Deployment of unsupervised learning in a search for new physics with ATLAS Open Data

Analysis of two lepton final state from OpenData at ATLAS

Sakarias Frette

Spring 2022

Abstract

University of Oslo

CONTENTS

I. INTRODUCTION

II. THEORY

Some of theory sections about anomaly detection and machine learning algorithms are based on previous work done in other courses, such as [3] and [2].

A. Anomaly detection

Anomaly detection is a tool with a wide range of uses, from time series data, fraud detection or anomalous sensor data. Its main purpose is to detect data which does not conform to some predetermined standard for normal behavior. The predetermined standard varies from situation to situation, and can be set by the context it self, and what is expected as an anomaly. We typically classify anomalies in three categories[1]:

1. Point anomalies
2. Contextual anomalies
3. Collective anomalies

For the purpose of this report we will mostly consider collective anomalies, as anomalies in the standard model has to be collective to claim anything due to noise etc.

In high energy physics we can using machine learning separate anomaly detection into two categories, supervised and unsupervised searches.

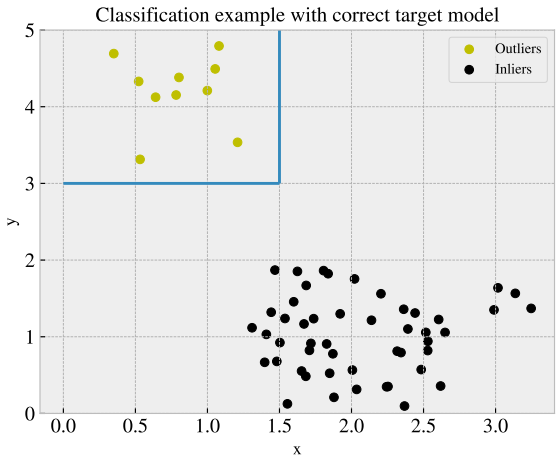

Figure 1: Example of supervised classification of anomaly where the target model is correct. Here the machine learning model manages to correctly identify the anomalies.

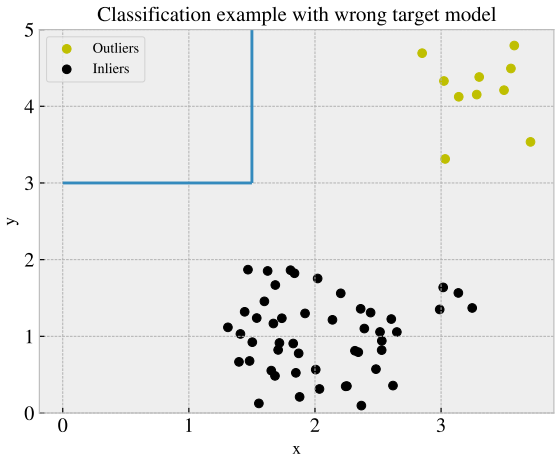

Figure 2: Example of supervised classification of anomaly where the target model is wrong. Here the machine learning model manages to wrongly identify the anomalies.

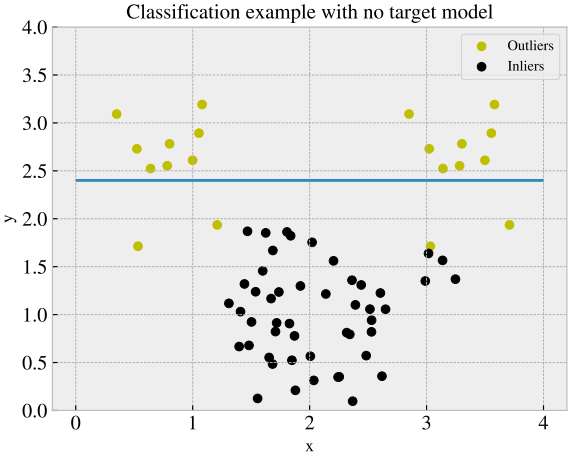

Figure 3: Example of unsupervised classification of anomaly with no target model. Here the machine learning model manages to with good precision identify the anomalies.

B. Stacked auto encoder

One method to attack the anomaly detection problem is the so called (stacked) auto encoder. The idea is based on reconstruction, and has been implemented for denoising of images, image compression, and anomaly detection. An auto encoder is a subgroup of feed forward neural networks, and the goal is to compress the information into fewer variables, called the latent space, which then can explain much of the data through decoding that information. This allows the algorithm to learn the most important components of the data. An

illustrative image of an auto encoder is shown below.

