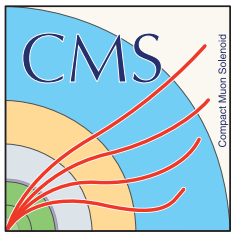
CERN Summer Student Report 2019

Machine Learning Based Outlier Detection for Data Certification



Patomporn Payoungkhamdee *

EP-CMG-CO

Supervisor: Marcel Andre Schneider †

A report submitted in partial fulfilment of the requirements
for the CERN summer student programme 2019

August 22, 2019

*Mahidol University
†European Organization for Nuclear Research (CERN)

Abstract

Compact Muon Solenoid (CMS) detector was built in the middle of collision from Large-Hadron Collider (LHC) which is one of the most powerful particle accelerators in the world. The mission is to collect the product from collision and decaying which happens 40 million times each second. The data taking in the CMS experiment is reconstructed to become a physics quantity 48 hours after a collision. The certification of data quality is made on run and lumisection levels. The criteria to certify are both from an automatic system as well as manual work from untraceable misbehaving of detector which is marked by offline shifter and detector experts. Approximately 95% of data are good and the rest of them are bad. It is not easy to say that all phenomena that cause misbehaving of a result are well understood. Then the aim of this work is to reduce the manual work for data qualification by exploding various types of semi-supervised learning by treating the outlier as bad in lumisection granularity.

Acknowledgement

- CERN Summer Student program 2019
- Especially
  - Marcel Andre Schneider
  - Francesco Fiori
  - Kaori Maeshima
  - Javier Fernandez
  - Adrian Alan Pol
  - Countless CMS DQM people
- GPU resources from IBM in collaboration with CERN Openlab

Contents

List of Figures

Chapter 1

Overview

Before the whole data be feeding to physics analysis, there is a procedure to certify the data quality in run and lumisection granularity. Data quality monitoring (DQM) team provides the tools and workflow where there is an offline and online section to consider which basically online is a real-time and the data that we get 2 days after collision would be inspected in offline section. The person who looking at a dozen of a histogram that demonstrates the occupancy of the detector in each sub-system and also the physics quantities in various perspectives of information. Most of Bad lumisection (LS) are automatically came from run tagged as bad by a human for the whole run and DCS bits in lumisection levels. In some cases, there is a small fraction bad LS that are manually marked as bad by data certification experts in lumisection levels because some kind of detector malfunction isnt traceable from the previous process. On the other hand, good LS is defined in Golden JSON which is literally taken from data that passes of those bad criteria.

The main objective of this work is to find a mathematical way to certify data quality lumisection granularity to reduce the manual work of data certification experts. As it can be seen that there are only a small fraction of data. Moreover, the bad data that are marked by the expert are relatively small compare to good data. Then we have to face to the imbalanced class problem and it is the reason why we choosing semi-supervised learning where we feed only good data to train the unsupervised model and testing with both kinds of data. Consequently, the data that are marked as bad by the model would be considered an outlier where the model does not familiar with. A more explicit analysis would be provided in this report.

Chapter 2

Background

2.1 Compact Muon Solenoid (CMS) Detector

The Compact Muon Solenoid (CMS) detector be designed for collect the collision and decaying data in forward and transverse of the beamline. It has more resolution in the transverse direction from the equipment that has been designed to focus in transverse direction as the Figure 2.1. According to [3], The detector itself consist of different sub-detector for their own purpose where the main components are

The Compact Muon Solenoid (CMS) detector be designed to collect the collision and decay-ing data in forward and transverse of the beamline. It has more resolution in the transverse direction from the equipment that has been designed to focus in transverse direction as Figure 2.3. According to [3], The detector itself consists of different sub-detector for their own purpose where the main components are

1. **Tracker** to trace the footprint of a charged particle by beginning at the hitting of the closest layer which is pixel detector and following by multiple layers of strip detector to gain more precision of particle tracks that correspond to momentum of the particle

2. **Electromagnetic Calorimeter (ECAL)** measure the momentum of leptons (especially electron) and photon where the main interaction is electromagnetic interaction

3. **Hadron Calorimeter (HCAL)** has been designed for measure the energy of hadronic particle where it also has QCD interaction rather than only electromagnatic

4. **Superconducting Solenoid** for generate a nearly-uniform magnetic field inside of the cylindrical shape and charged particle turn their heading around where it propagate in the outside of this radius like a muon track in Figure 2.2

5. **Muon Detectors** is one of the most important sub-detector for measuring the muon momentum and the track of them by taking a footprint of tracking system into account to get more precise information

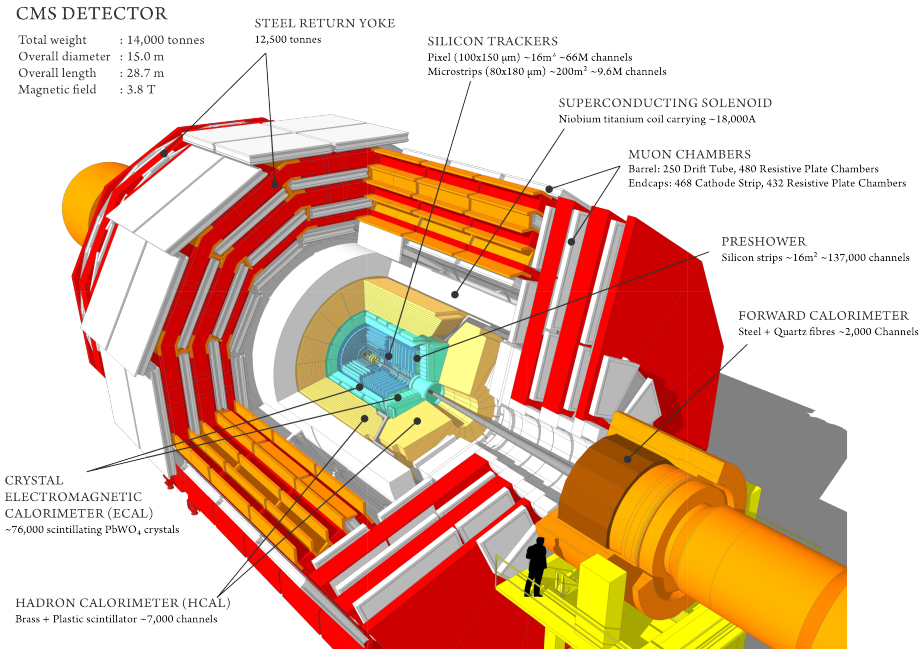


Figure 2.1: *Sectional view of the CMS detector. The LHC beams travel in opposite directions along the central axis of the CMS cylinder colliding in the middle of the CMS detector. Image retrieved from http://cms.web.cern.ch/news/cms-detector-design*

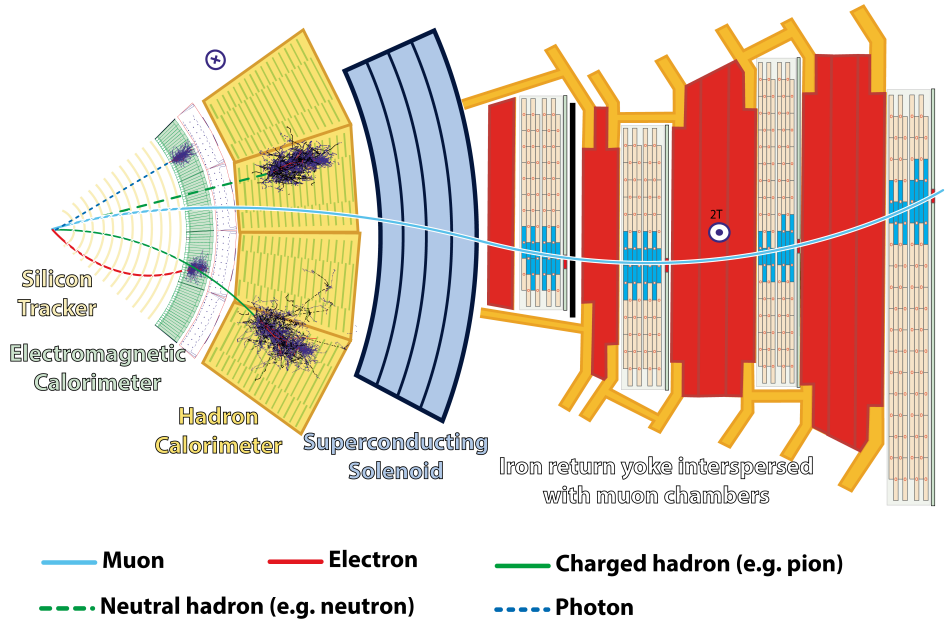


Figure 2.2: *Onion-like crossection of CMS. Image retrieved from [1]*

2.2 Data Acquisition (DAQ)

According to [2], the collision rate takes around 40Mhz (100 Tbyte/s) which is impossible for the instrument to collect all of those signals that happen at a time. The harvested signal

that CMS detector select is determined from low-level trigger where the detector frontend (electronic circuit determination) process to select an interesting signal. The level-1 (L1) trigger filter and produce the signal at 100 kHz (100 Gbyte/s). Then it has been sent to high-level trigger (HLT) where we could start to really see the interpretable physical quantities from here.

2.2.1 CMS Online System

CMS team also provide the tools for automate data acquisition as [7] where the big picture has been demonstrated in Figure 2.3. Apparently, the system still needs some people to double-check by using various tools e.g. Web GUI and so on during the running process of beam collider in LHC from the low-level system to high-level system.

