2020-10-4

Haowei Lou

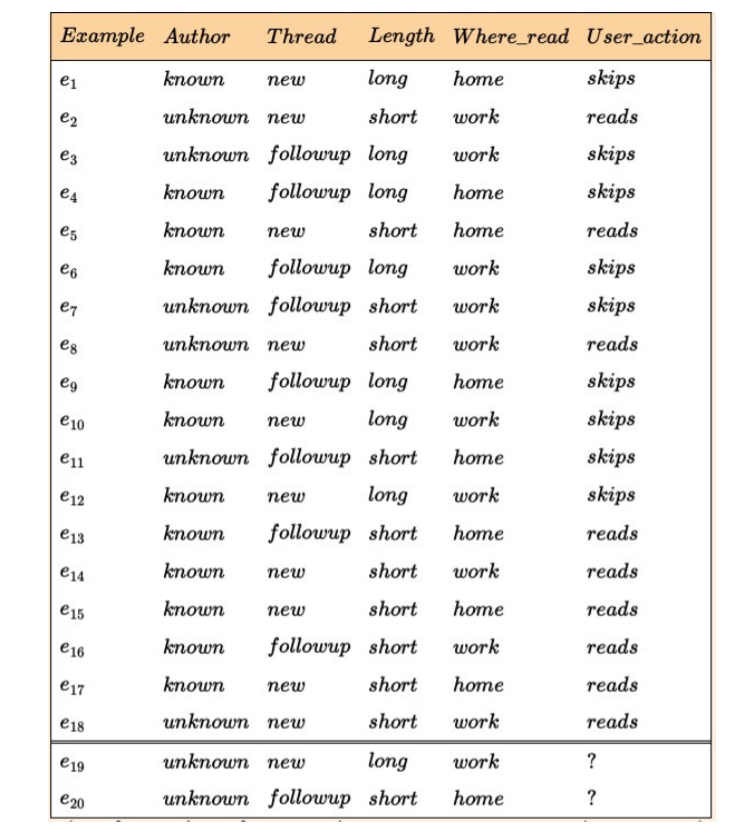
z5258575

COMP3411 Assignment 2

Haowei Lou

# Q1

## a)



The maximum information gain split tree is:

### Split on author:

There are 9 skips and 9 reads in the data set.

Entropy (parent) =

For the 12 knowns, 6 of them are skips and 6 are reads.

Entropy (known)==

For the 6 unknowns, 3 of them are skips and 3 are reads.

Entropy (unknown)==

The average entropy after splitting on ‘Author’, Entropy(Author) =

Information gained is 1-1 = 0

### Split on thread:

There are 9 skips and 9 reads in the data set.

Entropy (parent) =

For the 10 news, 3 of them are skip and 7 of them are read

Entropy (new) =

For the 8 followups, 6 of them are skip and 2 of them are read

Entropy (followup) =

The average entropy after splitting on ‘Thread, Entropy(Thread) =

Information gained by testing this attribute is: 1- 0.85 = 0.15, which is minor.

### Split by length:

There are 9 skips and 9 reads in the data set.

Entropy (parent) =

For the 7 longs, 7 of them are skip and 0 of them are read

Entropy(long)=

For the 11 shorts,2 of them are skip and 9 of them are read.

Entropy(short)=

The average entropy after splitting on ‘Length, Entropy(Length) =

Information gained by testing this attribute is: 1- 0.42 = 0.58, which is big.

### Split by Whereread:

There are 9 skips and 9 reads in the data set.

Entropy (parent) =

For the 8 homes, 4 of them are skip and 4 of them are read

Entropy(home)=

For the 10 works, 5 of them are skip and 5 of them are read

Entropy(work)=

The average entropy after splitting on ‘Whereread, Entropy(Whereread) =

Information gained is 1-1 = 0

By compare these four information gained. select Length to be the first node will maximise the total information gained.

Next node:

### Split by author:

There are 2 skips and 9 reads in the data set after 7 rows that include ‘long’ have been deducted.

