AWS-Market Share Analysis

June 10, 2019

1 Collection of Simple Random Sample from Population of Websites

1.1 Simple Random Sample

Below is a program that creates a simple random sample of websites according to Alexa Analytics' Top One Million sites CSV. The function `create_sample` creates a random integer between 0 and 999,999. That number acts as the index for the site we will get from the population CSV (e.g. The site at index 0 is google.com).

Once we have selected a site for our sample, we need to resolve its IP address (note: we will be using only IPv4 addresses as Alexa only offers those in their dataset and all websites should handle IPv4 and IPv6). If we are unable to resolve the address than we will randomly select another site to replace it (IS THAT OKAY?).

When sampling is completed, the sample dataset (domains and IPv4 addresses) is exported to a CSV file.

[2]:
```
from random import randint
import pandas as pd
import socket

# List of top one million sites according to Alexa Analytics/Website Ranking
# https://s3.amazonaws.com/alexa-static/top-1m.csv.zip
top_sites = pd.read_csv('top-1m.csv', header=None)[1]

# n is sample size
n=1000

# Dictionary used for stored sample data
sample = {
    'Website Domain' : [],
    'IPv4 Address' : []
}

def create_sample(n):
    i = 0
    while i < n:
        i += 1
        # Get random number between 0 and 999,999
        random_index = randint(0, len(top_sites) - 1)
```

```
        # If the site has not already been selected, add it to our data set
        if not top_sites[random_index] in sample['Website Domain']:
            try:
                # print("\033[0mGetting IPv4 Address for %s..." %␣
↪top_sites[random_index])
                ipv4 = socket.gethostbyname(top_sites[random_index])
            # If we can't resolve the IP from the host name, replace it with a␣
↪different host name
            except:
                # print("\033[1mFailed. Selecting new site for sample.")
                i -= 1
                continue
            sample['Website Domain'].append(top_sites[random_index])
            sample['IPv4 Address'].append(ipv4)

create_sample(n)

# Save sample to a CSV file
dataset = pd.DataFrame.from_dict(sample)
dataset.to_csv('website_sample.csv')

dataset
```

[2]:

	Website Domain	IPv4 Address
0	ipos.vn	94.237.76.49
1	sectorul4news.ro	89.42.219.210
2	newprint.ca	37.218.253.61
3	destinationhotels.com	23.100.83.213
4	flutetoday.com	97.74.55.1
5	serialcrush.com	62.149.142.158
6	mybmtc.com	202.71.129.225
7	hmi.edu	192.249.121.112
8	flashkit.com	70.42.23.121
9	fucktubes.xxx	104.28.23.71
10	thetodaypost.com	104.31.66.246
11	avpgalaxy.net	162.211.84.48
12	scwtenor.com	74.208.236.209
13	ecom.ly	104.18.47.2
14	redwolfwildernessadventures.com	67.59.136.110
15	wesounds.com	107.6.153.170
16	bloganten.ru	92.53.114.3
17	karinto.in	175.134.120.229
18	myscopeoutreach.org	182.160.163.245
19	landsurveyor.blogfa.com	149.56.201.253
20	finministry.com	104.18.41.100
21	odeontravel.rs	195.252.107.131

22	greeningtheblue.org	104.27.147.128
23	jerusalemperspective.com	104.199.115.212
24	bazi-oksana.ru	5.101.152.32
25	leupold.com	52.88.153.55
26	cloud9.gg	23.227.38.32
27	goldsgym.com	162.209.117.196
28	tagbox.in	52.172.54.225
29	sexmamki.org	151.80.209.25
..	...	...
970	jb.com.br	152.199.54.25
971	vietadsonline.com	171.244.34.197
972	itftkd.ir	185.128.81.85
973	rouxroamer.com	104.31.95.99
974	filejo.co.kr	43.255.255.83
975	peraichi.com	54.230.89.244
976	hw3d.net	192.99.14.211
977	nekoshop.ru	37.140.192.198
978	maxbestwork.com	66.199.189.51
979	hindihelpguru.com	199.250.213.223
980	exertion-fitness.com	23.227.38.32
981	aktualnacenabytu.sk	212.57.38.25
982	geoequipos.cl	192.140.57.10
983	removenotifications.com	192.64.119.93
984	matbit.net	5.61.47.250
985	homebasedonlinevehiclemarketing.com	146.66.96.176
986	xn--v8j5erc590uusnxox.com	183.90.253.8
987	nflsport.icu	198.54.121.189
988	izithakazelo.blog	192.0.78.191
989	irinabiz.ru	138.201.199.38
990	intercity.pl	46.174.180.162
991	bongacams3.com	31.192.123.62
992	twinstrangers.net	52.214.239.109
993	textgeneratorfont.com	162.241.133.121
994	silversaints.com	212.188.174.246
995	evassmat.com	104.28.24.228
996	mpets.mobi	136.243.25.36
997	londongateway.com	65.52.130.1
998	derangler.shop	85.236.56.247
999	tavirekini.lv	94.100.11.185

