Using DigitalGlobe's Open Data Against Ebola

In this notebook we'll be looking at how to use DigitalGlobe's open data to find areas in the Democratic Republic of the Congo that are at an elevated risk of an ebola outbreak.

Getting Started with DigitalGlobe's Open Data

The data for this notebook can be downloaded from the following link: https://www.digitalglobe.com/opendata/ebola-response-democratic-republic-of-the-congo/vector-data

If you haven't already, you will have to install sklearn to run this notebook.

In [1]:
```
!pip install sklearn
```

1]:
```
Requirement already satisfied: sklearn in /anaconda/envs/gbdx_py3/lib/python3.6/site-packages (0.0)
Requirement already satisfied: scikit-learn in /anaconda/envs/gbdx_py3/lib/python3.6/site-packages (from skl
[31mvcrpy 1.13.0 requires yarl, which is not installed.[0m
[31mgbdxtools 0.15.7 requires mock, which is not installed.[0m
```

[33mYou are using pip version 10.0.1, however version 18.0 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.[0m

In [2]:
```
import gbdxtools
gbdx = gbdxtools.Interface()
import json
from sklearn.cluster import DBSCAN
#import geopandas as gpd
import shapely
from matplotlib import pyplot as plt
```

We can use the open data by converting it to a GeoJSON file. I did this (on a mac) by installing GDAL, cd-ing into the equateur-dynamic folder in Terminal, and running the command line `ogr2ogr -f GeoJSON equateur-dynamic.geojson equateur-dynamic.shp` (see this link for other ways to do this: https://www.igismap.com/different-ways-to-convert-shapefile-to-geojson-online-or-offline-or-api/).

To use the data in your notebook, you have to upload it to the Jupyter file tree. You can do this by clicking on the Jupyter logo in the upper left corner, then clicking the upload button in the upper right corner of the redirected page.

In [3]:
```
with open('equateur-dynamic.geojson') as f:
    dynamic = json.load(f)
```

Now we can show the vectors on a slippy map. Note that it may take a while to load, which is why I'm only showing the first 10000 vectors here.

In [4]:
```
gbdx.vectors.map(features=dynamic['features'][:10000], zoom = 11)
```

4]:

Analyzing the Building Footprints

First, let's run a clustering algorithm on all the buildings so we can distinguish between different cities.

Since clustering can take a lot of memory, I'm only running it on the first 50000 buildings in the GeoJSON file. For simplicity, I'm only using the first point in each vector.

In [5]:
```
points = [i['geometry']['coordinates'][0][0] for i in dynamic['features'][
```

In [6]:
```
#arbitrarily deciding that two buildings .005 degrees apart (roughly a thi
dbs = DBSCAN(eps=.005)
clusters = dbs.fit_predict(points)
```

Our new list `clusters` is 50000 elements long, where each element is the id of a cluster, and each index is the id of a vector.

In [7]:
```
cluster_sizes = [0 for _ in range(max(clusters)+1)]
for i in range(50000):
    cluster_sizes[clusters[i]] += 1
```

Now we have a list (`cluster_sizes`), where each element corresponds to the size of a cluster.

In [8]:
```
small_clusters = [i for i in range(max(clusters)+1) if cluster_sizes[i] <=
```

We can now map all the buildings that were in small clusters. (You may have to pan and zoom a bit to find them).

In [9]:
```
gbdx.vectors.map(features=[dynamic['features'][i] for i in range(50000) if
```

9]:

By plotting the clusters as a histogram (the x-axis represents the cluster ID, and the y-axis represents the cluster's size), we can see that most clusters are small with a few exceptions.

In [10]:
```
plt.hist(clusters, bins = max(clusters) + 1)
```

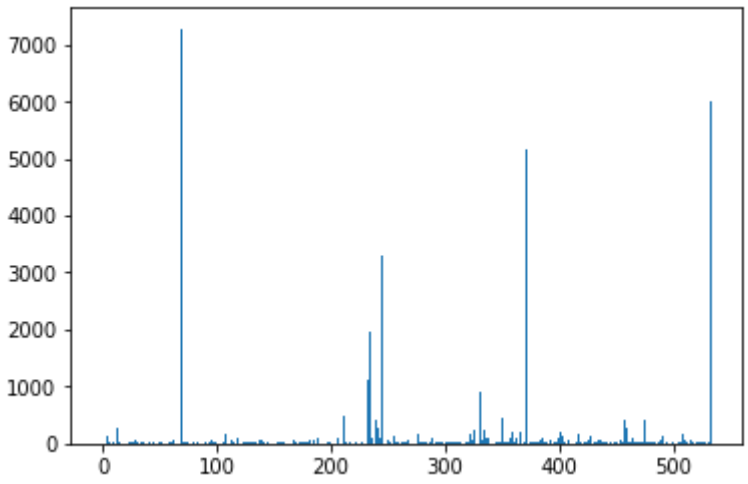
Out[10]:
```
(array([9.860e+02, 1.000e+01, 2.300e+01, 9.000e+00, 6.000e+00, 1.230e+02,
        1.400e+01, 8.000e+00, 5.000e+00, 3.200e+01, 2.400e+01, 2.200e+01,
        6.000e+00, 2.610e+02, 1.100e+01, 1.600e+01, 8.000e+00, 5.000e+00,
        9.000e+00, 9.000e+00, 3.300e+01, 5.000e+00, 9.000e+00, 1.300e+01,
        1.800e+01, 1.700e+01, 2.200e+01, 6.000e+00, 3.900e+01, 5.600e+01,
        6.000e+00, 1.700e+01, 6.000e+00, 6.000e+00, 4.300e+01, 3.800e+01,
        2.100e+01, 1.900e+01, 5.000e+00, 1.700e+01, 6.000e+00, 1.500e+01,
        4.300e+01, 7.000e+00, 1.900e+01, 1.200e+01, 1.390e+02, 7.000e+00,
        7.000e+00, 6.000e+00, 1.800e+01, 9.000e+00, 1.500e+01, 1.400e+01,
        7.000e+00, 2.700e+01, 7.000e+00, 9.000e+00, 6.000e+00, 1.800e+01,
        1.300e+01, 1.600e+01, 7.000e+00, 6.500e+00, 7.000e+00, 1.700e+01,
        8.000e+00, 6.000e+00, 8.000e+00, 2.340e+02, 7.283e+03, 1.700e+01,
        5.000e+00, 3.200e+01, 5.000e+00, 1.200e+01, 1.500e+01, 8.000e+00,
        6.100e+01, 6.000e+00, 1.700e+01, 5.000e+00, 8.000e+00, 1.500e+01,
        1.400e+01, 2.000e+01, 6.000e+00, 6.000e+00, 2.300e+01, 7.000e+00,
        1.600e+01, 2.700e+01, 3.100e+01, 6.000e+00, 1.700e+01, 9.400e+01,
        5.700e+01, 7.000e+00, 2.600e+01, 1.280e+02, 1.700e+01, 8.000e+00,
        2.020e+02, 1.000e+01, 2.600e+01, 1.000e+01, 5.000e+00, 1.800e+01,
        1.750e+02, 5.100e+01, 6.000e+00, 1.000e+01, 9.000e+00, 4.100e+01,
        5.200e+01, 1.300e+01, 6.000e+00, 5.000e+00, 4.300e+01, 1.000e+02,
        2.200e+01, 8.000e+00, 7.000e+00, 3.800e+01, 3.800e+01, 5.000e+00,
        1.600e+01, 4.400e+01, 1.100e+01, 1.600e+01, 2.700e+01, 1.700e+01,
        2.200e+01, 3.200e+01, 2.800e+01, 1.400e+01, 1.300e+01, 8.000e+00,
        5.100e+01, 6.000e+00, 8.000e+01, 8.400e+01, 3.600e+01, 2.300e+01,
```

