C200 Programming Assignment № 4

Dr. M.M. Dalkilic
Computer Science
School of Informatics, Computing, and Engineering

Indiana University, Bloomington, IN, USA

September 29, 2022

Introduction

Due Date: 10:59 PM, Thursday, October 6, 2022

Please submit your work **using** the new autograder before the deadline and remember to add, commit and push to Github. For this HW, we have not provided the unit-test file. There are print statements in the starter code that you can uncomment to see the function output. We encourage you to test the functions by using your own values as function parameters in the print statements.

Suggestions: Many of the functions can use previous functions you've implemented, so rely upon those to shorten your work-time and build better code. Remember, reviewing the slides **always** helps (this is absolutely necessary and also a hint).

This is the last homework before the midterm.

Problem 1: Longest run of ones

Given a list of zeros and ones, return the longest count of consecutive ones. For example,

```
x = [[0,1,1,0,0,0,1,1,1,1],[0,0],[1,1,0,1],[1,1,0,1,1,1,1],
     [1,1,1,1,1,1,1,1,1,1,1], [0,0,1,1,1,1,0,0]]

def ls(x):
    pass

for x in x:
    print(f"{x}  {ls(x)}")
```

produces

```
[0, 1, 1, 0, 0, 0, 1, 1, 1, 1]  4
[0, 0]  0
[1, 1, 0, 1]  2
[1, 1, 0, 1, 1, 1, 1]  4
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]  11
[0, 0, 1, 1, 1, 1, 0, 0]  4
```

Deliverables for Problem 1

- Complete the functions described above.

Problem 2: Computing Relationships

Assume you're building a friendship site. A person fills out a survey:

No or Yes	Question
(0 or 1)	You love dogs.
(0 or 1)	Your favorite color is a prime color.
(0 or 1)	If there's a choice between movie and book, you choose movie.

A person answer with three zero,one values. You do love dogs, your favorite color is blue, and you'd always read before watching the crappy movie version, so you list would be [1,1,0]. To match people, companies use trigonometry. Assume you have a list of people $[p_0, p_1, \ldots, p_m]$ where each person is a list of 0s,1s. We find the two different people who have the smallest angle. We need three basic functions: inner product, magnitude, and $\cos^{-1}$. We assume two lists $x = [x_0, x_1, \ldots, x_n], y = [y_0, y_1, \ldots, y_n]$ of 0s and 1s of the same length:

$$inner\_prod(x,y) = x_0y_0 + x_1y_1 + \cdots + x_ny_n \tag{1}$$

$$mag(x) = \sqrt{inner\_prod(x,x)} \tag{2}$$

For the last function (where we calculate the angle), we know that:

$$\cos(\theta) = \frac{inner\_prod(x,y)}{mag(x)mag(y)} \tag{3}$$

In class we learned that we can invert a function and the inverted function is denoted as f^{-1}. So we can:

$$\cos^{-1}(\cos(\theta)) = \cos^{-1}(\frac{inner\_prod(x,y)}{mag(x)mag(y)}) \tag{4}$$

$$\theta = \cos^{-1}(\frac{inner\_prod(x,y)}{mag(x)mag(y)}) \tag{5}$$

The math module has math.acos() for $\cos^{-1}()$. Further, Python returns θ in radians. We have:

$$\pi \text{ radians} = 180 \text{ degrees} \tag{6}$$

Thus, to convert from radians to degree, you **must** multiply your answer by $\frac{180}{\pi}$.

Your task is to first complete the functions “inner_prod“, “mag“ and “angle“ and then use them to write the “match“ and “best_match“ functions. The match function takes a list of people p_i where each person is a list of 0s 1s, and returns all unique pairs with the angle in degrees. **Note:** “match“ returns the output in the format: [[person 1, person2, angle], [person2, person3, angle], [person1, person3, angle]], where each person i.e., person1, person2 is represented by a list for example, [1,1,1] or [0,1,0]. Hence “match“ returns a list of lists.