Entropy(parent) = = = 0.6840384356

For the 6 knowns, 0 of them are skip and 6 of them are read

Entropy(known)= = =0

For the 5 unknowns, 2 of them are skip and 3 of them are read

Entropy(unknown)= = =0.9709505945

The average entropy after splitting on ‘Author, Entropy(author) =

Information gained is 0.68-0.44=0.24

### Split by thread:

There are 2 skips and 9 reads in the data set after 7 rows that include ‘long’ have been deducted.

Entropy(parent) = = = 0.6840384356

For the 7 news, 0 of them are skip and 7 of them are read

Entropy (new) =

For the 4 followups, 2 of them are skip and 2 of them are read

Entropy (followup) =

The average entropy after splitting on ‘Thread, Entropy(Thread) =

Information gained is 0.68-0.36=0.32

### Split by Whereread:

There are 2 skips and 9 reads in the data set after 7 rows that include ‘long’ have been deducted.

Entropy(parent) = = = 0.6840384356

For the 5 homes, 1 of them are skip and 4 of them are read

Entropy (home) =

For the 6 works, 1 of them are skip and 5 of them are read

Entropy (work) =

The average entropy after splitting on ‘Thread, Entropy(Whereread) =

Information gained is 0.684-0.683=0.001

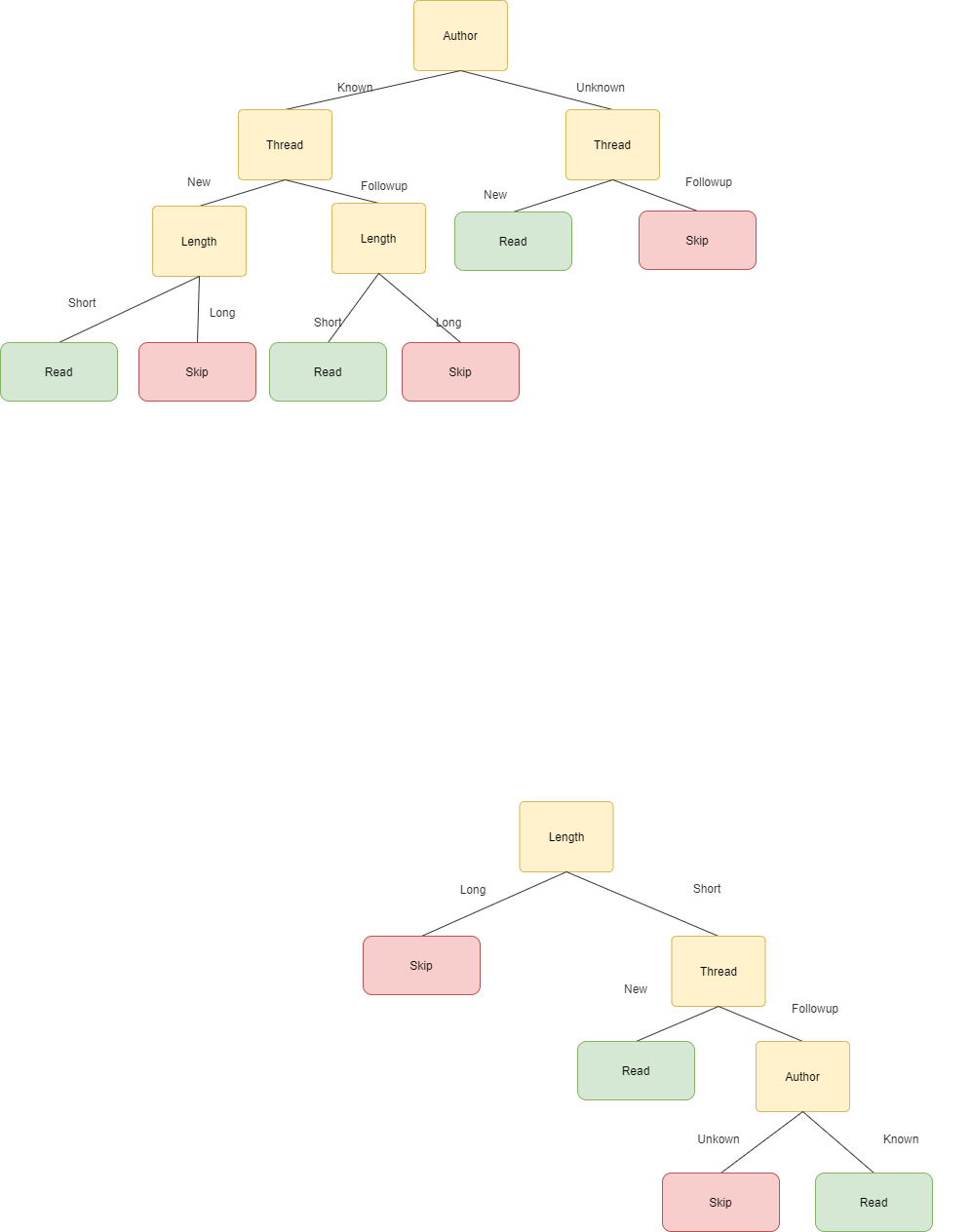
By compare these three information gained. select thread to be the second node will maximise the total information gained.