6.000e+00, 2.400e+01, 3.300e+01, 9.000e+00, 3.300e+01, 6.000e+00,
6.000e+00, 5.000e+00, 9.000e+00, 7.000e+00, 2.700e+01, 1.700e+01,
3.500e+01, 5.000e+01, 1.500e+01, 3.700e+01, 5.000e+00, 5.000e+00,
6.000e+00, 6.000e+00, 1.800e+01, 6.000e+00, 8.000e+00, 1.400e+01,
5.500e+01, 7.000e+00, 2.800e+01, 1.600e+01, 1.000e+01, 1.700e+01,
1.100e+01, 2.500e+01, 1.000e+01, 2.200e+01, 1.000e+01, 1.800e+01,
3.500e+01, 6.000e+00, 5.800e+01, 3.100e+01, 7.000e+00, 6.000e+00,
5.000e+01, 9.800e+01, 5.000e+00, 1.100e+02, 5.000e+00, 5.000e+00,
6.500e+01, 6.000e+00, 1.600e+01, 8.000e+00, 5.000e+00, 6.700e+02,
2.100e+01, 3.100e+01, 2.100e+01, 6.000e+00, 6.000e+00, 8.000e+00,
1.100e+01, 6.000e+00, 1.800e+01, 1.100e+02, 5.000e+00, 8.000e+00,
5.000e+00, 1.000e+01, 4.990e+02, 3.750e+02, 2.700e+01, 1.000e+01,
5.000e+00, 1.400e+01, 1.200e+01, 5.000e+00, 2.300e+01, 7.000e+00,
8.000e+00, 1.700e+01, 5.000e+00, 7.000e+00, 6.000e+00, 1.900e+01,
1.200e+01, 6.000e+00, 6.000e+00, 6.000e+00, 1.000e+01, 1.126e+03,
8.000e+00, 1.967e+03, 6.300e+01, 9.200e+01, 5.000e+00, 1.200e+01,
4.020e+02, 1.040e+02, 2.870e+02, 4.300e+01, 8.700e+01, 3.100e+01,
3.298e+03, 1.000e+01, 6.900e+01, 1.000e+01, 8.900e+01, 7.800e+01,
2.710e+02, 1.700e+01, 6.000e+00, 8.900e+01, 1.260e+02, 8.500e+01,
2.000e+01, 6.000e+00, 1.500e+01, 1.000e+01, 2.800e+01, 2.600e+01,
5.000e+00, 2.100e+01, 4.100e+01, 2.200e+01, 4.600e+01, 7.000e+00,
9.000e+00, 1.000e+01, 8.000e+00, 1.400e+01, 7.000e+00, 5.000e+00,
5.000e+00, 1.790e+02, 1.650e+02, 1.600e+01, 5.000e+00, 1.900e+01,
2.600e+01, 1.400e+01, 1.700e+01, 1.400e+01, 7.000e+00, 6.000e+00,
1.200e+01, 8.800e+01, 8.600e+01, 7.000e+00, 5.000e+00, 1.700e+01,
7.000e+00, 1.600e+01, 1.300e+01, 5.000e+00, 1.600e+01, 6.000e+00,
5.000e+00, 2.400e+01, 1.100e+01, 8.500e+01, 1.200e+01, 2.000e+01,
3.100e+01, 2.300e+01, 1.600e+01, 3.500e+01, 2.700e+01, 2.500e+01,
4.400e+01, 2.600e+01, 1.400e+01, 2.500e+01, 5.000e+00, 9.000e+00,
8.000e+00, 2.200e+01, 2.210e+02, 1.100e+01, 2.900e+01, 1.770e+02,
1.280e+02, 4.800e+01, 2.530e+02, 2.500e+01, 5.000e+00, 9.000e+00,
6.000e+00, 6.000e+00, 9.060e+02, 7.400e+01, 9.800e+01, 2.410e+02,
1.050e+02, 1.110e+02, 7.000e+00, 9.300e+01, 5.000e+00, 6.500e+01,
6.000e+00, 5.000e+00, 9.000e+00, 1.300e+01, 2.500e+01, 2.900e+01,
1.200e+01, 1.700e+01, 1.100e+01, 4.370e+02, 1.020e+02, 4.100e+01,
2.300e+01, 1.260e+02, 2.100e+01, 1.380e+02, 1.090e+02, 3.300e+01,
2.110e+02, 1.500e+02, 2.600e+01, 9.800e+01, 1.500e+01, 6.000e+00,
3.840e+02, 1.910e+02, 1.000e+01, 6.000e+00, 1.500e+01, 3.200e+01,
5.178e+03, 8.000e+00, 6.000e+00, 9.000e+00, 2.900e+01, 3.800e+01,
6.000e+00, 1.500e+01, 1.424e+03, 1.300e+01, 1.100e+01, 1.500e+01,
6.100e+01, 6.000e+00, 9.900e+01, 1.170e+02, 3.300e+01, 2.020e+02,
1.900e+01, 8.000e+00, 3.600e+01, 7.400e+01, 1.700e+01, 9.000e+00,
6.000e+00, 1.100e+01, 1.200e+01, 4.700e+01, 9.600e+01, 1.470e+02,
1.970e+02, 2.800e+01, 1.350e+02, 3.100e+01, 3.400e+01, 5.000e+00,
5.000e+00, 7.500e+01, 1.100e+01, 9.000e+00, 9.000e+00, 1.000e+01,
7.000e+00, 6.000e+00, 3.700e+01, 1.150e+02, 1.760e+02, 1.600e+01,

```
        1.800e+01, 5.000e+00, 1.800e+01, 2.200e+01, 2.800e+01, 3.300e+01,
        1.500e+01, 5.100e+01, 1.370e+02, 3.140e+02, 1.000e+01, 1.500e+01,
        1.300e+01, 8.500e+01, 1.700e+01, 4.900e+01, 6.000e+00, 4.900e+01,
        5.000e+00, 1.700e+01, 1.100e+01, 1.500e+01, 3.700e+01, 7.200e+01,
        7.000e+00, 2.000e+01, 1.500e+01, 1.180e+02, 5.000e+00, 3.700e+01,
        9.500e+01, 4.400e+01, 1.500e+01, 5.000e+00, 3.600e+01, 5.100e+01,
        1.300e+01, 1.720e+02, 4.280e+02, 3.300e+01, 2.900e+02, 5.000e+00,
        2.100e+01, 1.100e+01, 8.000e+00, 8.100e+01, 6.300e+01, 1.300e+01,
        7.000e+00, 2.600e+01, 6.100e+01, 1.200e+01, 2.000e+01, 3.600e+01,
        1.300e+01, 5.000e+00, 4.080e+02, 1.190e+02, 2.000e+01, 1.500e+01,
        7.000e+00, 2.500e+01, 7.000e+00, 2.200e+01, 8.000e+00, 1.800e+01,
        1.000e+01, 6.400e+01, 3.600e+01, 1.070e+02, 6.100e+01, 3.400e+01,
        1.340e+02, 6.000e+00, 2.200e+02, 1.200e+01, 1.200e+02, 9.000e+00,