[1000 rows x 2 columns]

2 Determining Proportion of Websites Running AWS

2.1 Proportion of IPv4 addresses owned by AWS

The program takes a list of all IPv4 addresses owned by AWS and compares them to the list of addresses in our sample. AWS does not give a list of IPv4 address but instead gives a subnet of their addresses, this means they've purchased addresses in bulk so they're grouped together. In order to properly compare an IPv4 address to a subnet, python offers a library called `ipaddress` that breaks up subnets and ip addresses into a data format that can easily be compared.

If an address appears in Amazon's IPv4 range (their owned addresses) than the domain associated with the IP address is appended to a list. The list of websites is then exported as a CSV file.

[3]:
```
import json, requests, ipaddress

# List of IP Ranges (IPv4 and IPv6) owned by Amazon and used for AWS
# https://ip-ranges.amazonaws.com/ip-ranges.json
aws_ip_ranges = json.loads(requests.get('https://ip-ranges.amazonaws.com/
 ↪ip-ranges.json').text)

# Determine if given IP address (ip_input) shows uo in AWS IPv4 Range
def check_aws(ip_input):
    # Compare given IP to all AWS IP addresses within AWS IPv4 Subnetwork
    for i in range(len(aws_ip_ranges['prefixes'])):
        # Parse IPv4 address for comparison
        site_ip = ipaddress.ip_address(ip_input)

        # Parse AWS IPv4 Subnet
        aws_subnet = ipaddress.
 ↪ip_network(aws_ip_ranges['prefixes'][i]['ip_prefix'])

        # If IP is within the AWS IPv4 Range, the website is run on AWS
        if site_ip in aws_subnet:
            return True
    # If the website is not within the range, the
    # website operates independnetly of AWS
    return False

# List of websites using AWS
websites_using_aws = []

def get_aws_domains():

    # Check every IP within our sample against AWS IPv4 Range
    for i in range(len(dataset)):
        if check_aws(dataset['IPv4 Address'][i]):
            websites_using_aws.append(dataset['Website Domain'][i])
```

```
get_aws_domains()

# Save dataset of AWS websites to a CSV file
aws_df = pd.DataFrame({'AWS Websites':websites_using_aws})
aws_df.to_csv('websites_using_aws.csv')

aws_df
```

[3]:

	AWS Websites
0	leupold.com
1	wanasatime.com
2	simplesdental.com
3	monetixwallet.com
4	10tv.in
5	rosedalecenter.com
6	margstacobistro.com
7	shanghainavi.com
8	keaweather.net
9	lion.co.nz
10	moomii.jp
11	figleafapp.com
12	maharajamultiplex.in
13	conchovalleyhomepage.com
14	honkmedia.net
15	willowtreeapps.com
16	playvod.ma
17	tigosports.gt
18	araelium.com
19	boostnote.io
20	echofoodshelf.org
21	obonsai.com.br
22	atcost.in
23	profsnhcadmission.in
24	manalonline.com
25	nj211.org
26	conta.no
27	cldmail.co.uk
28	obo.se
29	soft32.es
..	...
31	juegosmesa.cl
32	localone.app
33	grjapan.jp
34	peachysnaps.com
35	knottyladyyarns.com
36	amormaturo.com
37	hltmag.co.uk