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, consectetuer 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

$$A = \begin{bmatrix} A_{11} & A_{21} \\ A_{21} & A_{22} \end{bmatrix} \tag{1}$$

Lorem ipsum dolor sit amet, consectetuer 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

277

Heading on level 2

Lorem ipsum dolor sit amet, consectetuer 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, consectetuer 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 :
# autopep8: ignore E20,E401,E226,E302,E41
```

```
import re, sys, math, argparse, itertools
from argparse import ArgumentParser as parse
from argparse import RawTextHelpFormatter as textual
Float = Str = Int = Bool = lambda *l: l[0]

def keys(
  BINS    : Float("bins are of size n**BINS") = .5,
  COLS    : Str("columns to use for inference") = "x",
  DATA    : Str("where to read data") = "../data/auto2.csv",
  EPSILON: Float("small = sd**EPSILON") = .3,
  FAR     : Float("where to look for far things") = .9,
  GOAL    : Str("learning goals: best—rest—other") = "best",
  K       : Int("bayes low class frequency hack") = 2,
  M       : Int("bayes low range frequency hack") = 1,
  P       : Int("distance calculation exponent") = 2,
  SAMPLE : Int("#samples to find far things?") = 20,
  VERBOSE: Bool("set verbose") = False,
  TOP     : Int("focus on this many") = 20,
  XAMPLE : Str("egs: fl-x lsfl lists all, fl-x allfl runs all") = "" ):
  """
   ,-˙—“      (c) Tim Menzies, 2021, unlicense.org.
  /      “    The delta between things is
  “˙,-.˙*    simpler than the things.
       v """

  GOAL = –flbestfl : lambda b, r: b**2/b+r,
          flrestfl : lambda b, r: r**2/(b+r),
          flotherfl: lambda b, r: 1/(b+r)     ˝[GOAL]
  #------------------------------------------------------------------------
  class Col(o):
    "Store columns in `Col`, `Skip`, `Sym`, `Num`."
    def ¨init¨(i, at=0, txt="", inits=[]):
      i.n, i.at, i.txt = 0, at, txt
      i.w = -1 if "-" in txt else 1
      [i.add(x) for x in inits]

    def add(i, x, n=1):
      if x != "?": i.n += 1; x = i.add1(x, n)
      return x
  # -----------------------------------
  class Skip(Col):
    def add1(i, x, n=1): return x
  # -----------------------------------
  class Sym(Col):
    def ¨init¨(i, **kw): i.has = –˝; super().¨init¨(**kw)

    def add1(i, x, n=1): inc(i.has, x, n); return x

    def bins(i, j):
      for k in (i.has — j.has):
        yield i.has.get(k, 0), True, (i.at, (k, k))
        yield j.has.get(k, 0), False, (j.at, (k, k))

    def dist(i, x, y): return 0 if x == y else 1

    def ent(i):
      return sum(-v/i.n * math.log(v/i.n) for v in i.has.values())

    def merge(i, j):
      k = Sym(at=i.at, txt=i.txt)
      [k.add(x, n) for has in (i.has, j.has) for x, n in has.items()]
      return k

    def merged(i, j):
      k = i.merge(j)
      e1, n1, e2, n2, e, n = i.ent(), i.n, j.ent(), j.n, k.ent(), k.n
      if e1 + e2 ¡ 0.01 or e * .95 ¡ n1 / n * e1 + n2 / n * e2:
        return k
  # -----------------------------------
  class Num(Col):
    def ¨init¨(i, **kw):
      i.˙all, i.ok = [], False
      super().¨init¨(**kw)

    def add1(i, x, n):
      x, i.ok = float(x), False
      for ˙ in range(n): i.˙all += [x]
      return x

    def all(i):
      if not i.ok: i.ok = True; i.˙all = sorted(i.˙all)
      return i.˙all

    def bins(i, j):
      xy = [(z, True) for z in i.˙all]+[(z, False) for z in j.˙all]
      eps = EPSILON * (i.n*i.sd() + j.n*j.sd()) / (i.n + j.n)
      for ((lo, hi),s) in bins(xy,epsilon=eps,size=len(xy)**BINS):
        for klass, n in s.has.items():
          yield n, klass, (i.at, (lo, hi))

