

```

1  #!/usr/bin/env lua
2  -- vim: filetype=lua ts=2 sw=2 et:
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```

(c) 2022, Tim Menzies, opensource.org/licenses/Fair
 Usage of the works is permitted provided that this instrument is
 retained with the works, so that any entity that uses the works is
 notified of this instrument. DISCLAIMER: THE WORKS ARE WITHOUT WARRANTY.

-- xxxx kill cloning
 -- add back here the shorter doc string and maom amd go.rogue

local b4={}; for k,v in pairs(_ENV) do b4[k]=v end
 local any,coerce,csv,ent,fails,fmt,fo,go,id,it,main,many,map,cbj,push
 local no,o,oo,ok,per,r,rnd,rnds,runDemo,same,sd,settings,shuffle,sort,sum

local the,help={}, [(
 wicket: explore the world better, explore the world for good.
 (c) 2022, Tim Menzies, opensource.org/licenses/Fair

```

35  USAGE:
36  wicket.lua [OPTIONS]
37
38  OPTIONS:
39  --cohen      -c cohen              = .35
40  --K          -K manage low class counts = 1
41  --M          -M manage low evidence counts = 2
42  --far        -f how far to go for far = .9
43  --p          -p coefficient on distance = 2
44  --seed       -S seed              = 10019
45  --some       -s sample size for distances = 512
46  --stop       -T how far to go for far = 20
47  --min        -m size of min space      = .5
48  --best       -B best percent          = .05
49
50  OPTIONS (other):
51  --dump       -d dump stack+exit on error = false
52  --file       -f file name              = ../etc/data/auto93.csv
53  --help       -h show help              = false
54  --rnd        -r rounding numbers       = %5.3f
55  --todo       -t start up action        = nothing ]]
56
57

```