Note: For “angle“ function-round your answer to 2 decimal digits.

The function “best_match“ takes the result from match and returns the pair with the “best“ match– the smallest degree. We can assume there’s only one best match. Here is a sample run (with some extra output) for your perusal.

```
people0 = [[0,1,1],[1,0,0],[1,1,1]]
print(match(people0))
print(best_match(match(people0)))
```

gives an output

```
[[[0, 1, 1], [1, 0, 0], 90.0],
 [[0, 1, 1], [1, 1, 1], 35.26],
 [[1, 0, 0], [1, 1, 1], 54.74]]
[[0, 1, 1], [1, 1, 1], 35.26]
```

We're displaying the result of match so you can see the structure. Each unique pair has an angle. For example:

$$\begin{aligned} inner\_prod([1,0,0],[1,1,1]) &= 1(1)+0(1)+0(1)=1 \quad (7)\\ mag([1,0,0]) &= \sqrt{inner\_prod([1,0,0],[1,0,0])}=\sqrt{1}=1 \quad (8)\\ mag([1,1,1]) &= \sqrt{inner\_prod([1,1,1],[1,1,1])}=\sqrt{3} \quad (9)\\ \theta &= (\frac{180}{\pi})\text{math.acos}(\frac{1}{1\sqrt{3}}) \approx 54.74 \text{ degrees} \quad (10) \end{aligned}$$

Deliverables for Problem 2

- Round the output of angle function to 2 decimal digits.
- Complete all the functions for this problem.

Problem 3: Intersecting Lines

Given two intersecting lines $y = m_0x + b_0, y = m_1x + b_1$ at a single point (x', y') means that $y' = m_0x' + b_0$ and $y' = m_1x' + b_1$. The function $intersect$ takes two lines described by the slope and intercept, and returns their point of intersection.

For example, $y = 2x + 3$ and $y = -\frac{1}{2}x + 2$ we have

$$\begin{aligned} \ell_0 &= [2,3] \quad (11)\\ \ell_1 &= [-\frac{1}{2},2] \quad (12)\\ intersect(\ell_0,\ell_1) &= [-0.4,2.2] \quad (13)\\ intersect([1,4],[-1/2,1/2]) &= [-2.33,1.67] \quad (14) \end{aligned}$$

Deliverables for Problem 3

- Complete the function.
- Remember to round your result to 2 decimal places.

Problem 4: Probablity Mass Function

In this problem you'll build a probability mass function (pmf) from data. A pmf is a mapping p from a set of values $x_0, x_1, \ldots, x_n$ to the interval $[0,1]$ such that:

$$1 = p(x_0) + p(x_1) + \cdots + p(x_n) \quad (15)$$

Using members of, say, a list, we can find the frequency of occurence of each member, and treat the entirety as a pmf. The frequency is the relative fraction of the numbers. For example, [1,0,1,1,0] has a total count of 5, count here is the length of the list! The relative frequency of 0 is 2/5, since 0 occured twice in the entire list. The relative frequency of 1 is 3/5 (occured thrice in the entire list). Frequency is simply $\frac{Occurence}{count}$. We can see that 2/5 + 3/5 = 1 i.e., the pmf is 1 as shown in equation-15. As an additional note, you can also think about relative frequency as probability of occurence of a member, for example, the probability of 0 is $\frac{2}{5}$ this makes sense becauase essentially that's how we calculate probability.

Your function should return a dictionary whose keys are members of the list and corresponding values are the relative frequencies for the members. For example,

```python

def make_prob(xlst):
    pass

data = [[1,1,0,0],[1,2,3,1,1,2,1]]

for d in data:
    print(f"{d} {make_prob(d)}")
```

produces

```
[1, 1, 0, 0] {1: 0.5, 0: 0.5}
[1, 2, 3, 1, 1, 2, 1] {1: 0.57, 2: 0.29, 3: 0.14}
```

Deliverables for Problem 4

- Complete the function.
- Round the probability to 2 decimal places.