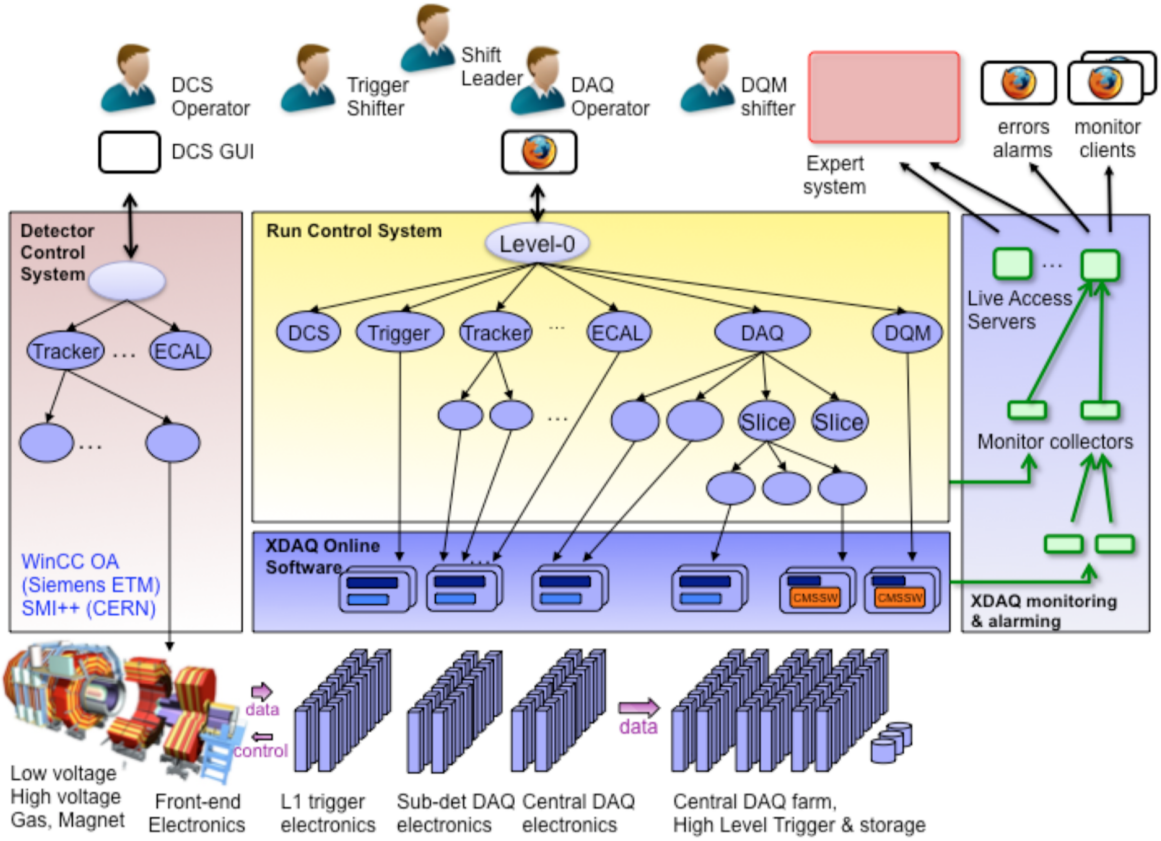


Figure 2.3: *Overview of the CMS online systems. Image retrieved from [7]*

2.2.2 Data Granularity

We decide to divide a big chunk of data into one run and each run contains multiple lumisection which has 23 seconds of interval due to the beam does not change much where we consider 23 seconds of time. Moreover, each lumisection contain multiple events. If we consider in terms of time to reconstruction, there are Express and PromptReco where data are reconstructed at nearly real-time and two days after a collision.

2.3 Data Quality Monitoring (DQM)

In order to make sure that data quality is nearly perfectly well collected, there is another story called DQM where it's actually the subset of the run control system in Figure 2.3. CMS DQM team provides the tool where there is online and offline shifter checking the result from beam collision real-time and 48 hours after collision orderly. Figure 2.4 is the schematic of a DQM workflow include the tools and person who responsible for each module. If some subsystems went weird such as the peak of the histogram drastically increase with no physical sense or some part of the detector turned off, they will report in the log of the system in the running process and calling detector experts to inspect the problem. In this work, we will only focus on the red box which is the offline world.

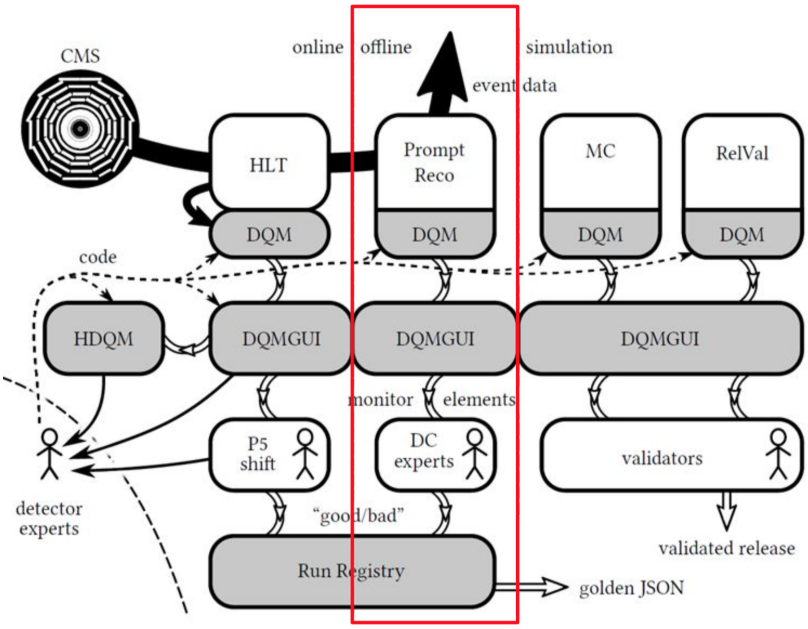


Figure 2.4: *Tools and Processes of DQM. Image retrieved from M. Schneider, CHEP 2018*

Regarding the scope that we want to mimic, offline shifter and detector experts check a multiple distribution histogram to inspect and certify data quality. The certification is made on run and lumisection granularity. The procedure to certify the data are the following list

1. Automatically filter by DCS bits, beam status and etc. (LS levels)
2. Runs tagged as bad by human (whole run)
3. In rare cases are marked by DC experts (LS levels)

Then the rest of them that pass all of those criteria are defined in Golden JSON which are good LS.

Chapter 3

Objectives

- Certify data quality in lumisection granularity
  - Classification on the basis of actual data distributions per LS
- Reduce manual work of DC Experts

3.1 Expectation

The most important aim of this work is to reduce the offline shifter work where we should provide the tools to reduce their work if some data are totally bad or perfectly good and let them inspect only for a few grey zones. In order to separate a kind of data quality, we have to define some decision values from some mathematical models to determine the data quality by finding some threshold to separate them. For the ideal case, we might see the good lumisection that looks perfectly good, bad lumisection that looks terribly skew and the grey zone where some of those two kinds of lumisection are overlapping.

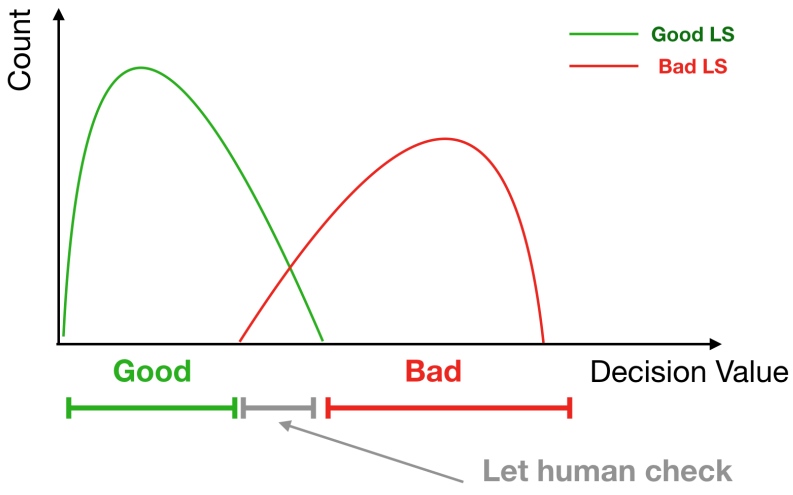


Figure 3.1: *Three possible regions of prediction*

3.2 Proposal For an Alternative Approach

Again, the purpose of this work is not to redefine the certification process but mimic a shifter and reduce their work if there is an obvious case. Then the automatic DCS bit flagging will stay but we apply the algorithm on top of it rather than remove the criteria. The quantity value of each data point are physical quantities such as

1. **Features** transverse momentum, azimuth angle from the beamline, etc.
2. **Objects** Mapped to the relevant promary dataset (i.e. trakcs to ZeroBias, muons to SingleMuon and so on)

Since each run contain a lot of lumisection which probably too much for processing the certification, [4] offers a way to certify data by run and lumisection levels where there is a supervised learning apply in the whole run and feeding only the grey zone to inspect by lumisection to investigate the outlier on step 2 as in the Figure 3.2.

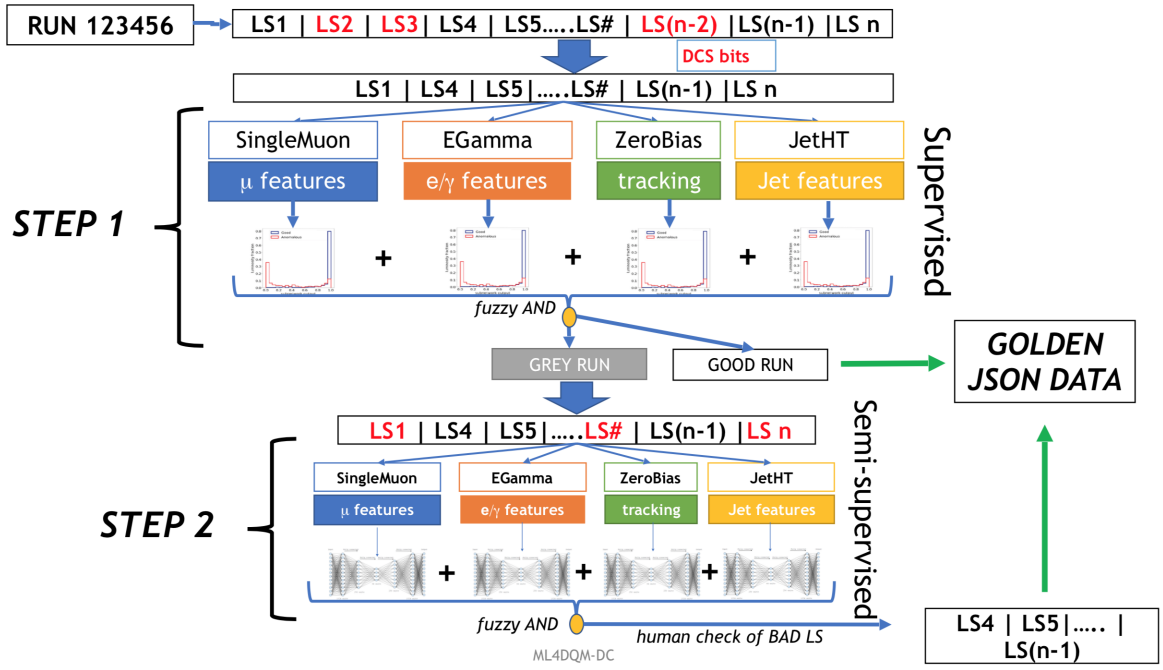


Figure 3.2: *ML certification procedure, image taken from [4]*

Chapter 4

Methodology

4.1 Datasets

Offline (PromptReco)

- pp collisions (Separately study 2016 and 2018 data)
- 4 different primary datasets: ZeroBias, JetHT, EGamma, SingleMuon
- Each lumisection (datapoint) contains
  - Selected n histograms of physics quantity e.g. JetPt, JetEta, JetPhi, etc.
  - Represent one histogram with 7 numbers
  - $n \times 7$ Features
- Good LS defined in Golden JSON else Bad LS

4.1.1 Histogram Representation

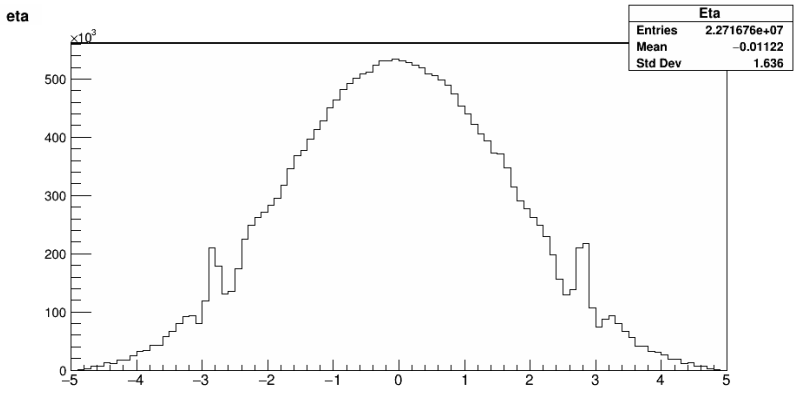


Figure 4.1: *η distribution, image taken from DQM GUI*

To mimic the offline shifter that looking at the histogram and certifies data by that. Then I decided to represent a single histogram with seven quantities instead of feeding all of those

to the model because it would be an overwhelming large feature. The explicit example is to consider one histogram as Figure 4.1

Here are the simple step for picking up our represented vector

- Quantize [10%, 30%, 50%, 70%, 90%] of the histogram (For 2016 data, we quantize [0%, 25%, 50%, 75%, 100%] of the histogram)
- Combines mean and rms
- Use these 7 values to represent one histogram

4.1.2 Data Preprocessing

For numerically convenient, we transform each data point by using

MinMaxScalar Transformation The mathematical expression is represented by considering lumisection i and features j

$$x'_{ij} \leftarrow \frac{x_{ij} - \min_{\forall i \in S_{\text{train}}}\{x_{ij}\}}{\max_{\forall i \in S_{\text{train}}}\{x_{ij}\} - \min_{\forall i \in S_{\text{train}}}\{x_{ij}\}} \tag{4.1}$$

In principle, each feature in our data should be in range zero to unity.