```

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

--- LIB
 r = math.random
 fmt = string.format
 --- t a b l e s
 function same(x) return x end
 function fu(x) return function(t) return t[x] end end
 function lt(x) return function(t,u) return t[x] < u[x] end end
 function push(t,x) t[#t+1]=x; return x end
 function map(t,f,u) u={}for _,v in pairs(t) do u[#u+1]=f(v) end;return u end
 function sort(t,f) table.sort(t,f); return t end
 function sum(t,f,n) n=0; for _,x in pairs(t) do n=n+(f or same)(x) end; return n end
 function shuffle(t, j) for i=#t,2,-1 do j=math.random(i); t[i],t[j]=t[j],t[i] end; return t end
 function any(a, i) i=r()*.5//1; i=math.max(1,math.min(i,#a)); return a[i] end
 function many(a,n,u) if n>#a then return shuffle(a) end u={}for j=1,n do push(u,any(a)) end;return u end
 function sd(t,f) f=f or same; return (f(per(t,.9)) - f(per(t,.1)))/2.56 end
 function per(t,p) return t[((p or .5)*#t) // 1] end
 --- t h i n g s t o s t r i n g s
 function oo(t) print(o(t)) end
 function o(t, u,one,sorted) sorted = #t>0 -- true when array's indexes are 1,2...#t
 one= function(k,v) return sorted and tostring(v) or fmt("%s%s",k,v) end
 u={}; for k,v in pairs(t) do u[#u+1]= one(k,v) end
 return (t.is or "").."["..table.concat(sorted and u or sort(u),",").."]" end
 function rnds(t,f) return map(t, function(x) return rnd(x,f) end) end
 function rnd(x,f) return fmt(type(x)=="number" and (x=-x//1 and f or the.rnd) or"%s",x) end
 --- s t r i n g s t o t h i n g s
 function coerce(x) x = x:match("%s*(-)%s*\$"
 if x=="true" then return true else x=="false" then return false end
 return math.tointeger(x) or tonumber(x) or x end
 function csv(src) src = io.input(src)
 return function(line, row) line=io.read()
 if not line then io.close(src) else row={} for x in line:match("(^[^,]+)") do row[#row+1]=coerce(x) end
 return row end end
 --- m a i n
 function main(todo, all) all={}; for k,_, in pairs(go) do push(all,k) end
 all = the.todo=="all" and sort(all) or {sd}o
 for _,x in pairs(all) do runDemo(x) end
 for k,v in pairs(_ENV) do if not b4[k] then print("?",k,type(v)) end end
 os.exit(fails) end
 function runDemo(x, b4) b4={}; for k,v in pairs(the) do b4[k]=v end
 math.randomseed(the.seed)
 if go[x] then print(x); go[x]() end
 for k,v in pairs(b4) do the[k]=v end end
 function settings(txt, d) d={} txt:gsub("\n ([^~|~|([%s+)][%s+]-[%s+)][%s+]*%s([%s+])",
 function(long,key,short,x) for n,flag in ipairs(arg) do
 if flag==short or flag==long then x = x=="false" and "true" or x=="true" and "false" or arg[n+1] end end
 d[key] = coerce(x) end
 if d.help then print(txt) end
 return d end
 --- O B J E C T S
 function obj(name, t,new,str) function new(kl,...) local x=metatable({},{},kl); kl.new(x,...); return x end
 t = {__tostring=o, is=name or "", __index=t
 return setmetatable(t, (__call=new)) end
 --- b i n s
 local Bin=obj"Bin"
 function Bin:new(t) self.pos, self.txt, self.n, self.has = t.pos, t.txt, t.n, ()
 self.lo, self.hi, self.ystats = t.lo, t.hi, t.stats end
 function Bin:tostring() local x,lo,hi,big = self.txt, self.lo, self.hi, math.huge
 if lo == hi then return fmt("%s==%s",x, lo)
 elseif hi == big then return fmt("%s>=%s",x, lo)