Problem 5: Entropy

Entropy (in signal processing) is a single measure indicating how uniform a pmf is. For a pmf $P = \{p_0, p_1, \ldots, p_n\}$, entropy is calculated:

$$H(P) \quad = \quad -\sum_{i=0}^{n} p_i \log_2(p_i) \tag{16}$$

Base two is used because it's easier to understand. You will write an entropy function that takes a list of objects and computes the entropy. Here is a sample run:

```python
data = [[1,1,0,0],[1,2,3,1,1,2,1]]

def entropy(lst):
    pass

for d in data:
    print(f"{entropy(d)}")
```

gives output:

```
1.0
1.38
```

If you have done Problem 4, you already have a function that calculates the probabilities-you can use that function to complete this problem.

Deliverables for Problem 5

- Complete the function.
- Round entropy to 2 decimal places.

Problem 6: Toward Statistical Analysis

In this problem, you'll write fundamental statistical functions. We assume a list of numbers $lst = [x_0, x_1, \ldots, x_n]$.

$$\begin{aligned}
mean(lst) &= (x_0 + x_1 + \cdots + x_n)/\text{len}(lst) \tag{17}\\
\mu &= mean(lst) \tag{18}\\
var(lst) &= \frac{1}{\text{len}(lst)}((x_0 - \mu)^2 + (x_1 - \mu)^2 + \cdots + (x_n - \mu)^2) \tag{19}\\
std(lst) &= \sqrt{var(lst)} \tag{20}
\end{aligned}$$

For example, $lst = [1, 3, 3, 2, 9, 10]$, rounding to two places

$$\begin{aligned}
mean(lst) &= 4.67 \tag{21}\\
var(lst) &= 12.22 \tag{22}\\
std(lst) &= 3.5 \tag{23}
\end{aligned}$$

The last function mean_centered takes a list of numbers $lst = [x_0, x_1, \ldots, x_n]$ and returns a new list $[x_0 - \mu, x_1 - \mu, \ldots, x_n - \mu]$. An interesting feature of the mean-centered list is that if you try to calculate its mean, it's zero:

$$\begin{aligned}
\mu &= (x_0 + x_1 + \cdots + x_n)/n \tag{24}\\
lst &= [x_0 - \mu, x_1 - \mu, \ldots x_n - \mu] \tag{25}\\
mean(lst) &= ((x_0 - \mu) + \cdots (x_n - \mu))/n \tag{26}\\
&= ((x_0 + \ldots + x_n) + n\mu)/n \tag{27}\\
&= \mu - \mu = 0 \tag{28}
\end{aligned}$$

Using the same list we have

$$mean(mean\_centered(lst)) \quad = \quad -0.0 = 0 \tag{29}$$

Deliverables for Problem 6

- For “mean“, “variance“ and “std“ functions-round the output to 2 decimal digits.
- Complete the functions.

Problem 7: Greatest difference and least difference

Given a list of numbers and an option 0 or 1, find the smallest absolute difference between any two numbers or absolute largest respectively. For example, the function blist(lst) returns

```python
data = [[[6,2,1,100],1],[[6,2,1,100],0],[[0,0,10,10],1],
        [[1,2,1,-4],0],[[1,2,1,-4],1],[[0,0,10,10],0]]

for d in data:
    print(f"{d} {blist(d)}")
```

gives output:

```
[[6, 2, 1, 100], 1] 99
[[6, 2, 1, 100], 0] 1
[[0, 0, 10, 10], 1] 10
[[1, 2, 1, -4], 0] 0
[[1, 2, 1, -4], 1] 6
[[0, 0, 10, 10], 0] 0
```

If 1 is given-we have to find the largest absolute difference. We have to compare:

$$\begin{aligned} & |6-2|, |6-1|, |6-100|, |2-1|, |2-100|, |1-100| && (30) \\ = \quad & 4, 5, 94, 1, 98, 99 && (31) \end{aligned}$$

The largest is 99. If 0 is given, we have to return the absolute smallest which would be 1 for the example above.

