Anomaly and Event Detection for Unsupervised Athlete Performance Data

Jim O' Donoghue[1 2], Mark Roantree[2], Bryan Cullen[3], Niall Moyna[2], Conor O Sullivan[1], and Andrew McCarren[1]

[1] School of Computing
[2] Insight: Centre for Data Analytics
[3] School of Health and Human Performance,
Dublin City University, Glasnevin, Dublin 9, Ireland.

Abstract. There are many projects today where data is collected automatically to provide input for various data mining algorithms. A problem with freshly generated datasets is their unsupervised nature, leading to difficulty in fitting predictive algorithms without substantial manual effort. One of the first steps in dataset preparation and mining is anomaly detection, where clear anomalies and outliers as well as events or changes in the pattern of the data are identified as a precursor to subsequent steps in data mining. In the research presented here, we provide a multi-step anomaly detection process which utilises different combinations of algorithms for the most accurate identification of outliers and events.

1 Introduction

Anomaly detection is an important component in data science. In many situations, researchers are confronted with datasets which possibly contain a large number of features and more often than not, incorporates outliers and missing data. Implementing dimensionality reduction and incorporating cluster analysis techniques such as K-means are commonly used in performing unsupervised learning tasks in such data. In fact, principal components are the continuous solutions to the discrete cluster membership indicators for K-means clustering [4].

Anomalies are generally defined as unusual events which occur within a dataset, where a subset of these events are *outliers*. Outliers are occurrences that make either no physical sense, or appear so extreme they are considered probabilistically infeasible. The identification of outliers and anomalies is critical in avoiding poorly-fitting models for many machine learning algorithms [17].

 In: R. Bergmann, S. Görg, G. Müller (Eds.): Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. Trier, Germany, 7.-9. October 2015, published at http://ceur-ws.org

1.1 Problem Background

The physiological demands of any sport are determined largely by the activity patterns of the game. Similar to other team sports, Gaelic football [1] involves repeated, short duration, high intensity bouts of anaerobic exercise interspersed with sustained light to moderate aerobic activity. The duration of these intervals of high intensity are largely unpredictable, due to the fact that they are imposed by the pattern of play, and can vary greatly from player to player and from one game to another. On average, senior players perform 96 bursts of high intensity activity lasting 6 seconds followed by an average recovery of 37 seconds. These players typically work at 80% of their maximum heart rate (HRmax) and cover an average distance of 8.5 km during competitive games. Superimposed on the physiological demands of match play are key technical activities such as winning possession of the ball, evading opponents and breaking tackles which involve high running velocities, agility, strength and power.

As part of the process for collecting data on each athlete, global positioning software (GPS) has become increasingly popular among sport scientists as a method of tracking movement patterns in many field based sports [13]. Modern GPS devices are portable, robust and lightweight making them particularly suited to field based sports. From a sports science perspective, the initial aim is to evaluate the characteristics and fitness levels of Gaelic football players and compare the physical and fitness characteristics relative to each playing position. The subsequent goal is to predict when these players are approaching or have reached optimal performance level. This requires the generation of a sufficiently rich dataset to build an initial model before it can be used in a real time environment. At each of 17 competitive games, 10 out of 15 players in the team were fitted with appropriate sensor devices to record heart rate, speed, distance covered, GPS location and accelerometer values, recording at multiple times per second. The resulting dataset contained in excess of 200 million values. Simple detection methods [14] can be useful for more obvious outliers, but encounter limitations in discerning more subtle anomalies. Due to the nature of contact sport, the devices incur a number of blows during each game, introducing many potential anomalies. The work presented here focuses on *anomaly detection* in unsupervised data.

Contribution. If one uses unsupervised clustering techniques such as K-means to determine an anomaly, then K (proposed number of clusters) can be calculated as part of the X-means cluster estimation technique [12]. However, such algorithms rely on the choice of good initial starting points [5] to find workable solutions. Our contribution is the development of an unsupervised outlier detection algorithm for time series data which employs both univariate and multivariate approaches for a more accurate detection rate and further our previously developed learning framework [11] to incorporate anomaly detection as well as classification. The univariate method is based on the approach taken in [15] but extends this work to manage time series data, while the multivariate approach builds upon the work of [16] and introduces a secondary decision statistic which detects when variables are unusually static. In dynamic environments

such as GAA matches extremely static measurements are equally as anomalous as those which are extremely varying.

