Sammy Pardes
IST 664
Homework 2
2/19/21

Introduction

My strategy for this assignment was to first generate a list of regular expressions to correctly match the Stanford faculty and staff email addresses. I went alphabetically down the list of false negatives, looking at each text file and taking note of the interesting ways each email address was written or obfuscated. At first, I was careful to only include regular expressions that added to the list of true positives, and did not contribute to the false positives list.

After going through the entire list of contacts for email addresses, I followed a near identical process for phone numbers. Once I had a list of all the true positives I could match, I went through the remaining contacts that could not be matched with the "two-group .edu" rule. I created regular expressions that yielded false positives for those emails that did not fit the mold.

Part I

My first step was to run the Contact Finder program with no values in the epattern or ppattern lists. This displayed 0 true positives, 0 false positives, and 117 false negatives.

Next, I added the three example patterns given in the Homework 2 Notes file. The first email pattern, **([A-Za-z]+)@([A-Za-z]+)\.edu** matched a group of at least one lowercase or uppercase alphabetic character, followed by an "@" symbol, another group of one or more alphabetic characters, and ending with ".edu". The second pattern given, **([A-Za-z.]+)@([A-Za-z.]+)\.edu**, was identical to the first one, but allowed for period characters in the first and second groups. The last expression provided, **([A-Za-z.]+)\s@\s([A-Za-z.]+)\.edu**, was very similar to the first two, but allowed for spaces around the "at sign", denoted by two "\s" escape characters. The result of these 3 expressions was 22 true positives, 1 false positive, and 95 false negatives. I decided to comment out the first regex because it gave one false positive, leaving tp=22, fp=0, fn=95.

Moving on to my own patterns, I created 7 expressions to match the true positive email addresses. Below, I have listed each regular expression, a short description of what the expression accomplishes, an example of its match, its obfuscation, and the resulting summary after running the ContactFinder program.

1. The first pattern of my own I added was **([A-Za-z.]+)\sat\s([A-Za-z.]+)\.EDU**. This matched with email addresses containing one or more upper/lowercase letters or a period, followed by the word "at" with spaces around it, and then the phrase ".EDU". This matched uma@cs.stanford.edu, which was written as "uma at cs.Stanford.EDU" in the cheriton file. After including this expression, the summary result was tp=23, fp=0, fn=94.

2. The expression **([A-Za-z.]+)\s+@\s+([A-Za-z.]+)\.edu** matched text that contained one or more uppercase/lowercase letters or periods, followed by an @ sign with one or more spaces on either side, another group of upper/lowercase letters or periods, and then ".edu". This matched the address dabo@cs.stanford.edu, which was written as "dabo @ cs.stanford.edu". 24 true positives, 0 false positives, and 93 false negatives remained.
3. **([A-Za-z]+)<del>@([A-Za-z.]+)\.edu** was added next. This accommodated at least one upper or lowercase letter, followed by the exact phrase "<del>@", and then more letter/period characters, ending in ".edu". latombe@cs.stanford.edu correctly matched this expression, which was originally denoted as latombe<del>@cs.stanford.edu. Now, there were 27 true positives, 0 false positives, and 90 false negatives.
4. **([A-Za-z]+)@([A-Za-z.]+)\.edu** was nearly identical to the previous regex, but utilized the unicode value for "@", or @. Our true positive group now included ada@graphics.stanford.edu, which was written as "ada@graphics.stanford.edu" in the levoy file, leaving tp=29, fp=0, fn=88.
5. Next, I took care of email addresses that used " <at symbol> " inplace of an actual @ sign, with this expression: **([A-Za-z]+)\s<at symbol>\s([A-Za-z.]+)\.edu**. dbarros@cs.stanford.edu matched this pattern since it was written as "dbarros <at symbol> cs.stanford.edu". The summary was then tp=31, fp=0, fn=86.
6. Chugging along, I added the regex pattern **([A-Za-z]+)\sAT\s([A-Za-z]+)\sDOT\sedu**. This matched emails where " AT " was after group one, and " DOT edu" was after group two. This left 32 true positives, 0 false positives, and 85 false negatives.
7. To produce 33 true positives, 0 false positives, and 84 false negatives, I added the RegEx **([A-Za-z]+)\sWHERE\s([A-Za-z]+)\sDOM\sedu**. This accounted for emails that used " WHERE " instead of "@" and " DOM edu" instead of ".edu", like "engler WHERE stanford DOM edu" for engler@stanford.edu.

