Part 2 - Feature Engineering - Version 1.0

1. Run the code

In the current directory, input the following command line at the terminal.

`python ./run.sh`

It is the same as:

```
python preprocess.py
python cctld.py
python ngram.py
python feature1.py
python feature2.py
```

2. Source Code

2.1 Feature Type

2.1.1 Linguistic Feature

`entropy.py`

- Calculate the mean value and standard value
- Count the number of vowels, digits, repeated letter, consecutive digit and consecutive consonant
- Calculate Shannon Entropy, length and counter of domain

2.1.2 ccTLD Feature

`cctld.py`

- Given the domain name, find the matched ccTLD

  e.g., `196.1.211.6.`, its ccTLD is none, set ccTLD_num=9999

  e.g., `sina.com.cn.`, its ccTLD is `com.cn`, find the corresponding ccTLD_num=579

- Reading file:

  `public_suffix_list.txt` : a file of collected ccTLD names

  ```
  ac
  com.ac
  ...
  (8,744 lines)
  ```

`ds_1m_p.txt` : a file of collected url, is_malicious, host, core_domain, path, cctld

```
myspace.com/ariya   0   myspace.com $myspace$   ariya   com
...
(1,223,675 lines)
```

- Writing file:

  `cctld.txt` : a file of url, is_malicious, cctld, cctld_num

```
oyarena.in  0   in  990
...
(1,223,675 lines)
```

2.1.3 N-Gram Feature

`ngram.py`

- Using unigram, bigram, trigram to calculate its frequency and rank

  e.g., `google.co.uk`

```
# unigram
defaultdict(int, {'g': 2, 'o': 2, 'l': 1, 'e': 1})
# bigram
defaultdict(int, {'$g': 1, 'go': 1, 'oo': 1, 'og': 1, 'gl': 1, 'le':
1, 'e$': 1})
# trigram
defaultdict(int, {'$go': 1, 'goo': 1, 'oog': 1, 'ogl': 1, 'gle': 1,
'le$': 1})
```

- Reading file:

  `ds_genuine_p.txt` : a file of genuine urls

```
myspace.com/everything/leonard-cohen    0   myspace.com $myspace$
everything/leonard-cohen    com
youtube.com/watch?v=sC8hOIjwZYY 0   youtube.com $youtube$   watch?
v=sC8hOIjwZYY com
amc.edu 0   amc.edu $amc$       edu
(444,800 lines)
```

  `cctld.txt` : a file of domain, binary classification and matched ccTLD

```
oyarena.in  0   in  990
...
(1,223,675 lines)
```

- Writing file:

`ngram1.txt` : ngram-type, rank, gram string, frequency

```
1   1   e   429971
1   2   a   379334
1   3   o   332601
...
3   27994   lvf 1
3   27995   lvh 1
3   27996   lvk 1
(29,489 lines)
```

`ngram2.txt` : url,is_malicious,s1,s2,s3,core_domain

```
url domain  is_malicious    unigram bigram  trigram core_domain
myspace.com/everything/leonard-cohen    0   8.71    73.88   139.29
$myspace$
youtube.com/watch?v=sC8hOIjwZYY 0   10.14   78.75   30.00   $youtube$
...
(1,223,676 lines)
```

2.2 Feature Processing

2.2.1 Feature Preprocess

`preprocess.py`

- original: url,is_malicous
- modified: url, is_malicous, host, core_domain, path, cctld
- e.g.

```
Before: "sgademexico.com/tmp/Inc,Dropbox/1/view.php",1
After:   sgademexico.com/tmp/Inc,Dropbox/1/view.php 1
sgademexico.com $sgademexico$   tmp/Inc,Dropbox/1/view.php  com
```

2.2.2 Feature Extractor

`feature1.py`

- Extract features:

  url, is_malicious, cctld_num, entropy, length, norm_entropy, uni_rank, bi_rank, tri_rank, uni_std, bi_std, tri_std, gib_value

- Reading file:

  `gib_model.pki`

  > Reference from https://github.com/rrenaud/Gibberish-Detector

  `ngram1.txt, cctld.txt` has been mentioned above.

- Writing file:

`feature1.txt`

```
url is_malicious    cctld_num   entropy length  norm_entropy    uni_rank
bi_rank tri_rank    uni_std bi_std  tri_std gib_value
www.soverial.fr 1   763 2.303   15.0    0.154   14.62   130.71  894.38
12.47   185.09  1730.47 1.00
ibuycountryhome.realestate  1   6933    2.651   26.0    0.102   9.25
103.80  895.42  8.06    133.41  1477.77 1.00
giraffeadvertising.com.au   1   165 2.678   25.0    0.107   10.70   109.21
886.87  10.57   104.45  988.89  1.00
victorcasino.com/g76ub76/   1   636 2.733   25.0    0.109   8.07    73.67
609.93  8.74    58.93   552.58  1.00
djjmzfcx9o.neliver.com  1   636 2.713   22.0    0.123   16.00   370.29
6266.65 11.95   410.68  9498.48 1.00
www.baisheng.co.nz  1   4940    2.447   18.0    0.136   15.19   135.76
941.25  12.49   192.61  1588.90 1.00
```

2.2.3 Feature Normalization

`feature2.py`

- Normalize the feature values to [0,1], except url, is_malicious
- Reading file:

  `feature1.txt` has been mentioned above
- Writing file

  `feature2.txt`

```

```

2.3 Feature Visualization

`visual.ipynb`

- Visual analysis to compare different features

2.4 Web Tool

2.5 File Table

Code file	Reading file	Writing file
preprocess.py	ds_1m.csv	ds_1m_p.txt
entropy.py	/	/
cctld.py	public_suffix_list.txt ds_1m_p.txt	cctld.txt
ngram.py	ds_genuine_p.txt cctld.txt	ngram1.txt ngram2.txt
feature1.py	gib_model.pki ngram1.txt cctld.txt	feature1.txt
feature2.py	feature1.txt	feature2.txt
visual.ipynb	to do	to do
web tool	to do	to do

3. Special Cases

3.1