4.2 Semi-Supervised Learning

We exploiding a various unsupervised model from a beautiful simple one to more complicate neural network which are

- Schölkopf's One-Class SVM
- Isolation Forest
- 4 Flavours of Autoencoder

Training the model by feeding only good LS as well as validate it with good LS to ensure the learning curve is appropriate for hyper-parameters configurations. After the training process is done, we test it with both good and bad LS. Consequently, it's falling into **Semi-supervised Learning** category.

4.2.1 Schölkopf's One-Class SVM

Support vector machine (SVM) is one of the most popular machine learning models since it has no randomness and beautiful straightforward way to express. There is a way to tweak the original one to interpret as the radius-like hyperplane where the inlier of data is feeding and mapped to more than one radius with a soft margin controlled by a factor ν as interpret in [8]. By minimizing

$$\frac{||w||^2}{2} + \frac{1}{\nu l}\sum_{i=1}^{l}\xi_i - \rho \tag{4.2}$$

Under conditions

$$w \cdot \Phi(x_i) \geqslant \rho - \xi_i \ , \ \xi_i \geqslant 0 \tag{4.3}$$

With Gaussian Base Radial function (GBF) as a kernel like equation 4.4

$$k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma||\mathbf{x} - \mathbf{y}||^2) \tag{4.4}$$

4.2.2 Isolation Forest

The idea of bagging the sample and construct a tree are incredibly beautiful are interpreted and adapt in a way that unsupervised learning comes to the place are study by [6]. Forest are constructed by picking up subsampling (Ψ) and Iteratively picking up features and random value to construct the node (equivalent to step function), then Anomaly score evaluate from average depth of the instance over a forest

$$s(x, \Psi) = \exp^{-<h(x)>/c(\Psi)} \tag{4.5}$$

where

- $h(x)$ is the depth in tree h
- $c(\Psi)$ normalization factor growing as $\log_2(\Psi)$ from branching

4.2.3 Autoencoder (AE)

Let start with the hyper-parameters and configurations of our model

Truncated normal initializer

For model weight initializer, we are using truncated normal initializer which basically by putting the cutoff only inside $\pm 2\sigma$ of gaussian distribution as in the Figure 4.2 to prevent some high absolute value that might leading to divergence of model in the training process.

In our case, we set up $\sigma = 1$ and $\mu = 1$

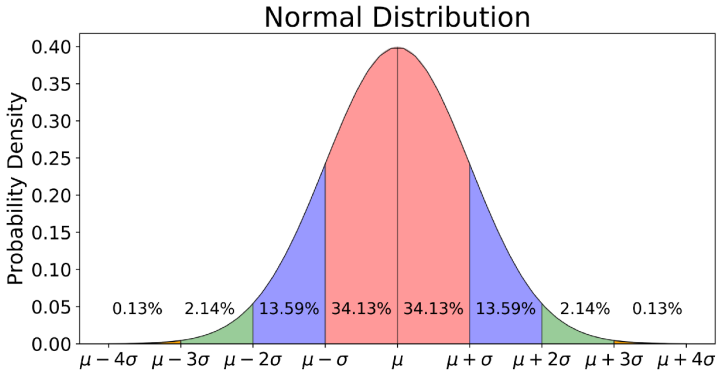


Figure 4.2: *Gaussian distribution, retrieved from https://towardsdatascience.com/understanding-the-68-95-99-7-rule-for-a-normal-distribution-b7b7cbf760c2*

Adam Optimizer

Adam stands for **adaptive moment estimation** [5]. It combines Momentum optimization and RMSProp to keep the residue of the gradients decaying from the previous update. We set up $lr = 10^{-4}$ (learning rate), $\beta_1 = 0.7$ and $\beta_2 = 0.9$.

There are 4 flavours of our autoencoder that we constructed

Vanilla AE

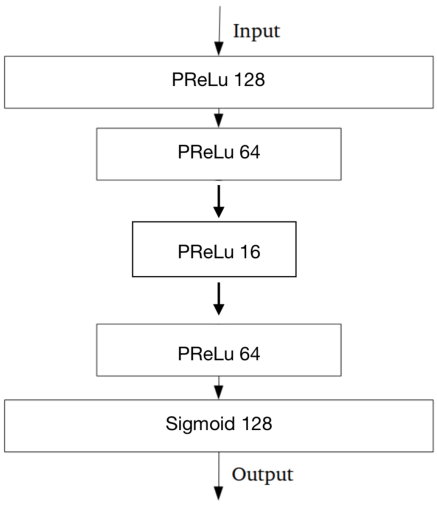


Figure 4.3: *Body of Vanilla AE*

- Concise the information into small latent space and reconstruct the vector $\tilde{x}$ from latent space as in the Figure 4.3
- Loss function is designed by taking the square different reconstructed vector from the latent space and origin vector as equation 4.6

$$\mathcal{L}_{\text{tot}} \equiv \frac{1}{N} \sum_{i}^{N} |x_i - \tilde{x}_i|^2 \tag{4.6}$$

Sparse AE

- Similar to Vanilla AE
- Tweak by L1 Regularizaion (Prevent overfitting)
- Loss function

$$\mathcal{L}_{\text{tot}} \equiv \frac{1}{N} \sum_{i}^{N} |x_i - \tilde{x}_i|^2 + \lambda_{\text{s}} \sum_{j} ||w_j|| \tag{4.7}$$

- where $\lambda_{\text{s}} = 10^{-5}$

Contractive AE

- Tweak by Jacobi Matrix (Prevent variation in dataset)
- Loss function

$$\mathcal{L}_{\text{tot}} \equiv \frac{1}{N}\sum_{i}^{N}|x_i - \tilde{x}_i|^2 + \lambda_{\text{c}}||J_h(x)||^2 \tag{4.8}$$

- where $\lambda_{\text{c}} = 10^{-5}$
- Definition

$$||J_h(x)||^2 \equiv \frac{1}{N}\sum_{ij}\left(\frac{\partial h_j}{\partial x_i}\right)^2 \tag{4.9}$$

- where h_j is activation function
- In our cases
  - PReLu activation function

$$||J_h(x)||^2 = \frac{1}{N}\sum_{i}^{N}\sum_{j}[\alpha_j H(-(w_{jk}x^{ik} + b_j)) + H(w_{jk}x^{ik} + b_j)]\sum_{k}(w_{jk})^2 \tag{4.10}$$

  - Sigmoid activation function

$$||J_h(x)||^2 = \frac{1}{N}\sum_{ij}[h_{ij}(1 - h_{ij})]\sum_{k}(w_{jk})^2 \tag{4.11}$$

Variational AE

- Random "new sampling" in latent space by gaussian random generator as demonstrated in Figure 4.4

$$\mathcal{Z} \equiv \mathcal{N}(\mu_i, \sigma_i) \tag{4.12}$$

- Tweak by reduce discontinuity in latent space
- Loss function

$$\mathcal{L}_{\text{tot}} = \frac{1}{N}\sum_{i}^{N}|x_i - \tilde{x}_i|^2 + \mathcal{D}_{\text{KL}}(p|q) \tag{4.13}$$

Theorem 1. *How much information is loss after represent data with function could be measured by Kullback-Leibler Divergence*

$$\mathcal{D}_{KL} \equiv < \log p - \log q > \tag{4.14}$$

Where p is observed value and q is approximation function
Since our q is Gaussian function, then D-KL term would looks like

$$\mathcal{D}_{KL,i} = \frac{1}{2}\sum_{k}^{n_{latent}}(\mu_{ik}^2 + \sigma_{ik}^2 - 2\log\sigma_{ik} - 1) \tag{4.15}$$

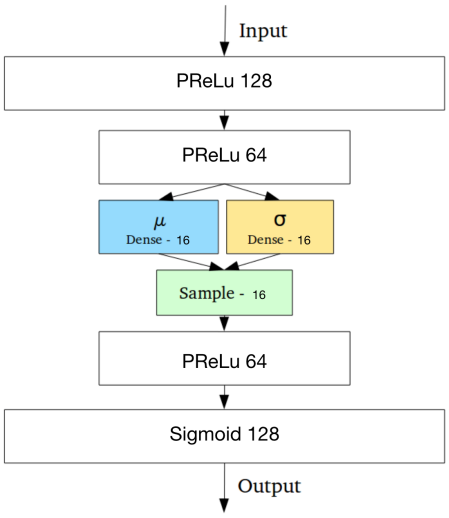


Figure 4.4: *Body of Variational AE retrieved from https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf*

Consequently, loss function of variational autoencoder could be written in the closed form as Equation 4.16

$$\mathcal{L}_{\text{tot}} = \frac{1}{N}\sum_{i}^{N} |x_i - \tilde{x}_i|^2 + \frac{1}{2N}\sum_{i}^{N}\sum_{k}^{n_{\text{latent}}} (\mu_{ik}^2 + \sigma_{ik}^2 - 2\log\sigma_{ik} - 1) \tag{4.16}$$

Chapter 5

Results and Interpretation

5.1 2016 Datasets

5.1.1 Primary Analysis

In order to roughly understand a group (similar patterns) of data, one way to do it is to reduce the dimension of data. In our case, there are 259 features which will be transformed into two-dimension on the basis of two eigenvectors (selected by two largest eigenvalues) belonging to covariance matrix which computed from the datasets.