    def dist(i, x, y):
      if    x == "?": y = i.norm(y); x = 1 if y ¡ 0.5 else 0
      elif y == "?": x = i.norm(x);y = 1 if x ¡ 0.5 else 0
      else           : x, y = i.norm(x), y.norm(y)
      return abs(x-y)

    def norm(i, x):
      if x == "?": return x
      a = i.all()
      return max(0, min(1, (x-first(a))/(last(a)-first(a)+1E-32)))

    def sd(i) : return (per(i.all(), .9) - per(i.all(), .1))/2.56
    def span(i) : return (first(i.all()), last(i.all()))
    def wide(i, n=0): return last(i.all()) - first(i.all()) ¿= n
  #------------------------------------------------------------------------
  class Row(o):
    "Data is in `Row`s which, in turn, are in `Table`s."
    def ¨init¨(i, lst, tab=None): i.tab, i.cells = tab, lst

    def dist(i, j):
      d = n = 1E-32
      for col in i.tab.cols[COLS]:
        n += 1
        x, y = i.cells[at], j.cells[at]
        d += 1 if x == "?" and y == "?" else col.dist(x, y) ** P
      return (d/n) ** (1/P)

    def far(i, rows):
      tmp = [(dist(i, j), j) for ˙ in range(SAMPLE)]
      return per(sorted(tmp, key=first), FAR)
  # -----------------------------------
  class Table(o):
    def ¨init¨(i, inits=[]):
      i.rows = []
      i.cols = o(all=[], names=[], x=[], y=[], klass=None)
      [i.add(x) for x in inits]

    def add(i, a): i.data(a) if i.cols.names else i.header(a)
    def clone(i, inits=[]): return Table([i.cols.names] + inits)

    def data(i, a):
      a = a.cells if type(a) == Row else a
      a = [col.add(a[col.at]) for col in i.cols.all]
      i.rows += [Row(a, tab=i)]

    def header(i, a):
      i.cols.names = a
      for at, x in enumerate(a):
        new = Skip if i.skipp(x) else (Num if i.nump(x) else Sym)
        new = new(at=at, txt=x)
        i.cols.all += [new]
        if not i.skipp(x):
          i.cols["y" if i.yp(x) else "x"] += [new]
          if i.klassp(x):
            i.cols.klass = new

    def klassp(i, x): return "!" in x
    def nump(i, x): return x[0].isupper()
    def skipp(i, x): return "?" in x
    def yp(i, x): return "-" in x or "+" in x or i.klassp(x)

  # -----------------------------------
  def stratify(src):
    all, klass = None,–˝
    for n,row in enumerate(src):
      if all:
        kl    = row[all.cols.klass.at]
        here = klass[kl] = klass.get(kl,None) or all.clone()
        here.add(row)
        all.add(row)
      else:
        all = Table([row])
    return o(all=all, klass=klass)
  #------------------------------------------------------------------------
  def bins(xy, epsilon=0, size=30):
    "Use `bins` to divide numeric data into ranges."
  def merge(b4):
    j, tmp, n = 0, [], len(b4)
    while j ¡ n:
      a = b4[j]
      if j ¡ n - 1:
        b = b4[j + 1]
        print(" “na",a[1])
        print("b",b[1])
        if cy := a[1].merged(b[1]):
          print("c",cy)
          a = ((a[0][0], b[0][1]), cy)
          j += 1
      tmp += [a]
      j += 1
    return merge(tmp) if len(tmp) ¡ len(b4) else b4

  def divide(xy):
    bin = o(x=Num(), y=Sym())
    bins = [bin]
    for i, (x, y) in enumerate(xy):
      if bin.x.n ¿= size:
        if x != b4 and i ¡ len(xy)-size and bin.x.wide(epsilon):
          bin = o(x=Num(), y=Sym())
          bins += [bin]
      bin.x.add(x)
      bin.y.add(y)
      b4 = x
    return bins

  return merge([(bin.x.span(), bin.y)
                for bin in divide(sorted(xy, key=first))])
#-----------------------------------------------------------------------------
def contrasts(here, there, t):
  "Report ranges that are most different in two classes."
  def like(d, kl):
    out = prior = (hs[kl] + K) / (n + K*2)
    for at, span in d.items():
      f = has.get((kl, (at, span)), 0)
      out *= (f + M*prior) / (hs[kl] + M)
    return out

  def val(d): return GOAL(like(d, True), like(d, False)), d
  def top(a): return sorted(a, reversed=True, key=first)[:TOP]

