Why is this an Anomaly? Explaining Anomalies using Sequential Explanations

Tshepiso Mokoena[a,b], Turgay Celik[a,b,*], Vukosi Marivate[c]

[a] *School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg, South Africa*
[b] *Wits Institute of Data Science, University of the Witwatersrand, Johannesburg, South Africa*
[c] *Department of Computer Science, University of Pretoria, Pretoria, South Africa*

Abstract

In most applications, anomaly detection operates in an unsupervised mode by looking for outliers hoping that they are anomalies. Unfortunately, most anomaly detectors do not come with explanations about which features make a detected outlier point anomalous. Therefore, it requires human analysts to manually browse through each detected outlier point's feature space to obtain the subset of features that will help them determine whether they are genuinely anomalous or not. This paper introduces sequential explanation (SE) methods that sequentially explain to the analyst which features make the detected outlier anomalous. We present two methods for computing SEs called the outlier and sample-based SE that will work alongside any anomaly detector. The outlier-based SE methods use an anomaly detector's outlier scoring measure guided by a search algorithm to compute the SEs. Meanwhile, the sample-based SE methods employ sampling to turn the problem into a classical feature selection problem. In our experiments, we compare the performances of the different outlier- and sample-based SEs. Our results show that both the outlier and sample-based methods compute SEs that perform well and outperform sequential feature explanations.

Keywords: Outlier explanation, Sequential feature explanation, Sequential explanation, Anomaly validation, Explainable AI

1. Introduction

Anomalies are known as "rare" data points generated by a process that is distinct from the process generating the "normal" data points (Siddiqui et al.,

*Corresponding author
Email addresses: Tshepisomokoena20@gmail.com (Tshepiso Mokoena), celikturgay@gmail.com (Turgay Celik)

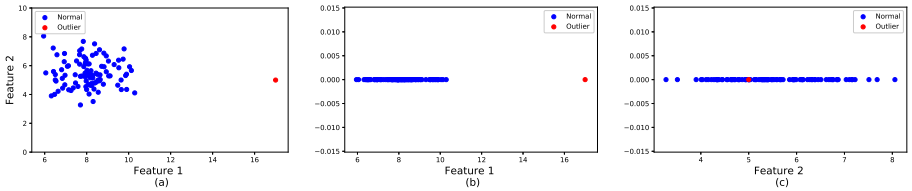

Figure 1: Plot (a) shows an outlier detected by an arbitrary anomaly detector in a 2-dimensional feature space, while plots (b) and (c) show the detected outlier visualised in Features 1 and 2, respectively. It is clear from plots (b) and (c) that the outlier detected in the plot (a) is outlying only in Feature 1.

2019a). Anomaly detection is the problem of identifying the process that generates the anomalies within a dataset. A variety of application domains have applied anomaly detection over the years. An anomaly in any one of these domains could represent an interesting phenomenon or a severe cause of concern which will need to be addressed urgently by a human analyst analysing the data. An anomaly in insurance claims data could represent fraudulent claims, whereas an anomaly in astronomical data could represent discoveries of new galaxies or new planets.

Analysts use a variety of anomaly detection methods in these domains to identify anomalies. In most cases, the process generating the anomalies is unknown, resulting in anomaly detection methods in real-world applications to operate in an unsupervised mode by looking for outliers in the dataset, hoping that they are anomalies (Siddiqui et al., 2019a; Emmott et al., 2015). An outlier is a data point that deviates so much from the other data points in the dataset. It arouses suspicions that a different process generated it. However, outliers do not always equate to anomalies. For example, a company's computer security application may exhibit an unusually high amount of copying and printing activities at the end of the month. These activities could cause concern compared to other days of the month, making them appear highly outlying and classified as anomalous activities. However, because these activities occurred at the end of the month and there are month-end processes such as payrolls and monthly audits that need to be copied and printed, there is a valid explanation behind the high amounts of copying and printing exempt them from being anomalous. Due to this, a human analyst must investigate the outliers to decide which ones are most likely to be true anomalies and deserve further investigation.

An analyst provided with an outlier point faces the challenge of identifying the features that makes it outlying. In most cases, the analyst needs manually browse through the outlier point's entire feature space to determine the outlying subset of features that will give them enough information to classify whether the outlier point is genuinely anomalous or not. Identifying the outlying subset can be extremely challenging even for a few features characterise the feature space, especially when feature interactions are critical in the decision making (Siddiqui et al., 2019a).

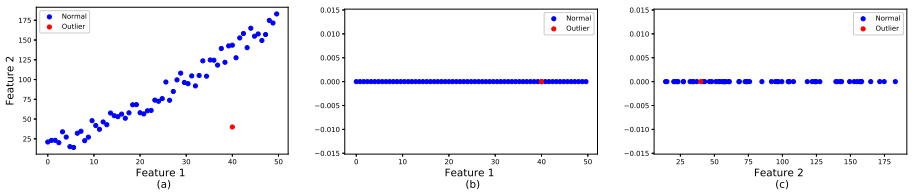

Figure 2: Similar to Figure 1, however, the detected outlier in this case is not outlying in Feature 1 or Feature 2. The outlier is outlying due to the feature interactions of Feature 1 and Feature 2

For example, Figures 1 (a) and 2 (a) show different scenarios of outliers in two-dimensional feature space datasets. For each figure, the red data point represents an outlier detected using an arbitrary anomaly detector, while the blue data points represent the normal data points. We observe from Figure 1 (a) that the outlier point is outlying because it does not form part of the cluster of blue data points. In contrast, from Figure 2 (a), we observe that the outlier is outlying because it does not form part of the linear trend followed by the blue data points. An analyst analysing these outlier points would want to know along which features the outliers are outlying to make a judgement on whether they are truly anomalous or not. Figures 1 (b) and (c) show visualisations of the outlier in Figure 1 (a) in Features 1 and 2 respectively. We observe from Figure 1 (c) that the outlier appears normal in Feature 2, whereas, from Figure 1 (b) the outlier is outlying in Feature 1. Therefore, Feature 1 is the outlying feature subspace that the analyst must identify to decide whether it provides enough information to classify the outlier as anomalous or not. On the other hand, we observe from Figures 2 (b) and (c) that the outlier in Figure 2 (a) is not outlying when considering Features 1 and 2 independently. The outlier is only outlying due to the interactions of Feature 1 and Feature 2 in Figure 2 (a).

To assist the analyst with identifying the outlying subset of features, Siddiqui et al. (2019a) developed an outlier explanation called a sequential feature explanation (SFE). An SFE for a detected outlier point is an ordered sequence of features. The order of the features in the SFE indicates their importance of explaining to the analyst, which features makes the outlier point outlying. An SFE for a detected outlier point incrementally presents the outlying features to the analyst by first presenting the first feature in the SFE. If the analyst can decide that the outlier point is anomalous or with only one feature presented, then the analyst's job is done. If the analyst cannot decide based on the single feature presented, then the next feature in the SFE is presented. Now the analyst has seen the two most prominent features that explain why the outlier point is outlying. The process will continue until the analyst has acquired enough information to decide whether the outlier point is an anomaly or not.

The current SFE computation methods (Siddiqui et al., 2019a) only work for anomaly detectors with an assumption that they can compute outlier scores for an outlier point in any given subset of features. Siddiqui et al. (2019a) em-

ployed an Ensemble Gaussian Mixture Model (EGMM) (Glodek et al., 2012) as their anomaly detector. They used the model's marginal densities to compute the outlier scores for an outlier point in any given subset of features. With this assumption, the SFEs generate explanations that explain why the EGMM judged an outlier point to be outlying and not why the outlier is outlying in general. In practice, most anomaly detectors do not provide these scores trivially. It then makes it impossible to compute SFEs that explain why the particular anomaly detector judged an outlier point to be outlying except for density-based anomaly detectors such as the EGMM (Siddiqui et al., 2019a). The dependence of the assumption on the anomaly detector is one of the disadvantages of SFEs because computing the outlier scores in any given feature space is a non-trivial task for most anomaly detectors. Another disadvantage of SFEs is their sequential greedy nature which results in the "nesting effect". Once a feature is added to the SFE, it cannot be removed, and this could provide sub-optimal explanations to the analyst. For example, the subset of the five best features in the SFE must contain the subset of the one, two, three and four best-chosen features in the SFE. In practice, the five best features may not contain any of the best one, two, three, or four best-chosen features (Nakariyakul and Casasent, 2009), which could lead to the analyst making a wrong decision or requiring the presentation of more features from the SFE to make their final decision.

The minimum feature prefix (MFP) (Siddiqui et al., 2019a) is used to evaluate SFE methods. Because there are no ground truth explanations, the approach's idea is to construct a simulated analyst for each anomaly detection benchmark using supervised learning on the ground truth about which points are anomalies and normal. The simulated analyst provides a probability score of how anomalous a data point is. The MFP is the minimum number of features required for the analyst to reach a probability threshold of how anomalous a data point is given the number of features presented in the SFE. However, the first problem we observe regarding the evaluation of SFEs using MFPs is that two or more explanations with the same MFP could be considered to be equal even if one of the SFEs returns significantly higher probabilities than the other. For example, if the first four features in one SFE returns probabilities of $0.1, 0.3, 0.5, 0.82$ and another SFE returns probabilities of $0.7, 0.75, 0.78, 0.95$, then they both have an MFP of 4 if the simulated analyst's probability threshold is 0.8. However, the second SFE is far more superior than the first SFE because, for every sequential feature presented to the analyst, it always returns higher probabilities. The second problem is that the supervised learning model used to compute the probabilities of the simulated analyst does not take class imbalances into account. Anomaly detection benchmarks are imbalanced because the anomalies are generally significantly fewer than the normal data points. Therefore, fitting a model that does not take class imbalance into account introduces a bias towards the normal class of data points (Krawczyk, 2016). The third problem we observed is that the outlier point whose probability is being predicted in the given feature space is used to train the simulated analyst model. Including the data point that requires a prediction in the training set of the model introduces a bias towards its probability prediction.

In this paper, we make two contributions. The first contribution is to introduce a new outlier explanation called a sequential explanation (SE) to address the shortcomings of the SFEs. The second contribution is introducing a new evaluation method called the area under the analyst certainty curve (AUCC) that addresses the shortcomings of the MFP. Similar to an SFE, an SE for a detected outlier point is presented sequentially to the analyst. The SE first presents the analyst with the first feature subset containing one feature. If the analyst can decide that the outlier point is anomalous or not based on that one feature, then the analyst's job is done. If the analyst cannot decide based on the single feature presented, then the next feature subset containing two features in the SE is presented. The process of presenting the analyst with the best feature subsets in increasing size will continue until the analyst has acquired enough information to decide whether an outlier point is an anomaly or not. The SE has three advantages over the SFE. Firstly, SEs do not enforce the SFE assumption, which restricted SFEs to only work on "score-based" anomaly detectors that can compute outlier scores for an outlier point in any given subset of features. Secondly, SEs do not suffer from the nesting effect. Thirdly, the explanations of the SEs are not subjective to the anomaly detector used, and the SE can be used as an additional step for any black-box anomaly detection algorithm to explain what makes the detected outlier points outlying. We introduce two methods of computing SEs called the sample and outlier-based SEs. Sample-based SEs turn the problem of computing SEs into a classical supervised feature selection problem by using a sampling method to create a balanced inlier and outlier class and then use the features that best separate its inlier and outlier class as the SE. While outlier-based SEs compute SEs by using an anomaly detector's outlier scoring measure guided by a search algorithm to identify the features that make the outlier point of interest outlying based on the outlier scoring measure used.

