**Logo

Description automatically generated**

**San Francisco Bay University**

**CS483 - Fundamentals of Artificial Intelligence**

**Homework Assignment #4**

**Instruction: Due day: 7/21/2022**

1. **Push the source code to Github**
2. **Overdue homework submission could not be accepted.**
3. **Take academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
4. Re-calculate the entropy for the feature selection in the example of file “*Gini Impurity Cal in Decision Tree*” rather than Gini impurity method. And then, compare the results from two different criteria

*Hint: taking the reference at the following link for your calculation*

[*https://towardsdatascience.com/entropy-how-decision-trees-make-decisions-2946b9c18c8*](https://towardsdatascience.com/entropy-how-decision-trees-make-decisions-2946b9c18c8)

|  |  |  |
| --- | --- | --- |
| **Color** | **Diameter** | **Label** |
| Green | 3 | Apple |
| Yellow | 3 | Apple |
| Red | 1 | Grape |
| Red | 1 | Grape |
| Yellow | 3 | Lemon |

Total pop = 5

Text

Description automatically generated

Features: Color, Diameter

Label: Apple, Grape, Lemon

P(A) = 2/5 = 0.4

P(G) = 2/5 = 0.4

P(L) = 1/5 = 0.2

**Solution:**

**Feature 1:** Diameter

**For diameter >=3**

P(A) = 2/3 = 0.67

P(L) = 1/3 = 0.33

**For diameter < 3**

P(A) = 0/2 = 0

P(G) = 2/2 = 1

D<3

d>= 3

Diameter Entropy (Ed) Calculation:

E (Parent) =

= = 1.52

E (d>= 3) = = 0.98

E (d< 3) = = 0

Weighted average of entropies

E(diameter) =

Information Gain = E(Parent) - E(diameter) = 1.52-0.588 = 0.932

|  |  |  |
| --- | --- | --- |
| **Color** | **Diameter** | **Label** |
| Green | 3 | Apple |
| Yellow | 3 | Apple |
| Red | 1 | Grape |
| Red | 1 | Grape |
| Yellow | 3 | Lemon |

**Feature 2:** color

**For color=green**

P(A) = 1/1

P(L) = 0

P(G) = 0

**For color=yellow**

P(A) = 1/2

P(L) = 1/2

P(G) = 0

**For color=red**

P(A) = 0

P(L) = 0

P(G) = 2/2

C = red

C = green

C = yellow

Color Entropy (Ed) Calculation:

E (C = green) =

=

E (C = yellow) = = 1

E (C = red) =

Weighted average of entropies

E(Color) =

Information Gain = E(Parent) - E(color) = 1.52-0.4 = 1.12

**Conclusion:**

The information gain from color feature is higher than the one from diameter, hence the first choice for classification is **color.** Comparing the Gini index and Entropy, Gini index requires less mathematical computation compared to entropy, but entropy is more accurate as shown in this example (Entropy achieved the same classification accuracy as Gini impurity in the first level)

1. Given a dataset as follows, please buildup a decision tree with max information gain comparing the different condition checking features by hand calculation **Gini impurity and information gain.** And predict "Profit" in the new data. After that, write Python program to verify your design through calling existing functions from the library

|  |  |  |  |
| --- | --- | --- | --- |
| Age | Competition | Type | Profit |
| Old | Yes | Software | Down |
| Old | No | Software | Down |
| Old | No | Hardware | Down |
| Mid | Yes | Software | Down |
| Mid | Yes | Hardware | Down |
| Mid | No | Hardware | Up |
| Mid | No | Software | Up |
| New | Yes | Software | Up |
| New | No | Hardware | Up |
| New | No | Software | Up |
| Mid | No | Hardware | ? |

**Solution:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Level 1: Impurity of root** | | | |  |
|  | **imp =** P(Down)\*(1-P(Down)) + P(up)\*(1-P(up)) | | | |
|  | = 5/10\*(1-5/10) + 5/10\*(1-5/10) | | | |
|  | = 0.5 | |  |  |
|  |  |  |  |  |
|  | **Ave. Imp** = 10/10 \* 0.5 = **0.5** | | |  |

**Level 2; Impurity of Age**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Old** | Down | Up |  | **Mid** | down | up |  | **New** | down | up |
| 3 | 3 | 0 |  | 4 | 2 | 2 |  | 3 | 0 | 3 |
|  |  |  |  |  |  |  |  |  |  |  |

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Imp= 3/3\*(1-3/3) + 0/3\*(1-0/3) | |  | Imp = 2/4\*(1-2/4) + 2/4\*(1-2/4) | | | |  | Imp = 0/3\*(1-0/3) + 3/3\*(1-3/3) | |
| = 0 |  |  | = 0.5 |  |  |  | | = 0 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ave. Imp = 3/10\*0 | | |  | Ave.Imp = 4/10\*0.5 | |  |  | Ave. Imp = 3/10\*0 | | |
| = 0 | |  |  | = 0.2 | |  |  | = 0 | |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tot. Ave. Imp.** = 0 + 0.2 + 0 = **0.2** | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Info. Gain** = 0.5 (from Ave.Imp of level 1) - 0.2(from Tot. Ave. Imp) = **0.3** | | | | | | | |

**Impurity of Competition:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Yes** | Down | Up |  | **No** | down | up |  |
| 4 | 3 | 1 |  | 6 | 2 | 4 |  |
|  |  |  |  |  |  |  |  |

