View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Technical Disclosure Common

Technical Disclosure Commons

Defensive Publications Series

November 29, 2018

MONITORING OF FEATURE-DISTRIBUTION SHIFT IN PRODUCTION ENVIRONMENT

Jan Brabec

Lukas Machlica

Cenek Skarda

Follow this and additional works at: https://www.tdcommons.org/dpubs_series

Recommended Citation

Brabec, Jan; Machlica, Lukas; and Skarda, Cenek, "MONITORING OF FEATURE-DISTRIBUTION SHIFT IN PRODUCTION ENVIRONMENT", Technical Disclosure Commons, (November 29, 2018)
https://www.tdcommons.org/dpubs_series/1726

This work is licensed under a Creative Commons Attribution 4.0 License.

This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

MONITORING OF FEATURE-DISTRIBUTION SHIFT IN PRODUCTION ENVIRONMENT

AUTHORS:

Jan Brabec
Lukas Machlica
Cenek Skarda

ABSTRACT

A fundamental problem faces with network traffic is that the traffic is dynamic and evolves over time. Presented herein is a tool for monitoring feature distribution shift and possible efficacy implications. The tool takes advantage of the fact that labels are available on the training set, but not available in the production environment where the labelling cannot be done on the fly. The tool implicitly provides estimates of feature robustness to time shift for each feature.

DETAILED DESCRIPTION

The concept of data-driven classification of network traffic to different malware classes is important to a number of different types of network traffic analytics. However, a fundamental problem associated with data-driven classification of network traffic is that network traffic is dynamic and evolves over time. For example, new domains are registered, domains expire, popularity/usage of domains changes, *etc*. Presented herein are techniques to detect shifts in distribution of features upon which a machine learning model operates, and for monitoring how the shifts affect the model once it is run in a production environment. A trivial solution could be to monitor the number of false alarms, correct and missed predictions, and relate the shift to actual efficacy of the system. However, this is not feasible, because it is not possible to label each individual network request to decide which requests are truly malicious. Moreover, this would not provide any additional insight in possible robustness of utilized features, which can be used once a new version of a classifier is trained so that, for example, features extremely sensitive to time shift will be removed from the training process.

Presented herein is a monitoring tool designed to monitor possible drops in efficacy related to distribution shift of features in relation to the deployed classifier. The techniques

presented herein use quantile based binning of feature values and availability of labels on the training set to generate a simple, yet valuable, insight into these feature distribution shifts.

For ease of description, the techniques presented herein are described in the context of network traffic analysis. However, it is to be appreciated that the techniques presented herein are also easily applicable to other domains with similar properties.

Short Introduction to Binning

Binning is a common practice in majority of training algorithms, such as decision or random forests. It decreases the effective number of unique feature values to a predefined size. For example, it is assumed that one-hundred (100) input numbers are uniformly sampled on interval (0, 1), a binning with four (4) equally-sized bins with lower and upper boundaries (0, 0.25), (0.25, 0.5), (0.5, 0.75), (0.75, 1) is applied, and each input number is represented by the center of the respective bin in to which it falls. This will lead to approximately a quarter ($1/4^{th}$) of the numbers having the value 0.125, a second quarter of the numbers having the value of 0.375, a third quarter of the numbers having the value 0.625, and the last quarter of numbers having the value 0.875.

Quantile based Binning

Continuing with the above example with 100 input numbers, the histogram of feature values for these 100 numbers would be flat only when the numbers are uniformly sampled. However, if the boundaries would be given by the 1st, 2nd and 3rd quantile, i.e. (0, 1st-q), (1st-q, 2nd-q), (2nd-q, 3rd-q), (3rd-q, 1) so that the bins are not necessarily placed equidistantly (i.e., each may have different widths), then the histogram will become flat again, as illustrated in the example in the Figures 1A-1C, below. The quantile binning is often preferred in machine learning over uniform binning because it focuses on densely occupied parts of the feature space.

