

## binr.py

Page 1/3

```

1 #!/usr/bin/env python3 -B
2 # vim: ts=2:sw=2:sts=2:et
3 """
4 binr.py : build rules via stochastic incremental XAI
5 (c) 2025, Tim Menzies, timm@ieee.org, mit-license.org
6
7 Options:
8 -h          Show help.
9 -b bins=7   Number of bins for discretization (int).
10 -B Budget=30 Max rows to eval (int).
11 -c era=10   Number of rows in an era (int)
12 -p p=2     Distance coefficient
13 -r repeats=20 Number of experimental repeats (int).
14 -s seed=42  Random number seed (int).
15 -f file=.../data/aut93.csv File to load (str).
16 """
17 from types import SimpleNamespace as o
18 from math import floor, sqrt, cos, log, exp, pi
19 from typing import Any, Iterable
20 import fileinput, random, sys, re
21 rand = random.random
22
23 the = o(bins=7, Budget=30, era=10, p=2, repeats=20, seed=42,
24        file="/data/aut93.csv")
25
26 QTY = float | int
27 ATOM = QTY | str
28 ROW = list[ATOM]
29 ROWS = list[ROW]
30 NUM, SYM, COLS = o, o, o # redefined below
31 COL = NUM | SYM # redefined below
32 DATA = tuple[ROWS, COLS] # redefined below
33
34 # -----
35 def SYM() -> SYM:
36     return o(it=SYM, n=0, has={}, bins={})
37
38 def NUM() -> NUM:
39     return o(it=NUM, n=0, mu=0, sd=0, m2=0, bins={})
40
41 def COL(at=0, of="") -> COL:
42     it = (NUM if of[0].isupper() else SYM)()
43     it.at = at
44     it.of = of
45     it.best = str(of)[-1] if "-" else ""
46     return it
47
48 def COLS(names: list[str]) -> COLS:
49     t = [COL(at=i, of=s) for i, s in enumerate(names)]
50     return o(it=COLS, names=names,
51            all = t, for c in t if c.of[-1] not in "4-X",
52            X = [c for c in t if c.of[-1] != "X" and c.of[-1] in "4-"],
53            y = [c for c in t if c.of[-1] != "X" and c.of[-1] in "4-"])
54
55 def DATA(rows: ROWS = None) -> DATA:
56     return adds(rows, o(it=DATA, n=0, rows=[], cols=None))
57
58

```

## binr.py

Page 2/3

```

59 def add(i: o, v: ATOM | ROW) -> ATOM | ROW:
60     if v=="?": return v
61     i.n += 1
62     if SYM is i.it: i.has[v] = 1 + i.has.get(v, 0)
63     elif NUM is i.it:
64         v = float(v)
65         d = v - i.mu
66         i.mu += d/i.n
67         i.m2 += d*(v - i.mu)
68         i.sd = 0 if i.n < 2 else sqrt(i.m2/(i.n - 1))
69     elif DATA is i.it:
70         if i.cols:
71             i.rows += [[add(c, v[c.at]) for c in i.cols.all]]
72         else: i.cols = COLS(v)
73         return v
74
75 def adds(src: str | Iterable = None, it: o | None = None) -> o:
76     it = it or NUM()
77     if str(src)[-4:]=="*.csv":
78         with open(src, encoding="utf-8") as f:
79             for line in f:
80                 if line: add(it, [s.strip() for s in line.split(",")])
81     else: [add(it, row) for row in (src or [])]
82     return it
83
84 # -----
85 def norm(num: NUM, v: ATOM) -> float:
86     return 1 / (1 + exp(-1.702 * (v - num.mu) / (num.sd + 1e-32)))
87
88 def bin(col: COL, v: ATOM) -> int | ATOM:
89     return floor( (the.bins * norm(col, v) ) if v!="?" and NUM is col.it else v)
90
91 def dist(src: Iterable) -> float:
92     d, n = 0, 0
93     for d1 in src:
94         n += 1
95         d += d1 ** the.p
96     return (d/n) ** (1/the.p)
97
98 def disty(data: DATA, row: ROW) -> float:
99     return dist(abs(norm(col, row[col.at]) - col.best) for col in data.cols.y)
100
101 def distx(data: DATA, row1: ROW, row2: ROW) -> float:
102     return dist(_aha(col, row1[col.at], row2[col.at]) for col in data.cols.x)
103
104 def _aha(col: COL, a: ATOM, b: ATOM) -> float:
105     if a=="?": return 1
106     if SYM is col.it: return a != b
107     a, b = norm(col, a), norm(col, b)
108     a = a if a != "?" else (0 if b>0.5 else 1)
109     b = b if b != "?" else (0 if a>0.5 else 1)
110     return abs(a - b)
111
112 # -----
113 def scoreGet(data, row):
114     return sum(x.bins[b].mu for x in data.cols.x
115              if (b := bin(x, row[x.at])) in x.bins)
116
117 def scorePut(data, row, score):
118     for x in data.cols.x:
119         if (b := bin(x, row[x.at])) != "?":
120             x.bins[b] = x.bins.get(b) or NUM(x.at, b)
121             add(x.bins[b], score)
122
123 def score(data, eps=0):
124     best_score, best_row = 1e32, None
125     seen, rows, model = set(), shuffle(data.rows), DATA([data.cols.names])
126     for j, row in enumerate(rows):
127         if len(seen) >= the.Budget: break
128         add(model, row)
129         scorePut(model, row, disty(model, rows))
130         seen.add(id(row))
131         if j % the.era == 0:
132             candidate = min(rows[j+1 : j+20], key=lambda r: scoreGet(model, r))
133             seen.add(id(candidate))
134             if (score := disty(model, candidate)) < best_score - eps:
135                 best_score, best_row = score, candidate
136     return best_row
137
138

```

## binr.py

Page 3/3

```

139 # -----
140 def test_h(_) -> None:
141     print(__doc__)
142
143 def test_the(_) -> None:
144     print(the)
145
146 def test_s(n: str) -> None:
147     the.seed = float(n); random.seed(the.seed)
148
149 def test_sym(_) -> None:
150     print(adds("aaabbc", SYM()))
151
152 def test_num(_) -> None:
153     def box_muller(mu, sd):
154         return mu + sd * sqrt(-2 * log(rand())) * cos(2 * pi * rand())
155     print(adds(box_muller(10, 2) for _ in range(100)))
156
157 def test_data(f) -> None:
158     data = DATA(the.file)
159     print(data.cols.x[-1])
160     print(len(data.rows))
161
162 def test_distx():
163     ys, data = NUM(), DATA(the.file)
164     Y=lambda row: floor(100*distx(data, row))
165     for r in sorted(data.rows, key=Y)[:20]:
166         print(Y(r), r)
167
168 def test_distx():
169     xs, data = NUM(), DATA(the.file)
170     X=lambda row1: floor(100*distx(data, row1, data.rows[0]))
171     for r in sorted(data.rows, key=X)[:20]:
172         print(X(r), r)
173
174 def test_score():
175     d = DATA(the.file)
176     _tests = {k:fun for k, fun in vars().items() if "test_" in k}
177
178 def test_all():
179     for k, fun in _tests.items(): print("n----- "+k); fun(_)
180
181 # -----
182 if __name__ == "__main__":
183     for n, g in enumerate(sys.argv):
184         if fn := vars().get(f"test{g.replace('-', '_')}"):
185             fn(sys.argv[n+1] if n < len(sys.argv)-1 else None)
186

```