 elseif lo == -big then return fmt("%s<=%s", x, hi)
 else return fmt("%s< %s< %s",lo,x,hi) end end
 function Bin:select(t) t = t.cells and t.cells or t
 local x, lo, hi = t[self.pos], self.lo, self.hi
 return x=="?" or lo == hi and lo == x or lo <= x and x < hi end

```

175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

--- s y m
 local Sym=obj"Sym"
 function Sym:new(pos,txt) self.pos = pos or 0
 self.txt = txt or ""
 self.n = 0
 self.has, self.mode, self.most = {},nil,0 end
 function Sym:sub(x) return self:add(x,-1) end
 function Sym:add(x,inc) if x ~= "?" then
 inc = inc or 1
 self.n = self.n + inc
 self.has[x] = (self.has[x] or 0) + inc
 if self.has[x] > self.most then self.most, self.mode = self.has[x], x end end
 return x end
 function Sym:mid() return self.mode end
 function Sym:div(e) e=0; for _,m in pairs(self.has) do
 if m>0 then e = e-m/self.n * math.log(m/self.n,2) end end
 return e end
 function Sym:dist(x,y) return x=="?" and y=="?" and 1 or x==y and 0 or 1 end
 function Sym:bins(rows, x,n,out,has,tmp,inc) n,out,tmp = 0,(),()
 function inc(x) n=n+1; return n end
 function has(x) tmp[x]=tmp[x] or Bin((txt=self.txt, pos=self.pos, n=inc(x),
 lo=x,hi=x, stats=Sym())) end
 for _,r in pairs(rows) do x = r.cells[self.pos]; has(x); tmp[x].ystats:add(r.klass) end
 for _,x in pairs(tmp) do push(out, x) end
 return out end
 --- m a i n
 local Num=obj"Num"
 function Num:new(pos,txt) self.pos = pos or 0
 self.txt = txt or ""
 self.n, self.mu, self.m2 = 0,0,0
 self.w = self.txt:find"\$" and -1 or 1
 self.lo, self.hi = math.huge, -math.huge end
 function Num:add(x, d) if x ~= "?" then
 self.n = self.n + 1
 self.lo = math.min(x, self.lo)
 self.hi = math.max(x, self.hi)
 d = x - self.mu
 self.mu = self.mu + d/self.n
 self.m2 = self.m2 + d*(x - self.mu) end
 return x end
 function Num:mid() return self.mu end
 function Num:div() return (self.m2/(self.n - 1))^0.5 end
 function Num:norm(x, lo,hi) lo,hi= self.lo, self.hi
 return x=="?" and x or hi-lo < 1E-9 and 0 or (x - lo)/(hi - lo) end
 function Num:dist(x,y) if x=="?" and y=="?" then return 1 end
 if x=="?" then y = self:norm(y); x = y<.5 and 1 or 0
 elseif y=="?" then x = self:norm(x); y = x<.5 and 1 or 0
 else x,y = self:norm(x), self:norm(y) end
 return math.abs(x - y) end
 local _bins
 function _bins(rows, xy,f) x=row.cells[self.pos]; if x=="?" then return {x=y=row.klass} end end
 xy = sort(map(rows,f),lt"x")
 return _bins(self.txt,self.pos,xy,sd(xy, fu"x")*the.cohen, (#xy)^the.min) end
 function _bins(txt,pos,xy,epsilon,small, div,b4,out) function div(lo,hi, x,y,cut,lhs,rhs,tmp,best,overall)
 lhs, rhs, overall = Sym(), Sym(), Sym()
 for i=lo,hi do overall:add(rhs:add(xy[i].y)) end
 best = rhs:div()
 for i=lo,hi do x, y = xy[i].x, xy[i].y
 lhs:add(rhs:sub(y))
 if lhs.n > small and rhs.n > small then
 if x ~= xy[i+1].x then
 if x - xy[lo].x > epsilon and xy[hi].x - x > epsilon then
 tmp = (lhs.n*lhs:div() + rhs.n*rhs:div()) / (lhs.n + rhs.n)
 if tmp < best then
 best,cut = tmp,i end end end end
 if cut
 then div(lo, cut)
 div(cut+1, hi)
 else b4 = push(out, Bin((txt=txt, pos=pos, n=1+#out,lo=b4,
 hi=xy[hi].x, stats=overall))).hi end
 end
 b4, out = -math.huge, ()
 div(1,#xy)
 out[#out].hi = math.huge
 return out end