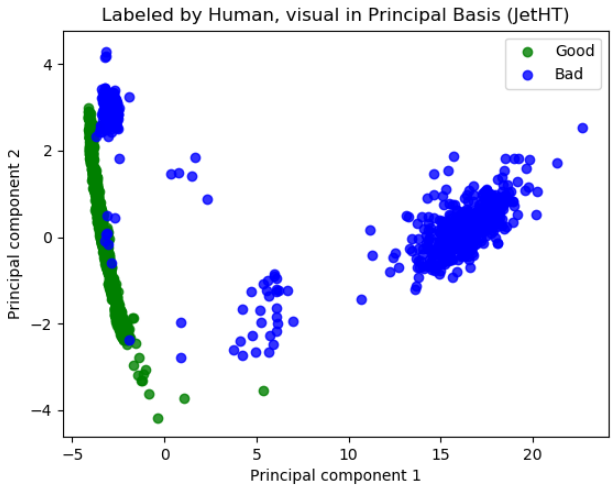
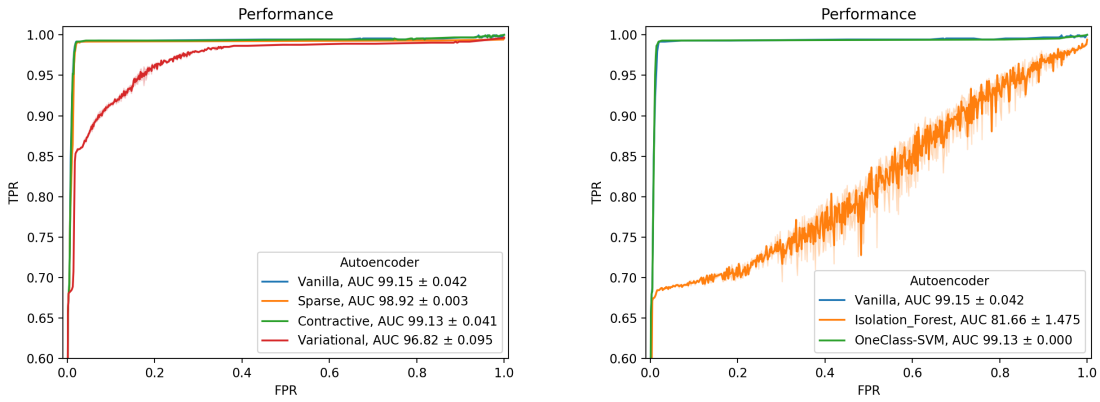


Figure 5.1: *Principal component with the labeled color from the system*

As it can be seen on the green line in Figure 5.1 that there are nice band which is good LS and a few weird LSs that located outside the tubular shape as well as bad LS that could be divided into the bad LS with some patterns and anomaly bad LS which I would call both of them as "outlier". That's essentially the punchline why I called outlier detection instead of anomaly detection.

5.1.2 Performance

By Iteratively retrain the model ten times to make sure that it's working systematically and plot the root mean square as a shady fluctuation in Figure 5.2



(a) *4 Flavours of AE* **(b)** *Vanilla AE vs SVM vs Isolation Forest*

Figure 5.2: *Comparative visualization of model performance*

To sum up, even there are a fancy mathematical expression of non-vanilla autoencoders but it does not guarantee that we would get the best performance out of it. On the other hand, the simplest AE has the performance among all AE. One other interesting spot is the performance of OneClass-SVM also yields the remarkable results as nearly compatible with Vanilla AE without any fluctuation since the model itself has no randomness and work very straightforward.

5.1.3 Distribution of decision value (to find the threshold)

The story behind the performance figure is genuinely extracted from the distribution of decision value from Figure 5.3 and slowly moving a threshold of minimal point in the overlapping region of good and bad LS from a label in the distribution until it got the maximal value. The below figures are the comparison between our two great candidates by considering to pick some threshold and see the contamination on each side.

For Vanilla AE, the contamination of bad LS falling into good LS is around 1% over the good LS below the cutoff and there are only a few of good LS falling into bad LS which might be ignorable.

For OneClass-SVM, the contamination LS that bad falling into good LS is almost the same as Vanilla AE does. There is no coincidence for a totally different approach of model train and spot the same thing. This might implicitly imply that it either came from some imperfection of data in the training and testing or some kind of malfunction in the sub-system could not propagate into JetHT physics objects.

As can be seen in the distribution, there is no clear grey zone for this study so far.

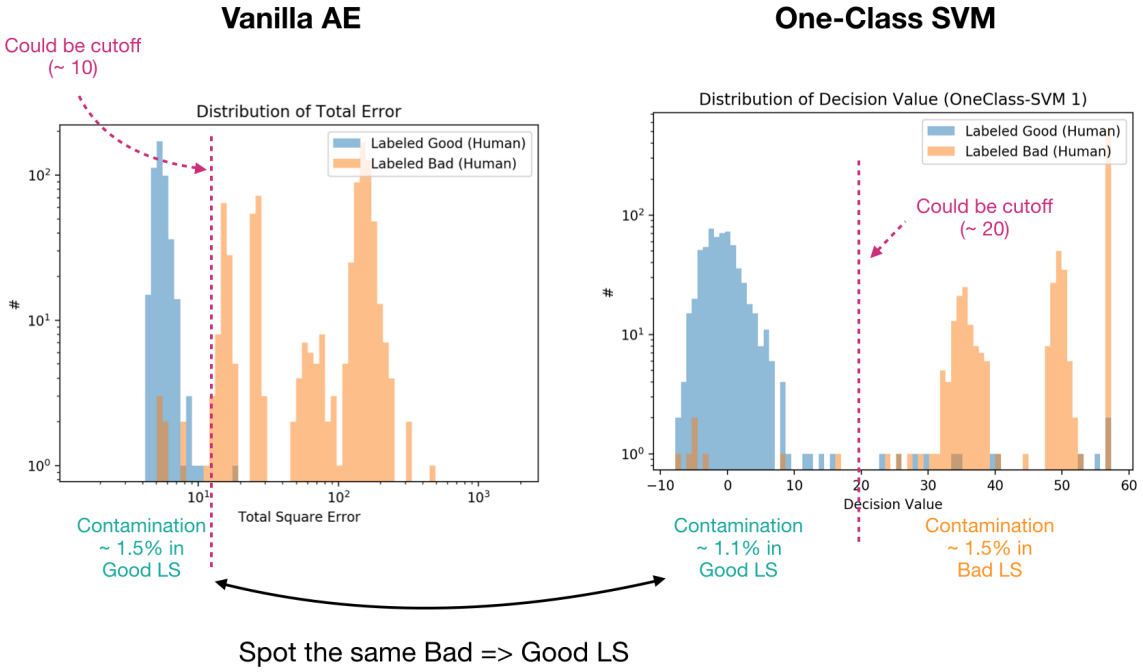


Figure 5.3: *Distribution of decision value*

5.1.4 Example of square error from reconstruction

Figure 5.4 shows the example of LS reconstruction which calculated from x and $\tilde{x}$ between good and bad LS.

5.1.5 Extended Investigation

It may be questioned why many of bad LS seems to have a group of bad LS as you have seen in the plot of hyperspace and few collections of bad LS in decision value distribution (As the black arrow that links between the distribution and 2D-hyperspace). In this section, I want to explicitly prove that the model really sees that the right cluster is the worse bad LS and closer to tubular is less bad LS which decision value has to be quite similar to good LS. In order to prove that, I choose our best candidate to shade the decision value as z-axis color to represent how bad LS in each data point is as in Figure 5.5. The result strongly agrees that there are obvious bad lumisection and less badness as it gets closer to the green band.

5.2 2018 Datasets

5.2.1 Primary Analysis

For 2018 data, we dig a bit more to understand which cause the badness of bad LS by taking sub-system label into account from RR's API. There is plenty of sub-systems in CMS detector. In order to roughly understand the malfunction of sub-system, we decided to pull label only for HCAL, ECAL, TRACKER and MUON detector which are the main part of the detector.

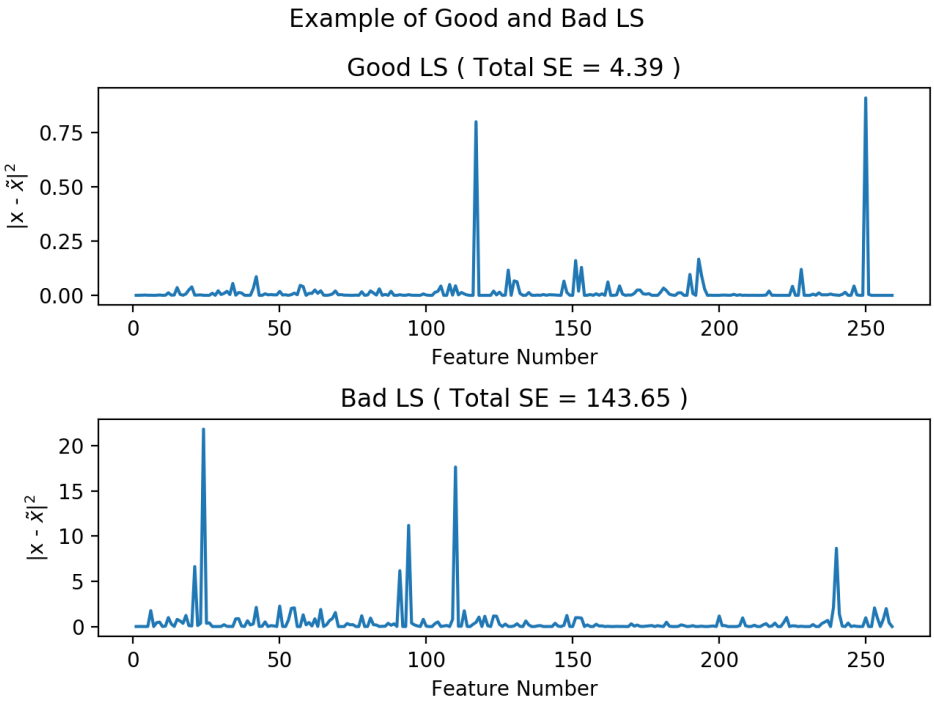


Figure 5.4: *Reconstruction error from Vanilla AE*

To roughly describe each feature contribute to each principal component, we extract the element in matrix transform (equivalent to an element in each eigenvector) and take the absolute value to consider only for the magnitude and ignore the direction in the space where it directly proportional to the contribution of each one. The spectrum of contribution will provide in the following hyperspace of each primary dataset.

According to Figure (5.6, 5.7, 5.8, 5.9), It's obviously to tell that the cluster of outlier are mainly consists of malfunction from MUON and TRACKER sub-detector. Not only the outlier that has an interesting pattern but clustering in inlier is also remarkably considerable as clustering mainly from a malfunction of ECAL and HCAL that located near or inside the green band.

Please note that the calculation of the matrix transform exclude failure scenario since it's a fake data and it might leading to a weird correlation in covariance matrix. The following list is the list of important features that highly correlated to the rest of them in our dataset

- From Figure 5.6, qpVt in transverse direction and qSigmalEta contribute in first component. Secondly, second component mostly consists of qPUEvt and qlumiEvt. Lastly, there are overlapping feature where both of them sharing the different value which are qSigmalPhi and qpVtxChi2.

- Regarding Figure 5.7, qglobTkChi2 and qpVtx in perpendicular direction of the beam are dominated in first component and second component mostly consists of qPUEvt, qMuN and qMuNCh orderly. The only overlapping feature in SingleMuon is qglobTkNHits.