        6.000e+00, 7.000e+00, 1.200e+01, 7.000e+00, 8.000e+00, 5.000e+00,
        8.000e+00, 1.100e+01, 2.100e+01, 1.100e+01, 1.680e+02, 1.760e+02,
        2.100e+01, 5.900e+01, 1.400e+01, 1.100e+01, 1.000e+01, 6.000e+00,
        4.900e+01, 7.000e+00, 1.300e+01, 1.800e+01, 8.000e+00, 1.300e+01,
        6.600e+01, 1.200e+01, 9.000e+00, 3.500e+01, 8.000e+00, 2.700e+01,
        6.000e+00, 1.800e+01, 1.000e+01, 1.100e+01, 4.000e+01, 5.000e+00,
        6.010e+03]),
 array([ -1.,   0.,   1.,   2.,   3.,   4.,   5.,   6.,   7.,   8.,   9.,
         10.,  11.,  12.,  13.,  14.,  15.,  16.,  17.,  18.,  19.,  20.,
         21.,  22.,  23.,  24.,  25.,  26.,  27.,  28.,  29.,  30.,  31.,
         32.,  33.,  34.,  35.,  36.,  37.,  38.,  39.,  40.,  41.,  42.,
         43.,  44.,  45.,  46.,  47.,  48.,  49.,  50.,  51.,  52.,  53.,
         54.,  55.,  56.,  57.,  58.,  59.,  60.,  61.,  62.,  63.,  64.,
         65.,  66.,  67.,  68.,  69.,  70.,  71.,  72.,  73.,  74.,  75.,
         76.,  77.,  78.,  79.,  80.,  81.,  82.,  83.,  84.,  85.,  86.,
         87.,  88.,  89.,  90.,  91.,  92.,  93.,  94.,  95.,  96.,  97.,
         98.,  99., 100., 101., 102., 103., 104., 105., 106., 107., 108.,
        109., 110., 111., 112., 113., 114., 115., 116., 117., 118., 119.,
        120., 121., 122., 123., 124., 125., 126., 127., 128., 129., 130.,
        131., 132., 133., 134., 135., 136., 137., 138., 139., 140., 141.,
        142., 143., 144., 145., 146., 147., 148., 149., 150., 151., 152.,
        153., 154., 155., 156., 157., 158., 159., 160., 161., 162., 163.,
        164., 165., 166., 167., 168., 169., 170., 171., 172., 173., 174.,
        175., 176., 177., 178., 179., 180., 181., 182., 183., 184., 185.,
        186., 187., 188., 189., 190., 191., 192., 193., 194., 195., 196.,
        197., 198., 199., 200., 201., 202., 203., 204., 205., 206., 207.,
        208., 209., 210., 211., 212., 213., 214., 215., 216., 217., 218.,
        219., 220., 221., 222., 223., 224., 225., 226., 227., 228., 229.,
        230., 231., 232., 233., 234., 235., 236., 237., 238., 239., 240.,
        241., 242., 243., 244., 245., 246., 247., 248., 249., 250., 251.,
        252., 253., 254., 255., 256., 257., 258., 259., 260., 261., 262.,
        263., 264., 265., 266., 267., 268., 269., 270., 271., 272., 273.,
        274., 275., 276., 277., 278., 279., 280., 281., 282., 283., 284.,
```

```
       285., 286., 287., 288., 289., 290., 291., 292., 293., 294., 295.,
       296., 297., 298., 299., 300., 301., 302., 303., 304., 305., 306.,
       307., 308., 309., 310., 311., 312., 313., 314., 315., 316., 317.,
       318., 319., 320., 321., 322., 323., 324., 325., 326., 327., 328.,
       329., 330., 331., 332., 333., 334., 335., 336., 337., 338., 339.,
       340., 341., 342., 343., 344., 345., 346., 347., 348., 349., 350.,
       351., 352., 353., 354., 355., 356., 357., 358., 359., 360., 361.,
       362., 363., 364., 365., 366., 367., 368., 369., 370., 371., 372.,
       373., 374., 375., 376., 377., 378., 379., 380., 381., 382., 383.,
       384., 385., 386., 387., 388., 389., 390., 391., 392., 393., 394.,
       395., 396., 397., 398., 399., 400., 401., 402., 403., 404., 405.,
       406., 407., 408., 409., 410., 411., 412., 413., 414., 415., 416.,
       417., 418., 419., 420., 421., 422., 423., 424., 425., 426., 427.,
       428., 429., 430., 431., 432., 433., 434., 435., 436., 437., 438.,
       439., 440., 441., 442., 443., 444., 445., 446., 447., 448., 449.,
       450., 451., 452., 453., 454., 455., 456., 457., 458., 459., 460.,
       461., 462., 463., 464., 465., 466., 467., 468., 469., 470., 471.,
       472., 473., 474., 475., 476., 477., 478., 479., 480., 481., 482.,
       483., 484., 485., 486., 487., 488., 489., 490., 491., 492., 493.,
       494., 495., 496., 497., 498., 499., 500., 501., 502., 503., 504.,
       505., 506., 507., 508., 509., 510., 511., 512., 513., 514., 515.,
       516., 517., 518., 519., 520., 521., 522., 523., 524., 525., 526.,
       527., 528., 529., 530., 531., 532., 533., 534.]),
    )
```

