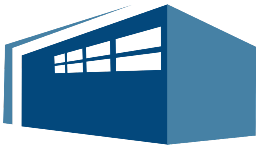
Implementation project
Lab Machine Learning for Data Science
Summer Semester 2024

G. Montavon
Institute of Computer Science
Department of Mathematics and Computer Science
Freie Universität Berlin

Lab ML for Data Science: Part I
Getting Insights into an Unsupervised Dataset



The goal of Part I of the project is to extract insights from a purely unsupervised dataset. We will focus on the following dataset:

UCI Wholesale customers dataset
`https://archive.ics.uci.edu/ml/datasets/Wholesale+customers`

This dataset collects the annual spending on different product categories by a collection of wholesale customers located in Portugal. Specifically, we are interested in identifying general or specific patterns about wholesale customers, for example, instances with anomalous spending behavior, or clusters of similarly behaving wholesale customers. In particular, we would like to leverage unsupervised ML techniques to identify anomalies and clusters.

In contrast to a pure prediction approach, the goal will be to go beyond the mere detection of anomalies and to provide in addition an explanation, for example, why specific instances are predicted to be anomalous. Often, the explanation is of greater interest than the prediction itself because it identifies the causes of the anomalies and allows meaningful action to be taken to address these causes.

Additionally, it is also important in a data science context, that the outcomes of the analysis are reproducible. For example, a lab that does not possess exactly the same data should ideally arrive at the same conclusions if it follows roughly the same methodology. To ensure the reproducibility of the outcomes, we will consider introducing mechanisms in the ML approach that favor robustness to resampling.

1 Loading the Data, Preprocessing, Initial Data Analysis

The first step will consist of downloading the dataset and converting it into numerical tables (e.g. numpy arrays). In practice, raw data is rarely directly usable as input to machine learning algorithms. In particular, there may be substantial heterogeneity between the different input features. Some features may be physical measurements, monetary measurements, while others may be category indicators or even non-numerical data such as text or images. Hence, a preliminary filtering of what is interesting for the analysis we would like to conduct is desirable.

In the context of the UCI wholesale dataset, one may, for example, want to base anomaly and cluster predictions on numerical data (annual spending per category) and drop meta-data such as Channel and the Region indicators or reserve it for an ulterior use. Once such a preliminary step has been taken, we have a standard dataset of size $N \times d$ where N is the number of instances (wholesale customers), d is the number of spending categories, and each value in the table can be expressed in monetary unit.

To verify the range and distribution of these values, we can generate some basic statistical visualizations of the data. This includes histograms showing for each category the distribution of spendings, or scatter plots showing the correlation between different product categories. A common observation from such basic statistical analysis is that the distributions are heavy tailed, with many instances having rather small spendings, whereas a few may have spendings one or two orders of magnitude above. Any anomaly detection algorithm would systematically highlight those high spenders as anomalous and not make a distinction between spending little and not spending at all.

To address this issue, it can be useful to apply some nonlinear transformation to the data, for example, applying the log function to the features so that the distribution becomes compressed for large values and expanded for small values.

$$x \mapsto \log(x + \theta)$$

Here, we add a positive offset θ in the logarithm so that zero spending ($x = 0$) does not get mapped to $-\infty$. You may experiment with different offsets, such as $\theta = 1$, $\theta = 10$ or $\theta = 100$. To verify the effect of the transformation, you can recompute the histograms and scatter plots in transformed space and check visually whether the transformation had the desired result, e.g. whether the distribution look Gaussian-distributed.

2 Detecting Anomalies

We now focus on a question of interest, specifically, whether there are anomalous instances in the given data. A basic form of anomaly detection is to verify for each data point whether it has neighbors. A simple measure of anomaly can then be given by the distance (or squared distance) to the nearest neighbor. For example, given some test point with index j in the dataset, one can get its outlier score by performing a minimum over the $N - 1$ remaining points in the data, i.e.:

$$\begin{aligned} z_{jk} &= \|\boldsymbol{x}_j - \boldsymbol{x}_k\|^2 \\ y_j &= \min_{k \neq j} z_{jk} \end{aligned} \tag{1}$$

With such anomaly scores, data points can now be ranked from most to least anomalous. For example, one may now be able to extract a top-10 list of the most anomalous instances in the dataset, and print for these 10 instances their recorded spending across categories.

2.1 Robust anomaly models

In practice, however, Eq. (1) may not identify top outliers in a way that is sufficiently *reproducible*. Imagine, for example, a point $\boldsymbol{x}_j$ whose nearest neighbor, call it $\boldsymbol{x}_i$ is at a distance of 1 but the second nearest neighbor is at a distance of 10. If the nearest point $\boldsymbol{x}_i$ was not in the dataset (e.g. due to slight variations in the data collection process), the outlierness score of $\boldsymbol{x}_j$ would have changed drastically. An efficient way of addressing this problem is to reconsider the notion of outlierness by considering a point to be an outlier based on multiple neighbors. In other words, a point may still be an outlier even if (by accident) some other point in the data shares roughly the same values.

Such a redefined notion of outlierness can be implemented by replacing the hard minimum in the original equation by a soft minimum, i.e.:

$$y_j = \mathrm{soft}\min_{k \neq j} \{z_{jk}\} \tag{2}$$

with

$$\mathrm{soft}\min_{k \neq j} \{z_{jk}\} = -\frac{1}{\gamma} \log \Big(\frac{1}{N-1} \sum_{k \neq j} \exp(-\gamma z_{jk}) \Big).$$

As a side note, the softmin can be interpreted as a generalized F-mean with $F(t) = \exp(-\gamma t)$. The output of such function is also related to the log-likelihood prediced by a kernel density estimator of the rest of the data (see e.g. [2]).