The remainder of the paper is outlined as follows. Section 2 reviews related work. Sections 3 and 4 present the outlier and sampled-based SEs respectively. Section 5 presents the data and the evaluation methods, while Section 6 contains our experiments. Lastly, Section 7 concludes the paper.

2. Related Work

There are a few anomaly detection methods that provide explicit outlier explanations that attempt to explain why the anomaly detector considered them as outliers. The idea behind these explanations is to make the analyst understand the inner workings of the anomaly detector itself by providing the analyst with outlier explanations that explain why a detected outlier point is outlying according to the anomaly detector used. ABOD (Kriegel et al., 2008), SOD (Kriegel et al., 2009), COP (Kriegel et al., 2012), LOGP (Dang et al., 2014) and LODI (Dang et al., 2013) are anomaly detection methods that provide explicit explanation that attempt to explain the inner workings of the anomaly detector.

In order to explain the inner workings of the EGMM, Siddiqui et al. (2019a) introduced SFEs. A length k SFE for a data point $\mathbf{x}$ is an ordered list of feature

indices $\mathcal{E} = (e_1, \ldots, e_k)$ where $k < d$ and d is the dimensionality of the data point $\mathbf{x}$. Features that appear earlier in the order are considered to be more important to the high outlier score of a point. The notation $\mathcal{E}_i$ denotes the set of the first i feature indices of $\mathcal{E}$. Also, for any set of feature indices $\mathcal{S}$ and a data point $\mathbf{x}$, $\mathbf{x}_\mathcal{S}$ denotes the projection of $\mathbf{x}_\mathcal{S}$ onto the feature subspace specified by $\mathcal{S}$. Therefore, given an SFE $\mathcal{E}$ for a data point $\mathbf{x}$, the point is incrementally presented to the analyst by first presenting feature $\mathbf{x}_{\mathcal{E}_1}$. If the analyst can make a judgement based on only that information, then the analyst's job is done. Otherwise, the analyst is shown the next feature together with the previous feature, which is $\mathbf{x}_{\mathcal{E}_2}$. The process of incrementally adding features to the set of presented features continues until the analyst can make a decision. Siddiqui et al. (2019a) developed the Marginal and Dropout SFE methods to explain to the analyst why an outlier point is outlying according to the anomaly detector and for the analyst also to understand the inner working of the EGMM.

Let f be a density-based anomaly detector that predicts the probability of a data point being normal in any given feature subspace. There are two Marginal SFEs, namely the sequential marginal SFE (SeqMArg) and the independent marginal SFE (IndMarg). SeqMarg adds one feature at a time to the SFE $\mathcal{E}$. At each step, it adds the feature that minimises the joint marginal density with the previously selected features in the SFE. More formally, SeqMarg computes the following explanation:

$$\textbf{SeqMarg}: \qquad e_i = \arg\min_{j \in \overline{\mathcal{E}}_{i-1}} f(\mathbf{x}_{\mathcal{E}_{i-1}+j}), \tag{1}$$

where $\overline{\mathcal{E}_i}$ is the complement $\mathcal{E}_i$ and represents the set the feature indices not selected in $\mathcal{E}_i$. SeqMarg takes into account feature interactions because it considers previously selected features in the SFE before it adds a new feature. IndMarg is a marginal method that does not take feature interactions into accounts and assumes features are independent. IndMarg only requires computation of individual marginals $f(\mathbf{x}_i)$ and adds features into the SFE $\mathcal{E}$ by sorting the features in increasing order of $f(\mathbf{x}_i)$.

There are two Dropout SFEs, namely the sequential dropout SFE (SeqDO) and the independent dropout SFE (IndDO). IndDO assigns a score of $f(\mathbf{x} - \mathbf{x}_i) - f(\mathbf{x})$ for each feature i, where we denote the removal of $\mathbf{x}_i$ from $\mathbf{x}$ by $\mathbf{x} - \mathbf{x}_i$. To obtain the IndDo SFE, the features are sorted in descending order of their scores. SeqDo is defined as:

$$\textbf{SeqDO}: \quad e_i = \arg\max_{j \in \overline{\mathcal{E}}_{1:i-1}} f(\mathbf{x}_{\overline{\mathcal{E}}_i - j}). \tag{2}$$

More recently Siddiqui et al. (2019b) applied SFEs to explain the inner workings of the Isolation Forest (Liu et al., 2008) anomaly detector. To use SFEs to explain the inner workings of an anomaly detector, the anomaly detector should compute outlier scores for an outlier point in any given subset of features. Therefore, Siddiqui et al. (2019b) were able to compute the outlierscores of an arbitrary outlier point $\mathbf{x}$ projected onto a feature subspace $\mathcal{S}$ (i.e $\mathbf{x}_\mathcal{S}$) by

considering the threshold tests involving features only from $\mathcal{S}$ across all trees in the Isolation Forest (Siddiqui et al., 2019b).

Most anomaly detection methods do not provide explanations and operate as black-boxes such that it is hard to identify the features that are responsible for a data point receiving a high outlier score. In general, obtaining outlier explanations that can explain the inner workings of each anomaly detection algorithm is a non-trivial task, and this is not the aim of our work.

The non-trivial task of explaining an anomaly detector's inner working has led to the development of outlier explanation methods that can explain any outlying data point regardless of the anomaly detector used to detect anomalies. These explanations are not subjective to the anomaly detector and do not explain their inner workings but rather attempt to explain what makes the detected outliers outlying in general. Therefore, the field of outlier explanations has led to the development of the following type of explanations: outlier explanations for groups of outliers (Kopp et al., 2020; Macha and Akoglu, 2018; Angiulli et al., 2013; Kuo and Davidson, 2016), pictorial explanations (Gupta et al., 2019), explaining outliers using decision rules (Pevnỳ and Kopp, 2014) and outlying aspects mining (Duan et al., 2015; Vinh et al., 2015, 2016).

Outlier explanations for groups of outliers compute explanations about what makes the identified outliers outlying as groups, instead of explaining the outlier points to the analyst one at a time. Explainer (Kopp et al., 2020), XPacs (Macha and Akoglu, 2018), EXPREX (Angiulli et al., 2013) and the work done in (Kuo and Davidson, 2016) all focus on identifying the subset of features that explain what makes outliers as groups outlying. Explaining outliers in a group or a set of groups has two key advantages (Macha and Akoglu, 2018). Firstly, it saves the analyst time, rather than the analyst having to go through the outliers one at a time. Secondly, it draws attention to outliers that form patterns, which are potentially more critical because they are repetitive. In our work, we focus on developing methods that can explain detected outliers one at a time rather than explaining them in groups. We are interested in explaining outliers one at a time to the analyst because firstly, outliers do not always fall into clusters or groups because they might be scattered. Secondly, the clusters could miss important features for certain outliers, which will be critical to the analyst's judgment. Lastly, the field of explaining outliers to analysts one at a time is still novel, and SEs are required to cover the gaps we have identified in our problem definition.

Pictorial explanations use visualisations to explain outliers. Gupta et al. (2019) introduced LookOut which explains outliers using visualisations. Given outliers from an arbitrary anomaly detector, LookOut finds two-dimensional scatter plots that visualise the feature subspace where the outliers are deviating from the rest of the dataset the most. Similarly, Liu et al. (2020), introduced LP-Explain whose key idea is to cluster outliers based on their behavior and identify the optimal two-dimensional subspaces to visualise each cluster of outliers (Liu et al., 2020). In our work, we are interested in identifying feature subspaces that are beyond two-dimensional. Also, LP-Explain visualises the outliers as groups while we are more interested in explaining outliers one at a time.

Explainer (Kopp et al., 2020) explains outliers using decision rules learned by training a random forest, where one class consists of the outlier point of interest and the second class consists of k randomly selected data points from the rest of the dataset. From this random forest, Explainer extracts individual classification rules which are represented in disjunctive normal form. The rules are extracted along the path from the root node to the leaf of the decision trees. Since each tree in the random forest is trained to separate a single point from k randomly selected data points, its training set is extremely imbalanced, and its height is very small, therefore resulting in very short explanations. In our work, we are interested in computing as many features in the explanation as required by the analyst. Explainer can also be used to explain groups of outliers (Kopp et al., 2020), however in our work we are only interested in explaining outliers one at a time

Outlying aspects mining (Duan et al., 2015; Vinh et al., 2015, 2016; Boukela et al., 2020) identify which subset of features make a data point different from the rest of the dataset. It is used to obtain the best feature subspace that distinguishes a data point from the rest of the dataset, whether it is an outlier or a normal data point. Outlying aspects mining tackles the problem from three different angles known as the score-and-search, feature selection and hybrid-based methods. Score-and-search based methods first define an outlier scoring measure to measure how outlying a data point is compared to the rest of the dataset in an arbitrary feature subspace. Score-and-search based methods obtain the outlying aspects by computing the outlier scoring measure of all possible feature subspaces and selecting the subspace that returns the best score as the outlying aspects. To avoid the dimensionality bias of comparing feature subspaces of different sizes, OAMiner (Duan et al., 2015) computes an expensive density score that requires the computation of the outlier scoring measure of every data point in every feature subspace in order to find a data points outlying aspects. Feature selection based methods (Micenkova et al., 2013) transform the outlying aspect mining problem into a classic two class feature selection problem and select the feature that best separate the classes as the outlying aspects. Hybrid based methods (Vinh et al., 2015) use a combination of a score-and-search a feature rankig method. The first stage ranks the features according to their potential to make a data point outlying, while the second stage is a search-and-score based approach on the top-ranked features from the first stage. Outlying aspect mining and our work are very similar with a few important differences to note. In our work, we are not interested in finding the most outlying feature subspace. In order to find the most outlying feature subspace the computationally expensive density score needs to be used in order to avoid dimensionality bias. To compute outlier-based SEs, we are not comparing outlier scoring measures of feature subspaces of different sizes because we are only interested in identifying the best feature subspace for each explanation of size i in the SE. SEs are therefore computationally cheaper than outlying aspect methods because we do not have to use the density score because we only compare feature subspaces of the same size against each other.