0]:



1 [11]:

```
plt.hist(cluster_sizes, bins = 300)
```

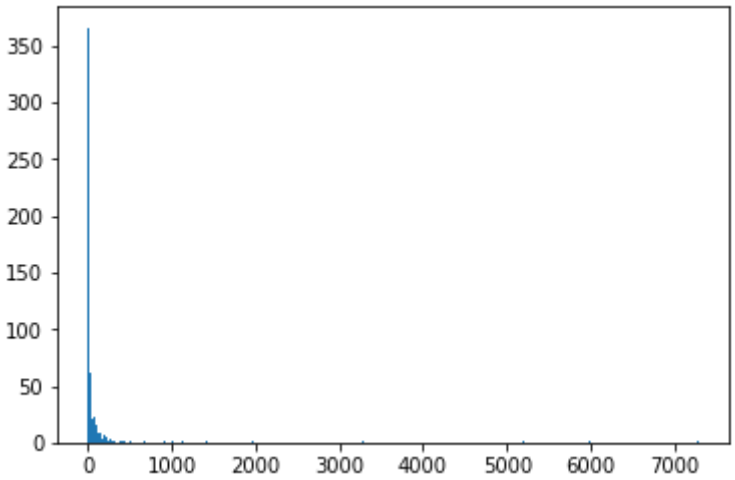
1]: (array([366., 61., 21., 23., 15., 9., 3., 7., 5., 2., 3.,

```
       2.,  1.,  0.,  0.,  2.,  2.,  2.,  0.,  0.,  1.,  0.,
       0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  1.,  0.,  0.,
       0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
```

```
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  1.]),
 array([5.00000e+00, 2.92600e+01, 5.35200e+01, 7.77800e+01, 1.02040e+02,
        1.26300e+02, 1.50560e+02, 1.74820e+02, 1.99080e+02, 2.23340e+02,
        2.47600e+02, 2.71860e+02, 2.96120e+02, 3.20380e+02, 3.44640e+02,
        3.68900e+02, 3.93160e+02, 4.17420e+02, 4.41680e+02, 4.65940e+02,
        4.90200e+02, 5.14460e+02, 5.38720e+02, 5.62980e+02, 5.87240e+02,
        6.11500e+02, 6.35760e+02, 6.60020e+02, 6.84280e+02, 7.08540e+02,
        7.32800e+02, 7.57060e+02, 7.81320e+02, 8.05580e+02, 8.29840e+02,
        8.54100e+02, 8.78360e+02, 9.02620e+02, 9.26880e+02, 9.51140e+02,
        9.75400e+02, 9.99660e+02, 1.02392e+03, 1.04818e+03, 1.07244e+03,
        1.09670e+03, 1.12096e+03, 1.14522e+03, 1.16948e+03, 1.19374e+03,
        1.21800e+03, 1.24226e+03, 1.26652e+03, 1.29078e+03, 1.31504e+03,
        1.33930e+03, 1.36356e+03, 1.38782e+03, 1.41208e+03, 1.43634e+03,
        1.46060e+03, 1.48486e+03, 1.50912e+03, 1.53338e+03, 1.55764e+03,
        1.58190e+03, 1.60616e+03, 1.63042e+03, 1.65468e+03, 1.67894e+03,
        1.70320e+03, 1.72746e+03, 1.75172e+03, 1.77598e+03, 1.80024e+03,
        1.82450e+03, 1.84876e+03, 1.87302e+03, 1.89728e+03, 1.92154e+03,
        1.94580e+03, 1.97006e+03, 1.99432e+03, 2.01858e+03, 2.04284e+03,
        2.06710e+03, 2.09136e+03, 2.11562e+03, 2.13988e+03, 2.16414e+03,
        2.18840e+03, 2.21266e+03, 2.23692e+03, 2.26118e+03, 2.28544e+03,
        2.30970e+03, 2.33396e+03, 2.35822e+03, 2.38248e+03, 2.40674e+03,
        2.43100e+03, 2.45526e+03, 2.47952e+03, 2.50378e+03, 2.52804e+03,
        2.55230e+03, 2.57656e+03, 2.60082e+03, 2.62508e+03, 2.64934e+03,
        2.67360e+03, 2.69786e+03, 2.72212e+03, 2.74638e+03, 2.77064e+03,
        2.79490e+03, 2.81916e+03, 2.84342e+03, 2.86768e+03, 2.89194e+03,
```

```
        2.91620e+03, 2.94046e+03, 2.96472e+03, 2.98898e+03, 3.01324e+03,
        3.03750e+03, 3.06176e+03, 3.08602e+03, 3.11028e+03, 3.13454e+03,
        3.15880e+03, 3.18306e+03, 3.20732e+03, 3.23158e+03, 3.25584e+03,
        3.28010e+03, 3.30436e+03, 3.32862e+03, 3.35288e+03, 3.37714e+03,
        3.40140e+03, 3.42566e+03, 3.44992e+03, 3.47418e+03, 3.49844e+03,
        3.52270e+03, 3.54696e+03, 3.57122e+03, 3.59548e+03, 3.61974e+03,
        3.64400e+03, 3.66826e+03, 3.69252e+03, 3.71678e+03, 3.74104e+03,
        3.76530e+03, 3.78956e+03, 3.81382e+03, 3.83808e+03, 3.86234e+03,
        3.88660e+03, 3.91086e+03, 3.93512e+03, 3.95938e+03, 3.98364e+03,
        4.00790e+03, 4.03216e+03, 4.05642e+03, 4.08068e+03, 4.10494e+03,
        4.12920e+03, 4.15346e+03, 4.17772e+03, 4.20198e+03, 4.22624e+03,
        4.25050e+03, 4.27476e+03, 4.29902e+03, 4.32328e+03, 4.34754e+03,
        4.37180e+03, 4.39606e+03, 4.42032e+03, 4.44458e+03, 4.46884e+03,
        4.49310e+03, 4.51736e+03, 4.54162e+03, 4.56588e+03, 4.59014e+03,
        4.61440e+03, 4.63866e+03, 4.66292e+03, 4.68718e+03, 4.71144e+03,
        4.73570e+03, 4.75996e+03, 4.78422e+03, 4.80848e+03, 4.83274e+03,
        4.85700e+03, 4.88126e+03, 4.90552e+03, 4.92978e+03, 4.95404e+03,
        4.97830e+03, 5.00256e+03, 5.02682e+03, 5.05108e+03, 5.07534e+03,
        5.09960e+03, 5.12386e+03, 5.14812e+03, 5.17238e+03, 5.19664e+03,
        5.22090e+03, 5.24516e+03, 5.26942e+03, 5.29368e+03, 5.31794e+03,
        5.34220e+03, 5.36646e+03, 5.39072e+03, 5.41498e+03, 5.43924e+03,
        5.46350e+03, 5.48776e+03, 5.51202e+03, 5.53628e+03, 5.56054e+03,
        5.58480e+03, 5.60906e+03, 5.63332e+03, 5.65758e+03, 5.68184e+03,
        5.70610e+03, 5.73036e+03, 5.75462e+03, 5.77888e+03, 5.80314e+03,
        5.82740e+03, 5.85166e+03, 5.87592e+03, 5.90018e+03, 5.92444e+03,
        5.94870e+03, 5.97296e+03, 5.99722e+03, 6.02148e+03, 6.04574e+03,
        6.07000e+03, 6.09426e+03, 6.11852e+03, 6.14278e+03, 6.16704e+03,
        6.19130e+03, 6.21556e+03, 6.23982e+03, 6.26408e+03, 6.28834e+03,
        6.31260e+03, 6.33686e+03, 6.36112e+03, 6.38538e+03, 6.40964e+03,
        6.43390e+03, 6.45816e+03, 6.48242e+03, 6.50668e+03, 6.53094e+03,
        6.55520e+03, 6.57946e+03, 6.60372e+03, 6.62798e+03, 6.65224e+03,
        6.67650e+03, 6.70076e+03, 6.72502e+03, 6.74928e+03, 6.77354e+03,
        6.79780e+03, 6.82206e+03, 6.84632e+03, 6.87058e+03, 6.89484e+03,
        6.91910e+03, 6.94336e+03, 6.96762e+03, 6.99188e+03, 7.01614e+03,
        7.04040e+03, 7.06466e+03, 7.08892e+03, 7.11318e+03, 7.13744e+03,
        7.16170e+03, 7.18596e+03, 7.21022e+03, 7.23448e+03, 7.25874e+03,
        7.28300e+03]),
    )
```

1]:



One factor in risk for Ebola outbreaks is population density. One way to estimate the density of a given cluster would be to draw a buffer around each building in a cluster, and find the number of buildings in this cluster divided by the area of the union of the buffers.

