Application of Machine learning on CDF Data

Candidate number: 998805

May 30, 2017

Abstract

Machine learning is the study of programming computers to learn from data without being explicitly programmed. One of the strengths of machine is pattern recognition, which can be very useful in high-energy physics where data of interest are at odds of one in a million. The primary objective of this report is to discuss the backgrounds of machine learning and application of machine learning techniques on experimental particle physics data. Specially, a semi-supervised learning algorithm involving k-means clustering is investigated, with the final program showing interesting results in a relation to background noises with respect to signals in dipion mass histograms of K_0^S.

1 Introduction

'Big Data' has been gaining a lot of traction in the technology sector for the past few years. This term has been repeatedly used to describe the immense amount of data that are being produced by the world since the conception of the internet, with the likes of companies such as Google and Amazon processing up to petabytes of data per day. While the world goes crazy about the amount of data produced by the Internet, particle physicists has already been dealing with incredibly large dataset for a long time. High-energy physics relies on particle colliders which are capable of recording millions of collisions events per second, gathering sufficient data such that the full spectrum of a quantum process can be mapped out. In fact, infrastructure of the modern internet was developed by CERN physicists to address the problem of sharing these data across the world. For today's side by side comparison, Cisco estimated an annual global internet traffic of 2.3 ZB[1], where as LHC produces 18 ZB per annum[C].

One of the challenges posed by this rate of data generation is to find a suitable way to make sense of all this data. Traditional computing methods requires explicit guidance in analysing data. But when seeking underlying structures in a dataset is often an NP-hard problem. At scales of modern datasets, it can be astronomically expensive to crunch the numbers. Machine learning grew out of the field of artificial intelligence and excels at performing complex high dimensional analysis on large datasets. It has become the de facto technique for companies of small to large industry data to discover patterns and insights that drives business decisions.

Physics was one of the early adopters of machine learning techniques. Since the advent of space telescopes in 1960s, astrophysics has been seeing an ever increasing influx of data. Analysing and classifying these imagery data is one of the strengths of machine learning, particular for a subfield called deep learning. Using neural networks, physicists can automatically and quickly identify interesting galaxies or galaxy clusters[2], replacing tedious human labour. However, since then, the use of machine learning in physics has not been as active then the commercial counterpart. Part of this is due to the black-box nature of many machine learning algorithms, where interpretability of a model is often sacrificed for precision and accuracy. This may be a major concern for most physics disciplines, where a better understanding of the laws of nature is often valued over a model that 'just works'.

1.1 Scope of this report

Taken from the success of machine learning in commercial applications and other fields of science, there are many aspects in which the same techniques can be applied on particle physics to various types of improvement. These could be detector resolution refinement, hadron jets classification, finding the Higgs Boson[3] and many more. Limited by the available time for the practical, this report shall work within a narrow scope. We start by asking a question: Using machine learning techniques on a dipion mass histogram at K_0^S mass range, can we prevent ourselves from removing real signals (false negatives) when reducing background noises?

2 Data Description

Information provided in this section is based on practical scripts of CO03[4] and NP10[5].

The Collider Detector Facility (CDF) is a multi-purpose detector at the Tevatron, a circular particle accelerator in the US. Protons and anti-protons are collided at a centre of mass energy of 1.8GeV. The report will be working with a relatively small dataset of 10 thousand events(21 MB), curated from data between 1985 and 2011.

The CDF experiment consisted of a tracking chamber of radius 1.45m, where charged particles leave curved tracks of motion under a magnetic field. A series of layered, cylindrical calorimeters within the chamber detect high-energy unstable particles that interact with strong force.

Each collision event is labelled with unique identifier, and the tracks are described by 5 geometric parameter which allows a perfect geometric reconstruction of the track. Important quantities such as momentum and energy are also retrievable from the parameters. Details of how to reconstruct the tracks can be referred to the report of a previous practical [6].

The program described in this report will be treating each track as a single data entry, with 5 variables (the track parameters).