Paper Structure. The paper is structured as follows: in Section 2, we discuss related work in the area; in Section 3 we provide a description of our approach and detail how we identify and classify anomalies; we describe our experimental setup together with an evaluation of the results in Section 4; and finally, we present our conclusions in Section 5.

2 Related Research

In [8], the authors present a novel approach for anomaly detection incorporating both density and grid-based clustering algorithms. Their primary focus is high dimensional data and they test their algorithm on the KDD Cup 1999 network dataset [9]. The approach taken was to optimise the pMafia algorithm, using a Frequency-Pattern tree in an intermediate step in order to improve the detection rate. Similar to our approach, they provide an unsupervised anomaly detection algorithm. However, in their evaluation it was shown that the improvement in detection rate had a negative side effect in generating a higher number of false positives. By their own admission, the algorithm works best for datasets with certain characteristics i.e. data points sought will be statistically different from normal data. This means that if there is an entire window of anomalous data, this may affect the performance of the detection method.

In [3], the authors present an algorithm for anomaly detection in multivariate time series data. Their goal is to capture relationships across variables and by doing so, identify different types of anomalies that occur in the time series dataset. As we use a real-world dataset, our comments concern their evaluation with the several time series datasets from [2] and not the experiments with synthetic data. The evaluation used a sliding window of length 6 and clearly demonstrates that for time series, a subsequence of data points outperforms the basic data point approach, which is similar to our findings. Apart from the fact that our research is based fully on a real-world dataset using unsupervised learning, we also employ both univariate and multivariate algorithms to deliver a higher performance in anomaly and outlier detection.

The authors of [17] developed the Robust Support Vector Machine that demonstrates its ability to identify images (bullet holes) when outliers exist. This algorithm is an improvement on the standard support vector machine (SVM) algorithm as the incorporation of the averaging technique to an SVM makes the decision function less susceptible to outliers and thus, avoids overfitting. The process could be used to identify outliers however it requires a supervised training dataset which, as with our work, is not always available.

In [12], the authors propose an extension to K-means algorithm called X-means to identify outliers in Gaussian datasets without specifying the initial number of suspected clusters. The algorithm performed exceptionally well with regard to identifying the exact number of clusters and functioned commensu-

rately against the K-means algorithm. However, the X-means algorithm is vulnerable to initial estimates and may attain a sub-optimal minima.

In [4], the authors demonstrate mathematically that principal components are the continuous solutions to the discrete cluster membership indicators for K-means clustering. This idea is extended in our work by using principal components as the basis for a decision based system to detect outliers in truly unsupervised data. In [16], the authors propose a novel Principal Components classifier in order to detect anomalies in the case of network intrusion identification on the KDD Cup 1999 network dataset, whose aim was to detect attacks on network access data. While they produced a false hit rate of only 1% and their PCC remained robust to false positives, all the other metrics degraded significantly in terms of quality. Our approach uses the PCC but extends it to detect highly static variables with a secondary chi-squared decision statistic. Furthermore, our training data is real-world, containing anomalous examples, whereas in [16] their classifier was trained on completely clean, non-anomalous data.

3 Outline Approach

Our anomaly detection algorithm has 4 primary components: Boundary Detection, Univariate Outlier Detection, Principal Component Transformation and Principal Component Classification. The role of the algorithm is to detect anomalies and classify these as outliers (data points which are far outside the expected norm) or events (samples which demonstrate a clear shift in the pattern of the data). Each of the algorithm's components detects anomalies within the dataset with the exception of Principal Component Transformation which transforms the data only. We now provide a brief overview of each component.

3.1 Boundary Detection

This represents a pre-processing stage where clearly erroneous data points are removed. This can only take place for those features where a domain expert has clearly specified boundaries, outside which data values make no sense. For example, if a player had a heart-rate below 40bpm or above 250bpm, the hardware has clearly malfunctioned. The process eliminates obvious errors so that later calculations are not affected.