After getting as many true positive email addresses as I felt was possible, I migrated my efforts to the Stanford faculty phone numbers. I used 6 expressions to correctly match all of the numbers.

1. The first phone number regex I implemented, **(\d{3})-(\d{3})-(\d{4})**, was provided in the Homework 2 Notes file. This matched a group of 3 digit characters, followed by a dash, another group of 3 digit characters, a second dash, and finally, a group of exactly 4 digit characters. This took care of 650-723-3642 from the eroberts file, written in the same format. After running this, the summary was tp=52, fp=0, fn=65.
2. More phone numbers became true positives with the regex **\((\d{3})\)(\d{3})-(\d{4})**. This was just like the first pattern, but demanded parentheses around the first group of digits and no dash between groups 1 and 2. From the ashishg file, 650-723-1614 matched which was marked as (650)814-1478 in the HTML. Now there were 60 true positives and 57 false negatives, still with 0 false positives.
3. Getting closer, I added **\((\d{3})\)\s(\d{3})-(\d{4})**. This pattern was similar to the one above, but added a space between groups 1 and 2. The 650-723-4539 number from bgirod was now a true positive. It was written in the file as "(650) 724-6354". The summary became tp=99, fp=0, fn=18.

4. The regex **\+1\s(\d{3})\s(\d{3})\s(\d{4})** made way for phone numbers with a preceding "+1" and an escaped space character between each group of digits. Jurafsky's phone number 650-723-5666, originally +1 650 723 5666, was now a true positive. 101 true positives, 0 false positives, and 16 false negatives were left.
5. **\[(\d{3})\]\s(\d{3})-(\d{4})** took in phone numbers with square brackets around the first group of digits, a space between groups 1 and 2, and a dash between groups 2 and 3. 650-725-2472 from the nass file "[650] 723-5499" now matched. The summary was tp=103, fp=0, fn=14.
6. To finish matching phone numbers, I added **\+1\s(\d{3})\s(\d{3})-(\d{4})**. Phone numbers with a "+1 " in the beginning, a space between digit groups 1 and 2, and a dash between groups 2 and 3 were counted. Now, all the phone numbers were true positives including 650-723-3432 from the shoham file, originally transcribed as +1 650 723-3432. The summary after running all the phone number patterns was tp=105, fp=0, fn=12.

Part II

Section A

After creating expressions to match the 105 true positive email addresses and phone numbers, I was left with 12 false negatives. I had difficulty matching these email addresses using expression with two groups of parentheses and ending in ".edu". I was able to write additional patterns to convert some of the obfuscated email addresses into false positives with some pattern rule-breaking.

1. The dlwh file contained an email address with dashes around each character: d-l-w-h-@-s-t-a-n-f-o-r-d-.-e-d-u. **([A-Za-z\-]+)@([A-Za-z\-]+)\.-e-d-u** matched this address by including an escaped dash character in the first and second groups and using ".-e-d-u" instead of ".edu". dlwh@stanford.edu was now a false positive, leaving tp=105, fp=1, fn=12. It did not meet the pattern criteria because it did not end in "\.edu".
2. I was able to falsely match emails such as hager@cs.jhu.edu by using 3 groups in the RegEx pattern **([A-Za-z\s]+)at([A-Za-z\s]+)dot([A-Za-z\s]+)edu**. These addresses needed 3 groups because the symbols were written as full words with spaces around them, like "hager at cs dot jhu dot edu". An expression with only 2 groups including space characters within groups 1 and 2 overmatched the email addresses and gave a handful of extra false positives. The expression with 3 groups yielded 105 true positives, 4 false positives, and 12 false negatives.
3. Next, I addressed jks@robotics.stanford.edu which was coded as "jks at robotics;stanford;edu". I added the expression **([A-Za-z]+)\sat\s([A-Za-z\;]+)edu** to the epatterns list. This broke the RegEx pattern rule because it did not end in "\.edu". Instead, I included the ";" character in the second group and matched " at " between the two groups. The summary became tp=105, fp=5, fn=12.
4. The email address ouster@cs.stanford.edu included the unicode value for a left quotation mark, "ouster (followed by “@cs.stanford.edu”)". This is not valid email address notation, but I could get a false positive with

([A-Za-z\(\s]+)“@([A-Za-z\.]+)\.edu. The summary after this expression was tp=105, fp=6, fn=12.

