3 - Rule Search

September 14, 2022

1 Rule Search

Rule search algorithms for identifying emergency shelter clients with the potential to become chronic shelter users.

[1]:
```
%load_ext autoreload
%autoreload 1
```

[2]:
```
import numpy as np
import pandas as pd
import datetime, copy, imp
import pickle
import time
import os
import re
import matplotlib.pyplot as plt
from sklearn.model_selection import StratifiedKFold
from importlib import reload
from dask.distributed import Client

from tqdm.auto import tqdm, trange
from tqdm.notebook import tqdm
tqdm.pandas()
```

```
import sys
sys.path.insert(0, '../util/')

from data_cache import CacheResult
import rules as rs
```

[ ]:
```

```

1.1 Load Coverage Tables

[6]:
```
winSizes = [ 30, 60, 90, 120 ]

covTblInfo = {}
covTbl = {}
labels = {}
clientIds = {}
idTimes = {}

for win in winSizes:

#    covFileStr = f'/hd2/data/di/plwh/cache/DiRules-CoverageTable-WinSz{win}.
 ↪pkl'
    covFileStr = f'/Users/gmessier/data/plwh/cache/
 ↪DiRules-CoverageTable-WinSz{win}.pkl'

    with open(covFileStr,'rb') as pklFile:
        dat = pickle.load(pklFile)

    covTblInfo[win] = rs.CoverageTableInfo(dat)
    covTbl[win] = dat['CoverageTable']
    labels[win] = dat['Labels']
    clientIds[win] = dat['ClientIds']
    idTimes[win] = dat['IdTimes']
```

[7]:
```
# Time to chronic event classification information.
# - Used to determine the time to identification for chronic clients who are␣
 ↪not identified by a rule.
#tte = pd.read_hdf('/hd2/data/di/plwh/cache/DiRules-CdnFedChronicTte___.hd5')
tte = pd.read_hdf('/Users/gmessier/data/plwh/cache/DiRules-CdnFedChronicTte___.
 ↪hd5')
```

[8]:
```
searchAttrCore = [ 0, 3, 1 ]
searchAttrExt = [ 0, 3, 1, 8, 13, 4 ]
```

[ ]:
```

```

1.2 Determine Parameter Settings

- Create cross-validation derived class to include identification time as a performance parameter.

[9]:
```
class DiRuleSetCrossValidation(rs.RuleSetCrossValidation):
    def␣
↪__init__(self,nSplit,ruleQual,ruleSearch,maxSetSize,debug=False,client=None):

        super().__init__(nSplit,ruleQual,ruleSearch,maxSetSize,debug,client)


    def cross_validate(self,covTblInfo,covTbl,labels,clientIds,idTimes,tte):

        skf = StratifiedKFold(n_splits=self.nSplit, random_state=None,␣
↪shuffle=True)

        covTblFtr = self.client.scatter(covTbl,broadcast=True)
        labelsFtr = self.client.scatter(labels,broadcast=True)

        futures = []
        for trainIdx, testIdx in skf.split(covTbl,labels):

            futures += [
                self.client.submit(
                    rs.rule_set_search,
                    self.ruleQual, self.ruleSearch,
                    covTblInfo,covTblFtr,labelsFtr,
                    idx=trainIdx, maxSetSize=self.maxSetSize)
            ]

        ruleSets = self.client.gather(futures)

        cnfMtx = np.zeros((2,2),dtype=int)
        self.ruleQual.size_head(len(testIdx))
        testIdTimes = []

        for ruleSet in ruleSets:

            cnfMtx += self.ruleQual.
↪confusion_matrix(ruleSet,covTbl[testIdx],labels[testIdx])

            # ID times for individuals who satisfy the rule set.
            self.ruleQual.calc_head(ruleSet,covTbl[testIdx],labels[testIdx])
            head = np.array(self.ruleQual.get_head(),dtype=bool)
            testIdTimes += list( idTimes[testIdx][head] )
            # ID times for individuals who have to wait for the chronic␣
↪definition.
            negId = tte.loc[ clientIds[testIdx][~head] ]
            testIdTimes += list( negId.loc[ negId.Flag == 'chr'].Time )

        return (cnfMtx,np.median(testIdTimes),np.mean(testIdTimes))
```