3.2 Univariate Outlier Detection

This stage is based on Chauvnet's method, an approach for univariate outlier detection found in [15]. Euclidean distance [16] has shown to be of little value with this type of time series data as it detects far too many outliers and thus, excludes large amounts of data. Early experiments with Euclidean distance with sport scientists for manual evaluation confirmed this assumption. A first step marks a sample as an *anomaly_low* or *anomaly_high*, while a second classifies these anomalies as either: *outlier*, *event* or *untrue* (not an anomaly). Before

detecting anomalous values, the algorithm first calculates summary statistics of mean, variance and standard deviation, denoted by $\bar{x}, \sigma^2$, and σ respectively, for each feature x_i in the dataset X. Unlike [15], where anomalies are detected with standard deviation alone, our univariate approach incorporates time differencing and compares the current time difference against *previous* time differences. If it is significantly different, we then compare with the *future* time difference to confirm the data point is an anomaly or outlier. This is achieved by iterating through each feature and examining every time point with a t-distribution coefficient for natural confidence intervals on the differenced data (which removes any non-stationary components of the data), a crucial factor for time series data as it is non-stationary.

The first steps of the algorithm presented get the dimensions of the dataset $|X| = (T \text{ x } n)$, where T is the number of time-points or samples and n is the number of features where $\forall x_{t,i} \in X$, $t \in (1, 2, \ldots, T-1, T)$ and $i \in (1, 2, \ldots, n-1, n)$. Subsequently a t-distribution coefficient β of 3 was chosen to give approximately a 99.9% confidence interval in order to detect anomalies with an α of 0.001 for each feature, where α is the probability of a false alarm. In concrete terms, this coefficient is multiplied by the standard deviation both positively and negatively ($\pm 3\sigma \Delta x_i$) for each feature in order to calculate upper and lower bounds for anomalies.

3.3 Principal Component Transformation

The aim of this step is to compute key characteristics and to transform the data for the final stage in the algorithm.

1. Recalculate the summary statistics from the previous stage. This is necessary due to removed outliers.
2. Impute the missing values.
3. Calculate the correlation matrix in order to provide input to the Principal Components Analysis.
4. Calculate eigenvectors and eigenvalues. The eigenvectors enable the creation of an orthogonal representation of the dataset, used to derive the principal components. Eigenvalues measure the energy contribution of each of the principal components, as well as providing input into the principal components classifier.
5. Standardise each feature to have unit variance. For each feature x, this involves subtracting the mean, $\bar{x}$ from x and dividing the result by the standard deviation, $\sigma^2 x$.
6. Compute the transformed dataset. Principal Component Analysis (PCA) provides an orthogonal representation of the data, describing it in terms of the axes of most variation for each component.

Before transforming the data into its principal components, it is differenced at a one second time lag and transformed into a 5 second moving average, centred on the value being transformed, essentially filtering noise from the data. Missing

values are imputed with the R Amelia [6] package. Amelia is a multi-variate imputation mechanism which infers missing data in a single-cross section from times series and is the only R component in a Python application. The use of R was necessary as Amelia was not available in Python and was evaluated to best suit our imputation needs. Employing bootstrapping and Expectation Maximisation, it allows for imputation from the posterior distribution of the data.

After imputation, it is necessary to determine how much the features change together by calculating the correlation matrix. The dimensions of this matrix are (p x p) where p is the number of features in the dataset.

Algorithm 1 Data Transform

function DATATRANSFORM(X)
  $X^{lagl} \leftarrow moving\_avrg(X)$ ▷ transform data into moving average at a lag of l
  $X^{imp} \leftarrow amelia(\Delta X^{lagl})$ ▷ impute missing data with Amelia on the differences
  $cov(X_{imp}, Y_{imp}) = \sum_{t=1}^{T} \frac{(x_{t,i}^{imp} - \bar{x_i}^{imp})(y_{t,i}^{imp} - \bar{y_i}^{imp})}{T-1}$ ▷ calculate covariance
  $corr(X^{imp}, Y^{imp}) = \frac{cov(X^{imp}, Y^{imp})}{\sigma X^{imp} Y^{imp}}$ ▷ calculate correlation
  $E = (e_1 \dots e_p)$ and $e_{vals} = \lambda_1 \dots \lambda_p$ ▷ calculate the eigenvalues and vectors
  for $x_{t,i}$ in x_i where $t \in 0 \dots T$ **do** ▷ standardise the data
    $Z_{t,i} = \frac{x_{i,t} - \bar{x}_i}{\sigma^2 x_i}$
  end for
  $P = (E^\top Z^\top)^\top$ ▷ calculate principal components
  return X^{imp}, E, e_{vals}, P
end function

The eigenvectors E are then calculated on the non-standardised data using the correlation matrix (whose calculation effectively standardises the data) where each eigenvector $e_i \in e_1, e_2, \dots, e_p$. Eigenvalues $\lambda_1, \dots, \lambda_p$ are also calculated. Once the data is standardised as Z, the result is multiplied with the eigenvectors which gives the principal components of the original dimensions (T x n).