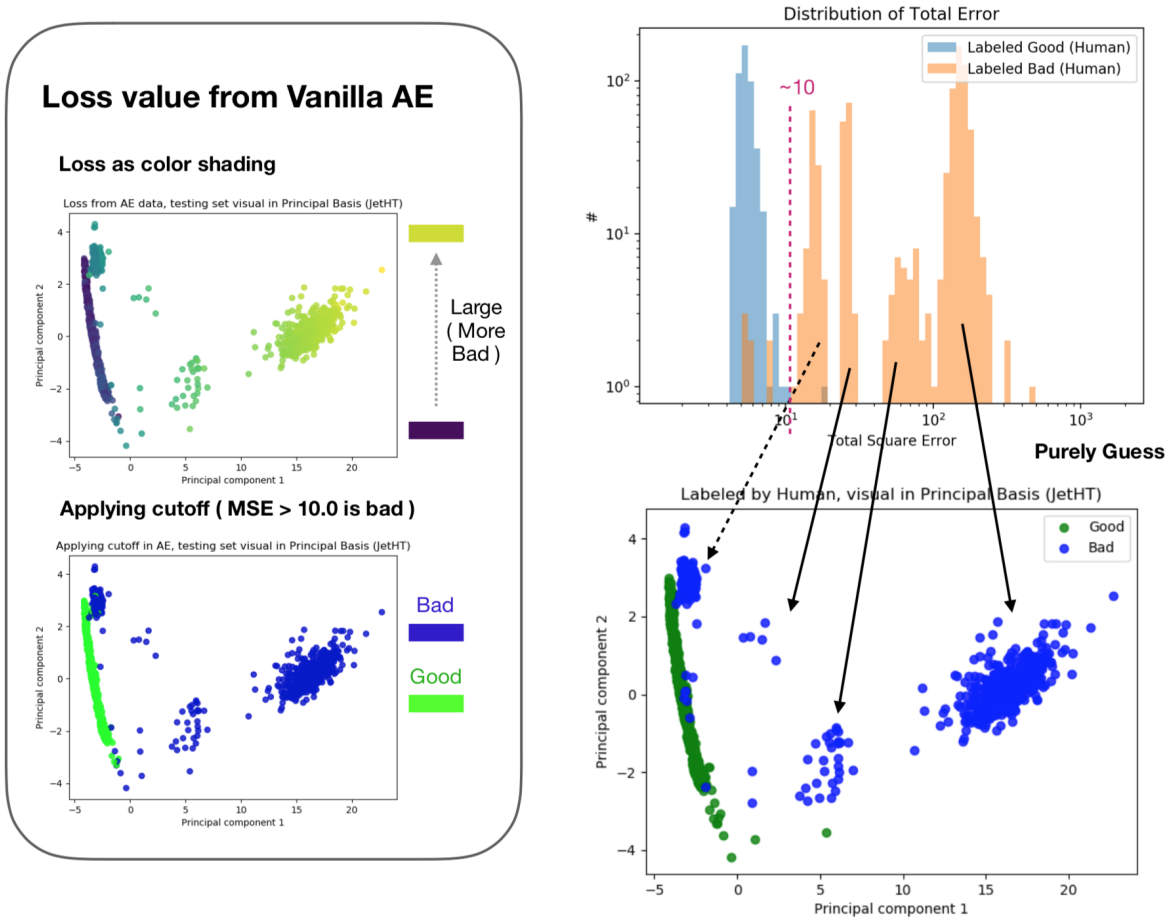


Figure 5.5: *Colorize reconstruction error from Vanilla AE*

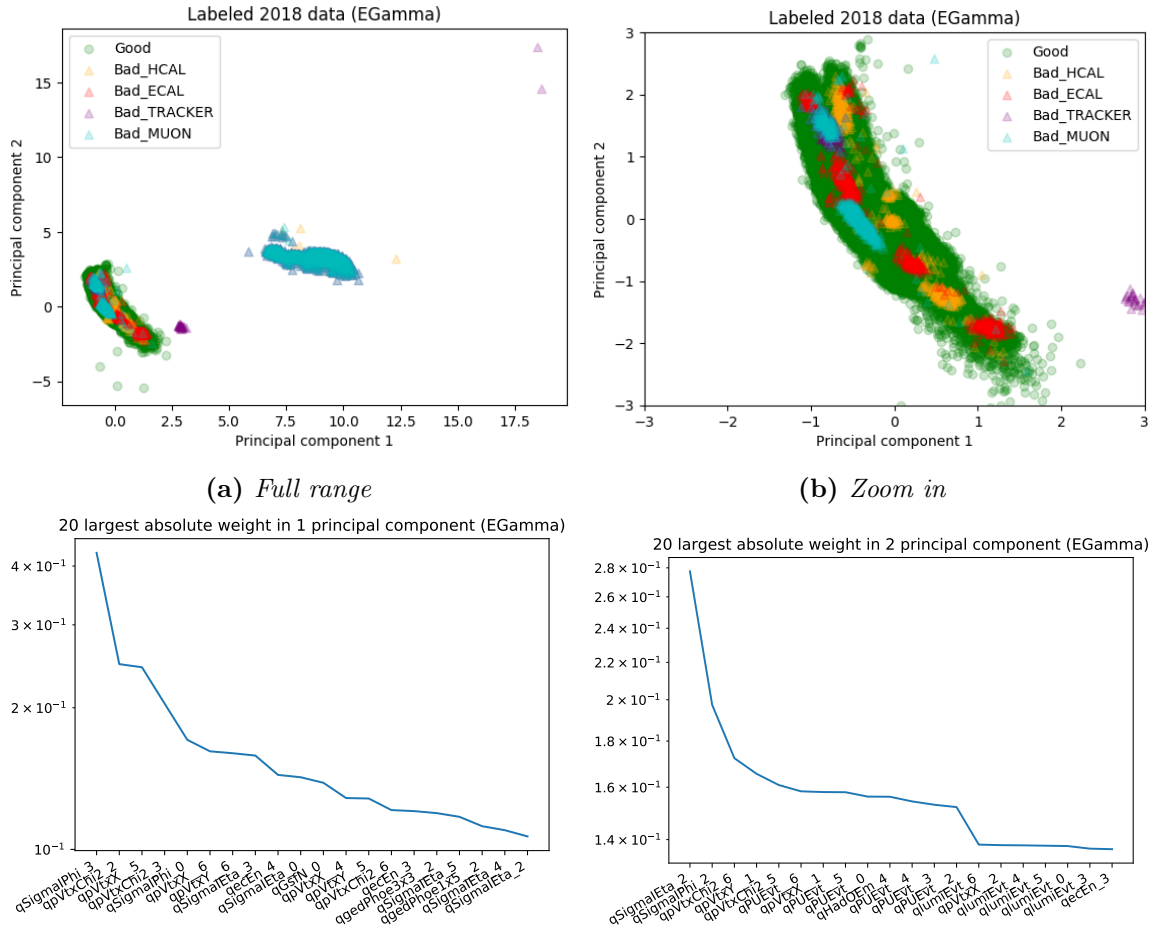
- For ZeroBias in Figure 5.8, both qgTkPt and qgTkPhi highly dominate in first component. Second component has smaller correlation which the faetures are qPUEvt, qlumiEvt and qgTkN.
- First component are constantly diminated to qCalJetN, qCalJetPt and qPUEvt orderly according to Figure 5.9. Feature qpVtxChi2, qPFMetPt and qPFJetEta also fairly equally contribute to the second component.

5.2.2 Performance

1) Include low statistics (fill null with zero) and testing with only bad LS form human

Train with feature set 1 and the result has shown in Figure 5.10.

The performance of AE for EGamma primary dataset is totally inefficient and even worse than randomly picking up which means that the model even saw most of bad LS even looks better than many good LS in the testing datasets. The rest of them is fairly acceptable but still not enough to exploit in the real system. Another interesting spot is the performance



(a) *Full range*

(b) *Zoom in*

(c) *Contribution of features on PC1 (explained variance ratio = 0.31)*

(d) *Contribution of features on PC2 (explained variance ratio = 0.25)*

Figure 5.6: *Two principal components of EGamma*

between a couple of AE in SingleMuon PD.

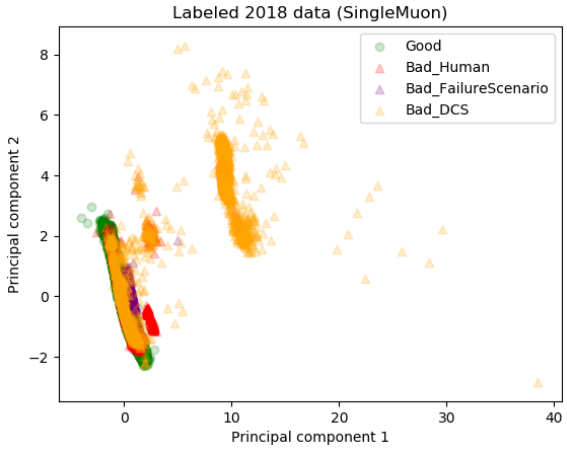
Figure 5.11 has demonstrated that even extended model has been combined various constraints that we know but it still not improves any further in terms of performance. Nevertheless, it has remarkable stability, especially for ContractiveVariational AE.

2) Exclude low statistics (Filter LS that has low EventsPerLs with value in the settings)

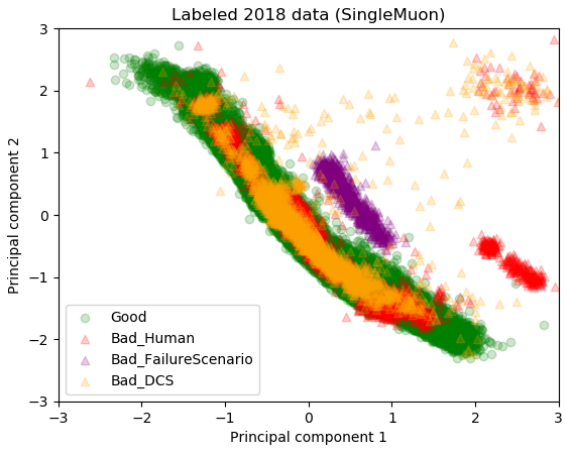
Train with feature set 1 and the result has shown in Figure 5.12. We also perform an extended autoencoder for testing with this case, Figure 5.13 has shown the stability and smoothness of the threshold as we have seen in Figure 5.11.

Distribution of decision value (to find the threshold)

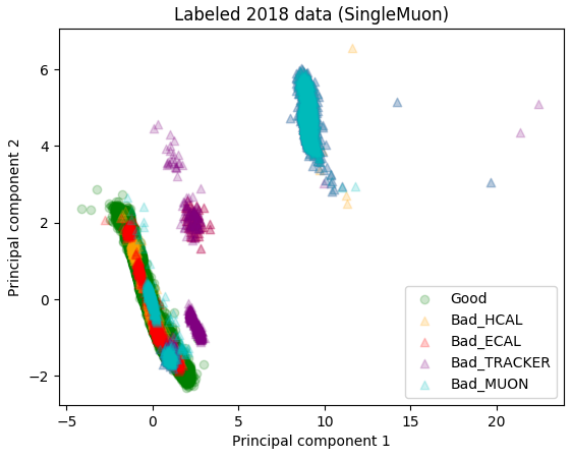
The distribution of decision value could be seen in Figure 5.14