  has = –(kl, (at, (lo, hi))): f
         for col1, col2 in zip(here.cols.x, there.cols.x)
         for f, kl, (at, (lo, hi)) in col1.bins(col2)˝
  n = len(here.rows, there.rows)
  hs = –True: len(here.rows), False: len(there.rows)˝
  solos = [val(dict(at=x)) for at, x in set([z for ˙, z in has])]
  ranges = –˝
  for ˙, d in top(solos):
    for k in d:
      ranges[k] = ranges.get(k, set()).add(d[k])
  for rule in top([val(d) for d in dict˙product(ranges)]):
    print(rule)
#-----------------------------------------------------------------------------
# Unit tests.
class Eg:
  def ls():
    "list  all examples."
    print(" “nexamples:")
    for k, f in vars(Eg).items():
      if k[0] != "˙":
        print(f"  –k:¡13˝ –f.˙˙doc˙˙˝")

  def data(file="../data/vote.csv"):
    "simple load of data into  a table"
    t = Table(csv(file))
    assert(435 == len(t.rows))
    assert(195 == t.cols.all[1].has[flyfl])

  def nclasses(file="../data/diabetes.csv", kl="positive"):
    ts = stratify(csv(file))
    assert(2   == len(ts.klass))
    assert(268 == len(ts.klass[kl].rows))
    assert(768 == len(ts.all.rows))

  def bins(file="../data/diabetes.csv",
          k1= "positive", k2= "negative"):
    ts = stratify(csv(file))
    goods, bads = ts.klass[k1], ts.klass[k2]
    for good,bad in zip(goods.cols.all, bads.cols.all):
        print(f" “n–good.at˝")
        [print(f" “t–x˝") for x in good.bins(bad)]
  #---------------------------------------------------------------------------
  # main program for keys
  if XAMPLE == "all":
    for k, f in vars(Eg).items():
      if k[0] != "˙": print(" “n"+k); f()
  else:
    if XAMPLE and XAMPLE in vars(Eg): vars(Eg)[XAMPLE]()

##############################
#----------------------------------------------------------------------------
# things that donflt use the config vars
# dictionaries
def has(d, k): return d.get(k, 0)
def inc(d, k, n=1): tmp = d[k] = n + d.get(k, 0); return tmp

def dict˙product(d):
  keys = d.keys()
  for p in itertools.product(*d.values()):
    yield dict(zip(keys, p))

# lists
def first(a): return a[0]
def last(a): return a[-1]  #
def per(a, p=.5): return a[int(p*len(a))]

class o(object):
  "object"
  def ˙˙init˙˙(i, **k): i.˙˙dict˙˙.update(**k)
  def ˙˙getitem˙˙(i, k): return i.˙˙dict˙˙[k]
  def ˙˙repr˙˙(i):  return i.˙˙class˙˙.˙˙name˙˙+str(
        –k:v for k, v in i.˙˙dict˙˙.items() if k[0] != "˙"˝)
  def ˙˙setitem˙˙(i, k, v): i.˙˙dict˙˙[k] = v

def csv(f=None, sep=","):
  "read csv files"
  def prep(s): return re.sub(rfl([“n“t“r ]—#.*)fl, flfl, s)
  if f:
    with open(f) as fp:
      for s in fp:
        if s := prep(s): yield s.split(sep)
  else:
    for s in sys.stdin:
      if s := prep(s): yield s.split(sep)

def cli(f):
  "Drive command line flags  from function annocations."
  p = parse(prog="./"+f.˙˙name˙˙, description=f.˙˙doc˙˙,
           formatter˙class=textual)
  used = –˝
  for (k, h),b4 in zip(
                   list(f.˙˙annotations˙˙.items()),f.˙˙defaults˙˙):
    used[k[0]] = c = k[0] if k[0] in used else k[0].lower()
    if b4 == False:
      p.add˙argument("-"+c, dest=k, help=h,
                     default=False, action="store˙true")
    else:
      p.add˙argument("-"+c, dest=k, default=b4,
                     help=h+" ["+str(b4)+"]", type=type(b4),
                     metavar=k)
  f(**p.parse˙args().˙˙dict˙˙)
# ------------------------------------
# Start up.
if ˙˙name˙˙ == "˙˙main˙˙": cli(keys)
```

Heading on level 1 again

Lorem ipsum dolor sit amet, consectetuer 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, consectetuer 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, consectetuer 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