3.4 Principal Component Classification

The final stage has three main steps.

1. Calculate the number of major and minor components to use by calculating the cumulative percentage of the total eigenvalue energy for each.
2. Calculate the classification value or *test statistic* for each sample.
   (a) Sum the major components divided by their eigenvalues, giving you the chi-squared test-statistic.
   (b) Calculate the same value for minor components.
3. Classify anomalies using significance values calculated with the chi-squared distribution

(a) Use the test statistic generated from step 2, compute the decision statistics with the chi-squared cumulative distribution function and compare this to a chi-squared distribution with `num_components` degrees of freedom and all false alarm rates α providing the confidence interval. The chi-squared distribution was employed as we observed that the distribution of differenced, standardised variables at the univariate stage demonstrated a normal distribution.
(b) If confidence interval exceeds either of the chosen decision statistic thresholds (e.g. $> 95\%$ or $< 0.001\%$), for either the major or the minor components test statistics (who have the same false alarm α pairs), the data instance is classified as anomalous.

In algorithm 3, we first calculate the number of principal components involved and determine how much variation in both the major and minor components is to be included in the classifier. The percentage variation thresholds are set and subsequently the number of components to use are calculated with Algorithm 2 which takes the eigenvalues, type of components being summed (string of 'major' or 'minor') and the desired percentage eigenvalue energy (variation) as parameters. Eigenvalues are initially summed to calculate the total variance and then the parameter num_components and current_sum (running total) are initialised to zero before being calculated.

In lines 5 to 12 of algorithm 2, for each eigenvalue there is a check to see if the current percent variance sum is less than the desired variance. If this is the case, the number of components is incremented and added to variance sum is the current eigenvalue divided by the eigenvalue total sum, as this gives the percentage variance. It is worth noting that if it is the major components sum, we begin at $i = 1$ to start with the major components but with the minor components, we begin with the last eigenvalue $i = p - 1$ where p is the total number of eigenvalues.

Once the number of major components q and the number of minor components r are found, the chi-squared test statistics are then calculated for both component types by summing the value for the component p_i at time-point t divided by the appropriate eigenvalue λ_i up until the number of components is exhausted. In the case of the major components, the process begins at 1 and stops at component q whereas in the case of the minor components, it begins at the last component p and sum to $p - r$ as shown in lines 7 to 12.

In lines 13 and 14, the decision statistic is calculated by $1 -$ the chi-squared cumulative distribution function with q degrees of freedom for the major components and r for the minor components. Given that our data follows a multivariate normal distribution the overall false alarm rate is given by Equation 1.

$$\alpha_{total} = \alpha_{major} + \alpha_{minor} - \alpha_{major}\alpha_{minor} \tag{1}$$

We then chose the varying and static false alarm rates α_{large} and α_{static} (line 15). If a particular decision statistic is greater than $1 - \alpha_{static}$ or less than $1 - \alpha_{large}$ i.e. a certain significance threshold, the row is classified as anomalous

Algorithm 2 Calculate Number of Components

function GETNUMCOMPONENTS(e_{vals}, *type*, *variance*)
  $\lambda_{total} = \lambda_1 + \lambda_2 + \ldots + \lambda_{p-1} + \lambda_p$
  $num\_components \leftarrow 0$
  $var\_sum \leftarrow 0$
  for all λ_i where $i \in 1, 2, \ldots, p$ **do**
    if $type == 'major'$ && $var\_sum < variance$ **then**
      $num\_components += 1$
      $var\_sum += \frac{\lambda_i}{\lambda_{total}}$
    else if $type == 'minor'$ && $var\_sum < variance$ **then**
      $num\_components += 1$
      $var\_sum += \frac{\lambda_{p-i}}{\lambda_{total}}$
    end if
  end for
  return *num_components*
end function