Next node:

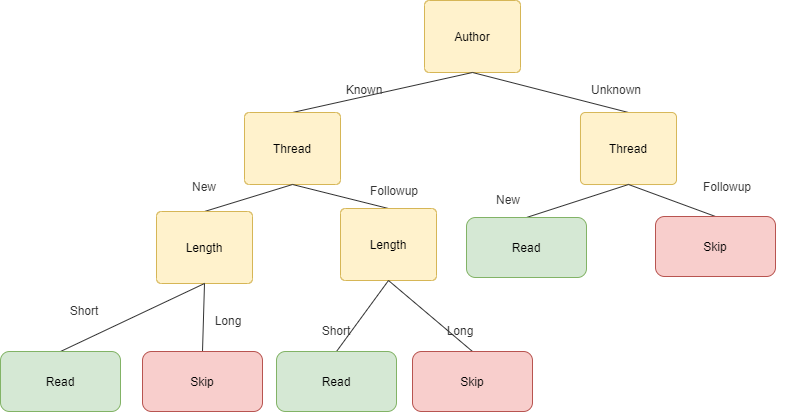
### Split by author:

There are 2 skips and 2 reads in the data set after 14 rows that include ‘long’ and ‘thread’have been deducted. And we have a clear classification if split by author, thus the third node should be author.

Optimal decision tree gained by maximizing information gain is:



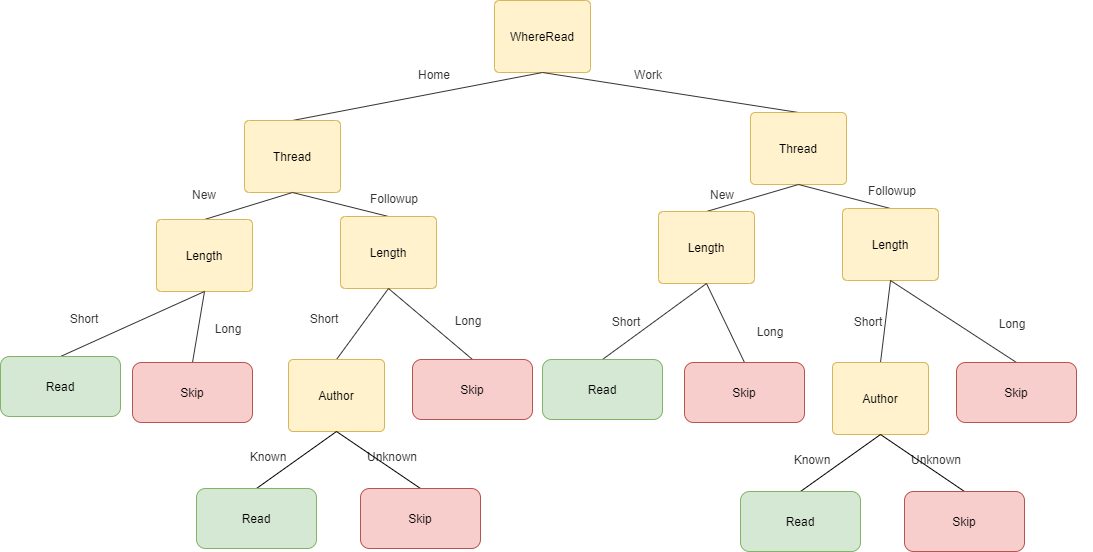
The order of features in new tree is: Author->Thread->Length->Where\_read->User\_actions



The tree found in this part is different with the tree found by maximum information gain. Functions that they represent are also different. For example, e19 in table, <unknown,new,long,work>, optimal tree will go through long->skip, which give answer ‘skip’. And the other one will go through unknown->new->read, which give answer ‘read’.