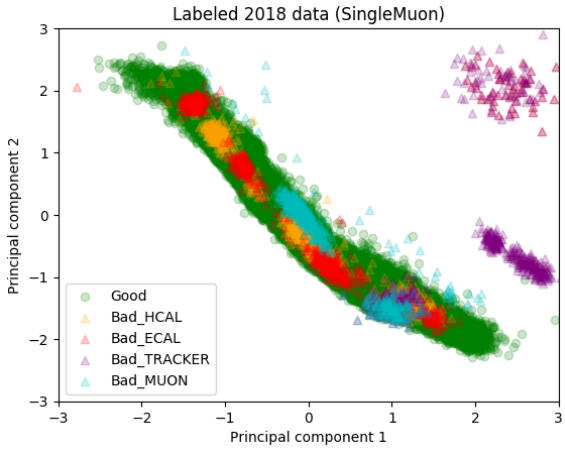
(a) *Full range*



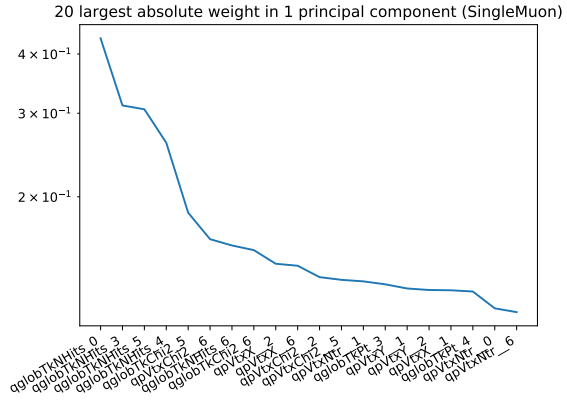
(b) *Zoom in*



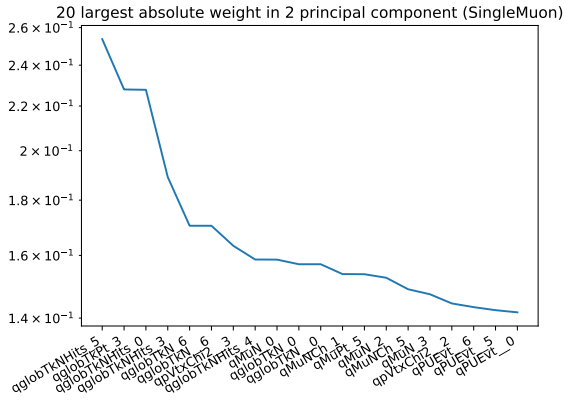
(c) *Full range*



(d) *Zoom in*

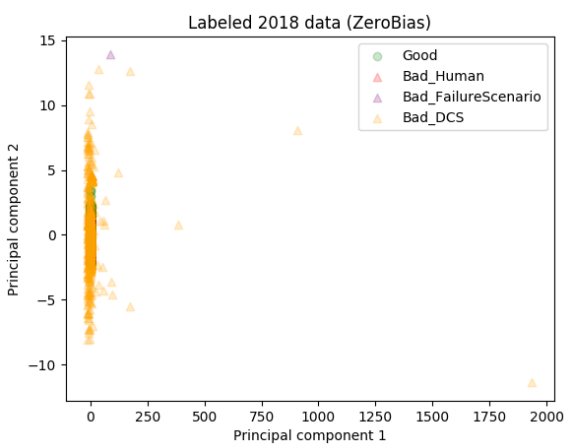


(e) *Contribution of features on PC1 (explained variance ratio = 0.37)*

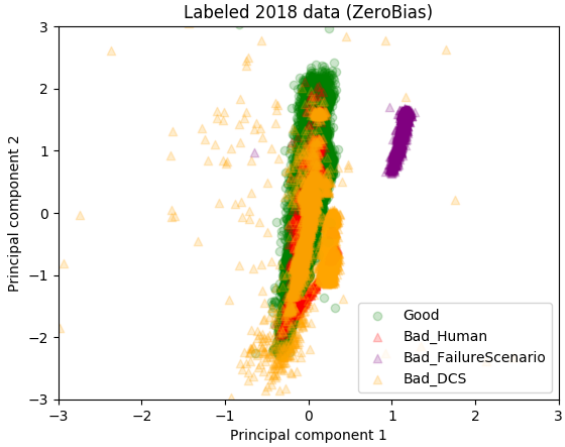


(f) *Contribution of features on PC2 (explained variance ratio = 0.22)*

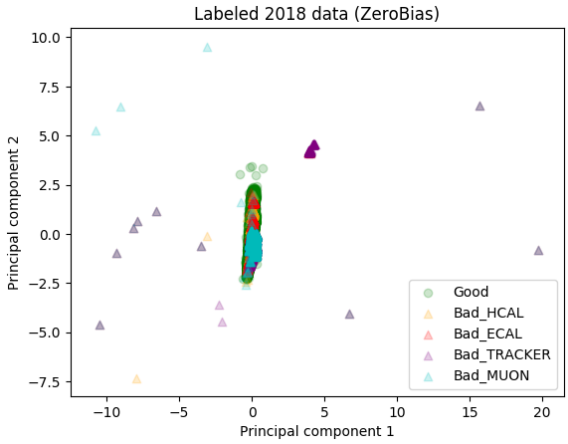
Figure 5.7: *Two principal components of Single Muon*



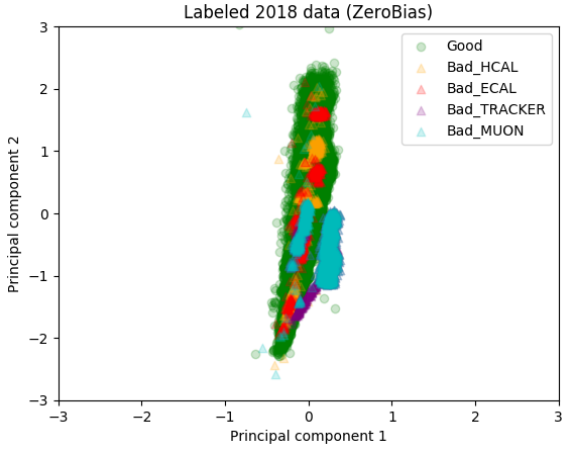
(a) *Full range*



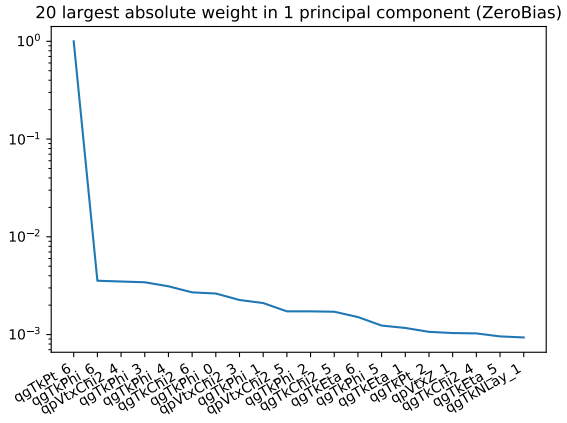
(b) *Zoom in*



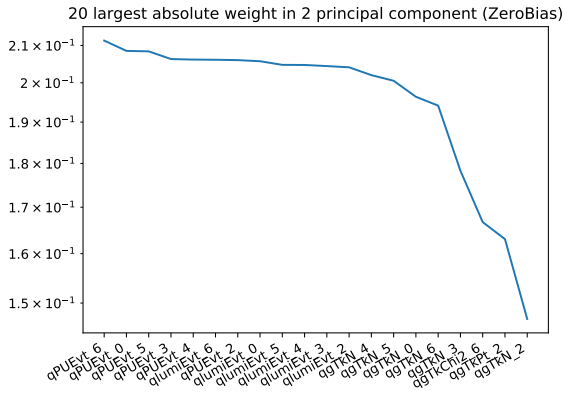
(c) *Full range*



(d) *Zoom in*

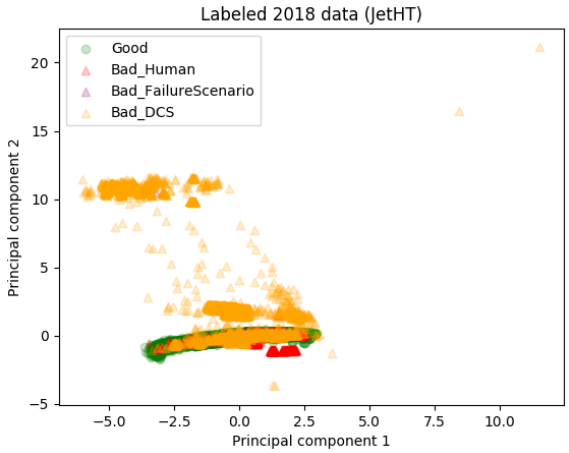


(e) *Contribution of features on PC1 (explained variance ratio = 0.89)*

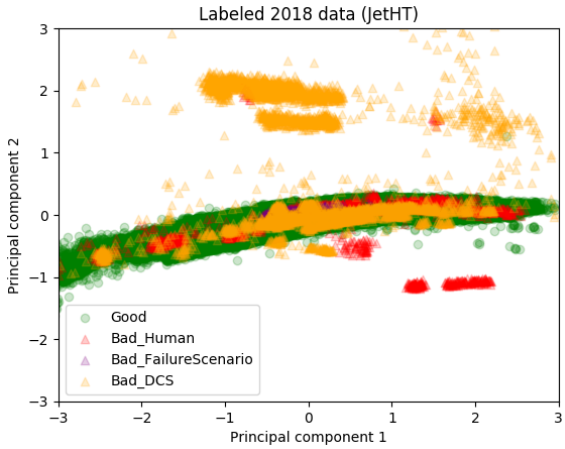


(f) *Contribution of features on PC2 (explained variance ratio = 0.0225)*

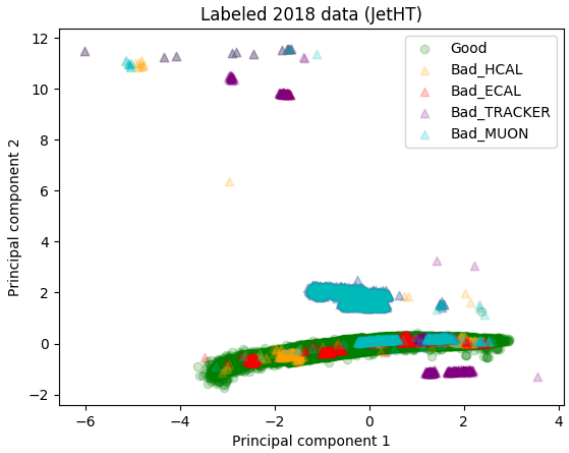
Figure 5.8: *Two principal components of ZeroBias*