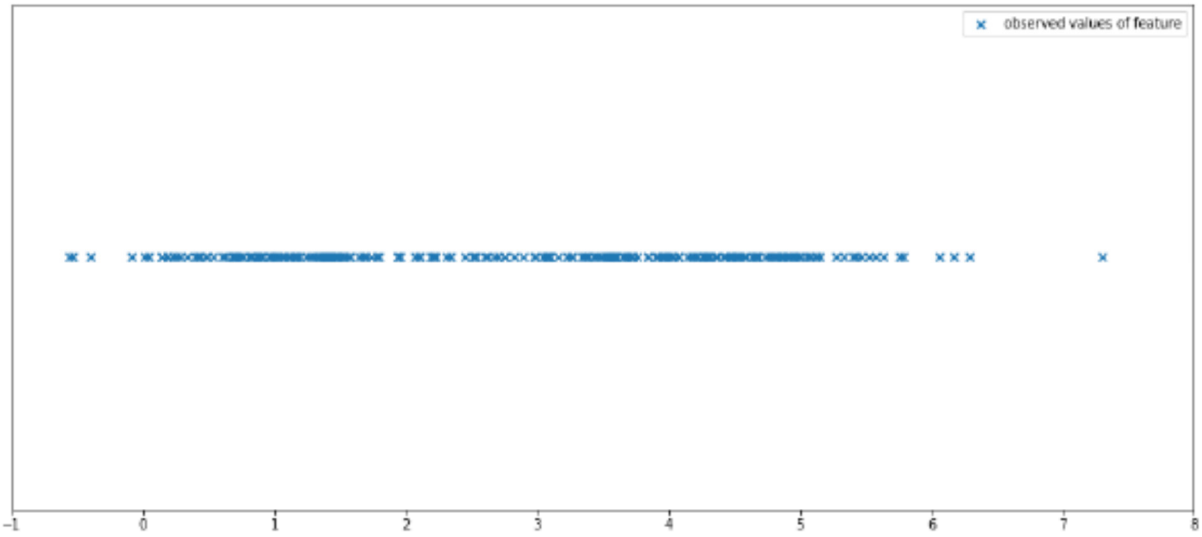

Figure 1A

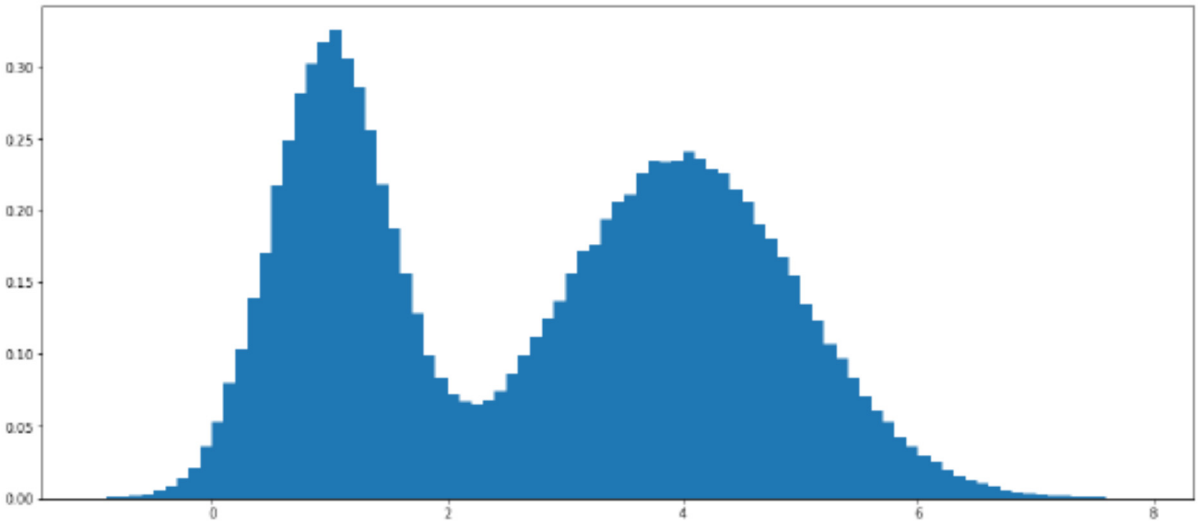

Figure 1B

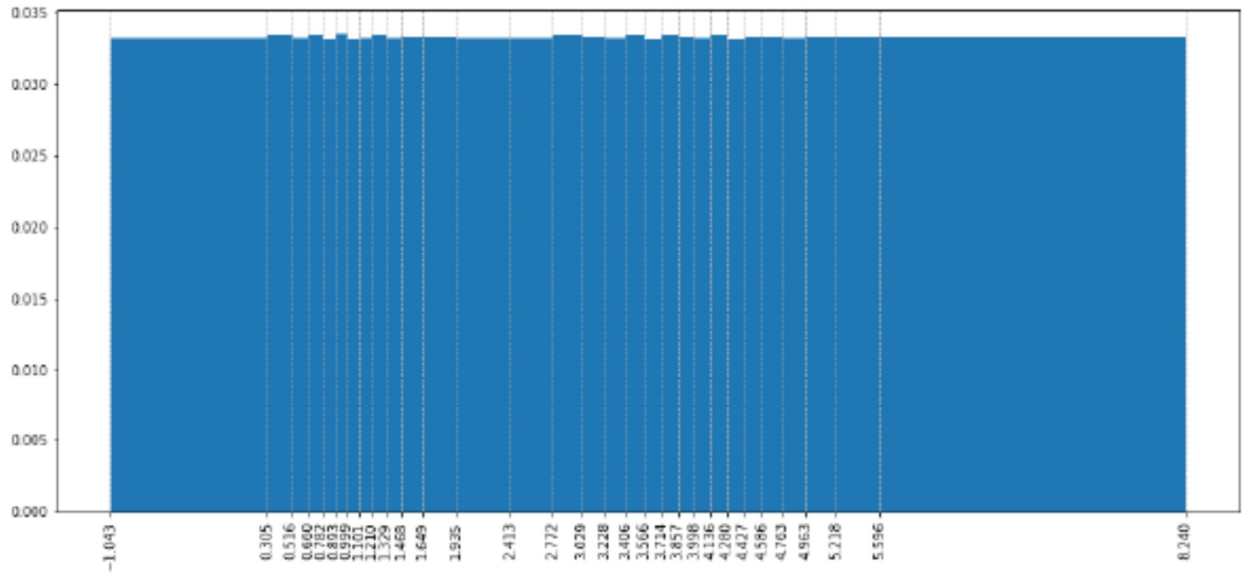

Figure 1C

Monitoring Tool Details

The facts described in the previous sections are used for the monitoring tool description, as follows. In particular, the "development dataset" is divided into a "training dataset" and a "validation dataset" and quantile binning is performed on the training dataset. Additionally, one histogram is computed for each feature: e.g., one histogram for the distribution of transferred bytes between client and server, another histogram for duration of connections in the training dataset, *etc*. The classifier is then trained on the binned feature set and, therefore, the classifier and the training feature value distributions are coupled. Moreover, according to the discussion above, the histogram of the feature values for the training set is flat/uniform. The monitoring tool, updated with the centers of the bins distributed according to the quantiles computed on the training set, is then deployed simultaneously with the model.

In general, the monitoring tool takes advantage of the fact that labels are available on the training dataset, but not available in the production environment where the labelling cannot be done on the fly. The monitoring tool can then perform a number of tasks. For example, one task of the monitoring tool is to continuously update the histogram of feature values on data processed in production environment for each feature.

Another task of the monitoring tool is to, in time intervals (e.g., once per day) and for each feature, compare the actual histogram to the flat histogram (e.g., to the distribution of values in the training set). As described below, this can be done for example by computing the information entropy of the actual histogram.

Another task of the monitoring tool is to, for each feature with similarity between the histograms (e.g., entropy of actual histogram) lower than a threshold:

- Adjust the distribution of the feature values in the validation set to match the distribution obtained in production environment, this can be done easily by proper scaling of each feature value to match the desired distribution (probability mass redistribution).
- Perform classification on the adjusted training set.
- Report decrease in efficacy on the training set (this is motivated by the concept of permutation feature importance), and eventually, raise alarm if the decrease is significant (based on user pre-specified threshold).

Shown below are Figures 2A, 2B, and 2C. Figure 2A depicts a possible distribution shift of feature values from Figure 1. In Figure 2B, the red area 200 relates to the transformed distribution according to the distribution shift from the first image, shown with equally sized bins. In Figure 2C, the transformed values according to the distribution shift are shown in the lower part (orange) 201, while the upper part (blue) 202 illustrates the original samples from Figure 1).