## b)

The order of feature is <WhereRead, Thread,Length,Author> and tree found is:



This tree a same function with the maximum information gained tree, where , <unknown,new,long,work> = skip which is same with the maximum given tree, but different with the tree found in part a.

## c)

There is no tree that can correctly classifies the training example but represent a different function than those found.

Let read be ‘True’ and Skip be ‘False’

The tree found in maximum information gained can be represent in logic by:

After simplification:

Tree found in a)

After simplification:

Tree found in b)

After simplification:

It has shown that features that matter are Length, Author and Thread in this decision tree.

Thus, ways to permutate these three features are:

[Author,Thread,Length],[Author,Length,Thread],[Thread,Author,Length],[Thread,Length,Author],[Length,Author,Thread],[Length,Thread,Author].

Both[Length, Thread, Author], [Thread, Length, Author], [Author, Length, Thread] are same in logic expression because they combine Length and Thread together, and then Author. And has already been found in preceding.

Then there are two possible different decision tree function for this question, which is [Author, Thread, Length], [Thread, Length, Author]. Which has already been found in a) and b). Therefore, there are no tree that correctly classifies the training examples but represents a different function than those found by the preceding algorithms.

# Q2

My code for developing decision tree is as follow:

import numpy as np

import pandas as pd

from sklearn import preprocessing

from sklearn import tree

from sklearn.datasets import load\_iris

from sklearn.preprocessing import LabelEncoder

data = pd.read\_csv('./adult.data',header=None) #load data

features = data.iloc[:,:-1].values

answer = data.iloc[:,-1].values

test = pd.read\_csv('./adult.test',header=None,skiprows=1) #load test

features\_test = test.iloc[:,:-1].values

answer\_test = test.iloc[:,-1].values

encode\_data = LabelEncoder()#encode data

encode\_data.fit(features[: , 1])

features[: , 1] = encode\_data.fit\_transform(features[: , 1])

features[: , 3] = encode\_data.fit\_transform(features[: , 3])

features[: , 5] = encode\_data.fit\_transform(features[: , 5])

features[: , 6] = encode\_data.fit\_transform(features[: , 6])

features[: , 7] = encode\_data.fit\_transform(features[: , 7])

features[: , 8] = encode\_data.fit\_transform(features[: , 8])

features[: , 9] = encode\_data.fit\_transform(features[: , 9])

features[: , 13] = encode\_data.fit\_transform(features[: , 13])

decision\_tree = tree.DecisionTreeClassifier(criterion='entropy',splitter='best',max\_depth=6,random\_state=0)

decision\_tree = decision\_tree.fit(features,answer)

encode\_data = LabelEncoder()#encode data

encode\_data.fit(features\_test[: , 1])

features\_test[: , 1] = encode\_data.fit\_transform(features\_test[: , 1])

features\_test[: , 3] = encode\_data.fit\_transform(features\_test[: , 3])

features\_test[: , 5] = encode\_data.fit\_transform(features\_test[: , 5])

features\_test[: , 6] = encode\_data.fit\_transform(features\_test[: , 6])

features\_test[: , 7] = encode\_data.fit\_transform(features\_test[: , 7])

features\_test[: , 8] = encode\_data.fit\_transform(features\_test[: , 8])

features\_test[: , 9] = encode\_data.fit\_transform(features\_test[: , 9])

features\_test[: , 13] = encode\_data.fit\_transform(features\_test[: , 13])

prediction = decision\_tree.predict(features\_test)

i = 0

correct = 0

while i < len(prediction):

    #replace the dot

    if prediction[i] == answer\_test[i][:-1]:

        correct += 1

    i += 1

print('The accuracy rate is:',"{:.2%}".format(correct/i))

The output for this program is:

The accuracy rate is: 84.84%

The graph is as follow.

