Decision_Tree_Implementation

January 31, 2020

For doing this assignment we use "chefboost" framework, which is a lightweight implementation of Decision Tree algorithms (including ID3, C4.5 and CART).

At the below cell we install the framework and import the necessary libraries.

[1]:
```
import pandas as pd
!pip install chefboost
from chefboost import Chefboost as chef
```

```
Collecting chefboost
  Downloading https://files.pythonhosted.org/packages/11/74/99026299127cd1095bcf
57e64668e4cc37358b1369c4d9e006333c893e1a/chefboost-0.0.3-py3-none-any.whl
Collecting tqdm>=4.30.0
  Downloading https://files.pythonhosted.org/packages/cc/2e/4307206db63f05
ed37e21d4c0d843d0fbcacd62479f8ce99ba0f2c0875e0/tqdm-4.42.0-py2.py3-none-any.whl
(59kB)
     || 61kB 3.5MB/s
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.6
/dist-packages (from chefboost) (1.17.5)
Requirement already satisfied: pandas>=0.22.0 in /usr/local/lib/python3.6/dist-
packages (from chefboost) (0.25.3)
Requirement already satisfied: python-dateutil>=2.6.1 in
/usr/local/lib/python3.6/dist-packages (from pandas>=0.22.0->chefboost) (2.6.1)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-
packages (from pandas>=0.22.0->chefboost) (2018.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-
packages (from python-dateutil>=2.6.1->pandas>=0.22.0->chefboost) (1.12.0)
Installing collected packages: tqdm, chefboost
  Found existing installation: tqdm 4.28.1
    Uninstalling tqdm-4.28.1:
      Successfully uninstalled tqdm-4.28.1
Successfully installed chefboost-0.0.3 tqdm-4.42.0
```

1 ID3

Here we use the attribute for which the resulting entropy after splitting is minimized; or, equivalently, information gain is maximum.

1.1 1st Expriment

[2]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
config = {'algorithm': 'ID3'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.021408389480547706
RMSE:  0.11584902000351505
Mean:  0.39404477287546186
MAE / Mean:  5.4329840044125755 %
RMSE / Mean:  29.399963653401695 %
finished in  115.94774627685547  seconds
```

1.2 2nd Expriment

[3]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'ID3'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  63.94912600517273  seconds
```

1.3 3rd Expriment

[4]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'ID3'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  69.36704921722412  seconds
```

1.4 4th Expriment

[5]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[1840:2760]
df[1840:2760]=tempDF
config = {'algorithm': 'ID3'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  68.0544261932373  seconds
```

1.5 5th Expriment

[6]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[2760:3680]
df[2760:3680]=tempDF
config = {'algorithm': 'C4.5'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  68.05437564849854  seconds
```

1.6 Final Results of ID3

Average MAE with 5-fold approach: $ $ $\frac{0.021408389480547706+0.199956531188872+0.199956531188872+0.20604216474679418+0.20604216474679418}{5} =$ 0.16668115627037601 32 $ $

Average Time to induce the tree with 5-fold approach: $ $ $\frac{115.94774627685547+63.94912600517273+69.36704921722412+68.0544261932373+68.05437564849854}{5} =$ 77.074544668197632 seconds

2 C4.5

The splitting criterion here is the normalized information gain. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the partitioned sublists.

2.1 1st Expriment

[17]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
```

```
config = {'algorithm': 'C4.5'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.021408389480547706
RMSE:  0.11584902000351505
Mean:  0.39404477287546186
MAE / Mean:  5.4329840044125755 %
RMSE / Mean:  29.399963653401695 %
finished in  136.82659888267517  seconds
```

2.2 2nd Expriment

[18]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'C4.5'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  75.611647605896  seconds
```

2.3 3rd Expriment

[19]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'ID3'}
```

```
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  74.76668620109558  seconds
```

