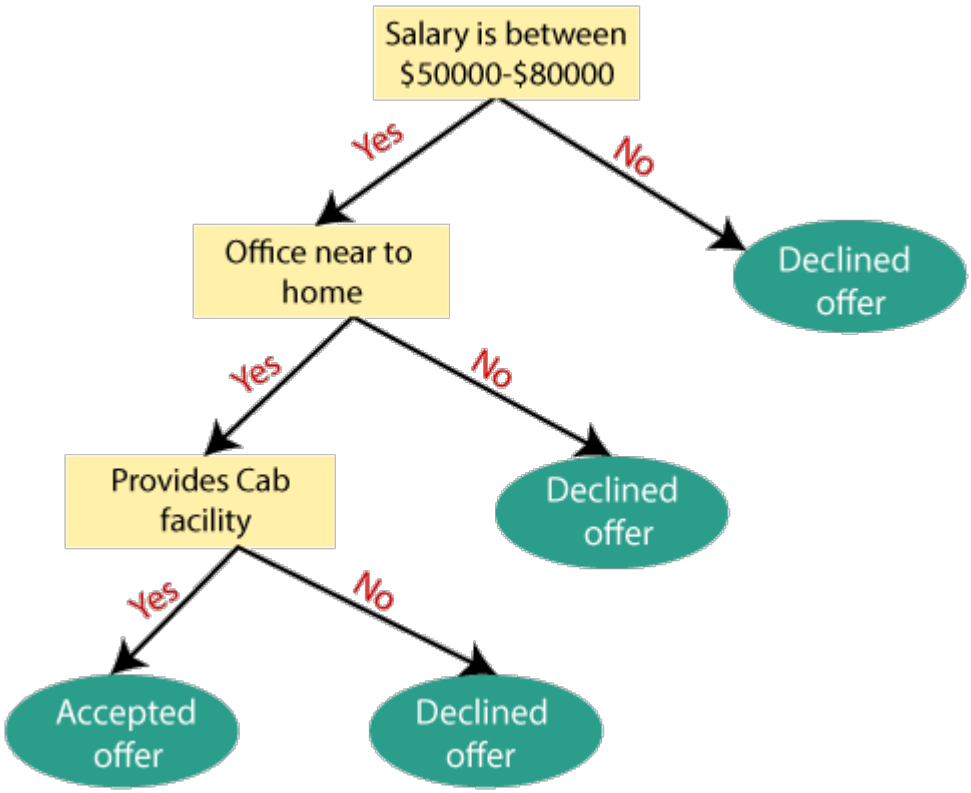


(2) How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contains possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.



3.

Decision tree model is suitable for dataset that has distinct difference between individual variables, and it requires the independent variables are correlated to dependent variable.

4.

(1) Data cleaning issue
The original dataset contains 10 variables, besides the dependent variable(survived), we should have tested the relationship between each factor and the dependent variable to determine which attributers are correlated to “survived”, but some of the data were missing and we failed to figure out a better method to fill in the missing data. As a consequence, we gave up those factors, merely selecting three factors based on our subjective opinion.

(2) Data evaluation
We initially attempted to use confusion matrix to evaluate the result but failed.

5.
To run the algorithm, we adapted the following Python library:
(1) sklearn:

- In python, sklearn is a machine learning package which include a lot of machine learning algorithms.
- We are using some of its modules like train_test_split, DecisionTreeClassifier and accuracy_score.

(2) NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

(3) Pandas:

- Used to read and write different files.
- Data manipulation can be done easily with dataframes.

We also adapted the following functions/methods
(1) Data Import:

- To import and manipulate the data we are using the pandas package provided in python

(2) Data Slicing:

- Before training the model we have to split the dataset into the training and testing dataset
- To split the dataset for training and testing we are using the sklearn module train_test_split
- Random-state variable is a pseudo-random number generator state used for random sampling

(3) Data Evaluating

- By exploring accuracy score, weighted average score of precision, weighted average score of recall, we evaluate the result

6.
a.
Library:
sklearn

Function:

```
from sklearn.metrics import classification_report
```

b.

In our model, we use the following indicators to evaluate the result.

(1) Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you have to look at other parameters to evaluate the performance of your model. For our model, we have got 0.803 which means our model is approx. 80% accurate.

$$\text{Accuracy} = TP+TN/TP+FP+FN+TN$$

(2) Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

$$\text{Precision} = TP/TP+FP$$

(3) Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the passengers that truly survived, how many did we label? We have got recall of 0.631 which is good for this model as it's above 0.5.

$$\text{Recall} = TP/TP+FN$$

(4) F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall. In our case, F1 score is 0.701.

$$\text{F1 Score} = 2*(\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

c.

Based on the metrix, the accuracy score is 0.78, we think that the score is satisfying because it does not over-fit, but the score is high enough to conclude that our model to a certain extent is able to predict the relationship between "pclass", "sex" and "age".

The weighted average score of precision is 0.81, the weighted average score of recall is 0.78. They almost reached to a balance point. Based on the result, we consider that the model is efficient.

```
0.7811550151975684
              precision    recall  f1-score   support

        died       0.91      0.78      0.84       236
    survived       0.58      0.80      0.67        93

    accuracy                           0.78       329
   macro avg       0.74      0.79      0.75       329
weighted avg       0.81      0.78      0.79       329
```

7.