Deliverables for Problem 7

- Complete the functions.

Problem 8: Trucking Company

Acme Trucking Company has data logs of their drives. It's a list of lists where each member is [name, [[speed,time],[speed,time],...]. Some of the truckers have recorded the time in hours, hours and minutes or only minutes. The [speed, time] is a list with mile per hour (speed) as the first member, and a list of a single number number [x] in hours or a pair [x, y] where x is hours and y is minutes. Since the speed is in miles per hour, you'll have to convert the [x, y] to hours only. For example, for [75, [0.2, 48]] then the distance is:

$$distance = 75\,\frac{\text{mile}}{\text{hr}} \times (.2\,\text{hr} + 48\,\text{min}(1/60\,\frac{\text{hr}}{\text{min}})) \tag{32}$$

$$= 75\,\frac{\text{mile}}{\text{hr}} \times (.2 + .8)\,\text{hr} = 75\,\text{mile} \tag{33}$$

All the speeds are in miles per hour. You are asked to write a Python program that will return the trucker that has driven the furthest along with the total miles driven. For example, if the data logs are:

```
truck_d = [['X', [ 55,[0,60]],[15,[2.5]],[75,[0.2, 48]]],
           ['Y', [55,[0,60]]],
           ['Z', [10,[1]],[10,[1]]],
           ['A', [30,[2]]]]
```

the you'll return

```
['X', 167.5]
```

since $55 + 37.5 + 75 = 167.5$ mile.

Deliverables for Problem 8

- Complete the function.
- You're encouraged to implement smaller functions to help.

C200 student pairs

rghafoor@iu.edu, svuppunu@iu.edu, chrinayl@iu.edu
adamshm@iu.edu, aketcha@iu.edu
dadeyeye@iu.edu, msmelley@iu.edu
aaher@iu.edu, abramjee@iu.edu
omakinfi@iu.edu, rvu@iu.edu
shakolia@iu.edu, bj13@iu.edu
abalbert@iu.edu, mccoyry@iu.edu
megalbin@iu.edu, chataway@iu.edu
waasali@iu.edu, howelcar@iu.edu
ahmalman@iu.edu, thamed@iu.edu
anders14@iu.edu, ianbaker@iu.edu
nsantoin@iu.edu, tdearbor@iu.edu
jaybaity@iu.edu, lvanjelg@iu.edu
nbalacha@iu.edu, nmccarry@iu.edu
aiballou@iu.edu, petcarmi@iu.edu
jabarbu@iu.edu, patevig@iu.edu
cmbeaven@iu.edu, zhangjoe@iu.edu
olibelch@iu.edu, rkabra@iu.edu
jadbenav@iu.edu, nolknies@iu.edu
sberck@iu.edu, maldowde@iu.edu
evberg@iu.edu, wjduncan@iu.edu
sbi@iu.edu, wilsdane@iu.edu
obianco@iu.edu, drewkimb@iu.edu
jbilbre@iu.edu, dboecler@iu.edu
aibitner@iu.edu, phiprice@iu.edu
jetblack@iu.edu, rilmhart@iu.edu
ablashe@iu.edu, kyeosen@iu.edu
pblasio@iu.edu, mvanworm@iu.edu
obowcott@iu.edu, jarabino@iu.edu
sabowe@iu.edu, shnaka@iu.edu
abrandtb@iu.edu, lcosens@iu.edu
owebrook@iu.edu, scclotea@iu.edu
browpr@iu.edu, jconcial@iu.edu
ttbrowne@iu.edu, jttrinkl@iu.edu
stebutz@iu.edu, bdzhou@iu.edu
ecaggian@iu.edu, jschlaef@iu.edu
jlcarrie@iu.edu, tchigudu@iu.edu
carcast@iu.edu, ceifling@iu.edu
chenjunx@iu.edu, notsolo@iu.edu
adhichin@iu.edu, nichhoff@iu.edu