5. The jurafsky@stanford.edu file had the stanford.edu domain before the "someone" portion of the email address: "obfuscate('stanford.edu','jurafsky');". I tackled this by breaking the pattern rule and putting the edu expression with additional symbols in the middle of the two groups: **([A-Za-z\(\']+)\.edu\'\,\'([A-Za-z\']+)**. After this, the summary was tp=105, fp=7, fn=12.
6. To turn support@gradiance.com from the ullman file into a false positive, I added to the code **([A-Za-z\s]+)dt\scom**. It was coded as "support at gradiance dt com". This was a false positive because it had a .com domain and it only needed one group. There were then 105 true positives, 8 false positives, and 12 false negatives.
7. pal@cs.stanford.edu, denoted as "pal at cs stanford edu", was another email address needing 3 groups for matching. I used **([A-Za-z]+)\sat\s([A-Za-z]+)\s([A-Za-z]+)\sedu** to get this in the false positive group. This encompassed 3 groups of at least one alphabetic character with " at " between the first two groups and a space between the 2nd and 3rd groups, ending in " edu". The false positive count was 9, still with 105 true positives and 12 false negatives.
8. Finally, I matched another email in the ouster file (teresa.lynn@stanford.edu) with **([A-Za-z.\s\(\"]+)@([A-Za-z]+)\.edu**. Now "teresa.lynn (followed by "@stanford.edu")" was a match. This added periods, spaces, left parentheses, and quotation marks to the first group of characters. An at sign had to follow group 1, and after that, a group of one or more A-Z or a-z characters, with the expression ending in ".edu". To finish my list of expressions, there were 105 true positives, 10 false positives, and 10 false negatives.

I was able to create expressions to match lam@cs.stanford.edu (lam at cs.stanford.edu) and vladlen@stanford.edu (vladlen at <!-- die!--> stanford <!-- spam pigs!--> dot <!-- die!--> edu) using RegExPal, but had difficulty adding the patterns into the ContactFinder program without overmatching and creating even more false positives. An expression that matched the vladlen email in RegExPal was **([A-Za-z]+)\sat\s([A-Za-z\<\!\-\>\s]+)\sedu**. This expression worked with the lam email in RegExPal: **([A-Za-z]+)\sat\s([A-Za-z.]+)\.edu**.

Section B

I think a really successful way to obfuscate email addresses and phone numbers is to break up the components into individual variables and keep them separate. Another way to create hard to match addresses would be to have the components out of order. Both of these methods make it challenging to find the right parts with RegEx, without overmatching. One example of the first tacticc would be something like: User = slpardes; site="syr.edu"; email = user + @ + site.

As I've discovered throughout this assignment, another way to obscure email addresses and phone numbers is to replace standard characters like "@", "-", and "." with words like "AT" and "DOT", making it difficult to differentiate the standard characters from the unique user data.

Conclusion

It's possible I could have combined some of my regular expressions to shorten the epatterns and ppatterns lists. For example, I could have made certain symbols, like spaces (\s) and parentheses (\(, \)), optional within phone numbers. I could have also taken care of more addresses with fewer expressions by making the spaces around the "@" in email addresses optional with the "?" notation.

However, I wanted to take a careful approach to not add to the false positives list and keep track of my work. Were I to do this assignment again, I would try more combinations of patterns to see if I could accomplish the same result, but with fewer rules.

