Phishing webpages, also referred to as “phishs”, are fake webpages that try to mimic target websites and deceive users into disclosing personal data or credentials. It is therefore a major security concern on the web.

Phishers aim to create believable phishing webpages, and for that they can use content taken directly from the target website – outgoing links pointing to the target website and keywords referring to the target (in different elements of the phishing webpage e.g., title, text, images, links). They will thus use relatively little content that they themselves host.

Most of the solutions today rely on static features, primarily taking the bag-of-words approach. This requires a huge amount of labeled data to train classification models and is language- and brand-dependent solution, and thus not very effective at identifying new phishing webpages, targeting brands that were not previously observed.

There are several “levels” in which phishs and legitimate webpages differs from each other, among them are the lexical composition of URLs, term usage consistency, usage of starting and landing mld, RDN usage and webpage content. Based on that, Marchal et al. suggested a list of features that can be used to determine if a webpage is a phish or not.

Here, I chose to focus on the URLs to create a relatively simple model, as proof of concept for the ability to distinguish between phishs and legitimate webpages. I chose ULRs for several reasons:

1. Phishing URL and domain name obfuscation techniques tend to produce long URLs composed of many terms.
2. URLs are relatively easy to extract and are relatively abundant in a webpage.
3. In Marchal et al., that relied on data from all the above-mentioned differences, URLs were the most significant, consisting 106 out of 212 total used features.

While working on this direction I identified a few difficulties and questions:

1. Phishs are identified and removed relatively fast, so to have enough data it could be useful to frequently check websites like “phish-tank” and extract the data while the phish is still active, to refine and improve the model.
2. The data extracted to train the model included only URLs available on the homepage, this is since they seem to be the most relevant, as phisher will often use links to the target webpage. It is however a question worth exploring – can we extract more **relevant** data if we “go deeper” into the link-chain of the outgoing links.

To create the classifier, I trained an XGboost model on 49 features extracted from 102 URLs:

- Protocol – http/https/other (1/2/3 value)
- Length of mld
- Length of URL
- Length of FQDN
- Count of dots in URL
- Count of terms in URL
- Count of terms in mld

For the main URL I extracted all the above listed features – 7 features. For the external and internal URLs I calculated the mean, median and standard deviation, resulting in
2 * 7 * 3 = 42 features and a total of 49 features on which the model was trained.

Once the model was trained, I tested its performance with features I extracted from a test set (35 URLs): the MSE was $1.145*10^{-9}$.
As a sanity check I tested the performance of a non-trained model, with the same test set, and got a MSE of 0.0571. At first glance it seemed that the trained model is not doing much better than the non-trained one, but a closer check of the prediction vector reveals it always predicts that the webpage is legitimate. Because there are much more legit pages in the data, the MSE is small.