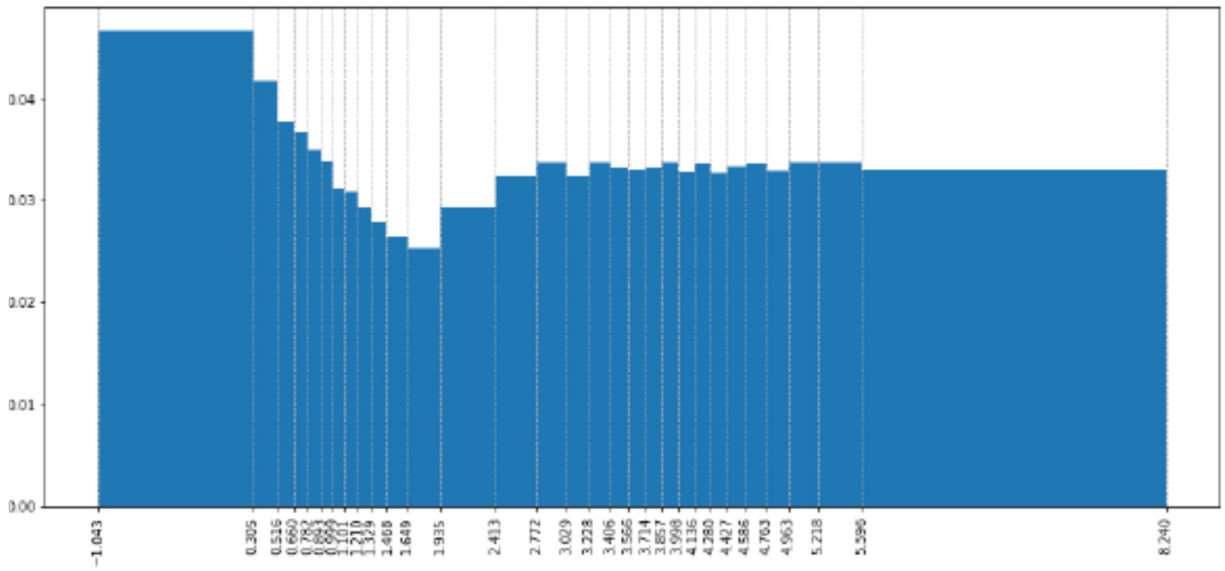

Figure 2A

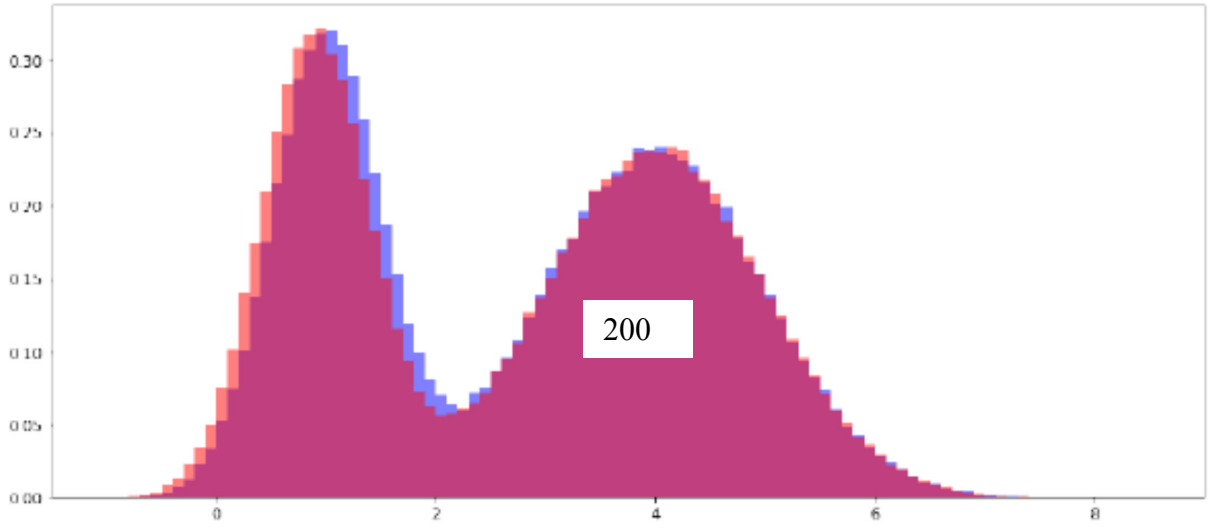

Figure 2B

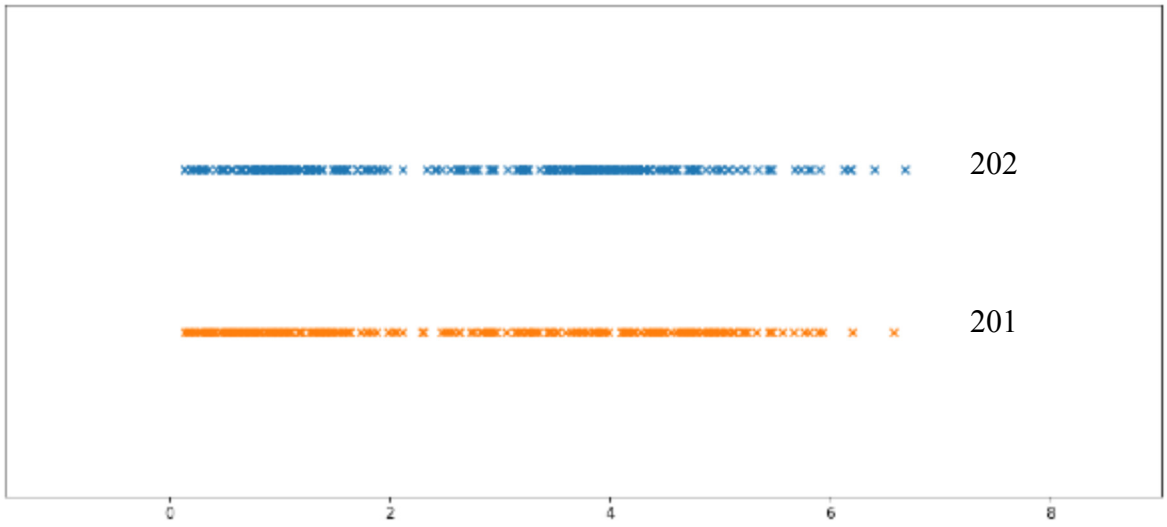

Figure 2C

Figures 3A, 3B, and 3C, below are similar to Figures 2A, 2B, and 2C, respectively except that the distribution shift is more significant.