ContactFinder Final Output

```
Assuming ContactFinder.py called in directory with data folder
True Positives (105):
{('ashishg', 'e', 'ashishg@stanford.edu'),
 ('ashishg', 'e', 'rozm@stanford.edu'),
 ('ashishg', 'p', '650-723-1614'),
 ('ashishg', 'p', '650-723-4173'),
 ('ashishg', 'p', '650-814-1478'),
 ('balaji', 'e', 'balaji@stanford.edu'),
 ('bgirod', 'p', '650-723-4539'),
 ('bgirod', 'p', '650-724-3648'),
 ('bgirod', 'p', '650-724-6354'),
 ('cheriton', 'e', 'cheriton@cs.stanford.edu'),
 ('cheriton', 'e', 'uma@cs.stanford.edu'),
 ('cheriton', 'p', '650-723-1131'),
 ('cheriton', 'p', '650-725-3726'),
 ('dabo', 'e', 'dabo@cs.stanford.edu'),
 ('dabo', 'p', '650-725-3897'),
 ('dabo', 'p', '650-725-4671'),
 ('engler', 'e', 'engler@lcs.mit.edu'),
 ('engler', 'e', 'engler@stanford.edu'),
 ('eroberts', 'e', 'eroberts@cs.stanford.edu'),
 ('eroberts', 'p', '650-723-3642'),
 ('eroberts', 'p', '650-723-6092'),
 ('fedkiw', 'e', 'fedkiw@cs.stanford.edu'),
 ('hager', 'p', '410-516-5521'),
 ('hager', 'p', '410-516-5553'),
 ('hager', 'p', '410-516-8000'),
 ('hanrahan', 'e', 'hanrahan@cs.stanford.edu'),
 ('hanrahan', 'p', '650-723-0033'),
 ('hanrahan', 'p', '650-723-8530'),
 ('horowitz', 'p', '650-725-3707'),
 ('horowitz', 'p', '650-725-6949'),
 ('jurafsky', 'p', '650-723-5666'),
 ('kosecka', 'e', 'kosecka@cs.gmu.edu'),
 ('kosecka', 'p', '703-993-1710'),
 ('kosecka', 'p', '703-993-1876'),
```

```
('kunle', 'e', 'darlene@csl.stanford.edu'),
('kunle', 'e', 'kunle@ogun.stanford.edu'),
('kunle', 'p', '650-723-1430'),
('kunle', 'p', '650-725-3713'),
('kunle', 'p', '650-725-6949'),
('lam', 'p', '650-725-3714'),
('lam', 'p', '650-725-6949'),
('latombe', 'e', 'asandra@cs.stanford.edu'),
('latombe', 'e', 'latombe@cs.stanford.edu'),
('latombe', 'e', 'liliana@cs.stanford.edu'),
('latombe', 'p', '650-721-6625'),
('latombe', 'p', '650-723-0350'),
('latombe', 'p', '650-723-4137'),
('latombe', 'p', '650-725-1449'),
('levoy', 'e', 'ada@graphics.stanford.edu'),
('levoy', 'e', 'melissa@graphics.stanford.edu'),
('levoy', 'p', '650-723-0033'),
('levoy', 'p', '650-724-6865'),
('levoy', 'p', '650-725-3724'),
('levoy', 'p', '650-725-4089'),
('manning', 'e', 'dbarros@cs.stanford.edu'),
('manning', 'e', 'manning@cs.stanford.edu'),
('manning', 'p', '650-723-7683'),
('manning', 'p', '650-725-1449'),
('manning', 'p', '650-725-3358'),
('nass', 'e', 'nass@stanford.edu'),
('nass', 'p', '650-723-5499'),
('nass', 'p', '650-725-2472'),
('nick', 'e', 'nick.parlante@cs.stanford.edu'),
('nick', 'p', '650-725-4727'),
('ok', 'p', '650-723-9753'),
('ok', 'p', '650-725-1449'),
('pal', 'p', '650-725-9046'),
('psyoung', 'e', 'patrick.young@stanford.edu'),
('rajeev', 'p', '650-723-4377'),
('rajeev', 'p', '650-723-6045'),
('rajeev', 'p', '650-725-4671'),
('rinard', 'e', 'rinard@lcs.mit.edu'),
('rinard', 'p', '617-253-1221'),
('rinard', 'p', '617-258-6922'),
('serafim', 'p', '650-723-3334'),
('serafim', 'p', '650-725-1449'),
('shoham', 'e', 'shoham@stanford.edu'),
('shoham', 'p', '650-723-3432'),
('shoham', 'p', '650-725-1449'),
('subh', 'e', 'subh@stanford.edu'),
('subh', 'p', '650-724-1915'),
('subh', 'p', '650-725-3726'),
('subh', 'p', '650-725-6949'),