[10]:
```
def Evaluate(client, resStr, covTblInfo, covTbl, labels, clientIds, idTimes,␣
↪tte, wSize, qualCalc, mxRuleLen, searchAttr, mxSetSize, nFolds):

    resStr += f' WinSize: {wSize}, MxRuleLen: {mxRuleLen}, MxSetSz:␣
↪{mxSetSize}, NFolds: {nFolds}\n'
    resStr += f' Searched Attr: {searchAttr}\n'

    rSrch = rs.OpusRuleSearch(qualCalc, mxRuleLen)
    rSrch.set_search_attributes(searchAttr)

    rsEval = DiRuleSetCrossValidation(nFolds, qualCalc, rSrch, mxSetSize,␣
↪client=client)

    (cnf,medIdTime,meanIdTime) = rsEval.
↪cross_validate(covTblInfo,covTbl,labels,clientIds,idTimes,tte)


    resStr += qual.confusion_summary_str(cnf)
    resStr += f'Median ID Time: {medIdTime}, Mean ID Time: {meanIdTime}'

    return resStr
```

[11]:
```
# Dumps results to a file.
resultFileStr = '../out/Results.txt'
def DumpResultStr(resStr,fileStr):
    f = open(fileStr,'a')
    f.write('----------------------------------------\n')
    f.write(resStr)
    f.write('\n')
    f.close()
```

[12]:
```
from dask.distributed import Client

client = Client("tcp://127.0.0.1:53547")
client
```

[12]:
```
<Client: 'tcp://127.0.0.1:53547' processes=4 threads=8, memory=16.00 GiB>
```

[13]:
```
_ = client.upload_file('../util/rules.py')
```

```
[14]: qual = rs.RuleQualFScore(covTblInfo[30].FtrStrs,betaSq=0.01)
rSrch = rs.OpusRuleSearch(ruleQuality=qual, maxRuleLen=1)
rsEval = DiRuleSetCrossValidation(8, qual, rSrch, maxSetSize=2, client=client)

(cnf,medIdTime,meanIdTime) = rsEval.
 ↪cross_validate(covTblInfo[30],covTbl[30],labels[30],clientIds[30],idTimes[30],tte)

print(qual.confusion_summary_str(cnf))
print(f'Median ID Time: {medIdTime}, Mean ID Time: {meanIdTime}')
```

```
Precision: 0.4395
Recall: 0.5214
Confusion:
 True Pos: 1289/2472
 False Neg: 1183/2472
 False Pos: 1644/9696
 True Neg: 8052/9696

Median ID Time: 54.0, Mean ID Time: 251.81073858114675
```

1.3 Determine Best Metric

```
[15]: # ID times using only the chronic definition (worst case benchmark values).?
chrDefIdTime = tte.loc[tte.Flag=='chr'].Time
mxMedIdTime = chrDefIdTime.median()
mxMnIdTime = chrDefIdTime.mean()
print(f'Default ID Time: {mxMedIdTime} (median), {mxMnIdTime} (mean)')
```

```
Default ID Time: 297.0 (median), 631.7966154810405 (mean)
```

Evaluate the following setting combinations (notes included based on results).

- ☐ Determine beta with long rules and extended features.
  - $\beta^2 = 0.25$ gives 0.73 recall and 0.57 precision. Cuts ID time in half with 90 day data.
- ☐ Compare extended features to core.
  - Going to fewer features degraded the selectivity of the rules. Adjusted β^2 to 0.1 to get similar performance.
- ☐ Compare short and long rules.
  - Reducing set size to 1 degraded sensitivity of the rules. Chagned β^2 to 1 to get 0.71 recall and 0.62 precision.
- ☐ Determine best time scale.
  - Going to 60 days with identical settings seemed to perform pretty much the same!
  - Some degradation seen in sensitivity and selectivity going to 30 days so 60 seems like the setting to beat.

```
[13]: nFolds = 10
```

Beta Value

```
[20]: %%time
wSize = 90
mxRuleLen = 3
mxSetSize = 2
searchAttr = searchAttrExt
resultFileStr = '../out/Beta.txt'
betaSqs = [ 0.25, 0.5, 1.0 ]

for betaSq in betaSqs:

    qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
    resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
    resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
 ↪labels[wSize],
                      clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                      mxRuleLen, searchAttr, mxSetSize, nFolds)

    print(resStr)
    DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 90, MxRuleLen: 3, MxSetSz: 2, NFolds: 10
 Searched Attr: [0, 3, 1, 8, 13, 4]
Precision: 0.6254
Recall: 0.6793
Confusion:
 True Pos: 1576/2320
 False Neg: 744/2320
 False Pos: 944/5120
 True Neg: 4176/5120
Median ID Time: 106.0, Mean ID Time: 284.98713235294116
FScore (betaSq = 0.50)
 WinSize: 90, MxRuleLen: 3, MxSetSz: 2, NFolds: 10
 Searched Attr: [0, 3, 1, 8, 13, 4]
Precision: 0.5236
Recall: 0.7978
Confusion:
 True Pos: 1851/2320
 False Neg: 469/2320
 False Pos: 1684/5120
 True Neg: 3436/5120
Median ID Time: 104.0, Mean ID Time: 205.21478521478522
FScore (betaSq = 1.00)
 WinSize: 90, MxRuleLen: 3, MxSetSz: 2, NFolds: 10
 Searched Attr: [0, 3, 1, 8, 13, 4]
Precision: 0.5273
Recall: 0.8569
Confusion:
 True Pos: 1988/2320
 False Neg: 332/2320
 False Pos: 1782/5120
 True Neg: 3338/5120
Median ID Time: 104.0, Mean ID Time: 157.52169673330084
CPU times: user 5.83 s, sys: 813 ms, total: 6.64 s
Wall time: 2h 14min 50s
```

