
This is the title of the template article

Firstname Lastname, University of Examples

Here is some sample text to show the initial in the introductory paragraph of this template article. The color and lineheight of the initial can be modified in the preamble of this document.

Heading on level 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi.

Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

$$A = \begin{bmatrix} A_{11} & A_{21} \\ A_{21} & A_{22} \end{bmatrix} \quad (1)$$

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus.

Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies ??

Heading on level 2

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.

- First item in a list
- Second item in a list
- Third item in a list

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim.

Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

```
#!/usr/bin/env python3.9
# vim: ts=2 sw=2 sts=2 et :
# autotep8: ignore E20,E401,E226,E302,E41
```

```

4 import re, sys, math, argparse, itertools
5 from argparse import ArgumentParser as parse
6 from argparse import RawTextHelpFormatter as textual
7 Float = Str = Int = Bool = lambda *l: l[0]
8
9 def keys(
10     BINS : Float("bins are of size n**BINS") = .5,
11     COLS : Str("columns to use for inference") = "x",
12     DATA : Str("where to read data") = "../data/auto2.csv",
13     EPSILON : Float("small = sd**EPSILON") = .3,
14     FAR : Float("where to look for far things") = .9,
15     GOAL : Str("learning goals: best—rest—other") = "best",
16     K : Int("bayes low class frequency hack") = 2,
17     M : Int("bayes low range frequency hack") = 1,
18     P : Int("distance calculation exponent") = 2,
19     SAMPLE : Int("#samples to find far things?") = 20,
20     VERBOSE : Bool("set verbose") = False,
21     TOP : Int("focus on this many") = 20,
22     XAMPLE : Str("egs: fl-x lsfl lists all, fl-x allfl runs all") = "" ):
23     """
24     /-—- (c) Tim Menzies, 2021, unlicense.org.
25     /-—- "The delta between things is
26     /-—- "simpler than the things.
27     /-—- v """"
28
29     GOAL = -flbestfl : lambda b, r: b**2/b+r,
30     flrestfl : lambda b, r: r**2/(b+r),
31     flotherfl : lambda b, r: 1/(b+r) "[GOAL]
32
33     # -----
34     class Col(o):
35         "Store columns in `Col`, `Skip`, `Sym`, `Num`."
36         def "init"(i, at=0, txt="", inits=[]):
37             i.n, i.at, i.txt = 0, at, txt
38             i.w = -1 if "-" in txt else 1
39             [i.add(x) for x in inits]
40
41         def add(i, x, n=1):
42             if x != "?": i.n += 1; x = i.add1(x, n)
43             return x
44
45     # -----
46     class Skip(Col):
47         def add1(i, x, n=1): return x
48
49     # -----
50     class Sym(Col):
51         def "init"(i, **kw): i.has = "-"; super()."init"(**kw)
52
53         def add1(i, x, n=1): inc(i.has, x, n); return x
54
55     def bins(i, j):
56         for k in (i.has — j.has):
57             yield i.has.get(k, 0), True, (i.at, (k, k))
58             yield j.has.get(k, 0), False, (j.at, (k, k))
59
60     def dist(i, x, y): return 0 if x == y else 1
61
62     def ent(i):
63         return sum(-v/i.n * math.log(v/i.n) for v in i.has.values())
64
65     def merge(i, j):
66         k = Sym(at=i.at, txt=i.txt)
67         [k.add(x, n) for has in (i.has, j.has) for x, n in has.items()]
68         return k
69
70     def merged(i, j):
71         k = i.merge(j)
72         e1, n1, e2, n2, e, n = i.ent(), i.n, j.ent(), j.n, k.ent(), k.n
73         if e1 + e2 > 0.01 or e * .95 > n1 / n * e1 + n2 / n * e2:
74             return k
75
76     # -----
77     class Num(Col):
78         def "init"(i, **kw):