Algorithm 3 Principal Components Classifier

function PRINCIPALCOMPONENTSCLASSIFY (PCC)(P, E, e_{vals})
  $p \leftarrow |P|$
  $var_{maj} \leftarrow$ percentage variance for major classifier
  $var_{min} \leftarrow$ percentage variance for minor classifier
  $q =$GetNumComponents($e_{vals}, 'major', var_{maj}$)
  $r =$GetNumComponents($e_{vals}, 'minor', var_{min}$)
  for $p_i \in P$ where $i \in 1, 2, \ldots, s-1, s$ **do** ▷ For each sample
    $test_t^{maj} += \frac{p_i}{\lambda_i}$
  end for
  for $p_j \in P$ where $j \in p, p-1, \ldots, p-d$ **do**
    $test_t^{min} += \frac{p_j}{\lambda_j}$
  end for
  $Decision_{maj} = 1 - P(\frac{q}{2}, \frac{test^{maj}}{2})$ ▷ 1 - chi-squared cumulative distribution function
  $Decision_{min} = 1 - P(\frac{r}{2}, \frac{test^{maj}}{2})$
  $c_{lrg} = 1 - \alpha_{large}$ ▷ false alarm rates $\alpha_{large}^{maj} = \alpha_{large}^{min}$
  $c_{stat} = 1 - \alpha_{static}$ ▷ $\alpha_{static}^{maj} = \alpha_{static}^{min}$
  for all $\delta_{major,t} \in Decision_{maj}$ and $\delta_{minor,t} \in Desicision_{min}$ **do**
    if $\delta_{major,t} < c_{large} \mid \delta_{minor,t} < c_{lrg} \mid \delta_{major,t} > c_{stat} \mid \delta_{minor,t} > c_{stat}$ **then**
      X_t is anomalous
      $Anomalies_t \leftarrow True$
    end if
  end for
  return *Anomalies*
end function

as in line 18 of Algorithm 3. This implies a very large or very small degree of variation at this time-point. Our extension to the PCC captures where there is very little or no variation from sample to sample. Our features should be non-static and should be constantly changing, this minor variation parameter was a very important factor and was not incorporated by [16]. Finally, as our aim was to identify anomalous time-periods as opposed to particular time-points (as a time-point itself is not anomalous) and due to our rolling average transformation, we incorporated time-points within an 11 point centred window as anomalous.

4 Evaluation and Analysis

In this section, we briefly describe our dataset, approach to evaluation and provide a detailed analysis of experiments and results.

4.1 Experiment Setup

Experimental Set-up. Experiments were performed on a Dell Optiplex 790 running 64-bit Windows 7 Home Premium SP1 with an Intel Core i7-2600 quad-core 3.40 GHz CPU and 16.0GB of RAM. The code for the experiments was developed in Python using the Enthought Canopy (1.5.4.3105) distribution of 64-bit Python 2.7.6 and developed in PyCharm 4.5 IDE, making use of NumPy 1.8.1-1[18], Pandas 0.16.0[10] and SciPy 0.15.1[7] for mathematical and statistical operations and data manipulation. The imputations were performed with R and Amelia, package version 1.0 [6].

Dataset. For our evaluation, we used the results of one match with 81,165 instances in the dataset. The hardware devices generate at least 10 sets of measures per second, which were averaged giving one set of measures per second and providing just under 8,000 instances. As only heart rate and distance covered were used in this experiment and only 9 players generated data, each instance had 18 features per second, namely 9 sets of heart rates and distances. We selected the data beginning at the pre-match warm-up, until 4 minutes and 30 seconds after the end of the match which left a total 6,211 instances.

Evaluation Design. The design of the outlier detection algorithm allows for evaluation of different processes in combination. The univariate (P2) and multivariate steps (P3 & P4) were evaluated in isolation and with bounds detection (P1) added. We also evaluated after the univariate step (P1 & P2), without the univariate step (P1, P3, & P4) and with both (P1, P2, P3, & P4).

Before evaluating combinations of processes, we first tested various numbers of major components in isolation, at various false alarm rates α before performing the same evaluation for the minor components in the PCC. The secondary alpha measure added to P4 to detect unusually static time-points was kept at 0.01% for all experiments, keeping the measure sensitive. Window-size was also not varied and kept at 11, to account for the rolling average transformation.

Each anomaly detection algorithm was trained on the dataset in its entirety without partition. For testing, a random subset of the dataset was presented

to the sports scientist involved in the original data collection for classification. He classified the subset as 69.89% anomalous with the remainder being non-anomalous. We then examined and cross-referenced the relevant subset of each result (from each configuration) and calculated an accuracy score for each. The evaluation metric use for all experiments was accuracy where $accuracy = \frac{TP+TN}{P+N}$.