('thm', 'e', 'pkrokel@stanford.edu'),
('thm', 'p', '650-725-3383'),
('thm', 'p', '650-725-3636'),
```

```
 ('thm', 'p', '650-725-3938'),
 ('tim', 'p', '650-724-9147'),
 ('tim', 'p', '650-725-2340'),
 ('tim', 'p', '650-725-4671'),
 ('ullman', 'e', 'ullman@cs.stanford.edu'),
 ('ullman', 'p', '650-494-8016'),
 ('ullman', 'p', '650-725-2588'),
 ('ullman', 'p', '650-725-4802'),
 ('widom', 'e', 'siroker@cs.stanford.edu'),
 ('widom', 'e', 'widom@cs.stanford.edu'),
 ('widom', 'p', '650-723-0872'),
 ('widom', 'p', '650-723-7690'),
 ('widom', 'p', '650-725-2588'),
 ('zelenski', 'e', 'zelenski@cs.stanford.edu'),
 ('zelenski', 'p', '650-723-6092'),
 ('zelenski', 'p', '650-725-8596'),
 ('zm', 'e', 'manna@cs.stanford.edu'),
 ('zm', 'p', '650-723-4364'),
 ('zm', 'p', '650-725-4671')}
False Positives (10):
('dlwh', 'e', 'd-l-w-h-@-s-t-a-n-f-o-r-d-.edu')
   gold:  ('dlwh', 'e', 'dlwh@stanford.edu')
('serafim', 'e', ' serafim @ cs dot stanford .edu')
   gold:  ('serafim', 'e', 'serafim@cs.stanford.edu')
   gold:  ('serafim', 'p', '650-723-3334')
   gold:  ('serafim', 'p', '650-725-1449')
('ouster', 'e', 'teresa.lynn (followed by "@stanford.edu')
   gold:  ('ouster', 'e', 'teresa.lynn@stanford.edu')
   gold:  ('ouster', 'e', 'ouster@cs.stanford.edu')
('jks', 'e', 'jks@robotics;stanford;.edu')
   gold:  ('jks', 'e', 'jks@robotics.stanford.edu')
('hager', 'e', 'hager @ cs dot jhu .edu')
   gold:  ('hager', 'e', 'hager@cs.jhu.edu')
   gold:  ('hager', 'p', '410-516-5521')
   gold:  ('hager', 'p', '410-516-5553')
   gold:  ('hager', 'p', '410-516-8000')
('subh', 'e', ' uma @ cs dot stanford .edu')
   gold:  ('subh', 'e', 'subh@stanford.edu')
   gold:  ('subh', 'e', 'uma@cs.stanford.edu')
   gold:  ('subh', 'p', '650-724-1915')
   gold:  ('subh', 'p', '650-725-3726')
   gold:  ('subh', 'p', '650-725-6949')
('pal', 'e', 'pal@cs.edu')
   gold:  ('pal', 'e', 'pal@cs.stanford.edu')
   gold:  ('pal', 'p', '650-725-9046')
('ullman', 'e', 'e@m.edu')
   gold:  ('ullman', 'e', 'support@gradiance.com')
   gold:  ('ullman', 'e', 'ullman@cs.stanford.edu')
   gold:  ('ullman', 'p', '650-494-8016')
   gold:  ('ullman', 'p', '650-725-2588')
   gold:  ('ullman', 'p', '650-725-4802')
('jurafsky', 'e', "obfuscate('stanford@jurafsky'.edu")
```

```
   gold:  ('jurafsky', 'e', 'jurafsky@stanford.edu')
   gold:  ('jurafsky', 'p', '650-723-5666')
('ouster', 'e', 'ouster (followed by @cs.stanford.edu')
   gold:  ('ouster', 'e', 'teresa.lynn@stanford.edu')
   gold:  ('ouster', 'e', 'ouster@cs.stanford.edu')
False Negatives (12):
{('dlwh', 'e', 'dlwh@stanford.edu'),
 ('hager', 'e', 'hager@cs.jhu.edu'),
 ('jks', 'e', 'jks@robotics.stanford.edu'),
 ('jurafsky', 'e', 'jurafsky@stanford.edu'),
 ('lam', 'e', 'lam@cs.stanford.edu'),
 ('ouster', 'e', 'ouster@cs.stanford.edu'),
 ('ouster', 'e', 'teresa.lynn@stanford.edu'),
 ('pal', 'e', 'pal@cs.stanford.edu'),
 ('serafim', 'e', 'serafim@cs.stanford.edu'),
 ('subh', 'e', 'uma@cs.stanford.edu'),
 ('ullman', 'e', 'support@gradiance.com'),
 ('vladlen', 'e', 'vladlen@stanford.edu')}
Summary: tp=105, fp=10, fn=12
```

Sources

https://blog.mailtrap.io/email-obfuscation/
https://wordpress.org/plugins/obfuscate-email/