2.2 Selecting a suitable parameter γ

To verify the gain in terms of reproducibility one can gain from using the softmin approach to detect anomalies, one can apply the bootstrap method in statistics. Bootstrap consists of simulating multiple variants of the same datasets by randomly sampling instances (with repetition) from the original dataset. The anomaly scores can then be computed for each variant of the dataset, and for each instance, one can characterize its anomalousness by two values, its average and its spread. In other words, the anomaly of an instance is modeled, not by a single value y_j, but by a random variable y_j of mean μ_j and spread σ_j.

A large value for the parameter γ of the softmin anomaly model of Eq. (2) may produce stark differences between the μ_j's of anomalous and non-anomalous instances, but also large spreads σ_j's within each instance. Conversely, a small value of γ can reduce these spreads, but also at the cost of reducing differences between instances. In practice, a neither too small nor too large value of γ is likely to achieve the maximum separability between anomalous and non-anomalous instances.

To find a suitable parameter γ, you may for example plot for each instance its anomaly average and spread, repeat the plot for different parameters γ, and then chose the parameter γ that produces the best separability on a visual level. Even better, you may formalize this visual intuition into some evaluation metric based on which you can derive an optimal parameter γ. Note that once you have selected a parameter γ, you can recompute a single anomaly model with this parameter γ on the whole data. This single model can then be used for the analyses in the next section.

3 Getting Insights into Anomalies

In a data science setting, only detecting anomalies may be of limited interest. While it helps the scientist to answer the basic question of whether there are anomalies in the data, one may wish for deeper insights, for example, whether anomalies are under/over-represented among specific types of retailers, or, for specific instances, which features (i.e. spending categories) contribute predominantly to instances being anomalous.

3.1 Relation Between Anomalies and Meta-Data

We may be interested to know whether different subsets of the data (e.g. retailers of a specific type or in a particular geographical locations) have more or less anomalies on average. Such insights can be obtained by taking into consideration the meta-data. Your analysis could take the form of plotting the distribution of anomaly scores on different subsets of data, where these subsets are determined based on the meta-data. You may then look for differences in distribution.

3.2 Indentifying Input Features that Drive Anomaly

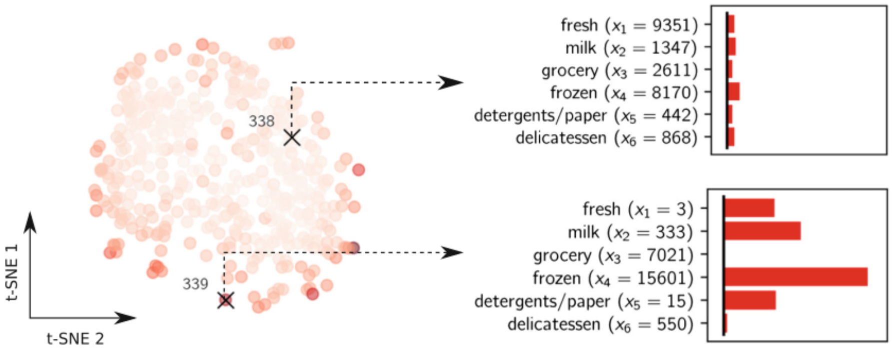
Identifying relevant input features can be achieved using a set of techniques commonly referred to as Explainable AI. We consider in particular the Layer-wise Relevance Propagation method (see [1] for an overview). In the context of the anomaly detection models considered here, attribution of anomaly scores to the input features can be achieved in two steps [2]: First, one identifies to what extent each data point k other than j has contributed to the anomaly score of instance j:

$$R_k^{(j)} = \frac{\exp(-\gamma z_{jk})}{\sum_{k \neq j} \exp(-\gamma z_{jk})} \cdot y_j \tag{3}$$

The variable $R_k^{(j)}$ indicates the contribution of data point k to the anomaly score of instance j. Then, these scores can be propagated back to the input features by observing that the (squared) Euclidean distance entering the anomaly score can be decomposed in terms of individual components:

$$R_i^{(j)} = \sum_{k \neq j} \frac{[\boldsymbol{x}_k - \boldsymbol{x}_j]_i^2}{\|\boldsymbol{x}_k - \boldsymbol{x}_j\|^2} \cdot R_k^{(j)} \tag{4}$$

The variable $R_i^{(j)}$ indicates the contribution of input feature i to the anomaly score of instance j. Implementing the attribution process described in Equations (3) and (4), and applying it to each data point enables us to build for each data point a histogram of the type one can see in the image below:



Note that Eqs. (3) and (4) embed the conservation property $\sum_{i=1}^{d} R_i^{(j)} = y_j$. This allows for interpreting each produced score as the share of the anomaly score that is explained by the associated input feature. For technical purposes, it can also serve as a unit test to verify the correct implementation of these propagation rules.

References

[1] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller. *Layer-Wise Relevance Propagation: An Overview*, volume 11700 of *Lecture Notes in Computer Science*, pages 193–209. Springer, 2019.

[2] G. Montavon, J. R. Kauffmann, W. Samek, and K.-R. Müller. *Explaining the Predictions of Unsupervised Learning Models*, volume 13200 of *Lecture Notes in Computer Science*, pages 117–138. Springer, 2020.