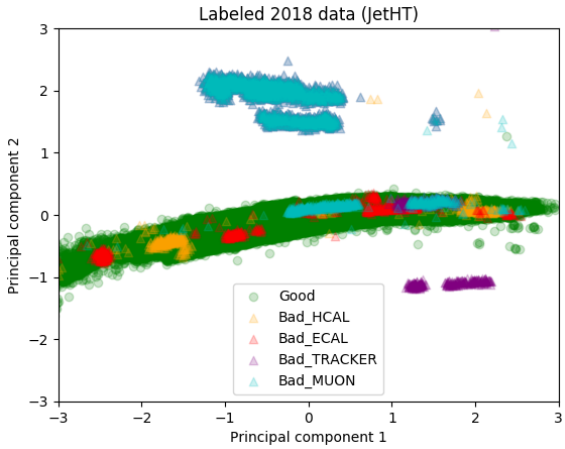
(a) *Full range*



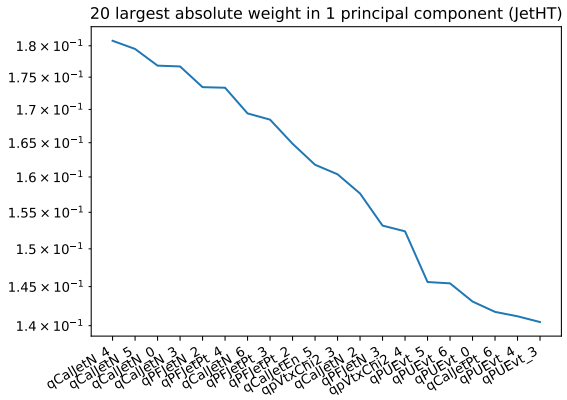
(b) *Zoom in*



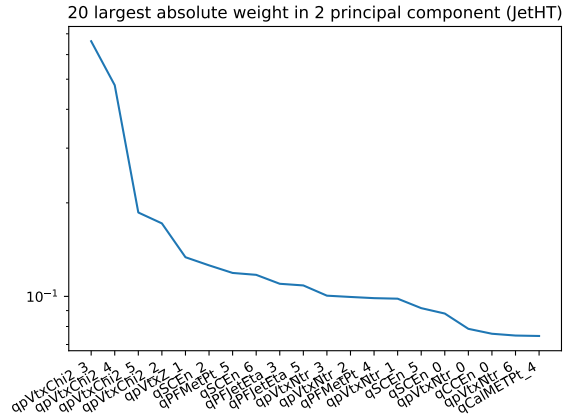
(c) *Full range*



(d) *Zoom in*

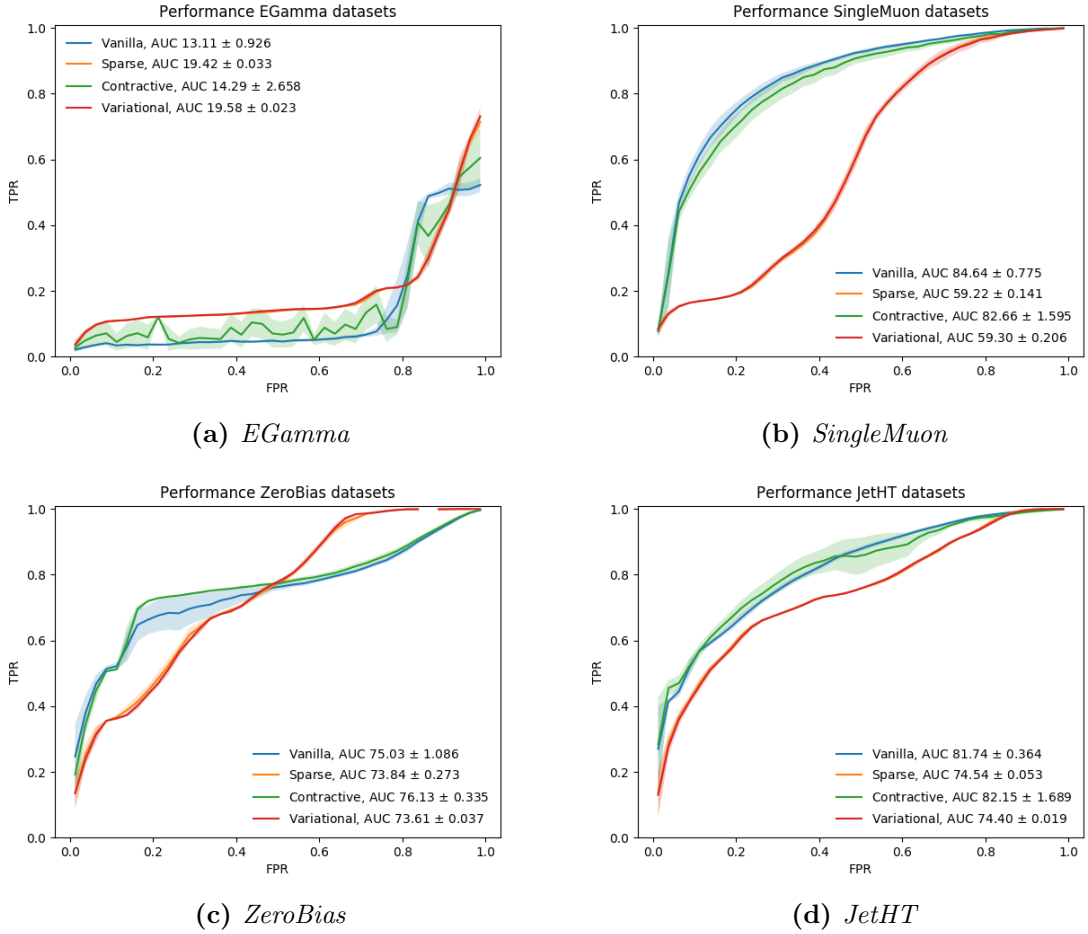


(e) *Contribution of features on PC1 (explained variance ratio = 0.45)*



(f) *Contribution of features on PC2 (explained variance ratio = 0.21)*

Figure 5.9: *Two principal components of JetHT*



(a) *EGamma* **(b)** *SingleMuon*

(c) *ZeroBias* **(d)** *JetHT*

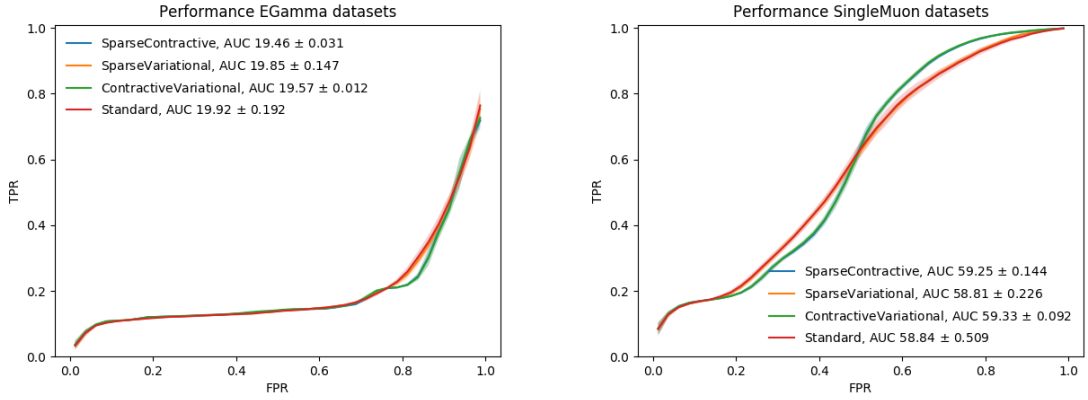
Figure 5.10: *Model performance for feature set 1 with 2018 data*

Reconstruction Error

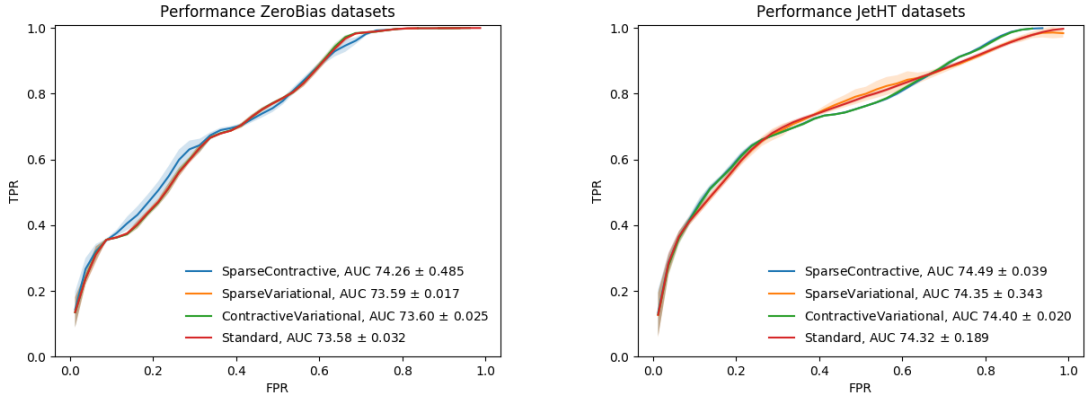
Regarding Figure 5.15, the peak around feature 50 in good LS is qglobTkN. Secondly, the couple clump around feature 80 is qglobTkChi2. The next pile is qglobTkNHits as well as last forky shape in around feature hundred dominated by qMuNCh.

According to Figure 5.16, the residue in feature number 20 to 30 is qpVtxY. There are two huddles in bad LS where it mainly consists of qgTkPt, qgTkEta, and qgTkPhi. The clump in good LS around 70 to 80 mostly is qgTkPhi and qgTkN.

Lastly, by considering Figure 5.17. Features that contain a very first peak in bad LS is qpVtxChi2. Secondly, around feature number 80 to 90 are qPFMetPt and qPFMetPhi. Lastly, there are the last two chunks of features (120-127 and 130-145) that behave like a noisy for both good and bad LS. Highly correlated features that show similar features (15-35) are qpVtxX, qpVtxY, and qpVtxZ.

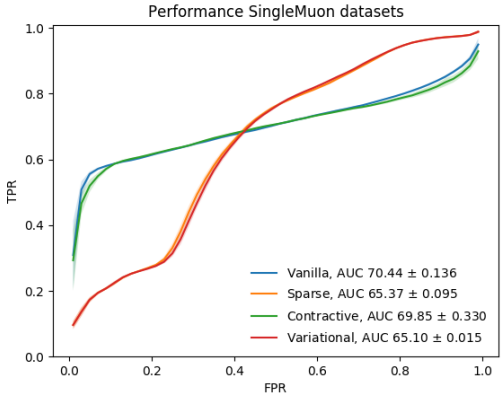


(a) *EGamma* **(b)** *SingleMuon*

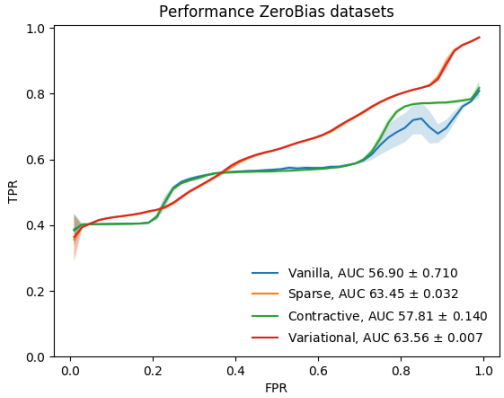


(c) *ZeroBias* **(d)** *JetHT*

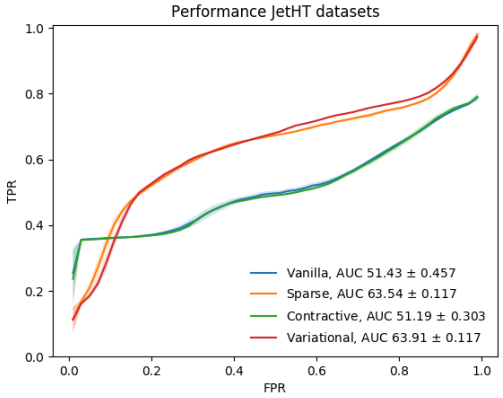
Figure 5.11: *Extended model performance for feature set 1 with 2018 data*