*Note that the location of the points is given in latitude and longitude. Since all of our data is near the equator, I'll just approximate a degree of latitude or longitude as 69 miles.

ı [12]:

```
densities = [0 for _ in range(len(cluster_sizes))]
buffers = [[] for _ in range(len(cluster_sizes))]
unions = [0 for _ in range(len(cluster_sizes))] #We'll use this later
for i in range(50000):
    buffers[clusters[i]].append(shapely.geometry.Point(points[i]).buffer(.
for i in range(len(buffers)):
    union = shapely.ops.cascaded_union(buffers[i])
    densities[i] = cluster_sizes[i] / (union.area * 69 * 69)
    unions[i] = union
```

Here `densities` is a list of the number of buildings per square mile for each cluster. Keep in mind that the densities may be too low because I did not incorporate all of the buildings in my analysis.

ı [13]:

```
plt.hist(densities, bins = 300)
```

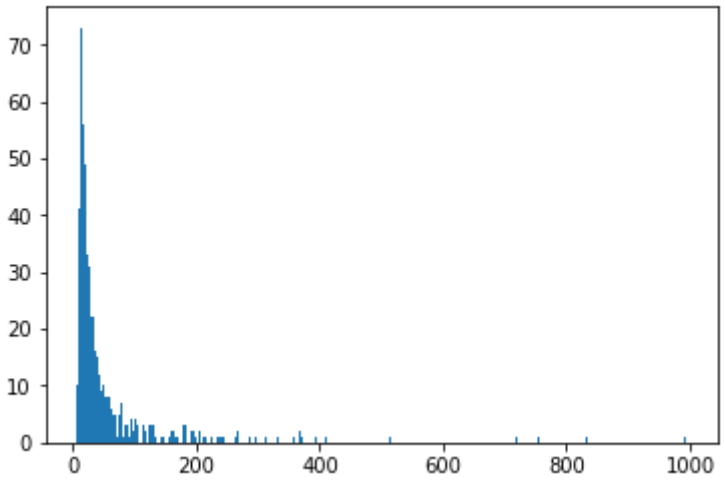
3]:

```
(array([10., 41., 73., 56., 49., 33., 31., 22., 22., 16., 15., 12.,  9.,
        10.,  8.,  8.,  8.,  6.,  5.,  5.,  1.,  5.,  7.,  1.,  3.,  3.,
         1.,  4.,  2.,  4.,  3.,  0.,  3.,  2.,  0.,  3.,  3.,  3.,  1.,
         0.,  0.,  1.,  1.,  0.,  0.,  1.,  2.,  2.,  1.,  1.,  0.,  0.,
         3.,  3.,  0.,  0.,  2.,  2.,  1.,  0.,  2.,  0.,  1.,  1.,  0.,
         0.,  1.,  0.,  0.,  1.,  1.,  1.,  1.,  0.,  0.,  0.,  0.,  0.,
         1.,  2.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  1.,  0.,  0.,
         0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,
```

```
       0.,  0.,  1.,  0.,  0.,  2.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,
       1.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       1.]), array([  5.23333451,   8.5324485 ,  11.8315625 ,  15.1306765 ,
       18.4297905 ,  21.72890449,  25.02801849,  28.32713249,
       31.62624648,  34.92536048,  38.22447448,  41.52358848,
       44.82270247,  48.12181647,  51.42093047,  54.72004447,
       58.01915846,  61.31827246,  64.61738646,  67.91650045,
       71.21561445,  74.51472845,  77.81384245,  81.11295644,
       84.41207044,  87.71118444,  91.01029843,  94.30941243,
       97.60852643, 100.90764043, 104.20675442, 107.50586842,
      110.80498242, 114.10409641, 117.40321041, 120.70232441,
      124.00143841, 127.3005524 , 130.5996664 , 133.8987804 ,
      137.19789439, 140.49700839, 143.79612239, 147.09523639,
      150.39435038, 153.69346438, 156.99257838, 160.29169237,
      163.59080637, 166.88992037, 170.18903437, 173.48814836,
      176.78726236, 180.08637636, 183.38549036, 186.68460435,
      189.98371835, 193.28283235, 196.58194634, 199.88106034,
      203.18017434, 206.47928834, 209.77840233, 213.07751633,
      216.37663033, 219.67574432, 222.97485832, 226.27397232,
      229.57308632, 232.87220031, 236.17131431, 239.47042831,
      242.7695423 , 246.0686563 , 249.3677703 , 252.6668843 ,
      255.96599829, 259.26511229, 262.56422629, 265.86334028,
      269.16245428, 272.46156828, 275.76068228, 279.05979627,
      282.35891027, 285.65802427, 288.95713827, 292.25625226,
      295.55536626, 298.85448026, 302.15359425, 305.45270825,
      308.75182225, 312.05093625, 315.35005024, 318.64916424,
      321.94827824, 325.24739223, 328.54650623, 331.84562023,
      335.14473423, 338.44384822, 341.74296222, 345.04207622,
      348.34119021, 351.64030421, 354.93941821, 358.23853221,
      361.5376462 , 364.8367602 , 368.1358742 , 371.43498819,
      374.73410219, 378.03321619, 381.33233019, 384.63144418,
      387.93055818, 391.22967218, 394.52878617, 397.82790017,
      401.12701417, 404.42612817, 407.72524216, 411.02435616,
```

```
414.32347016, 417.62258416, 420.92169815, 424.22081215,
427.51992615, 430.81904014, 434.11815414, 437.41726814,
440.71638214, 444.01549613, 447.31461013, 450.61372413,
453.91283812, 457.21195212, 460.51106612, 463.81018012,
467.10929411, 470.40840811, 473.70752211, 477.0066361 ,
480.3057501 , 483.6048641 , 486.9039781 , 490.20309209,
493.50220609, 496.80132009, 500.10043408, 503.39954808,
506.69866208, 509.99777608, 513.29689007, 516.59600407,
519.89511807, 523.19423206, 526.49334606, 529.79246006,
533.09157406, 536.39068805, 539.68980205, 542.98891605,
546.28803005, 549.58714404, 552.88625804, 556.18537204,
559.48448603, 562.78360003, 566.08271403, 569.38182803,
572.68094202, 575.98005602, 579.27917002, 582.57828401,
585.87739801, 589.17651201, 592.47562601, 595.77474  ,
599.073854  , 602.372968  , 605.67208199, 608.97119599,
612.27030999, 615.56942399, 618.86853798, 622.16765198,
625.46676598, 628.76587997, 632.06499397, 635.36410797,
638.66322197, 641.96233596, 645.26144996, 648.56056396,
651.85967796, 655.15879195, 658.45790595, 661.75701995,
665.05613394, 668.35524794, 671.65436194, 674.95347594,
678.25258993, 681.55170393, 684.85081793, 688.14993192,
691.44904592, 694.74815992, 698.04727392, 701.34638791,
704.64550191, 707.94461591, 711.2437299 , 714.5428439 ,
717.8419579 , 721.1410719 , 724.44018589, 727.73929989,
731.03841389, 734.33752788, 737.63664188, 740.93575588,
744.23486988, 747.53398387, 750.83309787, 754.13221187,
757.43132586, 760.73043986, 764.02955386, 767.32866786,
770.62778185, 773.92689585, 777.22600985, 780.52512385,
783.82423784, 787.12335184, 790.42246584, 793.72157983,
797.02069383, 800.31980783, 803.61892183, 806.91803582,
810.21714982, 813.51626382, 816.81537781, 820.11449181,
823.41360581, 826.71271981, 830.0118338 , 833.3109478 ,
836.6100618 , 839.90917579, 843.20828979, 846.50740379,
849.80651779, 853.10563178, 856.40474578, 859.70385978,
863.00297377, 866.30208777, 869.60120177, 872.90031577,
876.19942976, 879.49854376, 882.79765776, 886.09677176,
889.39588575, 892.69499975, 895.99411375, 899.29322774,
902.59234174, 905.89145574, 909.19056974, 912.48968373,
915.78879773, 919.08791173, 922.38702572, 925.68613972,
928.98525372, 932.28436772, 935.58348171, 938.88259571,
942.18170971, 945.4808237 , 948.7799377 , 952.0790517 ,
955.3781657 , 958.67727969, 961.97639369, 965.27550769,
968.57462168, 971.87373568, 975.17284968, 978.47196368,
981.77107767, 985.07019167, 988.36930567, 991.66841966,
994.96753366]), )
```