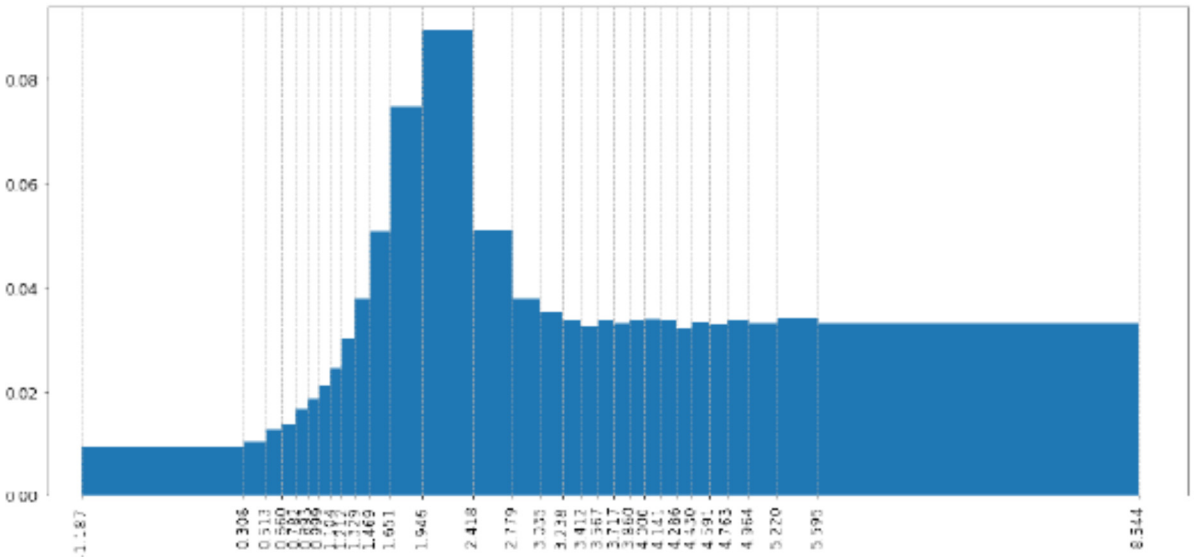

Figure 3A

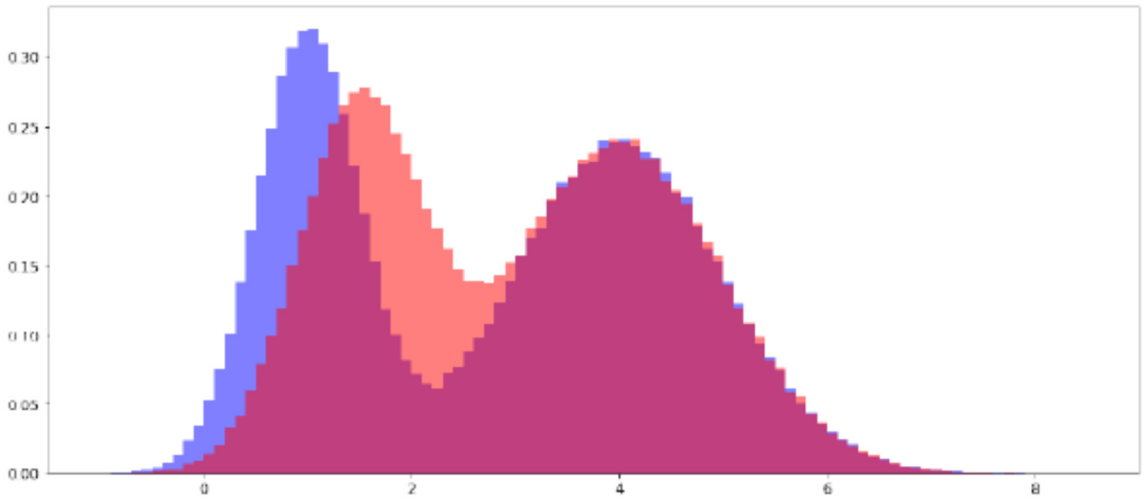

Figure 3B

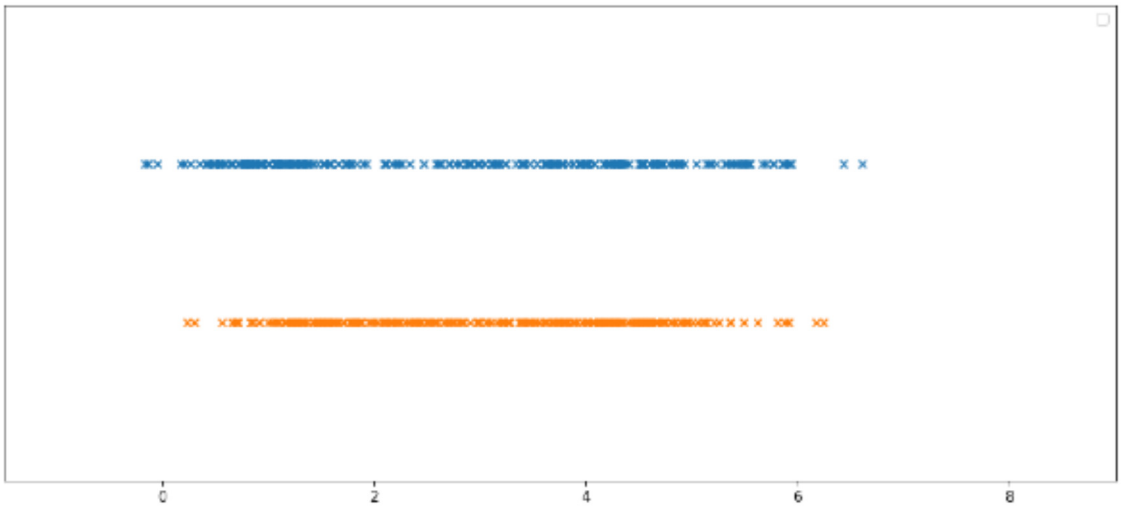

Figure 3C

The techniques presented herein implicitly provide estimates of feature robustness to time shift for each feature. It is given by the decrease of the efficacy of the system, when the distribution of feature values for a given feature is changed at some level of entropy (distance from the training distribution). The lower the decrease at a given level of entropy, the more robust is the feature.

For flat (uniform) distribution, the information entropy is maximal. That is, having a flat histogram with N bins with equal heights normalized so that the sum across all the bins is 1, the entropy is equal to: sum{n=1..N} (-1/N log (1/N)) = log(N). If a logarithm with base N is chosen, the entropy for the flat histogram will be equal to 1, while the lower

bound for entropy of discrete variables is 0. This will provide similarity score ranging in the interval (0, 1) with 0 being very different from training distribution (all the feature values are focused in a single bin) and 1 when the distributions are equal. The height of the bin is not important, only the fact that the shape of the histogram differs from being flat/uniform (i.e. does not have highest entropy).