Section	Links	Notes
1	https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/	How to evaluate the performance of a model in Azure ML and understanding "Confusion Metrics"
2	https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac	
3	https://www.datacamp.com/community/tutorials/decision-tree-classification-python	
4	https://scikit-learn.org/stable/modules/tree.html	
5	https://medium.com/machine-learning-101/chapter-3-decision-trees-theory-e7398adac567	
6	https://www.hackerearth.com/zh/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/	
7	https://www.kaggle.com/dmilla/introduction-to-decision-trees-titanic-dataset	
8	https://www.kaggle.com/kashnitsky/topic-3-decision-trees-and-knn	
9	https://www.geeksforgeeks.org/decision-tree-implementation-python/	
10	https://scikit-learn.org/stable/modules/tree.html	

8. Algorithm

In [46]:

```
# ADA_II
# HW2
# Team 2
# Huiwen Xu(Learder)
# Qi Liu
```

Dataset link: http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/titanic.txt

The dataset is divided into several variables including Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin and Embarked. By analysing the data and buidling the most

suitable model, the analysis should find out what kind of people were most likely to survive in the tragedy

In [47]:

```
import pandas as pd
```

In [50]:

```
df =
pd.read_csv('http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/
titanic.txt')
```

In [51]:

```
df.head()
```

Out[51]:

	row.names	pclass	survived	name	age	embarked	home.dest	room	ticket	boat	sex
0	1	1st	1	Allen, Miss Elisabeth Walton	29.0000	Southampton	St Louis, MO	B-5	24160 L221	2	female
1	2	1st	0	Allison, Miss Helen Loraine	2.0000	Southampton	Montreal, PQ / Chesterville, ON	C26	NaN	NaN	female
2	3	1st	0	Allison, Mr Hudson Joshua Creighton	30.0000	Southampton	Montreal, PQ / Chesterville, ON	C26	NaN	(135)	male
3	4	1st	0	Allison, Mrs Hudson J.C. (Bessie Waldo Daniels)	25.0000	Southampton	Montreal, PQ / Chesterville, ON	C26	NaN	NaN	female
4	5	1st	1	Allison, Master Hudson Trevor	0.9167	Southampton	Montreal, PQ / Chesterville, ON	C22	NaN	11	male

In [52]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1313 entries, 0 to 1312
Data columns (total 11 columns):
row.names    1313 non-null int64
pclass       1313 non-null object
survived     1313 non-null int64
name         1313 non-null object
age          633 non-null float64
```

```
embarked     821 non-null object
home.dest    754 non-null object
room         77 non-null object
ticket       69 non-null object
boat         347 non-null object
sex          1313 non-null object
dtypes: float64(1), int64(2), object(8)
memory usage: 71.9+ KB
```

In [53]:

```
X = df[['pclass','age','sex']]
y = df['survived']

X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1313 entries, 0 to 1312
Data columns (total 3 columns):
pclass    1313 non-null object
age       633 non-null float64
sex       1313 non-null object
dtypes: float64(1), object(2)
memory usage: 20.6+ KB
```

In [54]:

```
X['age'].fillna(X['age'].mean(),inplace=True)
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1313 entries, 0 to 1312
Data columns (total 3 columns):
pclass    1313 non-null object
age       1313 non-null float64
sex       1313 non-null object
dtypes: float64(1), object(2)
memory usage: 20.6+ KB
```

In [55]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test =
train_test_split(X,y,test_size=0.25,random_state=33)
```

In [56]:

```
from sklearn.feature_extraction import DictVectorizer
vec = DictVectorizer(sparse=False)
```

In [57]:

```
X_train = vec.fit_transform(X_train.to_dict(orient='record'))
print(vec.feature_names_)
X_test = vec.fit_transform(X_test.to_dict(orient='record'))
```

```
['age', 'pclass=1st', 'pclass=2nd', 'pclass=3rd', 'sex=female',
'sex=male']
```

In [58]:

```
from sklearn.tree import DecisionTreeClassifier

dtc = DecisionTreeClassifier()
dtc.fit(X_train,Y_train)
y_predict = dtc.predict(X_test)
```

In [59]:

```
from sklearn.metrics import classification_report

print(dtc.score(X_test,Y_test))
print(classification_report(y_predict,Y_test,target_names=['died','survived']))
```

```
0.7811550151975684
              precision    recall  f1-score   support

        died       0.91      0.78      0.84       236
    survived       0.58      0.80      0.67        93

    accuracy                           0.78       329
   macro avg       0.74      0.79      0.75       329
weighted avg       0.81      0.78      0.79       329
```

Conclusion

1. The accuracy of our prediction model is 0.78. The weighted average of precision is 0.81, the weighted average of recall is 0.78. We think the result is satisfying.

1. In this case, we just take "pclass", "age", "sex" into consideration, ignoring other factors.

Additional Notes

1. Because of the huge missing data of age, we merely use the average figure to fill in the missing data,which may have a non-negligible impact on the final result.

1. In this case, we only selected three factors based on our subject perspective, and owing to time, we did not analyse individual factors that may impact on survival.

In [ ]:

```

```