3]:



Naturally, since so many of the clusters were very small, it makes sense that their calculated density would be small as well (because each building gets buffered with a third of a mile).

Using the Human Landscape Data

We can find some interesting statistics by looking at the building footprints, but in order to really assess an area's risk of Ebola outbreaks, we need to look at access to hospitals, communication, and transportation, to name a few. We can find this information in the Human Landscape data.

ı [14]:
```
# with open('human-landscape/COD_MedicalFacilities_130215.json') as f:
#     health = json.load(f)
# with open('human-landscape/COD_InternetCafes_130215_WGS.json') as f:
#     internet = json.load(f)
with open('human-landscape/COD_Roads_130214_WGS84.json') as f:
    roads = json.load(f)
```

ı [15]:
```
gbdx.vectors.map(features = roads['features'], zoom = 11)
```

5]:

First let's take a look at road densities. Road densities have a U-shaped correlation with Ebola outbreaks, because more roads can make hospitals more accessible, but also facilitate the spread of the disease. (http://currents.plos.org/outbreaks/article/obk-16-0062-spatial-determinants-of-ebola-virus-disease-risk-for-the-west-african-epidemic/#ref12)

To find the road density of a cluster, let's buffer the roads so they have some width, and then find the percent of area they cover. This will give a rather arbitrary unit of measurement, but a consistent way to compare different clusters.

[16]:
```
road_lines = []
for i in range(len(roads['features'])):
    if isinstance(roads['features'][i]['geometry']['coordinates'][0][0], l
        for j in roads['features'][i]['geometry']['coordinates']:
            road_lines.append(shapely.geometry.LineString(j).buffer(.0001)
    else:
        road_lines.append(shapely.geometry.LineString(roads['features'][i]
all_roads = shapely.ops.cascaded_union(road_lines)
```

[17]:
```
road_densities = [(100*i.intersection(all_roads).area) / i.area for i in u
```

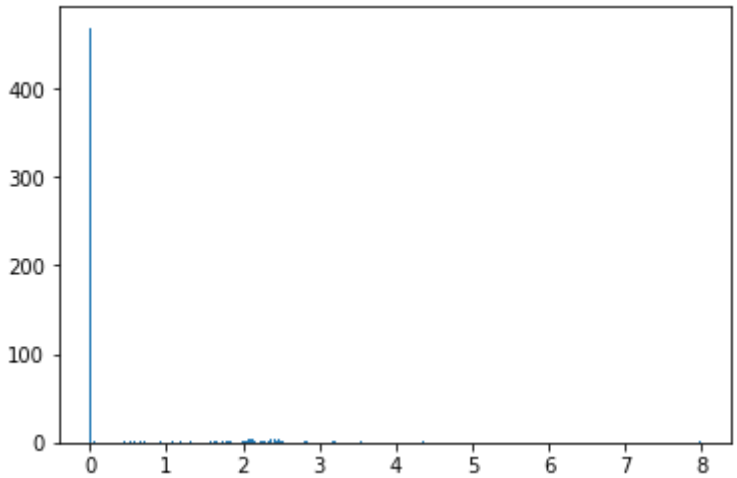
[18]:
```
plt.hist(road_densities, bins=300)
```