3 Methodology

This section describes the final program that cut dipion mass histograms by using k-means clustering to create tracks subset. Other exploratory attempts in applying other machine learning techniques are noted in Appendix B.

Before feeding the tracks into the k-means algorithm, the dataset is first normalised. Data normalisation in machine learning is a preprocessing method that standardises the range of the independent variables in the dataset. This step is particular important for the k-means algorithm. K-means relies heavily on Euclidean distance between data points and is very sensitive to variable scales. Normalising the data would prevent one single variable from dominating the clustering results. This importance will become more obvious when the algorithm is discussed in detail in subsection 3.2.

After preprocessing, the dataset is ready to be segmented by the k-means algorithm. The output would is a predetermined number of cluster centroids, with each track in the dataset assigned to one of the unique clusters.

We then retrieve a set of dipion mass reconstruction result from the CO03 practical. Tracks that are responsible for producing signals within the mass window of $0.48 - 0.52$ GeV[1] are tagged as kaon descendants. The CO03 practical uses an identical dataset as this report, so the selected tracks in the clusters can be tagged correspondingly. Clusters are classified according to their proportion of kaon descendants amongst the tracks, where clusters with higher proportions of kaon daughters are included in the new tracks subset.

3.1 Reconstruction

Provided with the new dataset, there are two new ways of which we can reconstruct the K^0_S. The trivial method would be simply to treat the new dataset as an fresh set of data, iterate through the events and reconstruct parent masses with track pairs, treating them all as pions. Another method would be to utilise both the new and original datasets. In each event iteration, we look for track pairs where at least one of the tracks belongs to the new subset that we had obtained. Both methods re-uses vertex reconstruction code from the CO03 practical report.

3.2 K-Means Clustering

The k-means algorithm belongs to a class of machine learning methods called unsupervised learning. This field of machine learning specialises in tasks of interring hidden structure from raw data with no classification or categorization. K-means clustering in particular is a method of identifying segments/clusters within a dataset.

A hyper-parameter, K, is needed as part of the algorithm input. In commercial settings, this parameter can usually be determined by business intelligence, such as the expected number of market/customer segments. In this report's context, physical intuitions may help rule out the effects of the uniformly distributed variables, such as the angle in transverse plane, but this still leaves a > 3 dimensional space to be considered. This leaves no good initial estimate for the value of K. Thus the value must be hand-picked from different results of k-means ran with different K.

The k-means algorithm starts by randomly initiating the cluster centroids. There are different ways to go about this step, each with their own ad-

[1] The kaon mass is 0.497 GeV

vantage. In this report we will be equating a centroid a random data point as initialisation. The next step involves two iterations. First all data points are assigned to the closest centroid, then the centroids are moved(updated) to the mean of points assigned to it. These two steps can be repeated indefinitely until a condition for convergence is reached.

For readers who are completely new to machine learning (the intended audience of this report) and would like to know about the algorithm more mathematical, a piece of commented pseudo code phrased in terms of physical quantities directly applicable to this report has been provided in Algorithm 1.

Algorithm 1 K-means clustering

```
Input:  $k \leftarrow$ Number of clusters, $T \leftarrow$ Tracks data
Output:  $\mu_1,\ \mu_2 \ldots \mu_k \leftarrow$ Cluster centroids
$\mu_i \leftarrow \text{random}(t \in T)$        ▷ Start with random guesses for cluster centroids
repeat
    for $t \in T$ do        ▷ Assign each track to the closest cluster
        $c_t \leftarrow \arg\min_j \left\{ ||\mu_j^{(i)} - t^{(i)}||^2 \right\}$
    end for
    for $\mu_j \in \mu$ do        ▷ Update centroid coordinates
        $\mu_j^{(i)} \leftarrow \frac{\sum_t^T 1\{c_t = j\} t^{(i)}}{\sum_t^T 1\{c_t = j\}}$
    end for
until $\mu$ converges
return $\mu_1,\ \mu_2 \ldots \mu_k$
```