4.2 Evaluation: Results and Analysis

Tuning the PCC (P3 & P4) Table 1 shows the results of our empirical search for the parameters to use in our PCC (P3 & P4). We first vary the percentage of total contribution to the classifier by the major components only ($r = 0$), in a range from 30% to 70% (columns 1 to 4) and then repeat the process with the minor components only ($q = 0$), in a range of 10% to 20%. We evaluate both at various false alarm rates α, in increments from 1% to 10%.

Table 1: Evaluation of Major only $r = 0$ and Minor only Components $q = 0$

α	MajC 30%	MajC 40%	MajC 50%	Maj 60%	MajC 70%	Min 10%	MinC 20%
1%	0.3656	0.3978	0.3979	0.3979	0.4409	0.4839	0.5807
2%	0.3979	0.3978	0.3871	0.4086	0.4409	0.5269	0.5807
4%	0.4409	0.4301	0.4301	0.4409	0.4409	0.5054	0.5914
6%	0.4624	0.4409	0.4409	0.4731	0.4731	0.5161	0.6237
8%	0.4839	0.4624	0.4731	0.4731	0.5054	0.5484	0.6237
10%	0.5054	0.4946	0.4946	0.4839	0.5054	0.5807	0.6022

The results in Table 1 furnishes us with interesting findings. First, the highest accuracy achieved with the major components (MajC) alone was just over 50% with an $\alpha = 10\%$. This was found at both the 30% and 70% contributions respectively. Increasing the number of major components actually decreased the accuracy for higher false alarm rates but slightly improved the values for lower false alarm rates. Our second finding is that when only the *minor* components were tested (MinC), a higher accuracy was immediately achieved. This is surprising as the major components explain the major variation in the dataset leading us to the conclusion that, at least for the dataset in question the components that contribute less to the dataset as a whole actually have greater capacity for modelling anomalies, suggesting that anomalous examples in this data are subtle and contribute to noise in the dataset as minor components generally contain a relatively high degree of the noise in a dataset.

After the analysis of Table 1 we chose the percentage contribution for the Major components to be 70% and those of the minor to be 20%. The ROC curve of this can be seen in Figure 1. Our first observation was the results were not optimal, at 66.67% accuracy this is just a 7% increase to using the minor components in isolation. When we again analysed the results of Table 1

we decided to test a configuration where an even greater emphasis was given to the minor components and less to the major components, as we realised that increasing the major components from 30% to 70% gave no great gains in accuracy. We chose to double the contribution from the minor components to 40%. The results again improved with an accuracy increase of just under 11% to 70.97% compared to using either of the major or minor components in isolation, and can also be seen in Figure 1.

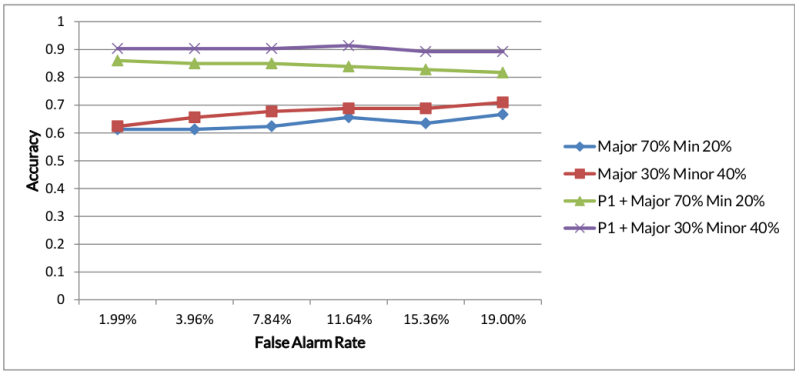

Fig. 1: P3/P4 and P1+P3/P4 Results at Various False Alarm Rates

Comparing Processes. Table 2 shows the comparison of the events found in P2 to the anomalies found with P3/P4, when taken as components in isolation, to those where P1 was coupled with P2 and with P3/P4. We chose to exclude the results from P1-P2-P3-P4 as only six clear outliers were found in P2 to be excluded from the P3/P4 processes and therefore did not have a great impact on the model. The results of the process in its entirety is also shown in Figure 1, demonstrating a clear gain in accuracy.