(a) *Full range*

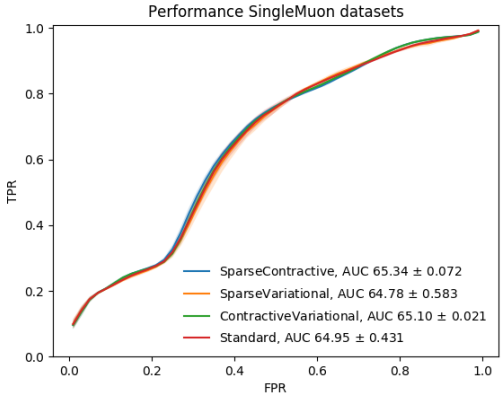


(b) *Zoom in*

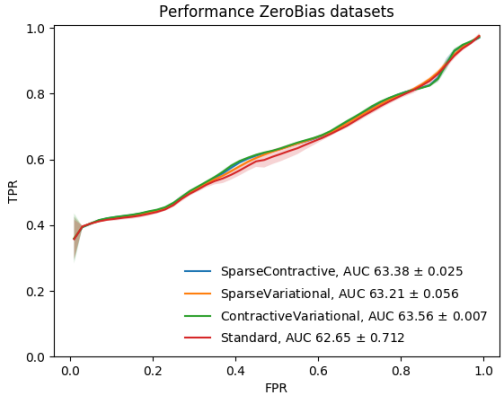


(c) *Full range*

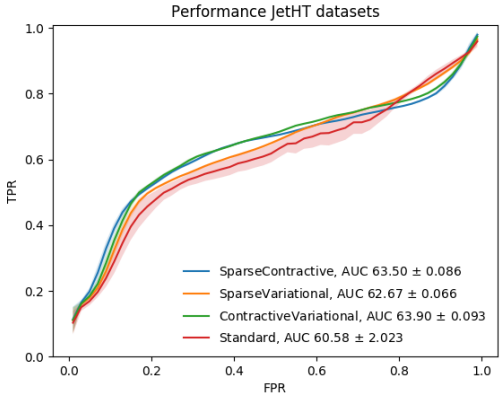
Figure 5.12: *Model performance for feature set 2 with 2018 data*



(a) *Full range*

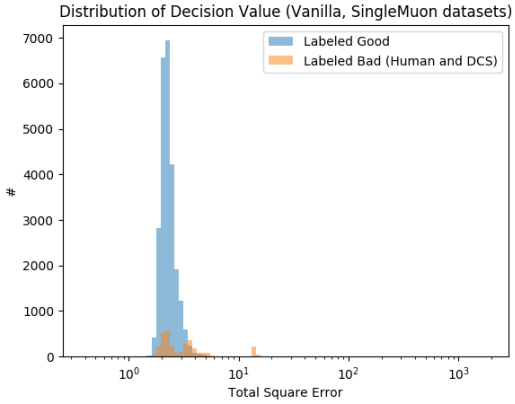


(b) *Zoom in*

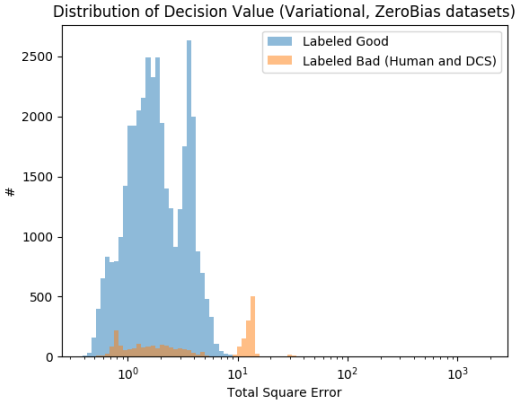


(c) *Full range*

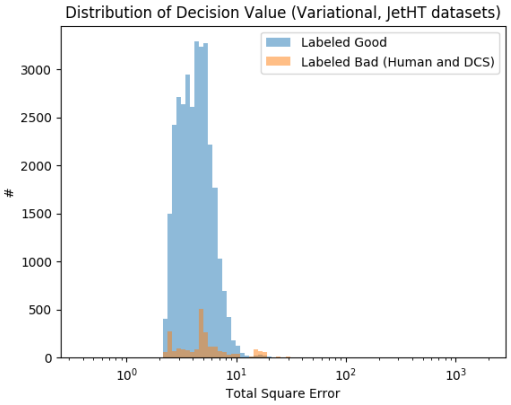
Figure 5.13: *Extended model performance for feature set 2 with 2018 data*



(a) *SingleMuon*

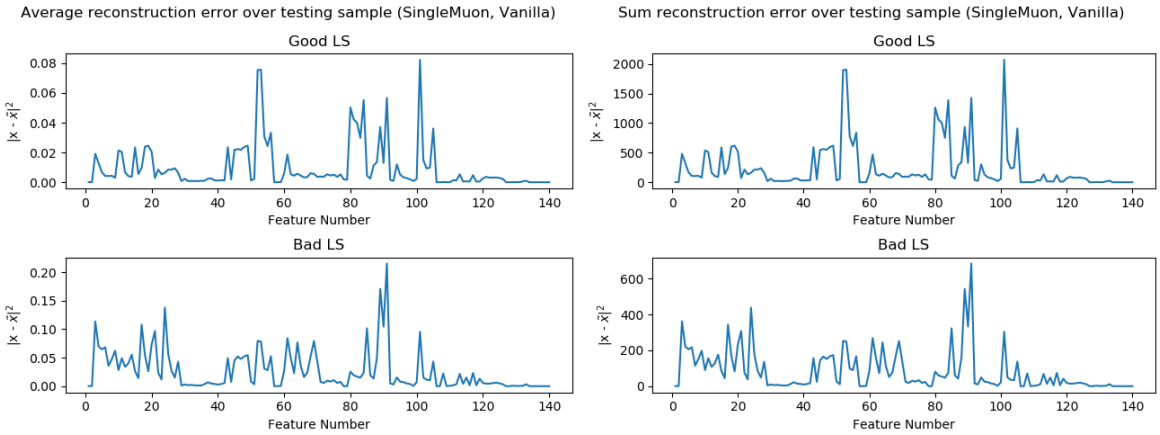


(b) *ZeroBias*



(c) *JetHT*

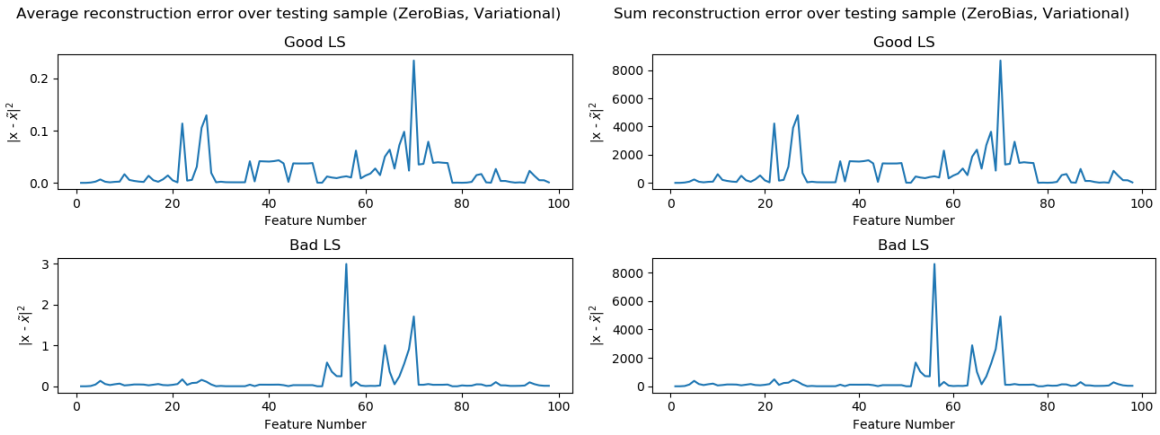
Figure 5.14: *Distribution of decision value*



(a) *Sum over sampling*

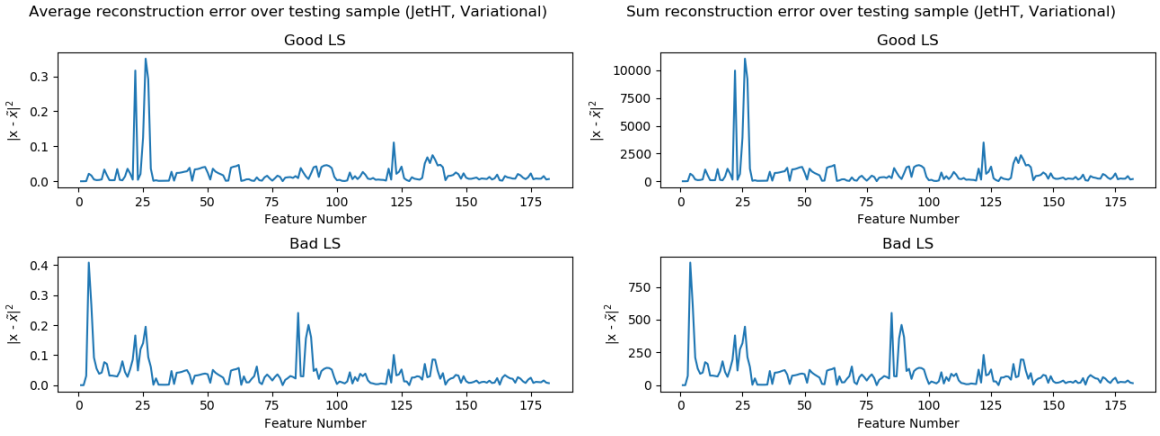
(b) *Average over sampling*

Figure 5.15: *Reconstruction error of SingleMuon*



(a) *Sum over sampling* **(b)** *Average over sampling*

Figure 5.16: *Reconstruction error of ZeroBias*



(a) *Sum over sampling* **(b)** *Average over sampling*

Figure 5.17: *Reconstruction error of JetHT*

Chapter 6

Conclusions

- Semi-supervised learning yields a remarkable result and well describe outlier LS
- So far, there is no grey zone from our model for this dataset
- Bad LS could be divided into two parts
  - Bad with some pattern
  - Anomaly

Bibliography

[1] Barney, D. [2016], CMS Detector Slice. CMS Collection.
URL: *https://cds.cern.ch/record/2120661*

[2] Cittolin, S., Rcz, A. and Sphicas, P. [2002], *CMS The TriDAS Project: Technical Design Report, Volume 2: Data Acquisition and High-Level Trigger. CMS trigger and data-acquisition project*, Technical Design Report CMS, CERN, Geneva.
URL: *http://cds.cern.ch/record/578006*

[3] Committee, C. G. L. E. [1997], 'The CMS muon project'.

[4] Fiori, F. [2019], ML Applied To Data CertificationStatus and Perspective.

[5] Kingma, D. P. and Ba, J. [2014], 'Adam: A Method for Stochastic Optimization', *arXiv e-prints* p. arXiv:1412.6980.

[6] Liu, F. T., Ting, K. M. and Zhou, Z. [2008], Isolation forest, *in* '2008 Eighth IEEE International Conference on Data Mining', pp. 413–422.

[7] P Bauer, G., Bawej, T., Behrens, U., Branson, J., Chaze, O., Cittolin, S., Coarasa, J., Darlea, G.-L., Deldicque, C., Dobson, M., Dupont, A., Erhan, S., Gigi, D., Glege, F., Gomez-Ceballos, G., Gomez-Reino, R., Hartl, C., Hegeman, J., Holzner, A. and Zejdl, P. [2014], 'Automating the cms daq', *Journal of Physics: Conference Series* **513**, 012031.

[8] Scholkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J. and Platt, J. [1999], Support vector method for novelty detection, *in* 'Proceedings of the 12th International Conference on Neural Information Processing Systems', NIPS'99, MIT Press, Cambridge, MA, USA, pp. 582–588.
URL: *http://dl.acm.org/citation.cfm?id=3009657.3009740*