4 Results and Discussion

A number of values ranging between $5 - 60$ for K were tested against the dataset. To visualise the results, tracks are selected randomly (to avoid cluttering the figure) and plotted in 3 dimensions with curvature, impact parameter, and polar angle at closest approach to z-axis as the independent variables. Figure 1 shows one set of the results with $K = 30$. We can observe from the plot that the clusters settled at several spots in the 3 dimensional space. A histogram with multiple cuts on track momentum, impact parameter, and intersections, is used to tag the tracks for cluster classification. The plotted graph is shown in the Figure 10. Details of cuts used to produce this histogram are available in the CO03 report.

In each set of centroid output, the tagged of kaon daughter tracks in each cluster is aggregated to compute the cluster kaon daughter proportion. The threshold for clusters to be included in the new track subset was set to be the kaon daughter proportion from the original histogram. An observation at this stage was that across the tested range of K values, roughly half the clusters passes the threshold kaon daughter proportion. This results in an average of 70% of tracks making it through into the subset.

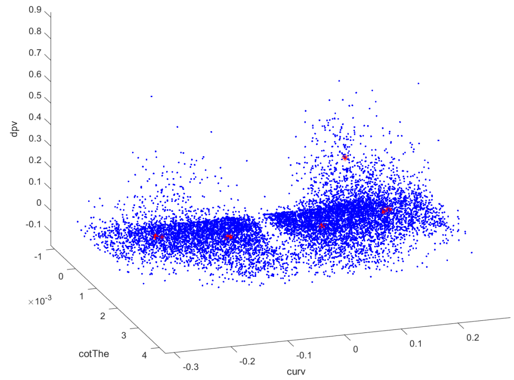

Figure 1: $K = 30$ Blue dot denote tracks, red dot denote centroids

Since the resultant subset is quite similar for different values of tested K, independently repeated analysis using subsets generated different values of K was regarded as redundant at this point of research. The from following parts of this section will thus be based on analyses using a subset generated from $K = 20$. Figure 3 and Figure 4 are the mass histogram produced by crossing pairs between the new track subset and the original tracks, or only in the new track subset, respectively, as laid out in subsection 3.1. The original uncut dipion mass histogram is show in Figure 2 for reference. The histograms will be referred to being the original histogram, crossed histogram, and subset histogram from this point on.

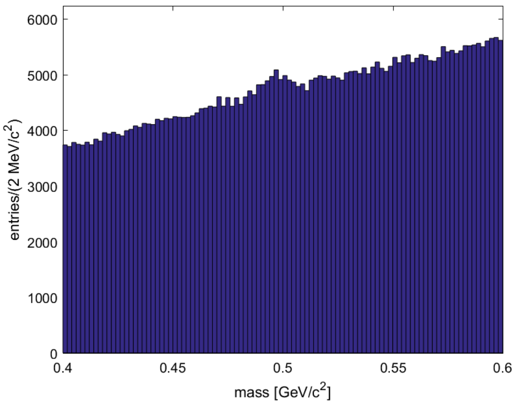

Figure 2: The original histogram

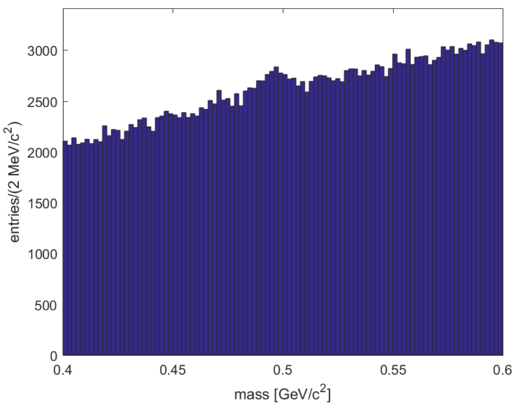

Figure 3: The cross section is roughly half of the original in the crossed histogram