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Imp= 3/4\*(1-3/4) + 1/4\*(1-1/4) | | Imp = 2/6\*(1-2/6) + 4/6\*(1-4/6) | | |
| = 0.375 |  |  |  | = 0.44 | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ave. Imp = 4/10\*0.375 | | | | Ave. Imp = 6/10\*0.44 | | | | |
| = 0.15 | | |  | = 0.264 | | | |
|  |  |  | |  |  |  |  | |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tot. Ave. Imp.** = 0.15 + 0.264 = **0.414** | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Info. Gain** = 0.5 (from Ave.Imp of level 1) - 0.414(from Tot. Ave. Imp) = **0.086** | | | | | | | |

**Impurity of Type:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Software** | Down | Up |  | **hardware** | down | up |  |
| 6 | 3 | 3 |  | 4 | 2 | 2 |  |
|  |  |  |  |  |  |  |  |

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Imp = 3/6\*(1-3/6) + 3/6\*(1-3/6) | | Imp = 2/4\*(1-2/4) + 2/4\*(1-2/4) | | |
| = 0.5 |  |  |  | = 0.5 | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ave. Imp = 6/10\*0.5 | | | | Ave. Imp = 4/10\*0.5 | | | | |
| = 0.3 | | |  | = 0.2 | | | |
|  |  |  | |  |  |  |  | |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tot. Ave. Imp.** = 0.3 + 0.2 = **0.5** | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Info. Gain** = 0.5 (from Ave.Imp of level 1) - 0.5(from Tot. Ave. Imp) = **0** | | | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Comparing Info Gains** | | |  |
| **Age** | **Competition** | **Type** |  |
| **0.3** | **0.086** | **0.0** |  |

**Taking "Age" will get highest info gain**

**Level 3:**

|  |  |  |  |
| --- | --- | --- | --- |
| Age | Competition | Type | Profit |
| Old | Yes | Software | Down |
| Old | No | Software | Down |
| Old | No | Hardware | Down |

|  |  |  |  |
| --- | --- | --- | --- |
| Age | Competition | Type | Profit |
| Mid | Yes | Software | Down |
| Mid | Yes | Hardware | Down |
| Mid | No | Hardware | Up |
| Mid | No | Software | Up |

Impurity of **Competition** in **Mid**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Yes** | Down | Up |  | **No** | down | up |  |
| 2 | 2 | 0 |  | 2 | 0 | 2 |  |
|  |  |  |  |  |  |  |  |

Impurity of **yes** in **Competition**

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

Imp = 2/2\*(1-2/2) + 0/2\*(1-0/2) = 0

Impurity of **No** in **Competition**

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

Imp = 0/2\*(1-0/2) + 2/2\*(1-2/2) = 0

**Tot. Ave. Imp.** = 0 + 0 = **0**

**Information gain = 0.3-0 = 0.3**

Impurity of **Competition** in **old**

**Avg.imp = 0**

**Information gain = 0.3-0 = 0.3**

Impurity of **type** in **old**

**Avg.imp = 0**

**Information gain = 0.3-0 = 0.3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Comparing Info Gains in old** | | |  |
| **Competition** | **Type** |
| **0.3** | **0.3** |
| **Both get the same info gain** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Age | Competition | Type | Profit |
| New | Yes | Software | Up |
| New | No | Hardware | Up |
| New | No | Software | Up |

Impurity of **New**

Impurity of **Type** in **Mid**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Software** | Down | Up |  | **Hardware** | down | up |  |
| 2 | 1 | 1 |  | 2 | 1 | 1 |  |
|  |  |  |  |  |  |  |  |

Impurity of **software** in **type**

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

Imp = 1/2\*(1-1/2) + 1/2\*(1-1/2) = 0.5

Impurity of **hardware** in **type**

Imp = P(Down)\*(1-P(Down)) + P(up)\*(1-P(up))

Imp = 1/2\*(1-1/2) + 1/2\*(1-1/2) = 0.5

**Tot. Ave. Imp.** = 0.5 + 0.5 = **1**

**Information gain = 0.3-1 = -0.7**

|  |  |
| --- | --- |
| **Comparing Info Gains in mid** | |
| **Competition** | **Type** |
| **0.3** | **-0.7** |

Impurity of **Competition** in **new**

**Avg.imp = 0**

**Information gain = 0.3-0 = 0.3**

Impurity of **type** in **new**

**Avg.imp = 0**

**Information gain = 0.3-0 = 0.3**

|  |  |
| --- | --- |
| **Comparing Info Gains in mid** | |
| **Competition** | **Type** |
| **0.3** | **0.3** |

**Competition has higher info gain**

**Both features have the same info gain**

**Final Decision Tree:**

Hence, the predicted profit for:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mid |  | No | Hardware | ? |

Predicted profit trend is Up

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mid |  | No | Hardware | up |

Age

Old

New

Mid

No

Yes

Competition

**Python Code Verification:**

|  |
| --- |
| #!/usr/bin/env python  # coding: utf-8  import pandas as pd  import numpy as np  from sklearn.tree import DecisionTreeClassifier  from sklearn import tree  """      Loading Data  """  col\_names = ['Age', 'Competition',  'Type', 'Profit']  # load dataset  df = pd.read\_csv("/content/fruits.csv")  df.head()  df=df.split('\n')  dat=[]  for data in df:      word=data.split(' ')      dat.append(word)  ndf=pd.DataFrame(dat,columns=['Age','Competition','Type','profit'])  x\_train=ndf.iloc[:,0:3]  y\_train=ndf.iloc[:,3]  print(x\_train)  clf\_tree=DecisionTreeClassifier(random\_state=0,max\_depth=3)  clf\_fit=clf\_tree.fit(x\_train,y\_train)  x\_test=np.array([1,1,1])  x\_test=x\_test.reshape(1,-1)  predicted=clf\_fit.predict(x\_test) |