Measure	**P2 Chvnts. Univariate**	**P3/P4 PCC**	**P1+P2**	**P1+P3/P4**
Accuracy	0.5376	0.7097	0.8495	0.8925

Table 2: Comparing Processes

As we can see from 2, the PCC multivariate anomaly detector is far more accurate than the univariate outlier process, but once P1 is added, the results approach in accuracy. This is primarily due to zero values now being classed as anomalies from the added boundary detection step. A number of samples given to the sports scientist were found to contain zeros which he immediately classed as anomalies. Further evaluation could perhaps exclude samples containing zeros as these classifications do not accurately test the algorithm as this fundamental

step is easy to compute and from the sports scientist's perspective, not difficult to identify via manual inspection. Given this caveat, once P1 was added we still achieved a final classification accuracy of 0.8925 once our algorithmic components were combined, showing the performance of the process as a whole was greater than any constituent process in isolation.

Some general findings include that anomalies (when the rolling window was not used), often occurred together in a sequential series. This gives credence to the hypothesis that in our time-series dataset, anomalies occurred in windows rather than at particular time-points. Furthermore, a subset of the events found from P2 and the anomalies found with P3/P4 overlapped at certain points, indicating strong events at these points. Finally, in an examination of the false positives, we found a number of the distance variables actually remained static, an unusual event for a GAA match and similarly for other sports events. Further evaluation will involve testing this hypothesis with the sports scientists, as we posit certain events even in the relatively small evaluation subset could have been missed upon manual inspection, due to fatigue or other factors incurred by examining vast swathes of data by eye. This secondary analysis could provide further motivation to this work, that is identifying anomalies in data that might have been missed by manual inspection.

5 Conclusions

Newly generated datasets often prove difficult for data mining as they can contain erroneous data-points and are often unsupervised (without classifications); datasets produced in areas such as sports science exemplify this. The first step in addressing these problems is anomaly detection, both to remove or adjust those values which are clear outliers, and to detect patterns which are signs of an event or change in the data. In this paper, we presented a novel anomaly detection algorithm which utilises both univariate and multivariate steps enabling us to determine which approach works best for a unsupervised time series dataset. Our results demonstrated the effectiveness of a combined approach when compared to even an improved univariate approach and that a multivariate approach outperforms it's univariate counterpart when used in isolation.

Acknowledgments

This work was partially funded by FP7 project Grant Agreement Number 304979 and also by Science Foundation Ireland under grant number SFI/12/RC/2289.

References

1. Rules of gaa football, 2015. Online; last accessed: 09/07/2015; `www.gaa.ie/about-the-gaa/our-games/football/rules`.

2. Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, July 2015. www.cs.ucr.edu/~eamonn/time_series_data/.
3. Haibin Cheng, Pang-Ning Tan, Christopher Potter, and Steven A Klooster. Detection and characterization of anomalies in multivariate time series. In *SDM*, pages 413–424. SIAM, 2009.
4. Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29. ACM, 2004.
5. AM Fahim, AM Salem, FA Torkey, G Saake, and MA Ramadan. An efficient k-means with good initial starting points. *Georgian Electronic Scientific Journal: Computer Science and Telecommunications*, 2(19):47–57, 2009.
6. Matthew Blackwell James Honaker, Gary King. *Amelia II: A Program for Missing Data*, 2014. R package version 1.0 — For new features, see the 'Changelog' file (in the package source).
7. Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. Online; accessed 2015-06-29; Available on: http://www.scipy.org/.
8. Kingsly Leung and Christopher Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, pages 333–342. Australian Computer Society, Inc., 2005.
9. Moshe Lichman. 1999 kdd cup dataset, UCI machine learning repository, 2013. Dataset; Available on: https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data.
10. Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.
11. Jim ODonoghue and Mark Roantree. A framework for selecting deep learning hyper-parameters. In *Data Science*, pages 120–132. Springer, 2015.
12. Dan Pelleg, Andrew W Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, pages 727–734, 2000.
13. Mark Roantree, Donall McCann, and Niall Moyna. Integrating sensor streams in phealth networks. In *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*, pages 320–327. IEEE, 2008.
14. Mark Roantree, Jie Shi, Paolo Cappellari, Martin F. OConnor, Michael Whelan, and Niall Moyna. Data transformation and query management in personal health sensor networks. *Journal of Network and Computer Applications*, 35(4):1191 – 1202, 2012. Intelligent Algorithms for Data-Centric Sensor Networks.
15. Stephen M Ross. Peirce's criterion for the elimination of suspect experimental data. *Journal of Engineering Technology*, 20(2):38–41, 2003.
16. Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, DTIC Document, 2003.
17. Qing Song, Wenjie Hu, and Wenfang Xie. Robust support vector machine with bullet hole image classification. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 32(4):440–448, Nov 2002.
18. Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.