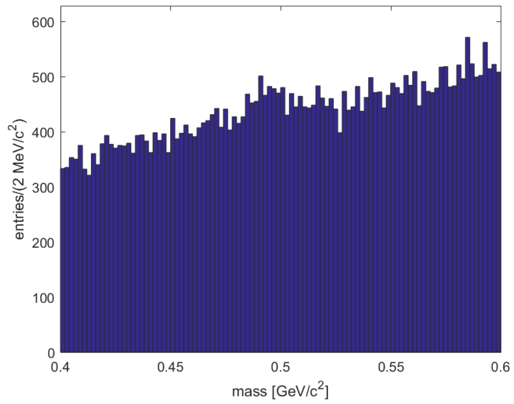

Figure 4: The background fluctuation is noticeably stronger in the subset histogram

To measure the background to signal ratio, we must first define the background. Driven from shape of mass distributions across the histograms, we shall make the assumption that the background is linearly proportional to the reconstructed mass. If the true background distribution of mass is more complicated than we have stated, we are only looking at a small range of mass where linear terms would be dominating in most functions, which justifies the simplification. The MATLAB polyfit function were used to fit the mass distribution. The formula can be expressed as:

$$y = mx + c$$

where y is the size of bin, m is the slope, x is the mass, and c is the constant. A list of residual signal is obtained by subtracting entries in the histogram by their respective background strength given by the formula.

Another MATLAB function, fit, is then used to fit the residual signal with the normal distribution:

$$y = A \exp\left\{\frac{(x-\mu)^2}{\sigma^2}\right\}$$

The fitted parameters are shown in Table 1.

	Original	Crossed	Subset
m	9615	5063	850
c	-110	67	9
A	326 ± 65	217 ± 50	51.6 ± 20
$\mu/10^{-3}$	496 ± 1	495 ± 1	494 ± 3
$\sigma^2/10^{-3}$	8.9 ± 2	9.8 ± 2	10 ± 4

Table 1: Fitted values of linear background and Gaussian signal of the three histograms

The fits are plotted over the scatter plots of the residual signals, shown in Figure 5, Figure 6, and Figure 7.