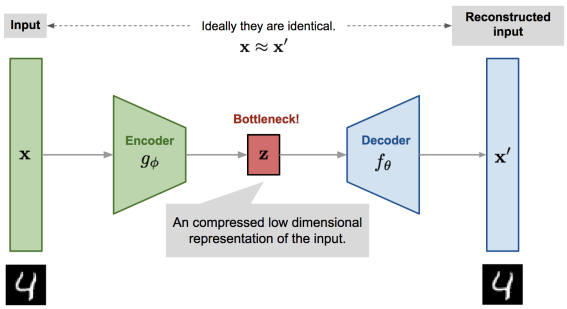

Figure 4: The architecture of an autoencoder with image reconstruction as the example use. Source, accessed 28.04.22

We see here in figure 4 that the input data is deconstructed through an encoder to a lower dimensional space called the latent space, depicted here as **z**, and then reconstructs the data with the decoder. It is important to note that the number of layers, nodes per layer and activation function per layer for the encoder does not need to match the structure of the decoder. The only requirement is that the input and output layer has the same shape. The end result is the reconstructed data **x′**. The training of the model is parameterized in the following way. We define the decoder as

$$\mathbf{z} = g_\xi(\mathbf{x}),$$

and the reconstructed information as

$$\mathbf{x'} = f_\theta(g_\xi(\mathbf{x})),$$

where the parameters (ξ, θ) are tuned to reconstruct the data as close to the original data as possible. The model then adjusts following the simple mean squared error of the reconstruction and the actual data, given below

$$L_{AE}(\xi, \phi) = \frac{1}{N}\sum_{i=0}^{N-1}\left(\mathbf{x}^i - f_\theta(g_\xi(\mathbf{x}^i))\right)^2$$

C. Monte Carlo simulated data

D. Beyond standard model physics

III. IMPLEMENTATION

A. Handling of data

In appendix **??** the features used in the data set are listed, with their respective C++ type and description.

1. *Cuts and pre-selection*

To preserve as much information and add as little bias as possible we did as few cuts as possible. The entire data gathering process is biased from the moment the data is collected at the collider, but even then it is good to try to minimize the bias.

The first cut we impose is to require only events with two leptons. We also require oppositely charged leptons, and that we get lepton pairs that are allowed, either two muons or two electrons. Then we impose cuts to filter out "good leptons".

2. *Choice of features*

- Which features to pick, amount etc
- How to handle features with no value/nan value (preserve laws of physics)

B. Tuning and training

IV. RESULTS AND DISCUSSION

A. Semi unsupervised

- Difference between brad and narrow search
- Bias
- bias with features to choose
- difference of how well the model detects different signal models in mc data signals
- any interesting in actual data?
- difference in scaling
- tuning effect

V. CONCLUSION

REFERENCES

[1] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly Detection: A Survey". In: *ACM Comput. Surv.* 41 (July 2009). DOI: 10.1145/1541880.1541882.

[2] Sakarias Frette and William Hirst. "Seaching for Higgs through supervised and unsupervised machine learning algorithms". In: (2022). URL: https://github.com/WilliamHirst/Advanced-Machine-Learning/blob/main/article/Project_1.pdf.

[3] Sakarias Frette, William Hirst, and Mikkel Metzch Jensen. "A computational analysis of a dense feed forward neural network for regression and classification type problems in comparison to regression methods". In: (2022), p. 5. URL: https://github.com/Gadangadang/Fys-Stk4155/blob/main/Project%202/article/Project_2_current.pdf.

Appendix A: Features

Table I: Features and their description

Feature	C++ Type	Description
lep_n	int	number of pre-selected leptons
lep_truthMatched	vector<bool>	boolean indicating whether the lepton is matched to a simulated lepton
lep_trigMatched	vector<bool>	boolean indicating whether the lepton is the one triggering the event
lep_pt	vector<float>	transverse momentum of the lepton
lep_eta	vector<float>	pseudo-rapidity, η, of the lepton
lep_phi	vector<float>	azimuthal angle, ϕ, of the lepton
lep_E	vector<float>	energy of the lepton
lep_z0	vector<float>	z-coordinate of the track associated to the lepton wrt. primary vertex
lep_charge	vector<int>	charge of the lepton
lep_type	vector<int>	number signifying the lepton type (e or μ)
lep_isTightID	vector<bool>	boolean indicating whether lepton satisfies tight ID reconstruction criteria
lep_ptcone30	vector<float>	scalar sum of track pT in a cone of R=0.3 around lepton, used for tracking isolation
lep_etcone20	vector<float>	scalar sum of track ET in a cone of R=0.2 around lepton, used for calorimeter isolation
lep_trackd0pvunbiased	vector<float>	d0 of track associated to lepton at point of closest approach (p.c.a.)
lep_tracksigd0pvunbiased	vector<float>	d0 significance of the track associated to lepton at the p.c.a.
met_et	float	transverse energy of the missing momentum vector