Rule Simplification

[71]:
```
wSize = 90
searchAttr = searchAttrCore
resultFileStr = '../out/Simplification.txt'
betaSq = 0.25

qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
```

[72]:
```
%%time
mxRuleLen = 3
mxSetSize = 2

resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
 ↪labels[wSize],
                  clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                  mxRuleLen, searchAttr, mxSetSize, nFolds)

print(resStr)
DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 90, MxRuleLen: 3, MxSetSz: 2, NFolds: 10
 Searched Attr: [0, 3, 1]
Precision: 0.6260
Recall: 0.6832
Confusion:
 True Pos: 1585/2320
 False Neg: 735/2320
 False Pos: 947/5120
 True Neg: 4173/5120
Median ID Time: 106.0, Mean ID Time: 239.5977961432507
CPU times: user 1.03 s, sys: 147 ms, total: 1.18 s
Wall time: 20min 58s
```

[73]:
```
%%time
mxRuleLen = 3
mxSetSize = 1

resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
↪labels[wSize],
                  clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                  mxRuleLen, searchAttr, mxSetSize, nFolds)

print(resStr)
DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 90, MxRuleLen: 3, MxSetSz: 1, NFolds: 10
 Searched Attr: [0, 3, 1]
Precision: 0.7565
Recall: 0.5034
Confusion:
 True Pos: 1168/2320
 False Neg: 1152/2320
 False Pos: 376/5120
 True Neg: 4744/5120
Median ID Time: 115.0, Mean ID Time: 337.12462908011867
CPU times: user 543 ms, sys: 55.2 ms, total: 599 ms
Wall time: 8min 31s
```

[74]:
```
%%time
mxRuleLen = 2
mxSetSize = 1

resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
↪labels[wSize],
                  clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                  mxRuleLen, searchAttr, mxSetSize, nFolds)

print(resStr)
DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 90, MxRuleLen: 2, MxSetSz: 1, NFolds: 10
 Searched Attr: [0, 3, 1]
Precision: 0.7427
Recall: 0.5336
Confusion:
 True Pos: 1238/2320
 False Neg: 1082/2320
 False Pos: 429/5120
 True Neg: 4691/5120
Median ID Time: 115.0, Mean ID Time: 319.4419789014187
CPU times: user 152 ms, sys: 22.4 ms, total: 175 ms
Wall time: 20.2 s
```

Window Sizes

```
[18]: mxRuleLen = 2
      mxSetSize = 1
      betaSq = 0.25
      #searchAttr = searchAttrCore
      searchAttr = [ 0 ]
      resultFileStr = '../out/WinSizeSleepOnly.txt'
```

```
[19]: %%time
      wSize = 30

      qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
      resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
      resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
       ↪labels[wSize],
                        clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                        mxRuleLen, searchAttr, mxSetSize, nFolds)

      print(resStr)
      DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 30, MxRuleLen: 2, MxSetSz: 1, NFolds: 10
 Searched Attr: [0]
Precision: 0.4780
Recall: 0.5846
Confusion:
 True Pos: 1444/2470
 False Neg: 1026/2470
 False Pos: 1577/9700
 True Neg: 8123/9700
Median ID Time: 52.0, Mean ID Time: 179.5670867309118
CPU times: user 188 ms, sys: 11.9 ms, total: 199 ms
Wall time: 4.63 s
```

```
[20]: %%time
      wSize = 60

      qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
      resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
      resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
       ↪labels[wSize],
                        clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                        mxRuleLen, searchAttr, mxSetSize, nFolds)

      print(resStr)
      DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 60, MxRuleLen: 2, MxSetSz: 1, NFolds: 10
 Searched Attr: [0]
Precision: 0.5591
Recall: 0.5077
Confusion:
 True Pos: 1188/2340
 False Neg: 1152/2340
 False Pos: 937/6330
 True Neg: 5393/6330
Median ID Time: 83.0, Mean ID Time: 308.8800732377174
CPU times: user 167 ms, sys: 8.22 ms, total: 175 ms
Wall time: 5.28 s
```