```

279 ---  i  o  w
280
281 local Row=obj"Row"
282 function Row:new(t)
283     self.evaluated, self.class, self.cells = false,false,t end
284
285
286 ---  c  o  l  s
287
288 local Cols=obj"Cols"
289 function Cols:new(names, col)
290     self.names, self.all, self.x, self.y, self.class = names, {}, {}, {}, nil
291     for pos,txt in pairs(names) do
292         col = push(self.all, (txt:find"^[A-Z]" and Num or Sym) (pos,txt))
293         if not txt:find"3" then
294             if txt:find"5" then self.class=col end
295             col.indep = not txt:find"[+]"
296             push(col.indep and self.x or self.y, col) end end end
297
298 function Cols:add(row)
299     for _,col in pairs(self.all) do col:add(row[col.pos]) end end
300
301
302 ---  e  g  s
303
304 local Egs=obj"Egs"
305 function Egs:new() self.rows,self.cols = {}, nil end
306
307 function Egs:clone(rows, out)
308     out = Egs():add(self.cols.names)
309     for _,row in pairs(rows or {}) do out:add(row) end
310     return out end
311
312 function Egs:load(file)
313     for row in csv(file) do self:add(row) end; return self end
314
315 function Egs:add(t, row)
316     if self.cols
317     then row = t.cells and t or Row(t)
318         self.cols:add(row.cells)
319         push(self.rows, row)
320     else self.cols=Cols(t) end
321     return self end
322
323 function Egs:better(row1,row2)
324     local s1, s2, n, e = 0, 0, self.cols.y, math.exp(1)
325     for _,col in pairs(self.cols.y) do
326         local a = col:norm(row1.cells[col.pos])
327         local b = col:norm(row2.cells[col.pos])
328         s1 = s1 - e^(col.w * (a - b) / n)
329         s2 = s2 - e^(col.w * (b - a) / n) end
330     return s1 / n < s2 / n end
331
332 function Egs:betters(rows)
333     return sort(rows or self.rows, function(a,b) return self:better(a,b) end) end
334
335 function Egs:mid(cols)
336     return rnds(map(cols or self.cols.y, function(col) return col:mid() end)) end
337
338 function Egs:dist(row1,row2, d,n)
339     d = sum(self.cols.x, function(col)
340         return col:dist(row1.cells[col.pos], row2.cells[col.pos])^the.p end)
341     return (d / (#self.cols.x)) ^ (1/the.p) end
342
343 function Egs:around(row1, rows, around)
344     function around(row2) return (dist=self:dist(row1,row2),row=row2) end
345     return sort(map(rows or self.rows,around), lt"dist") end
346
347 function Egs:far(row, rows)
348     return per(self:around(row, rows or many(self.rows,the.some)),the.far).row end
349
350 function Egs:sway(rows,stop, x,rest, some,y,best,a,b,c)
351     rows = rows or self.rows
352     rest = rest or {}
353     stop = stop or 2*the.best*#self.rows
354     if #rows <= stop then return rows,rest end
355     some = many(rows,the.some)
356     x = x or self:far(any(some), some)
357     y = self:far(x, some)
358     if self:better(y, x) then x,y = y,x end
359     x.evaluated = true
360     y.evaluated = true
361     c = self:dist(x,y)
362     for _,row in pairs(rows) do
363         a,b = self:dist(row,x), self:dist(row,y)
364         row.x = (a^2+c^2-b^2)/(2*c) end
365     best = {}
366     for i,row in pairs(sort(rows, lt"x")) do
367         push(i<#rows//2 and best or rest, row) end
368     return self:sway(best, stop, x,rest) end
369
370 function Egs:leaves(rows,stop,leaves, best,w,bw)
371     leaves= leaves or {}
372     rows = rows or self.rows
373     stop = stop or 2*(#self.rows)^the.min
374     print(1)
375     function w(bin) return bin.ystats.n/#rows * bin.ystats:div() end
376     function bw(bins) return (bins=bins, worth=sum(bins,w)) end
377     print(3)
378     if #rows < stop then
379         return push(leaves,self:clone(rows)) end
380     print(3.1)
381     tmp=map(self.cols.x,function(c) return bw(c:bins(rows))end)
382     oo(tmp[1].bins[1].ystats.has)
383     os.exit()
384     best=sort(map(self.cols.x,function(c) return bw(c:bins(rows))end),lt"worth") [1]
385     print(4)
386     for _,row in pairs(rows) do
387         for _,bin in pairs(best.bins) do
388             if bin:select(row) then push(bin.has, row); break; end end end
389     for _,bin in pairs(best.bins) do
390         if #bin.has < #rows then bin.has= self:leaves(bin.has,stop,leaves) end end
391     return leaves end
392
393
394

```