8]:
```
(array([469.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  1.,  0.,  2.,
        0.,  0.,  1.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  1.,  0.,  0.,  0.,  0.,  0.,  2.,  0.,  0.,  0.,
        1.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  1.,  0.,  1.,  1.,  0.,  0.,  1.,
        0.,  1.,  1.,  1.,  0.,  0.,  0.,  0.,  0.,  2.,  2.,
        2.,  4.,  3.,  3.,  1.,  0.,  2.,  2.,  1.,  0.,  2.,
        4.,  0.,  5.,  2.,  3.,  1.,  1.,  0.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  1.,  1.,  0.,  0.,  0.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  1.,
        0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
```

```
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
       0.,  0.,  1.]),
 array([0.        , 0.02661624, 0.05323249, 0.07984873, 0.10646497,
        0.13308121, 0.15969746, 0.1863137 , 0.21292994, 0.23954618,
        0.26616243, 0.29277867, 0.31939491, 0.34601115, 0.3726274 ,
        0.39924364, 0.42585988, 0.45247612, 0.47909237, 0.50570861,
        0.53232485, 0.55894109, 0.58555734, 0.61217358, 0.63878982,
        0.66540606, 0.69202231, 0.71863855, 0.74525479, 0.77187103,
        0.79848728, 0.82510352, 0.85171976, 0.878336  , 0.90495225,
        0.93156849, 0.95818473, 0.98480097, 1.01141722, 1.03803346,
        1.0646497 , 1.09126594, 1.11788219, 1.14449843, 1.17111467,
        1.19773091, 1.22434716, 1.2509634 , 1.27757964, 1.30419588,
        1.33081213, 1.35742837, 1.38404461, 1.41066085, 1.4372771 ,
        1.46389334, 1.49050958, 1.51712582, 1.54374207, 1.57035831,
        1.59697455, 1.62359079, 1.65020704, 1.67682328, 1.70343952,
        1.73005576, 1.75667201, 1.78328825, 1.80990449, 1.83652073,
        1.86313698, 1.88975322, 1.91636946, 1.9429857 , 1.96960195,
        1.99621819, 2.02283443, 2.04945067, 2.07606692, 2.10268316,
        2.1292994 , 2.15591564, 2.18253189, 2.20914813, 2.23576437,
        2.26238061, 2.28899686, 2.3156131 , 2.34222934, 2.36884558,
        2.39546183, 2.42207807, 2.44869431, 2.47531055, 2.5019268 ,
        2.52854304, 2.55515928, 2.58177552, 2.60839177, 2.63500801,
        2.66162425, 2.68824049, 2.71485674, 2.74147298, 2.76808922,
        2.79470546, 2.82132171, 2.84793795, 2.87455419, 2.90117043,
        2.92778668, 2.95440292, 2.98101916, 3.0076354 , 3.03425165,
        3.06086789, 3.08748413, 3.11410037, 3.14071662, 3.16733286,
        3.1939491 , 3.22056535, 3.24718159, 3.27379783, 3.30041407,
        3.32703032, 3.35364656, 3.3802628 , 3.40687904, 3.43349529,
        3.46011153, 3.48672777, 3.51334401, 3.53996026, 3.5665765 ,
        3.59319274, 3.61980898, 3.64642523, 3.67304147, 3.69965771,
        3.72627395, 3.7528902 , 3.77950644, 3.80612268, 3.83273892,
        3.85935517, 3.88597141, 3.91258765, 3.93920389, 3.96582014,
        3.99243638, 4.01905262, 4.04566886, 4.07228511, 4.09890135,
```

```
        4.12551759, 4.15213383, 4.17875008, 4.20536632, 4.23198256,
        4.2585988 , 4.28521505, 4.31183129, 4.33844753, 4.36506377,
        4.39168002, 4.41829626, 4.4449125 , 4.47152874, 4.49814499,
        4.52476123, 4.55137747, 4.57799371, 4.60460996, 4.6312262 ,
        4.65784244, 4.68445868, 4.71107493, 4.73769117, 4.76430741,
        4.79092365, 4.8175399 , 4.84415614, 4.87077238, 4.89738862,
        4.92400487, 4.95062111, 4.97723735, 5.00385359, 5.03046984,
        5.05708608, 5.08370232, 5.11031856, 5.13693481, 5.16355105,
        5.19016729, 5.21678353, 5.24339978, 5.27001602, 5.29663226,
        5.3232485 , 5.34986475, 5.37648099, 5.40309723, 5.42971347,
        5.45632972, 5.48294596, 5.5095622 , 5.53617844, 5.56279469,
        5.58941093, 5.61602717, 5.64264341, 5.66925966, 5.6958759 ,
        5.72249214, 5.74910838, 5.77572463, 5.80234087, 5.82895711,
        5.85557335, 5.8821896 , 5.90880584, 5.93542208, 5.96203832,
        5.98865457, 6.01527081, 6.04188705, 6.06850329, 6.09511954,
        6.12173578, 6.14835202, 6.17496826, 6.20158451, 6.22820075,
        6.25481699, 6.28143323, 6.30804948, 6.33466572, 6.36128196,
        6.38789821, 6.41451445, 6.44113069, 6.46774693, 6.49436318,
        6.52097942, 6.54759566, 6.5742119 , 6.60082815, 6.62744439,
        6.65406063, 6.68067687, 6.70729312, 6.73390936, 6.7605256 ,
        6.78714184, 6.81375809, 6.84037433, 6.86699057, 6.89360681,
        6.92022306, 6.9468393 , 6.97345554, 7.00007178, 7.02668803,
        7.05330427, 7.07992051, 7.10653675, 7.133153  , 7.15976924,
        7.18638548, 7.21300172, 7.23961797, 7.26623421, 7.29285045,
        7.31946669, 7.34608294, 7.37269918, 7.39931542, 7.42593166,
        7.45254791, 7.47916415, 7.50578039, 7.53239663, 7.55901288,
        7.58562912, 7.61224536, 7.6388616 , 7.66547785, 7.69209409,
        7.71871033, 7.74532657, 7.77194282, 7.79855906, 7.8251753 ,
        7.85179154, 7.87840779, 7.90502403, 7.93164027, 7.95825651,
        7.98487276]),
    )
```

8]:



The heavy skew to 0 is probably due to the fact that most of the clusters were very small, and therefore have a small chance of intersecting the roads in our data. For these clusters, it may be more useful to look at how close they are to roads.

[19]:
```
distance_to_roads = [i.distance(all_roads) * 69 for i in unions] #factor c
```

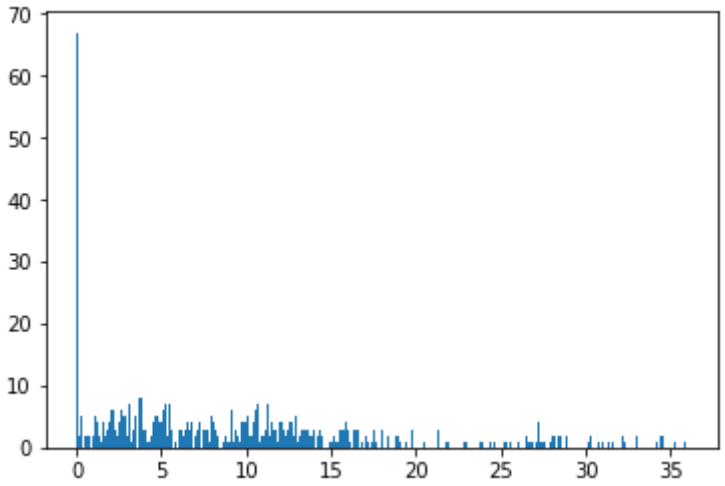
[20]:
```
plt.hist(distance_to_roads, bins = 300)
```

0]:
```
(array([67., 2., 5., 0., 2., 2., 2., 0., 2., 5., 4., 2., 1.,
        4., 2., 3., 4., 6., 6., 3., 2., 4., 6., 5., 5., 2.,
        7., 1., 3., 5., 0., 8., 3., 3., 1., 1., 2., 4., 5.,
        5., 4., 4., 6., 7., 2., 7., 3., 0., 1., 0., 3., 3.,
        2., 3., 4., 3., 4., 0., 2., 3., 4., 0., 3., 3., 3.,
        1., 5., 4., 3., 2., 0., 0., 1., 2., 1., 1., 6., 1.,
        3., 2., 1., 4., 4., 4., 5., 2., 2., 4., 6., 7., 1.,
        2., 2., 3., 7., 2., 4., 3., 3., 1., 4., 3., 2., 3.,
        4., 4., 2., 5., 1., 2., 3., 3., 3., 3., 2., 2., 3.,
        0., 2., 3., 2., 0., 0., 0., 1., 1., 2., 1., 1., 3.,
        3., 3., 4., 3., 1., 0., 3., 3., 3., 0., 1., 0., 2.,
        1., 0., 1., 3., 1., 0., 0., 3., 0., 0., 2., 0., 0.,
        0., 2., 2., 1., 0., 0., 1., 0., 0., 3., 0., 0., 0.,
        0., 1., 0., 0., 0., 0., 0., 0., 3., 0., 0., 0., 1.,
        1., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0.,
        0., 0., 0., 1., 1., 0., 0., 0., 1., 0., 1., 0., 0.,
        0., 0., 1., 1., 0., 1., 0., 0., 0., 1., 0., 0., 0.,
        2., 1., 1., 1., 0., 1., 4., 1., 1., 1., 0., 0., 1.,
        2., 2., 0., 2., 0., 0., 2., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 1., 2., 0., 0., 0., 1., 0., 1., 0.,
        0., 1., 0., 1., 0., 0., 0., 0., 2., 1., 0., 0., 0.,
        0., 0., 2., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 2., 2., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
        1.]),
 array([ 0.        , 0.11981382, 0.23962764, 0.35944146, 0.47925528,
        0.5990691 , 0.71888293, 0.83869675, 0.95851057, 1.07832439,
        1.19813821, 1.31795203, 1.43776585, 1.55757967, 1.67739349,
        1.79720731, 1.91702113, 2.03683496, 2.15664878, 2.2764626 ,
        2.39627642, 2.51609024, 2.63590406, 2.75571788, 2.8755317 ,
        2.99534552, 3.11515934, 3.23497316, 3.35478698, 3.47460081,
        3.59441463, 3.71422845, 3.83404227, 3.95385609, 4.07366991,
        4.19348373, 4.31329755, 4.43311137, 4.55292519, 4.67273901,
        4.79255284, 4.91236666, 5.03218048, 5.1519943 , 5.27180812,
        5.39162194, 5.51143576, 5.63124958, 5.7510634 , 5.87087722,
        5.99069104, 6.11050487, 6.23031869, 6.35013251, 6.46994633,
        6.58976015, 6.70957397, 6.82938779, 6.94920161, 7.06901543,
```

7.18882925, 7.30864307, 7.42845689, 7.54827072, 7.66808454,
7.78789836, 7.90771218, 8.027526 , 8.14733982, 8.26715364,
8.38696746, 8.50678128, 8.6265951 , 8.74640892, 8.86622275,
8.98603657, 9.10585039, 9.22566421, 9.34547803, 9.46529185,
9.58510567, 9.70491949, 9.82473331, 9.94454713, 10.06436095,
10.18417478, 10.3039886 , 10.42380242, 10.54361624, 10.66343006,
10.78324388, 10.9030577 , 11.02287152, 11.14268534, 11.26249916,
11.38231298, 11.5021268 , 11.62194063, 11.74175445, 11.86156827,
11.98138209, 12.10119591, 12.22100973, 12.34082355, 12.46063737,
12.58045119, 12.70026501, 12.82007883, 12.93989266, 13.05970648,
13.1795203 , 13.29933412, 13.41914794, 13.53896176, 13.65877558,
13.7785894 , 13.89840322, 14.01821704, 14.13803086, 14.25784469,
14.37765851, 14.49747233, 14.61728615, 14.73709997, 14.85691379,
14.97672761, 15.09654143, 15.21635525, 15.33616907, 15.45598289,
15.57579671, 15.69561054, 15.81542436, 15.93523818, 16.055052 ,
16.17486582, 16.29467964, 16.41449346, 16.53430728, 16.6541211 ,
16.77393492, 16.89374874, 17.01356257, 17.13337639, 17.25319021,
17.37300403, 17.49281785, 17.61263167, 17.73244549, 17.85225931,
17.97207313, 18.09188695, 18.21170077, 18.3315146 , 18.45132842,
18.57114224, 18.69095606, 18.81076988, 18.9305837 , 19.05039752,
19.17021134, 19.29002516, 19.40983898, 19.5296528 , 19.64946662,
19.76928045, 19.88909427, 20.00890809, 20.12872191, 20.24853573,
20.36834955, 20.48816337, 20.60797719, 20.72779101, 20.84760483,
20.96741865, 21.08723248, 21.2070463 , 21.32686012, 21.44667394,
21.56648776, 21.68630158, 21.8061154 , 21.92592922, 22.04574304,
22.16555686, 22.28537068, 22.40518451, 22.52499833, 22.64481215,
22.76462597, 22.88443979, 23.00425361, 23.12406743, 23.24388125,
23.36369507, 23.48350889, 23.60332271, 23.72313653, 23.84295036,
23.96276418, 24.082578 , 24.20239182, 24.32220564, 24.44201946,
24.56183328, 24.6816471 , 24.80146092, 24.92127474, 25.04108856,
25.16090239, 25.28071621, 25.40053003, 25.52034385, 25.64015767,
25.75997149, 25.87978531, 25.99959913, 26.11941295, 26.23922677,
26.35904059, 26.47885442, 26.59866824, 26.71848206, 26.83829588,
26.9581097 , 27.07792352, 27.19773734, 27.31755116, 27.43736498,
27.5571788 , 27.67699262, 27.79680645, 27.91662027, 28.03643409,
28.15624791, 28.27606173, 28.39587555, 28.51568937, 28.63550319,
28.75531701, 28.87513083, 28.99494465, 29.11475847, 29.2345723 ,
29.35438612, 29.47419994, 29.59401376, 29.71382758, 29.8336414 ,
29.95345522, 30.07326904, 30.19308286, 30.31289668, 30.4327105 ,
30.55252433, 30.67233815, 30.79215197, 30.91196579, 31.03177961,
31.15159343, 31.27140725, 31.39122107, 31.51103489, 31.63084871,
31.75066253, 31.87047636, 31.99029018, 32.110104 , 32.22991782,
32.34973164, 32.46954546, 32.58935928, 32.7091731 , 32.82898692,
32.94880074, 33.06861456, 33.18842838, 33.30824221, 33.42805603,
33.54786985, 33.66768367, 33.78749749, 33.90731131, 34.02712513,
34.14693895, 34.26675277, 34.38656659, 34.50638041, 34.62619424,

```
        34.74600806, 34.86582188, 34.9856357 , 35.10544952, 35.22526334,
        35.34507716, 35.46489098, 35.5847048 , 35.70451862, 35.82433244,
        35.94414627]),
    )
```

0] :



In []:

No comments... yet

Dare to be the first?

Comment

Share