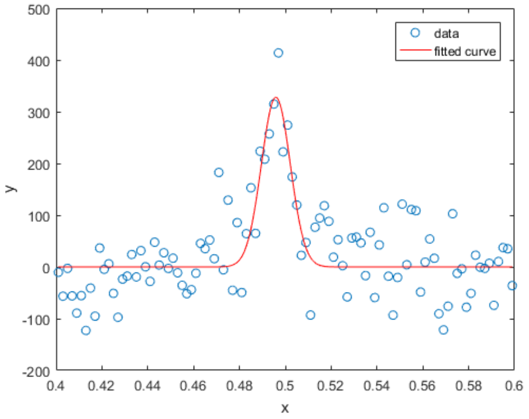

Figure 5: Fit of original mass histogram

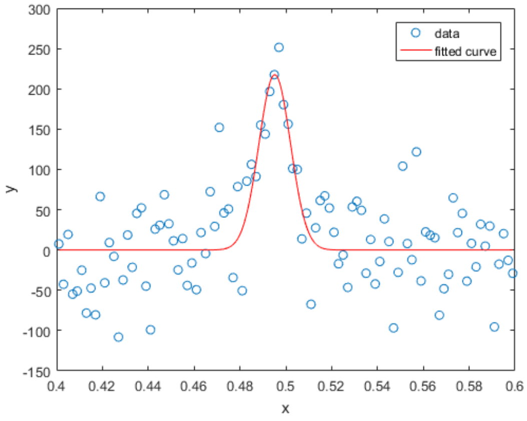

Figure 6: Fit of mass from the cross of the original tracks and new subset

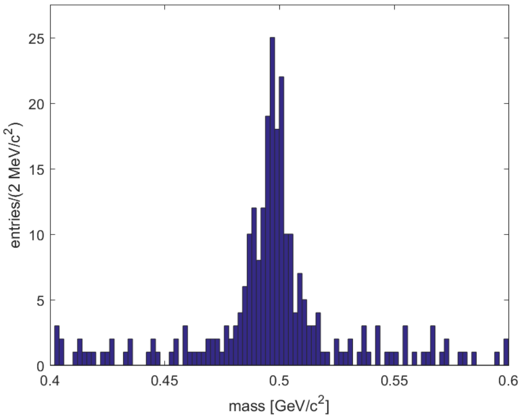

Figure 8: Cross histogram with cuts to identify kaon peak

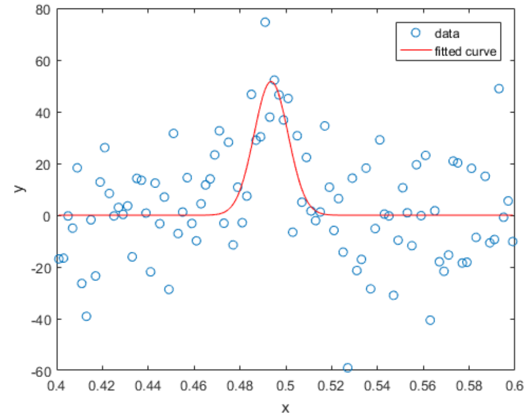

Figure 7: Fit of mass from only the new track subset

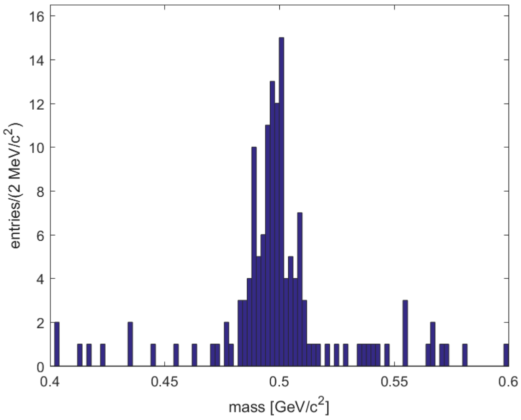

Figure 9: Subset histogram with cuts to identify kaon peak

Finally the background to signal ratios can be computed from the fitted values. Here the strength of signal is taken to be the value of A in the Gaussian fit, giving ratios of 0.034, 0.043 and 0.06 for the original, cross, and subset histograms respectively. We have found that there indeed is an improvement in the relative strength between signal to background in both methods of reconstruction. However, in the subset reconstruction, the signal strength error is significantly larger then the other reconstruction methods, with 39% error, whereas the other two methods both only have about 20% error.

A sanity check would be good at this point to verify that the new methods still provide adequate performance at identifying the K^0_S peak in a full cut histogram. This is shown in Figure 8 and Figure 9, which when compared to Figure 10, only a few entries are missing and the peaks are still very much intact.

4.1 Error analysis

One of the weaknesses of the k-means algorithm is it's interpretability. Given any dataset of random variables and a value for K, even if there were no real inherent structure in the data distribution, the k-means algorithm will always produce a set of centroids. This is may not be a problem in business applications where the K is known. But when used to discover structure in unfamiliar datasets, k-means can raise false positives for results, so care must be taken in interpreting the algorithm's output. In this report, we approached problem by visualisation, then labelling cluster with well-understood physics results.

Another problem is though there was only one hyper-parameter in the program, K, the model is still prong to overfitting. The solution to this can be to split the dataset into a training set and a cross validation set. A robust analysis be carried by comparing the results in cross validation set when different K values are used, and to determine the value of hyper-parameter in based on the cross validation performance.

In addition, since the same dataset were used to label the clusters and to reconstruct the mass histograms, this report may suffer heavily from data snooping bias. Such over fitting can be avoided by having started by setting aside another portion of the dataset(events) to be used later only in the reconstruction, where the remaining data can be fed into the k-means algorithm to compute the centroids. In this practical, the dataset was relatively small in a particle physics context. Separating the dataset may cause the training sample or test sample to not be representative of the distribution, which would render the method ineffective either way.