emiclar@iu.edu, parksdr@iu.edu
lizcoro@iu.edu, rpogany@iu.edu
giancost@iu.edu, daknecht@iu.edu
bcrick@iu.edu, conthom@iu.edu
cwcrotty@iu.edu, ryou@iu.edu
mattcrum@iu.edu, rjjorge@iu.edu
pcullum@iu.edu, aalesh@iu.edu
edeporte@iu.edu, limingy@iu.edu
jacdick@iu.edu, envu@iu.edu
dixonjh@iu.edu, nmwaltz@iu.edu
adolata@iu.edu, aareads@iu.edu
tdonoho@iu.edu, lyncsara@iu.edu
ecdruley@iu.edu, oschwar@iu.edu
majdunc@iu.edu, tavalla@iu.edu
ebya@iu.edu, agvore@iu.edu
seckardt@iu.edu, gbharlan@iu.edu
gavedwar@iu.edu, cgoeglei@iu.edu
augeike@iu.edu, jrosebr@iu.edu
jpenrigh@iu.edu, eg8@iu.edu
jjepps@iu.edu, davthorn@iu.edu
jfahrnow@iu.edu, zaschaff@iu.edu
nfarhat@iu.edu, awestin@iu.edu
chafiel@iu.edu, jwa14@iu.edu
riflemin@iu.edu, dl61@iu.edu
foxjust@iu.edu, comojica@iu.edu
tfriese@iu.edu, mkirolos@iu.edu
jofuen@iu.edu, darisch@iu.edu
magacek@iu.edu, agesas@iu.edu
landgarr@iu.edu, mihough@iu.edu
dgodby@iu.edu, cmcclar@iu.edu
goel@iu.edu, mahgree@iu.edu
gonzavim@iu.edu, esisay@iu.edu
wgranju@iu.edu, kusgupta@iu.edu
krgrohe@iu.edu, albperez@iu.edu
jgruys@iu.edu, ssetti@iu.edu
kegupta@iu.edu, bhmung@iu.edu
dgusich@iu.edu, vkethine@iu.edu
rhaghver@iu.edu, sarmayo@iu.edu
jhaile@iu.edu, mitcchar@iu.edu
halejd@iu.edu, ashmvaug@iu.edu
chaleas@iu.edu, bizzo@iu.edu

hallzj@iu.edu, egshim@iu.edu
alehami@iu.edu, jawashi@iu.edu
hardenja@iu.edu, jensprin@iu.edu
harpebr@iu.edu, aubhighb@iu.edu
brohelms@iu.edu, anuttle@iu.edu
hernaga@iu.edu, mysoladi@iu.edu
lohernan@iu.edu, etprince@iu.edu
mijherr@iu.edu, oakagzi@iu.edu
jonhick@iu.edu, matzhang@iu.edu
jchobbs@iu.edu, jnjeri@iu.edu
phoen@iu.edu, sjvaleo@iu.edu
tahoss@iu.edu, stefschr@iu.edu
gmhowell@iu.edu, sndashi@iu.edu
thuhtoo@iu.edu, lanounch@iu.edu
leghuang@iu.edu, mahajenk@iu.edu
ellhuds@iu.edu, snsung@iu.edu
milahusk@iu.edu, rorshiel@iu.edu
jackssar@iu.edu, wemurray@iu.edu
abdjimoh@iu.edu, skalivas@iu.edu
brkapla@iu.edu, skmcmaho@iu.edu
rdkempf@iu.edu, patelkus@iu.edu
nekern@iu.edu, benrmitc@iu.edu
keysa@iu.edu, cadwinin@iu.edu
rkkhouri@iu.edu, zwoolley@iu.edu
arkirt@iu.edu, bzurbuch@iu.edu
abvekoes@iu.edu, tanaud@iu.edu
nakoon@iu.edu, dmetodie@iu.edu
arykota@iu.edu, marebey@iu.edu
fdkussow@iu.edu, amurli@iu.edu
nkyryk@iu.edu, luilmill@iu.edu
lewiserj@iu.edu, martiro@iu.edu
eliantu@iu.edu, kyrhod@iu.edu
jolindse@iu.edu, istorine@iu.edu
jonllam@iu.edu, perkcaan@iu.edu
isclubia@iu.edu, antreye@iu.edu
joluca@iu.edu, ereno@iu.edu
vimadhav@iu.edu, ntatro@iu.edu
wodmaxim@iu.edu, jthach@iu.edu
namcbrid@iu.edu, mr86@iu.edu
emcgough@iu.edu, morrmaja@iu.edu
kmcinto@iu.edu, cjvanpop@iu.edu