79             i.all, i.ok = [], False
80             super(). "init"(**kw)
81
82         def add1(i, x, n):
83             x, i.ok = float(x), False
84             for _ in range(n): i.all += [x]
85             return x
86
87         def all(i):
88             if not i.ok: i.ok = True; i.all = sorted(i.all)
89             return i.all
90
91     def bins(i, j):
92         xy = [(z, True) for z in i.all] + [(z, False) for z in j.all]
93         eps = EPSILON * (i.n*i.sd() + j.n*j.sd()) / (i.n + j.n)
94         for ((lo, hi), s) in bins(xy, epsilon=eps, size=len(xy)**BINS):
95             for klass, n in s.has.items():
96                 yield n, klass, (i.at, (lo, hi))
97
98     def dist(i, x, y):
99         if x == "?": y = i.norm(y); x = 1 if y < 0.5 else 0
100         elif y == "?": x = i.norm(x); y = 1 if x < 0.5 else 0
101         else : x, y = i.norm(x), y.norm(y)
102         return abs(x-y)
103
104     def norm(i, x):
105         if x == "?": return x
106         a = i.all()
107         return max(0, min(1, (x-first(a))/(last(a)-first(a)+1E-32)))
108
109     def sd(i) : return (per(i.all(), .9) - per(i.all(), .1))/2.56
110     def span(i) : return (first(i.all()), last(i.all()))
111     def wide(i, n=0): return last(i.all()) - first(i.all()) i = n
112
113     # -----
114     class Row(o):
115         "Data is in `Row`s which, in turn, are in `Table`s."
116         def "init"(i, lst, tab=None): i.tab, i.cells = tab, lst
117
118     def dist(i, j):
119         d = n = 1E-32
120         for col in i.tab.cols[COLS]:
121             n += 1
122             x, y = i.cells[at], j.cells[at]
123             d += 1 if x == "?" and y == "?" else col.dist(x, y) ** P
124         return (d/n) ** (1/P)
125
126     def far(i, rows):
127         tmp = [(dist(i, j), j) for _ in range(SAMPLE)]
128         return per(sorted(tmp, key=first), FAR)
129
130     # -----
131     class Table(o):
132         def "init"(i, inits=[]):
133             i.rows = []
134             i.cols = o(all=[], names=[], x=[], y=[], klass=None)
135             [i.add(x) for x in inits]
136
137         def add(i, a): i.data(a) if i.cols.names else i.header(a)
138         def clone(i, inits=[]): return Table([i.cols.names] + inits)
139
140         def data(i, a):
141             a = a.cells if type(a) == Row else a
142             a = [col.add(a[col.at]) for col in i.cols.all]
143             i.rows += [Row(a, tab=i)]
144
145         def header(i, a):
146             i.cols.names = a
147             for at, x in enumerate(a):
148                 new = Skip if i.skipp(x) else (Num if i.nump(x) else Sym)
149                 new = new(at=at, txt=x)
150                 i.cols.all += [new]
151                 if not i.skipp(x):
152                     i.cols["y"] if i.y(x) else "x" += [new]
153                     if i.klassp(x):
154                         i.cols.klass = new
155
156         def klassp(i, x): return "!" in x
157         def nump(i, x): return x[0].isupper()
158         def skipp(i, x): return "?" in x
159         def yp(i, x): return "-" in x or "+" in x or i.klassp(x)
160
161     # -----
162     def stratify(src):
163         all, klass = None, "-"
164         for n, row in enumerate(src):
165             if all:
166                 kl = row[all.cols.klass.at]
167                 here = klass[kl] = klass.get(kl, None) or all.clone()
168                 here.add(row)
169                 all.add(row)
170             else:
171                 all = Table([row])
172         return o(all=all, klass=klass)
173
174     # -----
175     def bins(xy, epsilon=0, size=30):
176         "Use `bins` to divide numeric data into ranges."