5 Conclusion

This report has proposed a method of strengthening the signal in a mass histogram for any particle reconstruction. A program built in MATLAB has demonstrated the effectiveness of the method. Confirming this report's hypothesis opens up a new way to reduce background noises in mass reconstructions, which is vital in computing properties of particle accurately such as decay lifetime. In essence, this method replaces explicit physical parameter bounds with trained cluster centroids to serve as cut for tracks to be used for mass reconstruction.

A different approach of subsetting the dataset based on events instead of tracks could also show great possibility. This is in fact the method of choice by the LHC in the very first round of raw data filtering, down at the detector level, where only one in a million events are selected to be transmitted and reconstructed at their Tier 0 data centre[7].

Using this program to look for other particles would be another important validation of this method. Further comparisons of computing time for creating mass histograms between using the program in this report and the one found in CO03 would be a metric of accessing the practicality of this method.

References

[1] Cisco. The zettabye era - trends and analysis. `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html`, June 2016.

[2] M. Banerji, O. Lahav, C. J. Lintott, F. B. Abdalla, K. Schawinski, S. P. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar, A. Szalay, D. Thomas, and J. Vandenberg. Galaxy Zoo: reproducing galaxy morphologies via machine learning. *mnras*, 406:342–353, July 2010.

[3] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, 5:4308, July 2014.

[4] University of Oxford. Co03: Objects and classes. `https://www-teaching.physics.ox.ac.uk/CO-MATLAB/html/co03/co03.html`.

[5] University of Oxford. Np10: Search for mesons containing b-quarks. `https://www-teaching.physics.ox.ac.uk/practical_course/scripts/srv/local/rscripts/trunk/Nuclear/NP10/NP10.pdf`.

[6] Alan Chan. Finding the strange meson and baryon with object-oriented programming, March 2016.

[7] CERN. Processing: What to record? `http://cds.cern.ch/record/1997399`, August 2012.

A Code repository

All code used in this report is available at a Github repository at `https://github.com/pinealan/co03`

B Other results

The beginning of this piece of extra practical work was extremely exploratory. Machine learning in experimental physics is cross discipline in it's core, so it is rarely mentioned to undergraduates even in practical sessions, save included in a physics course syllabus. This leads to a lack of reference material suited at the level of this report, resulting in many iterations of the program until one came up with interesting results, which was extensively reported in the end.

One of the failed attempts was applying basic anomaly detection with a Gaussian kernel on tracks data, and to label tracks that are to be anomalies/tracks of interest. Those with some knowledge about particle physics in CDF colliders or anomaly detection would quickly see how this had failed. The distribution of multiple tracks parameters are in fact trivial. One such example would be the angle in transverse plane at closest approach, which is uniformly distributed across 0 to 2π. The same is true for daughter particle of any interaction, so there is no way to classify an anomaly with only this information.

Another trivial attempt was to simply treat the 5 track parameters

C Calculation for LHC data generation rate

The 18 ZB per annum were calculated from the numbers provided by CERN. As published on the CERN website[7], 600 million collisions happens every second, and each event averages to about 1 MB of data. Assuming that the LHC runs 24/7 for a whole year, without filtering or selection of any events/data, the number roughly comes out to be 18 ZB.

D Additional plot

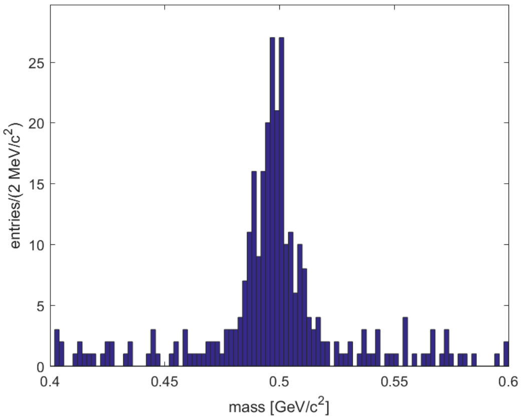

Figure 10: Original mass histogram with cuts to identify the K_S^0 mass peak