gmeinerd@iu.edu, alenmurp@iu.edu
mooralec@iu.edu, eawidema@iu.edu
kpmorse@iu.edu, petersgm@iu.edu
jnmroch@iu.edu, ptstorm@iu.edu
maxmuens@iu.edu, pyahne@iu.edu
jamundy@iu.edu, jpascov@iu.edu
joelna@iu.edu, sunreza@iu.edu
davingo@iu.edu, vsivabad@iu.edu
laynicho@iu.edu, blakruss@iu.edu
kninnema@iu.edu, cmw26@iu.edu
gokeefe@iu.edu, stusinha@iu.edu
jtohland@iu.edu, sampopek@iu.edu
aokhiria@iu.edu, sehpark@iu.edu
coolds@iu.edu, bwinckle@iu.edu
arnpate@iu.edu, msisodiy@iu.edu
patel88@iu.edu, sharpky@iu.edu
tcpatel@iu.edu, as145@iu.edu
mmpettig@iu.edu, samstuar@iu.edu
spletz@iu.edu, rtrujill@iu.edu
csradtke@iu.edu, askrilof@iu.edu
rraguram@iu.edu, sousingh@iu.edu
skrasher@iu.edu, jurzheng@iu.edu
kreddiva@iu.edu, kereidy@iu.edu
uzrivera@iu.edu, azaporo@iu.edu
mwroark@iu.edu, nysach@iu.edu
zekerobe@iu.edu, shawwan@iu.edu
rogerju@iu.edu, branwade@iu.edu
jbromers@iu.edu, avincelj@iu.edu
elyryba@iu.edu, dazamora@iu.edu
jscrogha@iu.edu, samsteim@iu.edu
burshell@iu.edu, nrs5@iu.edu
samsieg@iu.edu, btao@iu.edu
csmalarz@iu.edu, kt10@iu.edu
owasmith@iu.edu, owinston@iu.edu
nasodols@iu.edu, ianwhit@iu.edu
johsong@iu.edu, ayuraiti@iu.edu
rsstarli@iu.edu, jeffsung@iu.edu
evmtaylo@iu.edu, mew17@iu.edu
derthach@iu.edu, ertrice@iu.edu
ttsegai@iu.edu, xujack@iu.edu
kviele@iu.edu, jomayode@iu.edu

eweidne@iu.edu, joeywill@iu.edu
lufayshi@iu.edu, zhaofan@iu.edu

H200 (Honors) student pairs

ethcarmo@iu.edu, snresch@iu.edu
ligonza@iu.edu, alindval@iu.edu
nagopi@iu.edu, swa5@iu.edu
zfhassan@iu.edu, joshprat@iu.edu
tkefalov@iu.edu, nzaerhei@iu.edu
venguyen@iu.edu, bbcolon@iu.edu
marafoth@iu.edu, jarenner@iu.edu
arangwan@iu.edu, huntang@iu.edu
avreddy@iu.edu, aktumm@iu.edu
lorivera@iu.edu, sturaga@iu.edu