2.4 4th Expriment

[20]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[1840:2760]
df[1840:2760]=tempDF
config = {'algorithm': 'C4.5'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  73.99158716201782  seconds
```

2.5 5th Expriment

[21]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[2760:3680]
df[2760:3680]=tempDF
config = {'algorithm': 'C4.5'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  76.66375184059143  seconds
```

2.6 Final Results of C4.5

Average MAE with 5-fold approach: $ $
$\frac{0.021408389480547706+0.199956531188872+0.199956531188872+0.20604216474679418+0.20604216474679418}{5} =$
0.16668115627037601 32 $ $
Average Time to induce the tree with 5-fold approach: $ $
$\frac{136.82659888267517+75.611647605896+74.76668620109558+73.99158716201782+76.66375184059143}{5} =$
87.5720543384552 seconds

3 CART

CART decision tree algorithm uses Gini index as a metric for classification tasks. It stores sum of squared probabilities of each class.

3.1 1st Expriment

[22]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
config = {'algorithm': 'CART'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.021408389480547706
RMSE:  0.11584902000351505
Mean:  0.39404477287546186
MAE / Mean:  5.4329840044125755 %
RMSE / Mean:  29.399963653401695 %
finished in  139.17147850990295  seconds
```

3.2 2nd Expriment

[23]:
```
df = pd.read_csv('/content/spambase.data', header=None)
```

```
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'CART'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  76.28004765510559  seconds
```

3.3 3rd Expriment

[24]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[920:1840]
df[920:1840]=tempDF
config = {'algorithm': 'CART'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.199956531188872
RMSE:  0.4471649932506703
Mean:  -0.199956531188872
finished in  78.20051908493042  seconds
```

3.4 4th Expriment

[25]:
```
df = pd.read_csv('/content/spambase.data', header=None)
```

```
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[1840:2760]
df[1840:2760]=tempDF
config = {'algorithm': 'CART'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  76.31681728363037  seconds
```

3.5 5th Expriment

[26]:
```
df = pd.read_csv('/content/spambase.data', header=None)
df.rename(columns={0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:
 ↪'9',10:'10',11:'11',12:'12',13:'13',14:'14',15:'15',16:'16',17:'17',18:
 ↪'18',19:'19',20:'20',21:'21',22:'22',23:'23',24:'24',25:'25',26:'26',27:
 ↪'27',28:'28',29:'29',30:'30',31:'31',32:'32',33:'33',34:'34',35:'35',36:
 ↪'36',37:'37',38:'38',39:'39',40:'40',41:'41',42:'42',43:'43',44:'44',45:
 ↪'45',46:'46',47:'47',48:'48',49:'49',50:'50',51:'51',52:'52',53:'53',54:
 ↪'54',55:'55',56:'56',57:'Decision'},inplace=True)
tempDF=df[3680:4600]
df[3680:4600]=df[2760:3680]
df[2760:3680]=tempDF
config = {'algorithm': 'CART'}
model = chef.fit(df, config)
```

```
Regression  tree is going to be built...
MAE:  0.20604216474679418
RMSE:  0.45391867635821526
Mean:  -0.005868289502282112
finished in  80.05369997024536  seconds
```

3.6 Final Results of CART

Average MAE with 5-fold approach: $ $
$\frac{0.021408389480547706+0.199956531188872+0.199956531188872+0.20604216474679418+0.20604216474679418}{5} =$
0.16668115627037600132 $ $
Average Time to induce the tree with 5-fold approach: $ $

$\frac{139.17147850990295+76.28004765510559+78.20051908493042+76.31681728363037+80.05369997024536}{5} =$
90.004512500762938 seconds

4 Comparison Between Tree algorithms

In accuracy context the average results between these three algorithms are not much different (they all get average MAE as 0.16668115627037601 32. $ $

In time consumption context the average results between these three algorithms are ranked respectively as folows: $ $

$ID3(77.074544668197632) < C4.5(87.5720543384552) < CART(90.004512500762938)$