[21]:
```
%%time
wSize = 90

qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
 ↪labels[wSize],
                  clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                  mxRuleLen, searchAttr, mxSetSize, nFolds)

print(resStr)
DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 90, MxRuleLen: 2, MxSetSz: 1, NFolds: 10
 Searched Attr: [0]
Precision: 0.7080
Recall: 0.4974
Confusion:
 True Pos: 1154/2320
 False Neg: 1166/2320
 False Pos: 476/5120
 True Neg: 4644/5120
Median ID Time: 117.0, Mean ID Time: 357.2525035765379
CPU times: user 166 ms, sys: 8.65 ms, total: 175 ms
Wall time: 6.84 s
```

[22]:
```
%%time
wSize = 120

qual = rs.RuleQualFScore(covTblInfo[wSize].FtrStrs,betaSq)
resStr = 'FScore (betaSq = {:.2f})\n'.format(betaSq)
resStr = Evaluate(client,resStr, covTblInfo[wSize], covTbl[wSize],␣
 ↪labels[wSize],
                     clientIds[wSize], idTimes[wSize], tte, wSize, qual,
                     mxRuleLen, searchAttr, mxSetSize, nFolds)

print(resStr)
DumpResultStr(resStr,resultFileStr)
```

```
FScore (betaSq = 0.25)
 WinSize: 120, MxRuleLen: 2, MxSetSz: 1, NFolds: 10
 Searched Attr: [0]
Precision: 0.7961
Recall: 0.5216
Confusion:
 True Pos: 1210/2320
 False Neg: 1110/2320
 False Pos: 310/4330
 True Neg: 4020/4330
Median ID Time: 145.0, Mean ID Time: 337.5809885931559
CPU times: user 160 ms, sys: 12.2 ms, total: 173 ms
Wall time: 7.71 s
```

Window Size Rules

[14]:
```
mxRuleLen = 2
mxSetSize = 1
betaSq = 0.25
searchAttr = searchAttrCore
```

[15]:
```
win = 30

ruleQual = rs.RuleQualFScore(covTblInfo[win].FtrStrs,betaSq=betaSq)
ruleSearch = rs.OpusRuleSearch(ruleQuality=ruleQual, maxRuleLen=mxRuleLen)
ruleSearch.set_search_attributes(searchAttrCore)

ruleSet = rs.rule_set_search(
    ruleQual, ruleSearch,
    covTblInfo[win],covTbl[win],labels[win],
    idx=None, maxSetSize=mxSetSize
)

print('WinSz: {}, {}'.format(win,ruleQual.ruleset_str(ruleSet)))
```

```
WinSz: 30, ['A0 >= 28' 'A3 < 0.5']
```

[16]:
```
win = 60

ruleQual = rs.RuleQualFScore(covTblInfo[win].FtrStrs,betaSq=betaSq)
ruleSearch = rs.OpusRuleSearch(ruleQuality=ruleQual, maxRuleLen=mxRuleLen)
ruleSearch.set_search_attributes(searchAttrCore)
```

```
ruleSet = rs.rule_set_search(
    ruleQual, ruleSearch,
    covTblInfo[win],covTbl[win],labels[win],
    idx=None, maxSetSize=mxSetSize
)

print('WinSz: {}, {}'.format(win,ruleQual.ruleset_str(ruleSet)))
```

```
WinSz: 60, ['A0 >= 54' 'A1 < 10.5']
```

[17]:
```
win = 90

ruleQual = rs.RuleQualFScore(covTblInfo[win].FtrStrs,betaSq=betaSq)
ruleSearch = rs.OpusRuleSearch(ruleQuality=ruleQual, maxRuleLen=mxRuleLen)
ruleSearch.set_search_attributes(searchAttrCore)

ruleSet = rs.rule_set_search(
    ruleQual, ruleSearch,
    covTblInfo[win],covTbl[win],labels[win],
    idx=None, maxSetSize=mxSetSize
)

print('WinSz: {}, {}'.format(win,ruleQual.ruleset_str(ruleSet)))
```

```
WinSz: 90, ['A0 >= 78' 'A3 < 3.5']
```

[18]:
```
win = 120

ruleQual = rs.RuleQualFScore(covTblInfo[win].FtrStrs,betaSq=betaSq)
ruleSearch = rs.OpusRuleSearch(ruleQuality=ruleQual, maxRuleLen=mxRuleLen)
ruleSearch.set_search_attributes(searchAttrCore)

ruleSet = rs.rule_set_search(
    ruleQual, ruleSearch,
    covTblInfo[win],covTbl[win],labels[win],
    idx=None, maxSetSize=mxSetSize
)

print('WinSz: {}, {}'.format(win,ruleQual.ruleset_str(ruleSet)))
```

```
WinSz: 120, ['A0 >= 99' 'A3 < 4.5']
```

[ ]:

[ ]:

[ ]: