Credit Task 8.2C: DGA domain name clustering

Task description:

According to Wikipedia, **Domain generation algorithms** (DGA) are algorithms seen in various families of malware that are used to periodically generate a large number of domain names that can be used as rendezvous points with their command and control servers. The large number of potential rendezvous points makes it difficult for law enforcement to effectively shut down botnets, since infected computers will attempt to contact some of these domain names every day to receive updates or commands. Domains generated by dictionary DGA tend to be more difficult to detect due to their similarity to legitimate domains. The following are some sample domains:

V6PNSC80LL.COM
B9U5R3RJMPP.COM
YM5R99EX5Q8.COM
MBSIGLGFQIH2.COM
GSJZNQCOHIKO.COM
VEG2671WMX88.COM
DLNOYYVQSOZHH.COM
BFZFLQEJOHXMQ.COM
AJFSZWOMNHDFCYY.COM
EXAGQLXTMOPSFT8.COM
FWOGZPAGLGOVLIMY.COM

JVRRMMKYEJDEYLCQ.COM
LKLHJONIUDKKHCWO.COM
CADDBSGSCNYDZOH5F.COM
CEUNNFOHGWJYAUA9H.COM
NQZHTFHRMYMTVBQJE.COM
OVLREWGRHHVAJBOTX.COM
OTPWFJOKPOZOOMNK2O.COM
CNEISZDKHZEKQEUBUT.COM
EMUXMJDBTNWCQRFN0G.COM
OWASALWIGURWYVNNPV.COM
PMNYPARTDBVYHCZDJS.COM

In this task, you are given legitimate domain names and DGA domain names and try to use k-Means to cluster them.

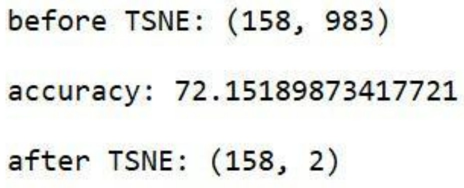
You are given:

- The top 100 legitimate domains provided by www.alexa.com site, saved in “Top-100.csv”
- Cryptolocker family DGA domain names, saved in “dga-cryptolocke-50.txt”
- post-tovar-goz family DGA domain names, saved in “dga-post-tovar-goz-50.txt”
- Code that pre-processes these data files has been provided in task8_2.py (using CountVectorizer with 2-gram)
- Other settings of your choice

You are asked to:

- use KMeans(n_clusters=2, random_state=170) to fit and predict the pre-processed data
- use TSNE(learning_rate=100, random_state=170) to reduce dimensionality to 2 to visualize the clusters found
- print the data shape before and after TSNE
- print clustering accuracy using np.mean(y_pred ==y)*100

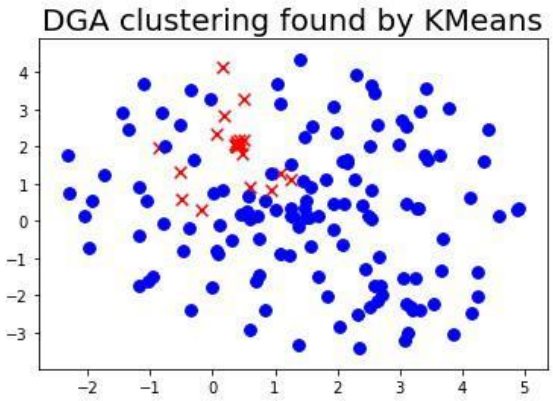
Sample output as shown in the following figures are **for demonstration purposes only**. Yours might be different from the provided.



```
before TSNE: (158, 983)

accuracy: 72.15189873417721

after TSNE: (158, 2)
```



Submission:

Submit the following files:

1. Your program source code (e.g. updated task8_2.ipynb)
2. A screen shot of your program running

Check the following things before submitting:

1. Add proper comments to your code