```

395
396 --- DEMOS
397
398 fails,go,no = 0, {}, {}
399 function ok(test,msg)
400     print("", test and "PASS"or "FAIL", msg or "")
401     if not test then
402         fails = fails+1
403         if the.dump then assert(test,msg) end end end
404
405 function go.sum( t)
406     print(sum((1,2,3),same)) end
407
408 function go.list( t)
409     t={}; for txt,_, in pairs(go) do if txt=="list" then push(t,txt) end end
410     for _,txt in pairs(sort(t)) do print(fmt("lua wicket.lua -t%s",txt)) end end
411
412 function go.div( s)
413     s=Sym()
414     for _,x in pairs("a","a","a","a","b","b","c") do s:add(x) end
415     ok(math.abs(1.376 - s:div()) < 0.01, "cm") end
416
417 function go.symbols( eg,rows)
418     eg = Egs():load(the.file)
419     rows = eg:betters()
420     for _,row in pairs(rows) do row.class=false end
421     for i=1,(#rows)*.2 do rows[i].class=true end
422     for _,col in pairs(eg.cols.x) do
423         for k,v in pairs(col:bins(rows)) do print(v) end end end
424
425 function go.leaves( eg,rows,s,tree)
426     eg = Egs():load(the.file)
427     rows = eg:betters()
428     for i=1,(#rows)*.2 do rows[i].class=true end
429     s=Sym()
430     for _,row in pairs(rows) do s:add(row.class) end
431     for _,egl in pairs(eg:leaves(eg.rows,10)) do
432         oo(egl:mid()) end
433     end
434
435 function go.many()
436     oo(many((10,20,30,40,50,60,70,80,90,100),100)) end
437
438 function go.sway( eg,best,gusses,rest)
439     local used = function(row) if row.evaluated then return true end end
440     eg = Egs():load(the.file)
441     oo(map(eg.cols.y, function(col) return col.txt end))
442     oo(map(eg.cols.y, function(col) return col.w end))
443     print("before",o(eg:mid()))
444     best,rest = eg:sway()
445     print("sway", o(eg:clone(best):mid()))
446     print("eval",#map(eg.rows,used), #best, #rest)
447     take2={}
448     for _,row in pairs(best) do row.class=true; push(take2,row) end
449     for _,row in pairs(many(rest,3*#best)) do row.class=false; push(take2,row) end
450     oo(take2)
451     eg:leaves(take2,5)
452     -- for _,row in pairs(rest) do row.class=false end
453     -- for _,row in pairs(best) do row.class=true end
454     -- for _,row in pairs(many(rest,3*#best)) do push(best,row) end
455     -- for _,egl in pairs(eg:leaves(best)) do
456     --     print(
457     --         for _,row in pairs(many(rest, 3*#gusses)) do push(
458     --             best= eg:clone()
459     --             for i,row in pairs(eg:betters()) do if i< the.best*#eg.rows then best:add(row)
460     --             ) else break end end
461     --             print("best",o(best:mid()))
462     --         end
463     --     end
464     function go.egl( eg)
465         eg = Egs():load(the.file)
466         print(#eg.rows, eg.cols.y[1]) end
467
468 function go.far( eg)
469     eg = Egs():load(the.file)
470     print(eg:far(eg.rows[1],eg.rows)) end
471
472 function go.around( eg)
473     eg = Egs():load(the.file)
474     print(eg:around(eg.rows[1])) end
475
476 function go.dist( eg,row2,t)
477     eg = Egs():load(the.file)
478     t={}; for i=1,20 do
479         row2= any(eg.rows)
480         push(t, (dist=eg:dist(eg.rows[1],row2), row = row2)) end
481     oo(eg.rows[1].cells)
482     print("—")
483     for _,two in pairs(sort(t,lt"dist")) do oo(two.row.cells) end end
484
485 function go.mids( eg,hi,lo,out)
486     eg = Egs():load(the.file)
487     oo(map(eg.cols.y, function(col) return col.txt end))
488     oo(map(eg.cols.y, function(col) return col.w end))
489     print("all",o(eg:mid()))
490     lo,hi = eg:clone(), eg:clone()
491     for i,row in pairs(eg:betters()) do
492         if i < 20 then lo:add(row) end
493         if i > #eg.rows - 20 then hi:add(row) end end
494     print("lo",o(lo:mid()))
495     print("hi",o(hi:mid())) end
496
497
498 --- START
499
500 the = settings(help)
501 main(the.todo)
502
503
504
505
506
507
508
509
510
511
512
513
514

```

```

515 --
516 --
517 --
518 --

```