3. Outlier-based Sequential Explanations

Outlier-based SEs compute SEs by using an anomaly detector's outlier scoring measure guided by a search algorithm to identify the features that make the outlier point of interest outlying based on the outlier scoring measure used. An outlier scoring measure $\phi : \mathbb{R}^{|\mathcal{S}|} \longrightarrow \mathbb{R}$ is a measure of outlyingness that assigns a real number to the data point $\mathbf{x}$ in feature subspace S.

3.1. SE notation

The following notation is used in the paper, unless specified otherwise:

- Let $\mathbf{x} \in \mathbf{X}$ represents a d-dimensional data point from the anomaly benchmark dataset $\mathbf{X}$ that contains n data points. We represent the i^{th} data point in $\mathbf{X}$ by $\mathbf{x}^{(i)}$.
- Let $\mathcal{S}$ be any set of the feature indices. Then we denote $\mathbf{x}_\mathcal{S}$ as the projection of the data point $\mathbf{x}$ onto the feature subspace specified by $\mathcal{S}$, while $\mathbf{x}$ represents the data point in the full feature space.
- $SE(\mathbf{x})^k = (e_1, e_2, \ldots, e_k)$, represents an ordered SE list of size k for the data point $\mathbf{x}$, where each e_i represents a subset of feature indices of size i and e_1 contains the first best feature index to be presented to the analyst, e_2 contains the two best feature indices to present to the analyst and so on.
- $SE(\mathbf{x}) = SE(\mathbf{x})^d$, is the full SE for $\mathbf{x}$.
- $SE(\mathbf{x})_i = e_i$, is the i^{th} explanation of size i in $SE(\mathbf{x})$.
- $SE(\mathbf{x})_0 = \emptyset$.

3.2. Anomaly detectors

There are a variety of anomaly detection algorithms available in literature (Chandola et al., 2009). We selected a few unsupervised anomaly detection algorithms in our work that cover different approaches to scoring outliers. The selection of algorithms is not exhaustive, but it is a good representation of the field by trying to cover several different solution approaches that cover different assumptions and properties when scoring outliers. We use the outlier scoring measure of each anomaly detection algorithms to compute the outlier-based SEs. We selected the Kernel density estimation (KDE) (Chandola et al., 2009), One-Class Support Vector Machine (OCSVM) (Chandola et al., 2009), Local outlier factor (LOF) (Breunig et al., 2000), k-means and Isolation Forests (iForests) (Liu et al., 2012) anomaly detectors to compute our outlier scoring measure. Appendix A provides a detailed summary of these anomaly detectors and the computation of their outlier scoring measures.

3.3. Sequential search outlier-based SEs

3.3.1. Sequential forward selection

Forward selection:. Forward selection (FS), is a greedy sequential search method that adds one feature at a time to the previously selected subset of features in the SE. At each step, it adds the feature that maximises the outlier score of the previously selected features in the SE. More formally, FS computes the following outlier-based SE:

$$SE(\mathbf{x})_i = SE(\mathbf{x})_{i-1} \cup \underset{j \in \overline{SE}(\mathbf{x})_{i-1}}{\arg\max} \; \phi(\mathbf{x}_{SE(\mathbf{x})_{i-1} \cup j}), \tag{3}$$

where $\overline{SE}(\mathbf{x})_i$ are the subset of features not selected in $SE(\mathbf{x})_i$.

Independent forward selection:. Independent forward selection (IND-FS), requires computation of the outlier score for each individual feature $\phi(\mathbf{x}_j)$ for $j = 1, \ldots, d$. To compute the SE, IND-FS sorts the features in increasing order of their scores $\phi(\mathbf{x}_j)$. More formally, IND-FS computes the following outlier-based SE:

$$SE(\mathbf{x})_i = SE(\mathbf{x})_{i-1} \cup \underset{j \in \overline{SE}(\mathbf{x})_{i-1}}{\arg\max} \; \phi(\mathbf{x}_j). \tag{4}$$

3.3.2. Sequential backward selection

Backward selection:. Backward selection (BS), starts with a full set of features, and at each iteration removes the feature that returns the lowest decrease in the outlier score and adds it to the SE. More formally, BS computes the following explanation:

$$SE(\mathbf{x})_i = SE(\mathbf{x})_{i-1} \cup \underset{j \in \overline{SE}(\mathbf{x})_{i-1}}{\arg\min} \; \phi(\mathbf{x}_{\overline{SE}(\mathbf{x})_{i-1} \setminus j}). \tag{5}$$

Independent backward selection:. For each feature $\mathbf{x}_i$ of the data point $\mathbf{x}$, independent backward selection (IND-BS) assigns it with a score of $\phi(\mathbf{x}_{-i}) - \phi(\mathbf{x})$, where $\mathbf{x}_{-i}$ denotes the removal of feature $\mathbf{x}_i$ from $\mathbf{x}$. The removal of the features with the lower scores make $\mathbf{x}$ appear most outlying. IND-BS sorts the features in increasing order of their scores to compute the explanation. More formally, IND-BS computes the following outlier-based SE:

$$SE(\mathbf{x})_i = SE(\mathbf{x})_{i-1} \cup \underset{j \in \overline{SE}(\mathbf{x})_{i-1}}{\arg\min} \; (\phi(\mathbf{x}_{-j}) - \phi(\mathbf{x})). \tag{6}$$

3.4. Meta-heuristic search outlier-based SE

3.4.1. Particle swarm optimisation

The sequential search algorithms compute SEs that suffer from the nesting effect. To combat the nesting effect, we introduce a new novel Particle Swarm Optimisation (PSO) meta-heuristic search that restricts the search of the feature subspace to only subsets of a specific size instead of searching over a range of

Algorithm 1: PSO for a fixed size subset of size i

Input: $\mathbf{x}, \mathbf{X}, i$
Output: *gbest*
for $j = 1$ **to** P **do**
  Initialise the particle p_j by sampling without replacement i features from the full set of features $\{1, \ldots, d\}$
  Initialise the particle's best position to its initial position: $pbest_j = p_j$
  if $f(\mathbf{x}_{p_j}) < f(\mathbf{x}_{gbest})$ **then**
    Update the swarms best known position: $gbest = p_j$
for *g=1* **to** G **do**
  for $j = 1$ **to** P **do**
    Update(p_j):
      1. Pool all the unique features from $p_j, pbest_j$ and $gbest$;
      2. Randomly select i features from the unique pool of features;
      3. Each of the i randomly selected features has a fixed probability of mutation m ;
      4. Elements selected for mutation are assigned new feature values at random from all the features not included in the pool of unique features from step 1.

    if $f(\mathbf{x}_{p_j}) > f(\mathbf{x}_{pbest_j})$ **then**
      Update the particle j's best known position: $pbest_j = p_j$
    if $f(\mathbf{x}_{p_j}) > f(\mathbf{x}_{gbest})$ **then**
      Update the swarms best known position: $gbest = p_j$

subset sizes. Therefore, when we require $SE(\mathbf{x})_i$, the PSO meta-heuristic will restrict the search to only feature subspaces of size i. The PSO (Kennedy and Eberhart, 1995) meta-heuristic search is known for its fast convergence and its tendency to fall into sub-optimal solutions. In order to take the PSO out of sub-optimal solutions and improve exploration, we include a mutation factor which also reduces the convergence rate.

Let $f : \mathbb{R}^{|\mathcal{S}|} \longrightarrow \mathbb{R}$ represent the fitness function to be optimised by the PSO search algorithm, P represent the total number of particles in the swarm, $pbest_j$ be the best-known position of particle j, *gbest* be the best-known position of the entire swarm, and G represent the total number of generations. The PSO meta-heuristic algorithm is presented in Algorithm 1. For outlier-based SEs, to compute $SE(\mathbf{x})_i$ of a data point $\mathbf{x}$ based on the PSO meta-heuristic we replace the fitness function f with the outlier scoring measure ϕ. After the PSO has undergone G generations, we set $SE(\mathbf{x})_i = gbest$.

4. Sample-based Sequential Explanations

Sample-based SEs turn the problem of computing SEs into a classical supervised feature selection problem by using a sampling method to create a balanced inlier and outlier class and then using the features that best separate its inlier and outlier class as the SE. We now define the sample-based SEs adapted from the work by Micenkova et al. (2013).

4.1. Sampling method

We first provide the definitions of the k-distance and reference set. The k-distance of a data point $\mathbf{x} \in \mathbf{X}$, denoted by k-distance($\mathbf{x}$), is the distance $d(\mathbf{x}, \mathbf{x}')$ between $\mathbf{x}$ and its k-th nearest neighbour $\mathbf{x}'$ in the data set $\mathbf{X}$. The reference set of a data point $\mathbf{x}$, denoted by $R_k(\mathbf{x})$, is the set of points $\mathbf{p} \in \mathbf{X}$ whose distance from $\mathbf{x}$ is less than or equal to k-distance($\mathbf{x}$). More formally the reference set is defined as follows:

$$R_k(\mathbf{x}) = \{\mathbf{p} \in \{\mathbf{X} \setminus \{\mathbf{x}\}\} | d(\mathbf{x}, \mathbf{p}) \leq k\text{-distance}(\mathbf{x})\}. \tag{7}$$

For a data point $\mathbf{x}$, the following steps are followed to compute the inlier and outlier class of $\mathbf{x}$ (Micenkova et al., 2013).

1. We define the sampled inlier class of $\mathbf{x}$ from $\mathbf{X} \setminus \{\mathbf{x}\}$ as a union of its reference set $R_k(\mathbf{x})$ and a set of randomly drawn data points from $\mathbf{X} \setminus \{\{\mathbf{x}\} \cup R_k(\mathbf{x})\}$ as:
$$\mathcal{I}(\mathbf{x}) = R_k(\mathbf{x}) \cup \{\mathbf{q}_j\}_1^r, \tag{8}$$
where each $\mathbf{q}_j$ is randomly chosen from $\mathbf{X} \setminus \{\{\mathbf{x}\} \cup R_k(\mathbf{x})\}$ and $r = |R_k(\mathbf{x})|$.
2. We define the outlier class of $\mathbf{x}$ as:
$$\mathcal{O}(\mathbf{x}) = \{\mathbf{x}\} \cup \{\mathbf{z}_j\}_1^s, \tag{9}$$
where each $\mathbf{z}_j$ is randomly sampled from $\mathcal{N}(\mathbf{x}, \lambda^2 I)$, $s = |\mathcal{I}(\mathbf{x})|$, I is the $d \times d$ identity matrix and $\lambda = \alpha \cdot \frac{1}{\sqrt{d}} \cdot k$-distance($\mathbf{x}$), and α is a user defined parameter which controls the width of the distribution.
3. We obtain $SE(\mathbf{x})_i$ by selecting the best i features that separate the inlier and outlier classes using a feature selection algorithm.

4.2. Feature selection methods

Wrapper-based methods obtain a feature subset by using a classifier as a black-box and the classifier's performance as the objective function guided by a search algorithm to identify the best features that represent the data (Chandrashekar and Sahin, 2014). In our work we use Support Vector Machines (Hearst et al., 1998) (SVMs) as our classifier. The feature selection methods we use to compute the wrapper-based sample-based SEs are the Forward selection SVM (FS-SVM) (Chandrashekar and Sahin, 2014), Backward selection SVM (BS-SVM) (Chandrashekar and Sahin, 2014) and PSO-SVM. For FS-SVM to obtain $SE(\mathbf{x})_i$, the algorithm starts with an empty set. At each iteration, it classifies

Table 1: Summary of Goldstein and Uchida (2016) anomaly detection benchmark datasets.

Data set	Size	#Features	#Anomalies
ALOI	50000	27	1508
Annthyroid	6916	21	250
Satellite	5100	36	75
KDD99	620098	38	1052

Table 2: Summary of Emmott et al. (2015) motherset anomaly detection benchmark datasets.

Motherset	#Features	Anomaly class size	Normal class size	#Anomaly Benchmarks	#Data Points (Range)
Pageb	10	560	4913	129	195-4466
Shuttle	9	9074	48926	120	6000-6000
Abalone	7	2081	2096	295	885-1906
Concrete	8	515	515	180	249-468
Wine	11	4113	4113	180	3165-3739
Yeast	8	977	977	120	400-888
Magic Gamma	10	12332	12332	300	2000-6000

the training data over the 3 cross-validation folds using an SVM and adds the feature that maximises the average classification accuracy over the 3 folds with the current subset of already chosen features into $SE(\mathbf{x})_i$ until i features are selected. Conversely, BS-SVM starts with a full set of features and at each iteration, it removes the feature that returns the lowest decrease in the average classification accuracy. PSO-SVM uses Algorithm 1 to search for SEs that are not nested. The fitness function f of the PSO-SVM is simply the average classification accuracy of the SVM that separates the inlier and outlier classes over 3 cross-validation folds.

Embedded-based methods perform feature selection during the training process of the classifier (Chandrashekar and Sahin, 2014). In our work we use Recursive Feature Elimination (RFE) (Guyon et al., 2002) to take feature interactions into account. We use SVM Recursive Feature Elimination (SVM-RFE) (Guyon et al., 2002) and Random Forests Recursive Feature Elimination (RF-RFE) (Liaw and Wiener, 2002) as our embedded-based sample-based SEs.

Filter-based methods make use of a feature ranking function to measure the relevance of each feature. Features with the highest relevance measure are the chosen features to represent the data best. We use the information theoretic methods Maximum relevance minimum redundancy (mRmR) (Brown et al., 2012) and Mutual information feature selection (MIFS) (Brown et al., 2012) as our filter-based sample-based SEs.

5. Data and Evaluations

5.1. Data

Due to the lack of publicly available real-world labelled anomaly detection benchmark datasets, we use the constructed anomaly benchmark datasets from Goldstein and Uchida (2016) and Emmott et al. (2015) for our experiments. Table 1 and 2 present a summary of the Goldstein and Uchida (2016) and Emmott et al. (2015) anomaly benchmarks respectively.

Goldstein and Uchida (2016) constructed their anomaly benchmarks from supervised learning labelled datasets by selecting one class as the anomalous class and randomly sampling a small portion of the data points to create it, whereas, the normal class is created by randomly sampling a larger portion of the data points from another class. While, Emmott et al. (2015) named the original datasets that they were sampling from to create their anomaly benchmarks the "motherset" datasets. For each motherset, they created an anomalous and normal class. Then to create their corpus of anomaly detection benchmark datasets, they sampled data points from the motherset such that desired measured properties in the anomaly benchmarks are obtained. In the end, they created thousands of anomaly benchmark datasets each with its own set of desired properties from each motherset. We concentrated on anomaly benchmark datasets that have an anomaly frequency between $1\% - 2\%$. Table 2 gives a summary of the motherset datasets we used in our experiments. For example, the Abalone motherset was used to generate 295 anomaly benchmarks. The motherset contains 2081 and 2096 data points in its anomaly and normal classes, respectively. While the number of data points in the anomaly benchmark datasets created from the motherset range between $885 - 1906$.

5.2. Evaluations using the area under the analyst certainty curve

Similar to Siddiqui et al. (2019a), we let the function $A(\mathbf{x}, \mathcal{S}) = P(\mathbf{x} = anomaly|\mathbf{x}_{\mathcal{S}})$ represent the simulated analyst. $A(\mathbf{x}, \mathcal{S})$ returns the probability of the data point $\mathbf{x}$ being anomalous considering only the features specified in the feature space $\mathcal{S}$. Given $SE(\mathbf{x})^k$, we define the analyst certainty curve for the first k feature subsets of the SE of $\mathbf{x}$ as the curve that plots the coordinates $(i, A(\mathbf{x}, SE(\mathbf{x})_i)$ for $i = 1, \ldots, k$. Intuitively, the higher the analyst certainty curve the better the quality of the SE because the selected feature subsets in the SE are returning high probabilities of the analyst detecting the anomalies given the features presented to them. To measure the performance of the SE, we measure the area under the curve of the analyst certainty curve (AUCC) and divide it by k so that the AUCC is between 0 and 1.

We now specify how we obtain the analyst function $A(\mathbf{x}, \mathcal{S})$. For the Emmott et al. (2015) datasets we construct the training set over the motherset, while for the Goldstein and Uchida (2016) datasets we constructed the training set over the actual anomaly benchmark datasets. We used a brute force method to obtain $A(\mathbf{x}, \mathcal{S})$ by training a discriminative model only in the feature space specified by $\mathcal{S}$. It leads to multiple trained models because we train a model for every feature space $\mathcal{S}$ encountered in the SEs. For each model we train, we exclude that data point whose SEs we are evaluating in the training set so that we avoid any bias in our probability estimations. To obtain "reliable" probability estimates even in the case of imbalanced datasets, we used the sampling and bagging methods specified by Wallace and Dahabreh (2013) and a RF model to predict the probability estimates for $A(\mathbf{x}, \mathcal{S})$.

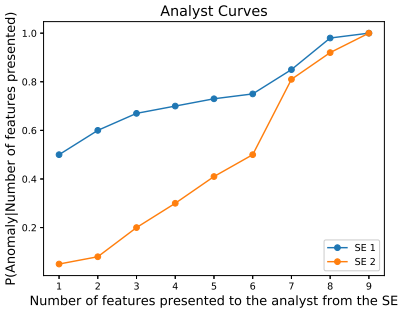

Figure 3: With a threshold of 0.8, SE 1 and 2 have an MFP of 7 and 8 if the threshold is 0.9. The AUCC of SE 1 is 0.67 and 0.46 for SE 2. The AUCC captures that SE 1 outperforms SE 2

6. Experimental Results and Discussions

We now present our experimental results. For the Goldstein and Uchida (2016) anomaly benchmark datasets, we averaged the AUCCs of the anomalies to obtain the benchmarks average AUCCs. Whereas for the Emmott et al. (2015) datasets we first averaged the AUCCs of the anomalies for each benchmark to obtain its average AUCC followed by averaging the average AUCCs of each benchmark dataset which was constructed from the same motherset to obtain an average of the AUCCs of the motherset's anomaly detection benchmark datasets. Each analyst certainty curve contains error bars which represent a 95% confidence interval of the results over five runs per SE. For the Emmott et al. (2015) datasets we generated the SEs until the analyst is presented with all the features. It enables us to analyse the analyst's conditional probability from the first feature subset up until the full set of features are presented. While for the Goldstein and Uchida (2016) datasets, we generated the SEs up to the first seven feature subsets because the dimensionality of the anomaly benchmark datasets was large. Appendix B details how we chose the parameters for the outlier and sample-based SEs in our experiments.

6.1. Analysis of results using the analyst certainty curves and AUCCs

We now show why the AUCC is a better evaluation measure than the MFP. Figure 3 presents a good arbitrary example of two competing SEs with the same MFP. If the simulated analyst's threshold of detection is 0.8, then SE 1 and SE 2 both have an MFP of 7. If the threshold is 0.9, then the MFP for both SEs would be 8. However, from observing the analyst certainty curves it is clear that SE 1 identifies feature subsets that return higher conditional probabilities than SE 2. In addition, the AUCC of SE 1 is 0.67, whereas, the AUCC for SE 2 is 0.46. The AUCC captures the difference in performances of the SEs much better than the MFP.

Figure 4 presents the analyst certainty curves of the top performing outlier-based SEs of the Wine, Pageb and Magic Gamma datasets. The average AUCC

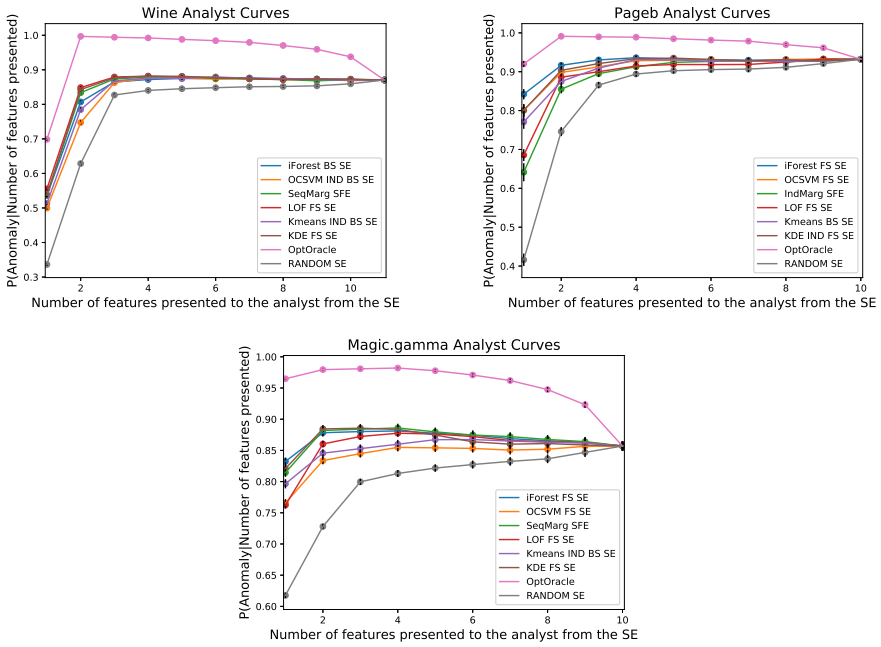

Figure 4: Analyst certainty curves of the top performing outlier-based SEs of the Wine, Pageb and Magic Gamma datasets

of the Random SE across the Pageb motherset anomaly benchmarks is 0.7753, while the best performing outlier-based SE, iForest FS has an average AUCC of 0.8329. From the analyst certainty curves we observe that the first feature that iForest FS SE selects returns the analyst with a conditional probability of 0.85 of being certain that the data points presented are anomalous. While the Random SE returns a probability of 0.42. Considering the next two features in the iForest FS and Random SEs, the conditional probability of the analyst's detection increases to 0.91 for the iForest FS SE, while it increases to 0.73 for the Random SE. The AUCCs provides a measure to compare which SEs perform better than the other, while a visual inspection of the analyst certainty curves provides insights on the trajectory analysis of the SEs performance for each feature subset.

From the analyst certainty curves of the Emmott et al. (2015) datasets, we observe three different types of curves. Type 1 curves are monotonically increasing with respect to the number of features presented to the simulated analyst until the SE presents the analyst with the full set of features. All of the Random SEs are type 1 analyst curves. Type 2 curves are monotonically increasing with respect to the number of features presented to the simulated analyst up until a peak is reached, after which it monotonically decreases until the SE presents the analyst with the full set of features. The Magic Gamma motherset iForest, KDE, LOF FS SEs are Type 2 SEs. Type 3 curves are

Table 3: Outlier-based SE AUCCs of the Emmott et al. (2015) datasets. The bold black AUCCs represent the best performing search methods for each outlier scoring measure, while the bold red AUCCs represent the overall best performing methods per dataset.

Dataset	Anomaly Detector	IND FS	FS	IND BS	BS	AD Oracle	Opt Oracle	Random
Abalone	iForest	0.7257	0.7316	**0.7348**	0.7337	0.7502	0.8096	0.6966
	OCSVM	0.7138	0.7255	**0.7316**	0.7300	0.7447		
	LOF	0.7353	**0.7407**	0.7335	0.7296	0.7520		
	Kmeans	0.7223	0.7210	**0.7245**	0.7244	0.7450		
	KDE	0.7220	**0.7291**	0.7029	0.7025	0.7438		
Concrete	iForest	0.8051	**0.8156**	0.8044	0.8042	0.8290	0.8605	0.7231
	OCSVM	0.7480	0.7562	**0.7675**	0.7645	0.7800		
	LOF	0.7844	0.7908	0.7839	**0.7932**	0.7991		
	Kmeans	0.7642	**0.7938**	0.7800	0.7905	0.8053		
	KDE	0.7984	**0.8083**	0.7820	0.7875	0.8229		
Magic Gamma	iForest	0.7753	**0.7837**	0.7746	0.7733	-	0.8641	0.7250
	OCSVM	0.7513	**0.7616**	0.7595	0.7598			
	LOF	0.7536	**0.7768**	0.7709	0.7645			
	Kmeans	0.7649	0.7685	**0.7714**	0.7688			
	KDE	0.7785	**0.7816**	0.7564	0.7630			
Pageb	iForest	0.8328	**0.8329**	0.8287	0.8300	-	0.8780	0.7753
	OCSVM	0.8163	**0.8266**	0.8162	0.8159			
	LOF	0.8133	**0.8139**	0.8033	0.8025			
	Kmeans	0.8199	0.8194	0.8200	**0.8214**			
	KDE	**0.8285**	0.8242	0.8031	0.8045			
Shuttle	iForest	0.8574	0.8565	0.8614	**0.8615**	-	0.8834	0.7680
	OCSVM	0.8471	0.8064	0.8459	**0.8571**			
	LOF	0.8477	**0.8673**	0.7983	0.7960			
	Kmeans	0.8677	**0.8694**	0.85106	0.8521			
	KDE	0.866	**0.8670**	0.615	0.6208			
Wine	iForest	0.7749	**0.7795**	0.7742	0.7752	-	0.8769	0.7305
	OCSVM	0.7525	0.757	**0.7664**	0.7635			
	LOF	0.7754	**0.783**	0.7638	0.762			
	Kmeans	0.7671	0.7717	**0.7729**	0.7716			
	KDE	0.7726	**0.7820**	0.7492	0.7555			
Yeast	iForest	0.6517	0.6168	0.6680	**0.6730**	0.6895	0.7992	0.5913
	OCSVM	0.6331	0.6483	**0.6515**	0.6293	0.6642		
	LOF	0.6410	**0.6526**	0.6309	0.6284	0.6701		
	Kmeans	**0.6687**	0.6655	0.6524	0.6578	0.6912		
	KDE	0.6787	**0.6851**	0.6706	0.6739	0.7035		

monotonically increasing with respect to the number of features presented up until they reached a plateau that is equal to the conditional probability of the presentation of the full set of features. All the Wine top performing outlier-based SEs are Type 3 SEs.

6.2. Random and Oracle SEs

In addition to the SEs presented, we compute the Random and Oracle SEs. A Random SE is a SE that randomly presents features to the analyst. Random SEs provide a lower bound on the attainable performance of our SEs. An OptOracle SE is an optimal SE that is allowed access to the simulated analyst (Siddiqui et al., 2019a), where $SE(\mathbf{x})_i$ is the set of i features that maximise the simulated analyst's conditional probability $A(\mathbf{x}, SE(\mathbf{x})_i)$. Lastly, an anomaly detector's Oracle (AD Oracle) SE is an outlier-based SE, where $SE(\mathbf{x})_i$ is the set of i features that maximise the anomaly detector's outlier scoring measure. An AD Oracle provides the best outlier-based SE that an AD's outlier scoring measure can compute.

6.3. Outlier-based SE results

Due to the small dimensionality of the Emmott et al. (2015) motherset anomaly benchmarks, we only used the sequential search outlier-based SEs and excluded the PSO meta-heuristic search algorithm because the dimensionality

Table 4: Outlier-based SE AUCCs of the Goldstein and Uchida (2016) datasets

Dataset	Anomaly Detector	IND FS	FS	IND BS	BS	AD PSO	Random
Aloi	iForest	0.5269	**0.5705**	0.5617	0.5552	0.5810	0.4075
	OCSVM	0.5278	0.5381	**0.5574**	0.5528	0.5685	
	LOF	**0.5344**	0.5241	0.5273	0.5073	0.5486	
	Kmeans	0.5294	0.5552	**0.5589**	0.5524	0.5771	
	KDE	0.5597	**0.5602**	0.4626	0.4900	**0.5858**	
Annthyroid	iForest	**0.8429**	0.8385	0.8375	0.8423	**0.8491**	0.5659
	OCSVM	**0.7653**	0.7604	0.6890	0.6703	0.7906	
	LOF	**0.8392**	0.8296	0.7990	0.7935	0.8431	
	Kmeans	0.7795	0.7856	0.7599	**0.7911**	0.8088	
	KDE	0.7447	**0.7705**	0.6954	0.7116	0.7936	
Kdd99	iForest	0.8476	**0.8500**	0.8472	0.8482	**0.8506**	0.6913
	OCSVM	0.8465	**0.8493**	0.7550	0.7708	0.8500	
	LOF	0.8458	0.8398	**0.8475**	0.8461	0.8485	
	Kmeans	0.8385	**0.8411**	0.8215	0.8206	0.8428	
	KDE	**0.8459**	0.8449	0.7267	0.7520	0.8505	
Satellite	iForest	0.8226	**0.8250**	0.7827	0.7919	**0.8307**	0.6435
	OCSVM	0.7953	**0.8089**	0.7783	0.7770	0.8209	
	LOF	**0.7838**	0.7758	0.7562	0.7634	0.8049	
	Kmeans	0.8124	**0.8163**	0.7954	0.7735	0.8292	
	KDE	0.8031	**0.8118**	0.7481	0.7716	0.8275	

of the anomaly benchmark datasets are small relative to the number of comparisons we make using the PSO meta-heuristic search. For example, the Magic Gamma motherset anomaly benchmark datasets have 11 features which are the most features from the Emmott et al. (2015) benchmarks. A dimensionality of 11 features has a total of 2048 combinations. In our work, we restrict the total number of comparisons in the PSO meta-heuristic search to a maximum of 3000 per feature subset of size i in the SE $SE(\mathbf{x})_i$. Therefore, in place of the PSO meta-heuristic search algorithm, we use the AD Oracle SE to obtain the optimal subset of features to add in the SE based on the AD's outlier scoring measure. Due to the large amount of the Emmott et al. (2015) anomaly benchmark datasets from each motherset, we only computed the AD Oracle SEs for the Abalone, Concrete and Yeast motherset anomaly benchmarks.

6.3.1. Comparison to Oracles

We observe from Table 3 that all of the AD Oracle SEs from the Abalone, Concrete and Yeast motherset anomaly benchmarks outperform their corresponding AD IND-FS, FS, IND-BS and BS SEs. These results show that a more extensive search in the search space yields improved results compared to using greedy sequential search methods. Also, the results show that there is a need for search methods that are not greedy and that a more extensive search improves the outlier-based SEs. The OptOracle SE outperforms all of the AD Oracle SEs for the Abalone, Concrete and Yeast motherset anomaly benchmark datasets. This shows that an exhaustive search across each AD's outlier scoring measure does not obtain the same SE as the OptOracle SE. The reason OptOracle SEs outperform the AD Oracle SEs could be due to the difference in the assumptions and properties of the anomaly detectors and OptOracle's notions of how to score outliers.

6.3.2. Comparison to Random, OptOracle and PSO

OptOracle significantly outperforms all of the SEs, whereas, all of the SEs significantly outperform the Random SE with an exception to the Shuttle moth-

erset KDE sequential backward selection SEs. Using the Goldstein and Uchida (2016) datasets, we compare the sequential search methods to the PSO meta-heuristic search method. From the AUCCs, we observe that given an AD's outlier scoring measure, the PSO meta-heuristic algorithm outperforms its corresponding sequential search SEs. The AUCCs show that the PSO meta-heuristic computes better SEs than the sequential search methods for the same AD scoring measure. The PSO meta-heuristic SE using an AD's outlier scoring measure can be outperformed by a sequential search method SE using a different AD outlier scoring measure. For example, the LOF PSO SE for the Aloi dataset has an average AUCC of 0.5489 and iForest FS has an average AUCC of 0.5705. Similar examples are observed throughout the results in Table 4. The PSO meta-heuristic search only outperforms the sequential search methods that make use of the same AD outlier scoring measure and does not necessarily perform better than all the sequential search methods that use different AD outlier scoring measures. The reason behind this could be because of the different assumptions and properties used by the different AD outlier scoring measures. A sequential search outlier-based SE using an AD outlier scoring measure that measures the correct properties and uses the correct assumptions for a specific dataset will outperform a PSO meta-heuristic SE using an AD outlier scoring measure that measures incorrect properties and uses the incorrect assumptions to score outliers on the dataset.

6.3.3. Sequential forward versus sequential backward selection SEs

To compare the sequential forward and backward selection search methods, we used a ranking measure to rank the methods in descending order of their AUCCs. Using Tables 3 and 4 we ranked each row according to the sequential forward and backward selection search methods from 1 to 4 in descending order, then for each dataset we computed the mean rank for each search method. Lastly, we computed the mean of the mean ranks of each sequential backward and forward selection search method across all the datasets. Overall, FS has the lowest mean rank followed by IND-FS, BS and IND-BS. The sequential forward selection SEs outperform the sequential backward selection SEs.

6.3.4. Comparison of AD outlier scoring measures

For the Goldstein and Uchida (2016) datasets, we ranked the results of each outlier-based SE for each dataset from 1 to 36 using the AUCCs in Table 4. For each AD outlier scoring measure, we calculate its mean rank by calculating the mean of each row. Then for each AD outlier scoring measure, we computed the mean of the mean ranks across the Goldstein and Uchida (2016) datasets. Overall, iForest has the lowest mean ranking followed by k-means, LOF, KDE and OCSVM. Similarly, we obtained the same results for the Emmott et al. (2015) motherset datasets in Table 3.

6.4. Sample-based SE results

We now analyse the results of the sample-based SEs. The AUCC results of the sample-based SEs can be found in Tables 5 and 6 for the Emmott et al.

Table 5: Sample-based SE AUCCs of the Emmott et al. (2015) datasets

Dataset	FS SVM	BS SVM	mRmR	MIFS	SVM RFE	RF RFE	Opt Oracle	Random
Abalone	0.7334	0.727	0.7274	0.7279	**0.7372**	0.7289	0.8096	0.6966
Concrete	0.8089	0.8061	0.8152	0.8015	0.8132	**0.8156**	0.8605	0.7231
Magic Gamma	**0.7873**	0.7783	0.7752	0.7788	0.7813	0.7857	0.8641	0.7250
Pageb	0.8305	0.8271	**0.8322**	0.8319	0.8286	0.8300	0.8780	0.7753
Shuttle	0.8474	0.8397	0.8410	0.8545	**0.8667**	0.8616	0.8834	0.7680
Wine	0.7761	0.7659	0.7659	0.7748	0.7758	**0.7770**	0.8769	0.7305
Yeast	0.6629	0.6698	0.6266	0.6627	**0.6761**	0.6177	0.7992	0.5913

Table 6: Sample-based SE AUCCs of the Goldstein and Uchida (2016) datasets

Dataset	FS SVM	BS SVM	mRmR	MIFS	SVM RFE	RF RFE	PSO SVM	Random
Annthyroid	0.8158	0.8130	0.8210	0.8217	**0.8348**	0.8143	0.8273	0.5659
Aloi	0.5656	0.5424	0.5065	0.5313	**0.5905**	0.5359	0.5742	0.4075
Kdd99	0.8069	0.7857	**0.8502**	0.8476	0.8490	0.8484	0.8145	0.6913
Satellite	0.8343	0.8343	0.8130	0.8069	0.8237	0.8218	**0.8368**	0.6435

(2015) and Goldstein and Uchida (2016) datasets respectively.

6.4.1. Comparison to Random, OptOracle and PSO-SVM

From the AUCCs and analyst certainty curves, we observe that all of the sample-based SEs significantly outperform the Random SE. However, from the Emmott et al. (2015) dataset AUCCs there is a large gap between the sample-based SEs and the OptOracle SEs. The gap shows that there is a need for improved feature selection methods that are not greedy. From the AUCCs of the Goldstein and Uchida (2016) datasets we observe that the PSO-SVM SE outperforms the greedy feature selection based FS-SVM and BS-SVM SEs. However, it does not always perform better than the other sample-based SEs. For example, in the Annthyroid, Aloi and KDD99 datasets, the SVM-RFE SE outperforms the PSO-SVM SE. A possible explanation for this would be that the PSO-SVM is a wrapper based feature selection method that relies on the accuracy of the SVM to select features. We observed in most cases that the SVM classifier was not able to obtain a 100% accuracy when splitting the inlier and outlier classes for the features that returned the highest classification accuracy, therefore resulting in sub-optimal solutions. The PSO-SVM SE improved on the greedy wrapper-based FS-SVM and BS-SVM SEs only, and it fails to outperform the other SE methods consistently.

6.4.2. Comparison of feature selection methods

To obtain the best performing feature selection methods, we used a ranking measure to compare the different methods. For each of the results in Tables 5 and 6, we ranked each feature selection method for each dataset from 1 to 6 excluding the PSO-SVM in the Goldstein and Uchida (2016) datasets. We then averaged the rankings for each feature selection method across the datasets to obtain the overall mean rank. Overall, SVM-RFE had the lowest rank, followed by FS-SVM, RF-RFE, mRmR, MIFS and BS-SVM. Considering the Goldstein and Uchida (2016) dataset alone and the including PSO-SVM SE, we found that SVM-RFE had the lowest rank, followed by PSO-SVM, FS-SVM, mRmR, RF-RFE, MIFS and BS-SVM.

Table 7: SFE AUCCs of the Emmott et al. (2015) datasets

Dataset	IndMarg	SeqMarg	IndDo	SeqDo	Opt Oracle	Random
Abalone	0.7212	**0.7298**	0.7133	0.7144	0.8096	0.6966
Concrete	0.8006	**0.8113**	0.7948	0.8001	0.8605	0.7231
Magic Gamma	0.7765	**0.7851**	0.7771	0.7723	0.8641	0.7250
Pageb	**0.8287**	0.8269	0.8103	0.8110	0.8780	0.7753
Shuttle	0.8661	**0.8670**	0.7269	0.7227	0.8834	0.7680
Wine	0.7628	**0.7794**	0.7618	0.7628	0.8769	0.7305
Yeast	0.6747	**0.6778**	0.6741	0.6736	0.7992	0.5913

Table 8: SFE AUCCs of the Goldstein and Uchida (2016) datasets

Dataset	IndMarg	SeqMarg	IndDo	SeqDo	Random
Aloi	0.5550	**0.5742**	0.5208	0.5343	0.4075
Annthyroid	0.7808	**0.7896**	0.7374	0.7436	0.5659
Kdd99	**0.8501**	0.8477	0.7139	0.7315	0.6913
Satellite	0.8200	**0.8234**	0.7901	0.8001	0.6435

6.5. Sample-based versus outlier-based SEs

For the Emmott et al. (2015) datasets, we used the same ranking method as in Section 6.4.2 and ranked the AUCCs of the outlier and sampled-based SEs from Tables 3 and 5 respectively and obtained the following rankings: SVM-RFE had the lowest rank, followed by FS-SVM, RF-RFE, the iForest SEs, mRmR, MIFS, BS-SVM, the EGMM SEs, the LOF SEs, the k-means SEs and the OCSVM SEs. For the Goldstein and Uchida (2016) datasets, we ranked the AUCCs of the outlier and sample-based SEs from Tables 4 and 6 respectively and obtained the following rankings: SVM-RFE had the lowest rank, followed by PSO-SVM, the iForest SEs, FS-SVM, RF-RFE, mRmR, MIFS, BS-SVM, the EGMM SEs, the LOF SEs, the k-means SEs and the OCSVM SEs. It is clear from these results that the sample-based SEs outperform the outlier-based SEs in terms of AUCCs.

6.6. SEs versus SFEs

Table 7 and Table 8 contain the SFE results for the Emmott et al. (2015) and Goldstein and Uchida (2016) datasets respectively. We observe from both tables that the dominant SFE method is the SeqMarg SFE, while from Table 8 in we observe that the PSO applied to the SFEs improves their results.

We now compare the best performing SFEs against the best performing outlier and sample-based SE methods for each data dataset. For the Emmott et al. (2015) datasets, the best performing outlier and sample-based SEs outperform the best performing SFE with the exception to the Wine and Yeast datasets where SeqMarg outperforms the best performing sample-based SE only and for the Magic gama dataset where SeqMarg outperforms the best performing outlier-based SE. For the Goldstein and Uchida (2016) datasets, all of the best performing outlier and sample-based SE outperform the best performing SFEs. The results show that our SEs are able to generate explanations that will guide the analyst to the features that will enable them to make judgements about whether detected outlier points are anomalous are not. In addition our SEs outperform SEs in terms of the quality of the explanations without being limited to the SFEs limitations.

Table 9: Annthyroid outlier-based SE running times

	IND_FS	FS	IND-BS	BS	PSO
iForest	9.45	19.65	10.73	23.37	681.27
OCSVM	1.52	7.02	3.17	11.22	7.77
LOF	0.82	13.80	31.60	60.01	17.32
k-means	1.47	19.30	7.19	21.70	23.97
KDE	0.01	0.27	0.05	0.57	4.64

Table 10: Annthyroid sample-based SE running times

FS-SVM	BS-SVM	mRmR	MIFS	SVM-RFE	RF-RFE	PSO-SVM
1.15	2.76	0.12	0.12	0.58	0.59	27.00

Table 11: Annthyroid SFE running times

IndMarg	SeqMarg	IndDo	SeqDo
0.86	1.39	10.73	2.28

6.7. Running times

We now compare the running times of the SEs and the SFEs. Tables 9, 10 and 11 contain the average running times over the first 100 anomalies of the Annthyroid dataset. We ran our experiments in Sections 6.3 and 6.4 using multiple servers that gave us access to between 64 and 128 CPUs and gigabytes RAM. The servers enabled us to use multiprocessing and caching of results in RAM in order to optimise the running times for the experiments. In order to fairly compare the running times without multiprocessing, we ran the SEs and SFEs using a single CPU for the running times measured in Tables 9, 10 and 11.

We observe from the running times that the iForest SE take the longest to compute. In particular, iForest PSO SE takes the longest time to compute. Also, we observe from the outlier-based SEs that the SEs that use the Ind-FS an Ind-BS sequential search are the fastest to compute, followed by the FS sequential search. From the LOF SE running times, we observe that LOF PSO runs faster than LOF BS and LOF IND-BS even though it searches through a much smaller subspace than the latter two SEs. The reason for this could be because the LOF outlier scoring measure takes long to compute when the feature subspace is large, and the IND-BS and BS sequential search methods search in the full feature space, unlike the PSO which only does its search up to the first 7 features. The sample-based SFEs had the fastest running times except for PSO-SVM. In particular, SVM-RFE, which is the best performing SFE overall has a fast running time. Some SEs may produce good results, but they may also take longer to compute the required features into the SEs. The analyst could then choose to trade off a good performing SE for alternative SEs that are computationally cheaper to compute even though they do return the best results.

6.8. Limitations and drawbacks

We have identified the following limitations and drawbacks of our research that leave room for further improvements:

1) In computing the SEs, we employed meta-heuristic search methods to prune the search space for each outlier data point. The use of meta-heuristic search methods is computationally expensive because we had to search for the SEs for each data point, which required many computational resources.
2) For the outlier-based SEs, each outlier detection method required us to choose the parameters we would use to compute the SEs. For each anomaly detection benchmark dataset, we computed the data points' outlier scores using the anomaly detection method. We then performed hyperparameter optimisation using grid search to find the parameters that return the highest area under the receiver operating characteristic curves (AUC ROC) in the whole feature space. The optimal parameters in the full feature space are used to compute the outlier-based SEs. The disadvantage of using the optimal parameters from the full feature space is that it might not be optimal for the lower-dimensional feature subspaces.
3) For the sample-based SEs, we used the grid search method to obtain the feature selection methods' optimal parameters. We then selected the parameters that return the highest average classification accuracy using 3-fold cross-validation in the full feature space. Similar to the drawbacks of the outlier-based SEs, using the optimal parameter from the full feature space might not be optimal for the lower-dimensional feature subspaces.

7. Conclusion

This paper introduced a new outlier explanation called a sequential explanation (SE) that addresses the shortcomings of the sequential feature explanations (SFEs). The SEs can be computed by either using the sample or outlier-based approaches. Outlier-based SEs compute SEs using an anomaly detector's outlier scoring measure guided by a search algorithm to identify the features that make the outlier point of interest outlying based on the outlier scoring measure used. Whereas sample-based SEs turn the problem of computing SEs into a supervised feature selection problem using a sampling method to create a balanced inlier and outlier class and use feature selection to select the features that best separate the two classes as the SEs. We also introduced a new evaluation method called the area under the analyst certainty curve (AUCC).

Our results showed that the AUCC is a better measure than the minimum feature prefix (MFP) and provides us with more insights into the performances of the SEs. We, therefore, used the AUCCs to evaluate and compare the SFEs, outlier and sample-based SEs.

For the outlier-based SEs, we found that the particle swarm optimisation (PSO) meta-heuristic search only outperforms the greedy search methods that use the same outlier scoring measure. However, it does not necessarily perform better than all of the greedy explanation methods that use different outlier scoring measures for their SEs. From the sequential search methods, forward selection provided the best SEs followed by independent forward selection, backward selection and IND-independent backward selection. The iForest anomaly detector outlier measure computes the best performing outlier-based SEs.

For the sample-based SEs, we found that the PSO meta-heuristic search only outperforms the greedy search methods that use the same outlier scoring measure. However, it does not necessarily perform better than all of the greedy explanation methods using different outlier scoring measures for their SEs. We also found that SVM-RFE outperformed PSO-SVM. A possible explanation for this would be that the PSO-SVM is a wrapper based feature selection method that relies on the SVM's accuracy to select features. We observed in most cases that the SVM classifier was not able to obtain a 100% accuracy when splitting the inlier and outlier classes for the features that returned the highest classification accuracy, therefore resulting in sub-optimal solutions.

Comparing the SFEs to the SEs, we found that in most cases, the best performing outlier and sample-based SEs outperformed the best performing SFE. It shows that our SEs provide better explanations than SFEs. Overall, the sample-based SE, SVM-RFE SE, returned the best performing SE, which also takes a reasonable time to compute.

Appendix A. Anomaly detector outlier scoring measures

We now summarise each of the anomaly detection algorithms and how their outlier scoring measure is computed.

Appendix A.1. Kernel density estimation

A Kernel density estimation (KDE) (Silverman, 1986) is a non-parametric statistical method used to estimate the probability density function (pdf) of a dataset. KDEs do this by using a kernel function to estimate the local densities of each data point in the dataset and then obtain the overall pdf by aggregating the local data densities of the data points. The outlier score of a data point is the probability of the data point not belonging to the fitted pdf. More formally, the KDE outlier scoring measure of a data point $\mathbf{x}$ in the feature subspace $\mathcal{S}$ is defined as (Silverman, 1986) :

$$\phi(\mathbf{x}_{\mathcal{S}}) = 1 - \frac{1}{n(2\pi)^{|S|/2}\prod_{s\in\mathcal{S}} h_s} \sum_{\mathbf{z}\in\mathbf{X}_{\mathcal{S}}\setminus\{\mathbf{x}_{\mathcal{S}}\}} e^{-\sum_{s\in\mathcal{S}} \frac{(\mathbf{x}_s-\mathbf{z})^2}{2h_s^2}}, \tag{A.1}$$

where h_s is the bandwidth of feature $s \in \mathcal{S}$.

Appendix A.2. One-Class Support Vector Machine

The One-Class Support Vector Machine (OCSVM) (Schölkopf et al., 2001) is a model based anomaly detection method. Similar to an SVM for classification, an OCSVM transforms a dataset in the feature space $\mathbb{R}^d$ to a higher-dimensional feature space $\mathbb{R}^p$, where $d < p$. In the transformed space, the OCSVM maximises the distance from the hyperplane to the origin, which results in learnt regions that contain the training dataset in the original space. For anomaly detection, we assume that a vast majority of the data is normal and that anomalies

contribute less to the computation of the region where the training data lies. The outlier score of each data point is the distance of the data point to the learnt region. Data points that fall within the region are considered to be inliers and have negative scores, whereas, data points outside the region have positive scores. More formally, the OSVM problem is reduced to the following optimisation problem:

$$\min_{\mathbf{w},\xi_i,\rho} \frac{1}{2}\|(\mathbf{w}\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho; \tag{A.2}$$

$$\text{subject to:} \tag{A.3}$$

$$\mathbf{w}\cdot\theta(\mathbf{x}_{\mathcal{S}}^{(i)}) \geq \rho - \xi_i \text{ for all } i = 1,\ldots,n; \tag{A.4}$$

$$\xi_i \geq 0 \text{ for all } i = 1,\ldots,n, \tag{A.5}$$

where $\mathbf{w}$ is the normed vector of the hyperplane, θ is a kernel map that transforms the training examples into the space $\mathbb{R}^p$, ξ_i are nonzero slack variables, ν is an upper bound on the fraction of outliers and a lower bound of the fraction of support vectors, and ρ is the intercept of the hyperplane.

Using Lagrange techniques and using a kernel function K for the dot-product calculations, the OCSVM outlier scoring function of a data point $\mathbf{x}$ in the feature subspace $\mathcal{S}$ is:

$$\phi(\mathbf{x}_{\mathcal{S}}) = (\mathbf{w}\cdot\theta(\mathbf{x}_{\mathcal{S}}^{(i)}) - \rho) = \sum_{i=1}^{n}\alpha_i K(\mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{S}}^{(i)}) - \rho \tag{A.6}$$

, where $\mathbf{x}_{\mathcal{S}}^{(i)} \in X_{\mathcal{S}}$, α_i are the Lagrange multipliers and $K(\mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{S}}^{(i)}) = \theta(\mathbf{x}_{\mathcal{S}})^T\theta(\mathbf{x}_{\mathcal{S}}^{(i)}), \forall i = 1,2,\ldots,n$.

Appendix A.3. Local outlier factor

Local outlier factor (LOF) (Breunig et al., 2000) is a nearest neighbourhood based anomaly detect method. For a data point $\mathbf{x}$, the LOF algorithm compares its local density to the local densities of its k-nearest neighbours (k-nn). A data point will get a high outlier score if its density is significantly lower than the density of its k-nn (Breunig et al., 2000). More formally the LOF outlier scoring function of a data point $\mathbf{x}$ in the feature subspace S is formulated as follows:

Let k-distance($\mathbf{x}_{\mathcal{S}}$) be the distance of the data point $\mathbf{x}$ to its k^{th} nearest neighbour in the feature subspace specified by $\mathcal{S}$. Then we define the reachability distance between $\mathbf{x}_{\mathcal{S}}, \mathbf{t}_{\mathcal{S}} \in X_{\mathcal{S}}$ as:

$$\text{reachability-distance}_k(\mathbf{x}_{\mathcal{S}}, \mathbf{t}_{\mathcal{S}}) = max\{\text{k-distance}(\mathbf{t}_{\mathcal{S}}), d(\mathbf{x}_{\mathcal{S}}, \mathbf{t}_{\mathcal{S}}))\} \tag{A.7}$$

and the local reachability density of $\mathbf{x}_{\mathcal{S}}$ is defined by

$$\text{lrd}(\mathbf{x}_{\mathcal{S}}) = 1/(\frac{\sum_{\mathbf{t}\in k\text{-nn}(\mathbf{x}_{\mathcal{S}})}\text{reachability-distance}_k(\mathbf{x}_{\mathcal{S}}, \mathbf{t})}{\mid k\text{-nn}(\mathbf{x}_{\mathcal{S}}) \mid}). \tag{A.8}$$

Then the LOF outlier score of $\mathbf{x}_{\mathcal{S}}$ is:

$$\phi(\mathbf{x}_{\mathcal{S}}) = \frac{\sum_{\mathbf{t} \in k\text{-nn}(\mathbf{x}_{\mathcal{S}})} \frac{\text{lrd}(\mathbf{t})}{\text{lrd}(\mathbf{x}_{\mathcal{S}})}}{\mid k\text{-nn}(\mathbf{x}_{\mathcal{S}}) \mid} \tag{A.9}$$

Appendix A.4. k-means anomaly detector

The k-means anomaly detector is a clustering based anomaly detector that works on the assumptions that normal data points lie close to their closest cluster centroid, while outliers are far away from it (Chandola et al., 2009). The anomaly detector first clusters the data into k clusters using Lloyds algorithm (Kanungo et al., 2002), then it assigns each data point the distance from its closest cluster as its outlier score (Chandola et al., 2009). Data points with the largest distances from their closest cluster centroids are considered to be outliers because they do not belong to a specific cluster. More formally, the k-means outlier scoring measure of a data point $\mathbf{x}$ in the feature subspace $\mathcal{S}$ is formulated as follows:

$$\phi(\mathbf{x}_{\mathcal{S}}) = \|\mathbf{x}_{\mathcal{S}} - c_{\mathbf{x}_{\mathcal{S}}}\|_2, \tag{A.10}$$

where $c_{\mathbf{x}_{\mathcal{S}}}$ represents the data point $\mathbf{x}_{\mathcal{S}}$'s closest cluster centroid and $\|\cdot\|$ is the Euclidean distance or L_2 vector norm.

Appendix A.5. Isolation Forests

Isolation Forests (iForests) (Liu et al., 2012) isolates data point by first randomly selecting a feature and a random split value between the maximum and minimum values of the selected feature in order to partition the data point into the region it belongs in. The process of randomly partitioning the data point continues until it is entirely isolated. Since a tree structure can represent recursive partitioning, the number of splittings required to isolate a data point is equivalent to the path length from the root node to the terminating node. Averaging the path length over a forest of random trees forms a measure of normality. Random partitioning produces shorter paths for outliers. Therefore, when a forest of random trees collectively produces shorter path lengths for particular data points, they are highly likely to be outliers. More formally, the iForest outlier scoring measure of a data point $\mathbf{x}$ in the feature subspace $\mathcal{S}$ is formulated as follows:

$$\phi(\mathbf{x}_{\mathcal{S}}) = -2^{\frac{E(h(\mathbf{x}_{\mathcal{S}}))}{c(\psi)}}, \tag{A.11}$$

where $h(\mathbf{x}_{\mathcal{S}})$ is the path length of $\mathbf{x}_{\mathcal{S}}$, $c(\psi)$ is the average path length of an unsuccessful search in a binary search tree, and ψ is the subsample size.

Appendix B. Hyperparameter Optimisation

Appendix B.1. Outlier-based SEs

We now discuss how we chose the parameters of the outlier-based SEs. For each anomaly detection benchmark dataset, we computed the data points' outlier scores using the respective anomaly detection method. We then performed hyperparameter optimisation using grid search to find the parameters that return the highest area under the receiver operating characteristic curves (AUC ROC) in the full feature space. The optimal parameters in the full feature space are used to compute the outlier-based SEs.

It is non-trivial to identify the range of values to perform grid search on. Therefore we chose to perform a grid search between a range of values that cover the parameter space when the parameter has a low value and a high value. For the k-means anomaly detector we use grid search to find the optimal k for $k = 2, \ldots, 8$. For each k, we calculated the AUC ROC using the ground truth of the benchmark. We chose the k value for each benchmark that gave us the highest AUC ROC and used it to compute the SEs for the k-means outlier scoring measure. We grid searched on $k = 15, \ldots, 20$ for each anomaly detection benchmark dataset for the LOF anomaly detector. For the bandwidth of the KDE we applied grid search on $[0.001, 0.01, 0.1, 1, 10]$ and also on Silvermans (Silverman, 1986) and Scotts (Scott, 2010) rule of thumbs. For OCSVM we used a linear kernel and a RBF kernel and applied grid search on $C = [0.01, 0.1, 1, 10]$ on both kernels and $\gamma = [0.001, 0.01, 0.1, 1]$ on the RBF kernel. For the EGMM anomaly detector, we used the same parameters as in Siddiqui et al. (2019a), while for iForests, we used the default parameters recommended in Liu et al. (2012) of 100 trees and a sub-sampling size of 256.

For the PSO, we limited the total number of computations in each search space to 3000 by setting $G = 75$ and $P = 40$. We set the mutation factor for each feature to $m = 0.1$. Also, to decrease the number of computations, we cached and saved all of our results in a hash table stored in RAM so that if the PSO algorithm generates the same particles we do not need to compute the value of their fitness function again, instead we do a quick look up the in the hash table.

Appendix B.2. Sample-based SEs

We now discuss how we chose the parameters of the sample-based SEs. To create the inlier and outlier class for a data point $\mathbf{x}$, we set its k for the reference set $R_k(\mathbf{x})$ to $k = 0.1 * n$ and set $\alpha = 0.35$ as suggested in Micenkova et al. (2013) . For each data points inlier and outlier class, we optimise the parameters of the feature selection methods that will compute its sample-based SEs by using the grid search method and selecting the parameters that return the highest average classification accuracy using 3-fold cross-validation in the full feature space.

For the feature selection methods that use SVM, we used a linear kernel and a RBF kernel and applied grid search on $C = [0.01, 0.1, 1, 10]$ on both kernels and $\gamma = [0.001, 0.01, 0.1, 1]$ on the RBF kernel. We selected the parameters from

the grid search that return the highest average classification accuracy using 3-fold cross-validation in the full feature space to compute the SEs. For MIFS, we used $\beta = 1$ as suggested in Battiti (1994) and used 100 trees for RF-RFE, while mRmR did not require any parameters.

References

Angiulli, F., Fassetti, F., Palopoli, L., 2013. Discovering characterizations of the behavior of anomalous subpopulations. IEEE Transactions on Knowledge and Data Engineering 25, 1280–1292. doi:10.1109/tkde.2012.58.

Battiti, R., 1994. Using mutual information for selecting features in supervised neural net learning. IEEE Transactions on Neural Networks 5, 537–550. doi:10.1109/72.298224.

Boukela, L., Zhang, G., Yacoub, M., Bouzefrane, S., Ahmadi, S.B.B., Jelodar, H., 2020. A modified LOF-based approach for outlier characterization in IoT. Annals of Telecommunications 76, 145–153. doi:10.1007/s12243-020-00780-5.

Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J., 2000. LOF: identifying density-based local outliers. ACM SIGMOD Record 29, 93–104. doi:10.1145/335191.335388.

Brown, G., Pocock, A., Zhao, M., Luján, M., 2012. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. Journal of Machine Learning Research 13, 27–66.

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection. ACM Computing Surveys 41, 1–58. doi:10.1145/1541880.1541882.

Chandrashekar, G., Sahin, F., 2014. A survey on feature selection methods. Computers & Electrical Engineering 40, 16–28. doi:10.1016/j.compeleceng.2013.11.024.

Dang, X.H., Assent, I., Ng, R.T., Zimek, A., Schubert, E., 2014. Discriminative features for identifying and interpreting outliers, in: IEEE International Conference on Data Engineering, pp. 88–99. doi:10.1109/icde.2014.6816642.

Dang, X.H., Micenková, B., Assent, I., Ng, R.T., 2013. Local outlier detection with interpretation, in: Advanced Information Systems Engineering. Springer Berlin Heidelberg, pp. 304–320. doi:10.1007/978-3-642-40994-3_20.

Duan, L., Tang, G., Pei, J., Bailey, J., Campbell, A., Tang, C., 2015. Mining outlying aspects on numeric data. Data Mining and Knowledge Discovery 29, 1116–1151. doi:10.1007/s10618-014-0398-2.

Emmott, A., Das, S., Dietterich, T., Fern, A., Wong, W.K., 2015. A meta-analysis of the anomaly detection problem `arXiv:1503.01158`.

Glodek, M., Schels, M., Schwenker, F., 2012. Ensemble gaussian mixture models for probability density estimation. Computational Statistics 28, 127–138. doi:10.1007/s00180-012-0374-5.

Goldstein, M., Uchida, S., 2016. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PLOS ONE 11, e0152173. doi:10.1371/journal.pone.0152173.

Gupta, N., Eswaran, D., Shah, N., Akoglu, L., Faloutsos, C., 2019. Beyond outlier detection: LookOut for pictorial explanation, in: Machine Learning and Knowledge Discovery in Databases. Springer International Publishing, pp. 122–138. doi:10.1007/978-3-030-10925-7_8.

Guyon, I., Weston, J., Barnhill, S., Vapnik, V., 2002. Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning 46, 389–422. doi:10.1023/a:1012487302797.

Hearst, M., Dumais, S., Osuna, E., Platt, J., Scholkopf, B., 1998. Support vector machines. IEEE Intelligent Systems and their Applications 13, 18–28. doi:10.1109/5254.708428.

Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., Wu, A., 2002. An efficient k-means clustering algorithm: analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 881–892. doi:10.1109/tpami.2002.1017616.

Kennedy, J., Eberhart, R., 1995. Particle swarm optimization, in: IEEEE International Conference on Neural Networks, pp. 1942–1948. doi:10.1109/icnn.1995.488968.

Kopp, M., Pevný, T., Holeňa, M., 2020. Anomaly explanation with random forests. Expert Systems with Applications 149, 113187. doi:10.1016/j.eswa.2020.113187.

Krawczyk, B., 2016. Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence 5, 221–232. doi:10.1007/s13748-016-0094-0.

Kriegel, H.P., hubert, M.S., Zimek, A., 2008. Angle-based outlier detection in high-dimensional data, in: ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08, pp. 444–452. doi:10.1145/1401890.1401946.

Kriegel, H.P., Kroger, P., Schubert, E., Zimek, A., 2012. Outlier detection in arbitrarily oriented subspaces, in: IEEE International Conference on Data Mining, pp. 379–388. doi:10.1109/icdm.2012.21.

Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A., 2009. Outlier detection in axis-parallel subspaces of high dimensional data, in: Advances in Knowledge Discovery and Data Mining. Springer Berlin Heidelberg, pp. 831–838. doi:10.1007/978-3-642-01307-2_86.

Kuo, C.T., Davidson, I., 2016. A framework for outlier description using constraint programming, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 1237–1243.

Liaw, A., Wiener, M., 2002. Classification and Regression by randomForest. R News 2, 18–22.

Liu, F.T., Ting, K.M., Zhou, Z.H., 2008. Isolation forest, in: IEEE International Conference on Data Mining, pp. 413–422. doi:10.1109/icdm.2008.17.

Liu, F.T., Ting, K.M., Zhou, Z.H., 2012. Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data 6, 1–39. doi:10.1145/2133360.2133363.

Liu, H., Ma, F., Wang, Y., He, S., Chen, J., Gao, J., 2020. LP-explain: Local pictorial explanation for outliers, in: IEEE International Conference on Data Mining, pp. 372–381. doi:10.1109/icdm50108.2020.00046.

Macha, M., Akoglu, L., 2018. Explaining anomalies in groups with characterizing subspace rules. Data Mining and Knowledge Discovery 32, 1444–1480. doi:10.1007/s10618-018-0585-7.

Micenkova, B., Ng, R.T., Dang, X.H., Assent, I., 2013. Explaining outliers by subspace separability, in: IEEE International Conference on Data Mining, pp. 518–527. doi:10.1109/icdm.2013.132.

Nakariyakul, S., Casasent, D.P., 2009. An improvement on floating search algorithms for feature subset selection. Pattern Recognition 42, 1932–1940. doi:10.1016/j.patcog.2008.11.018.

Pevnỳ, T., Kopp, M., 2014. Explaining anomalies with sapling random forests, in: Information Technologies-Applications and Theory Workshops, Posters, and Tutorials, pp. 1–15.

Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C., 2001. Estimating the support of a high-dimensional distribution. Neural Computation 13, 1443–1471. doi:10.1162/089976601750264965.

Scott, D.W., 2010. Scott's rule. Wiley Interdisciplinary Reviews: Computational Statistics 2, 497–502. doi:10.1002/wics.103.

Siddiqui, M.A., Fern, A., Dietterich, T.G., Wong, W.K., 2019a. Sequential feature explanations for anomaly detection. ACM Transactions on Knowledge Discovery from Data 13, 1–22. doi:10.1145/3230666.

Siddiqui, M.A., Stokes, J.W., Seifert, C., Argyle, E., McCann, R., Neil, J., Carroll, J., 2019b. Detecting cyber attacks using anomaly detection with explanations and expert feedback, in: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 2872–2876. doi:10.1109/icassp.2019.8683212.

Silverman, B.W., 1986. Density estimation for statistics and data analysis. Chapman & Hall/CRC monographs on statistics and applied probability, Chapman and Hall, London.

Vinh, N.X., Chan, J., Bailey, J., Leckie, C., Ramamohanarao, K., Pei, J., 2015. Scalable outlying-inlying aspects discovery via feature ranking, in: Advances in Knowledge Discovery and Data Mining. Springer International Publishing, pp. 422–434. doi:10.1007/978-3-319-18032-8_33.

Vinh, N.X., Chan, J., Romano, S., Bailey, J., Leckie, C., Ramamohanarao, K., Pei, J., 2016. Discovering outlying aspects in large datasets. Data Mining and Knowledge Discovery 30, 1520–1555. doi:10.1007/s10618-016-0453-2.

Wallace, B.C., Dahabreh, I.J., 2013. Improving class probability estimates for imbalanced data. Knowledge and Information Systems 41, 33–52. doi:10.1007/s10115-013-0670-6.