```

```

170 def merge(b4):
171     j, tmp, n = 0, [], len(b4)
172     while j < n:
173         a = b4[j]
174         if j < n - 1:
175             b = b4[j + 1]
176             print("na", a[1])
177             print("b", b[1])
178             if cy := a[1].merged(b[1]):
179                 print("c", cy)
180                 a = ((a[0][0], b[0][1]), cy)
181                 j += 1
182             tmp += [a]
183             j += 1
184     return merge(tmp) if len(tmp) < len(b4) else b4
185
186 def divide(xy):
187     bin = o(x=Num(), y=Sym())
188     bins = [bin]
189     for i, (x, y) in enumerate(xy):
190         if bin.x.n < size:
191             if x != b4 and i < len(xy)-size and bin.x.wide(epsilon):
192                 bin = o(x=Num(), y=Sym())
193                 bins += [bin]
194             bin.x.add(x)
195             bin.y.add(y)
196             b4 = x
197     return bins
198
199     return merge([(bin.x.span(), bin.y)
200                  for bin in divide(sorted(xy, key=first))])
201 #-----
202 def contrasts(here, there, t):
203     "Report ranges that are most different in two classes."
204     def like(d, kl):
205         out = prior = (hs[kl] + K) / (n + K*2)
206         for at, span in d.items():
207             f = has.get((kl, (at, span)), 0)
208             out *= (f + M*prior) / (hs[kl] + M)
209         return out
210
211     def val(d): return GOAL(like(d, True), like(d, False)), d
212     def top(a): return sorted(a, reversed=True, key=first)[-TOP]
213
214     has = -(kl, (at, (lo, hi))): f
215         for col1, col2 in zip(here.cols.x, there.cols.x)
216         for f, kl, (at, (lo, hi)) in col1.bins(col2)
217     n = len(here.rows, there.rows)
218     hs = -True: len(here.rows), False: len(there.rows)
219     solos = [val(dict(at=x)) for at, x in set([z for , z in has])]
220     ranges = -
221     for , d in top(solos):
222         for k in d:
223             ranges[k] = ranges.get(k, set()).add(d[k])
224     for rule in top([val(d) for d in dict'product(ranges)]):
225         print(rule)
226 #-----
227 # Unit tests.
228 class Eg:
229     def ls():
230         "list all examples."
231         print("nexamples:")
232         for k, f in vars(Eg).items():
233             if k[0] != "":
234                 print(f" -k:;13" -f.doc'')
235
236     def data(file="~/data/vote.csv"):
237         "simple load of data into a table"
238         t = Table(csv(file))
239         assert(435 == len(t.rows))
240         assert(195 == t.cols.all[1].has[flyfl])
241
242     def nclasses(file="~/data/diabetes.csv", kl="positive"):
243         ts = stratify(csv(file))
244         assert(2 == len(ts.klass))
245         assert(268 == len(ts.klass[kl].rows))
246         assert(768 == len(ts.all.rows))
247
248     def bins(file="~/data/diabetes.csv",
249             k1="positive", k2="negative"):
250         ts = stratify(csv(file))
251         goods, bads = ts.klass[k1], ts.klass[k2]
252         for good,bad in zip(goods.cols.all, bads.cols.all):
253
254             print(f"n-good.at'')
255             [print(f"t-x'') for x in good.bins(bad)]
256 #-----
257 # main program for keys
258 if XAMPLE == "all":
259     for k, f in vars(Eg).items():
260         if k[0] != "": print("n"+k); f()
261 else:
262     if XAMPLE and XAMPLE in vars(Eg): vars(Eg)[XAMPLE]()
263 #####
264 #-----
265 # things that don't use the config vars
266 # dictionaries
267 def has(d, k): return d.get(k, 0)
268 def inc(d, k, n=1): tmp = d[k] = n + d.get(k, 0); return tmp
269
270 def dict'product(d):
271     keys = d.keys()
272     for p in itertools.product(*d.values()):
273         yield dict(zip(keys, p))
274
275 # lists
276 def first(a): return a[0]
277 def last(a): return a[-1] #
278 def per(a, p=.5): return a[int(p*len(a))]
279
280 class o(object):
281     "object"
282     def __init__(i, **k): i.__dict__.update(**k)
283     def __getitem__(i, k): return i.__dict__[k]
284     def __repr__(i): return i.__class__.__name__+str(
285         -k:v for k, v in i.__dict__.items() if k[0] != " ")
286     def __setitem__(i, k, v): i.__dict__[k] = v
287
288 def csv(f=None, sep=","):
289     "read csv files"
290     def prep(s): return re.sub(rfl(["n"t"r ]-#.*)fl, flfl, s)
291     if f:
292         with open(f) as fp:
293             for s in fp:
294                 if s := prep(s): yield s.split(sep)
295     else:
296         for s in sys.stdin:
297             if s := prep(s): yield s.split(sep)
298
299 def cli(f):
300     "Drive command line flags from function annotations."
301     p = parse(prog="." + f.__name__, description=f.__doc__,
302              formatter'class=Textual)
303     used = -
304     for (k, h), b4 in zip(
305         list(f.__annotations__.items()), f.__defaults__):
306         used[k[0]] = c = k[0] if k[0] in used else k[0].lower()
307         if b4 == False:
308             p.add'argument("-" + c, dest=k, help=h,
309                          default=False, action="store'true")
310         else:
311             p.add'argument("-" + c, dest=k, default=b4,
312                          help=h+" [" + str(b4) + "]" , type=type(b4),
313                          metavar=k)
314     f(**p.parse'args().__dict__)
315 #-----
316 # Start up.
317 if __name__ == "__main__": cli(keys)
318

```

Heading on level 1 again

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus

Table 1: *Random table*

Name		
First name	Last Name	Grade
John	Doe	7.5
Richard	Miles	2

elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

Heading on level 2

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo.

First This is the first item

Last This is the last item

Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies