Machine Learning Outlier Detection in Safetica's Data Loss Prevention System

Jan Pluskal, ipluskal@fit.vutbr.cz

***Abstract*—Data loss prevention systems are becoming necessities in corporate computer system deployments. Nowadays, when everything is connected, and BYOD (Bring your own device) methodology is tolerated, even encouraged in many companies, network security administrators are obliged to keep with newest technologies to prevent threats to business resources. Threats might be parts of carefully planned corporate espionage, or simple malware encrypting all resources available to it. No matter which threat, data have to be kept safe and each interaction with critical business resources need to be monitored, authorized and logged for future analysis.**

In this paper, we discuss state of the art methods used for outlier detection, unsupervised learning and statistical analysis.

***Keywords*—Machine learning, Outlier detection, Data loss prevention**

I. Introduction

This paper aims to summarize State of the Art techniques usable in a classification of user behavior patterns with a goal of anomaly detection. This research is conducted on behest of Safetica Services s.r.o. to complete the first phase of research contract "Using network analysis techniques to prevent data loss." To achieve this, analysis Safetica's DLP platform is required to identify the key point(s) to be targeted by this research.

This paper is structured in following manner. The section II describes Safetica's Data Loss Prevention solution that monitors end-user stations and collects logs of conducted user activities. The section examines this platform and tries to discover key features and data connection point that might be used as data source for further analysis. The attention is paid to future feature candidates that will be further examined and evaluated.

The section III briefly introduces available machine learning techniques, divides them by their affection to a learning phase whether they require labeled data, i.e., supervised learning methods, or not, i.e., unsupervised learning methods. In further subsection, methods devised to increase accuracy are examined with consideration to be implemented in the proof-of-concept solution. Lastly, a comparison of performance measurement methods is analyzed to select one unified metric to be used in further performance comparisons.

The section IV examines types of outliers, how they are usually classified by literature review regarding their relation. Outliers might be non-contextual, contextual, sub-contextual, i.e., *Types I, II, and III*. Additionally, an exhaustive enumeration of outlier detection methods summarizing their affinity to detect outliers in the respective category, learning phase approach, and their strong sides and weaknesses.

II. Safetica's DLP platform

Safetica's Data-loss-prevention platform is a multiple client-server architectural based solution. There is a *Safetica Endpoint Client* application installed on End-user devices that monitors activity on each device. Safetica Endpoint Client application collects predefined information, notifies administrator, or restricts particular operation or action, based on rules defined on a server application. The *server application* functions as a controller and collector for *Safetica Endpoint Clients*. All information is stored in SQL based database.

This distributed architecture is very scalable, but still, every operation that could be computed on end-nodes that do not require contextual data related to multiple end-node devices would be much more economical to compute directly on an end-user device. This assumption should be kept in mind during the machine learning algorithm selection.

For purpose of this research, only *file based operations* will be inspected and further considered in a context of anomaly detection. For the preliminary study, *presentation_data* database was used for inner structure analysis. A *file_operation_log* table was used as a primary data source. Additional context was provided from other tables like *file_operation_log_dgid, file_operation_log_mail, mail_monitor_log, file_operation_log_print, print_log, file_operation_log_path_usb, file_operation_log_web.*

Features selected to be examined and tested for correlations are enumerated in Table I.

III. Machine Learning

In this section, we will focus on basic machine learning methods and categorization. We provide a brief overview of machine learning methods used for classification and clustering.

A. Classification of Machine Learning Methods

Machine learning methods that are appearing in the most latterly presented algorithms could be classified by their attitude to the data processing approach, and by their need of pre-classified data, to be learned from them or not. We divide those methods into three classes: *supervised*, *unsupervised* and *semi-supervised* learning [1].

B. Supervised learning

The supervised learning is machine learning task which infers a function from a labeled training data set [2]. In most cases, the training data consists of input pairs of an N-tuple and a label. The N-tuple is a vector of N variables describing the

Id	**Feature**	**Db**	**Type**
		files_view	
1	FileTimeStart	[files_view].[date_time_start_dtm]	S
2	FileTimeStop	[files_view].[date_time_end_dtm]	S
3	FileDuration		
4	FileApplication	[files_view].[id_application]	C
5	FileAction	[files_view].[action]	C
6	FileOperation Type	[files_view].[operation_type]	C
7	FileSourceType	[files_view].[source_type]	C
8	FileDestinationType	[files_view].[destination_type]	C
9	FileFileSize	[files_view].[file_size]	S
10	FileExtension	[extension]	C
11	FileDataCategory	[file_operation_log_dgid].[id_data_category]	C
		file_operation_log_path_usb	
12	UsbDeviceSerial	[device_usb].[serial]	C
		file_operation_log_mail	
13	EmailSender	[mail_monitor_log].[id_email_sender]	C
14	EmailReceiver	XXX	C
15	EmailSize	[mail_size]	S
		file_operation_log_print	
16	PrintPageCount	[print_log].[page_count]	S
17	PrintCopies	[print_log].[copies]	S
18	PrintDuplex	[print_log].[duplex]	C
		file_operation_log_web	
19	WebUploadDomain	[web].[domain]	C

TABLE I. CANDIDATE FEATURES SELECTION BASED ON *presentation_data* DATABASE. *Feature name* REPRESENTS A DESCRIPTIVE NAME OF A FEATURE FOLLOWING A SCHEMA – (FILE|USB|EMAIL|PRINT|WEB)<FEATUREDESCRIPTION>. *Db* LOCALIZE FEATURE IN A STRUCTURE OF *presentation_data* DATABASE. *Type* SYMBOLIZE FOR *C* A CATEGORICAL VALUE, FOR *S* A SEQUENTIAL ONE.

training data sample. The label is a target class label inferred by the supervisor. The learning algorithm tries to analyze input training set and produce an inferred function that will be used for classification of unknown data. Intended behavior would be using a relatively small training data set for learning the algorithm that will classify any number of unknown data sets different from the training set.

1) Bayesian Belief Networks: The simple algorithm of Bayesian belief networks is based on a Bayes' theorem and naive Bayesian classification that is also very fast and accurate as recent studies have shown [3]. Bayesian belief networks are probabilistic graphical models, which unlike Naive Bayesian Classifiers allow the representation of dependencies among subsets of attributes, that is, given a label of a tuple, the values of attributes are assumed to be conditionally dependent on one another. In the case, where is a priori known, that attributes are independent, the *Naive Bayesian classifier* is the most accurate in comparison with all other classifiers [1].

A *Belief Network* is defined by two components—a directed acyclic graph and a set of conditional probability tables. Nodes are representing a random variables—discrete or continuous. The probability table specifies the conditional distribution for each random variable, for example $P(X|Parents(X))$ [1].

Let $X = (x_1, ..., x_n)$ to be a set conversations represented by attribute tuple $Y_1, ..., Y_2$. Probability for each node in the network is computed as $P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i|Parents(Y_i))$ where $P(x_1, ..., x_n)$ is the probability of a particular combination of values of X, and the values for $P(x_i|Parents(Y_i))$ correspond to entries in the conditional probability table [1]. Class is identified as a node or set of nodes with the highest probability.

2) Neural Networks: Neural networks are used for a classification by backpropagation. The backpropagation is a neural network learning algorithm. The neural network used for classification in its simple form is built on three layers: input, hidden and an output layer. The input layer consists of nodes representing selected attributes. Output layer is formed by nodes representing classes. The hidden layer interconnects each node of the input layer with each node in the output layer. Each connection is parametrized by its weight.

Any node of hidden or output layer is calculating a weighted sum of its inputs as $O = \sum_{i=1} w_i f(I_i) + \theta$ where $f(a) = \frac{1}{1+\epsilon^{-a}}$. The $f(a)$ is called a squashing function, because it maps a larger input domain onto the smaller range of 0 to 1 [1]. The neural network is learning by updating weights and bias (θ) by back propagation of error that reflects the network's prediction.

By normalization of output layer values of trained neural network, we can obtain a probability vector that identifies occurrence as belonging to the particular class with a respective probability.

3) Support Vector Machine: The support vector machine algorithm could be used for classification of linear as well as non-linear data. When classifying a nonlinear data, the algorithm uses a nonlinear mapping function to transform the original data into a higher dimension that is separable by a linear hyperplane.

SVMs, even the fastest ones, could be extremely slow to learn, but after learning phase, they are highly accurate due to their ability to model complex nonlinear decision boundaries. They are also much less prone to overfitting than other methods [1].

4) Decision tree algorithms: The decision three is a flowchart like a tree structure, where each internal node detonates a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label [1]. The decision tree based algorithms might be slower then Bayes' theorem in the learning phase, but they are very fast classifiers. Very familiar representatives are *C4.5*[1] and *Random Forests* methods.

The C4.5 has adopted a greedy search approach of construction in a top-down manner with a splitting criterion using the concept of information entropy [4]. The algorithm works as follows: checks for base cases; for each attribute a finds the normalized information gain ration from splitting on a; creates a decision node that splits on the attribute with the highest normalized information gain; recurs on the sub-lists obtained by split, and adds those nodes as children of splitting node.

[1] Has been improved with the version of C5.0 in speed, memory usage and has been commercially deployed.

C. Unsupervised learning

Unsupervised learning, i.e., clustering, tends to gain more attention recently [5] because of its independence on the learning phase. All supervised algorithms need learning to build its model to classify, but user behavior is different for each job position, personal preferences and team. Therefore, the learned model of one user in one company might not be usable in another one.

Unsupervised learning methods are very useful in state of the art anomaly detection methods for their ability to detect a new previously unseen occurrence for future analysis. There are several categories of unsupervised learning according to their approach to cluster data.

Partitioning methods—this group of methods pre-create a k categories, therefore, before a method begins to cluster, there must be some classes a priori known. Methods varies in optimization heuristics representing a cluster (e.g., *K-means*, *K-menoids*).

Hierarchical methods—create a hierarchical decomposition of input data set. Two approaches can be used—agglomerative (i.e., bottom-up) and divisive(i.e., top-down) approach. In comparison to other methods, when the hierarchical method split or merge values, this step can never be undone therefore erroneous decision cannot be corrected.

Other methods like *density*, *model* and *grid* based have not yet been tested…

1) K-means: The *K-means* method is together with *K-menoids* the most commonly partitioning method used. The method selects k objects randomly as centroids representing the cluster centers of gravity. All other objects are assigned to the nearest cluster by the distance computed by the chosen metric function. Each cluster recalculates its mean followed by recalculation distances of all objects and their reassignment to the nearest cluster. This process iteratively continues until convergence is reached.

2) K-menoids: The *K-menoids* method has the same iteration algorithm as the *K-means* with a different representation of the center of gravity. The *K-menoids* remedies the biggest disadvantages of the *K-means* method which is outliers[2]. The outlier with an enormously bigger value than other cluster objects might distort the mean, therefore *K-menoids* uses one of those objects to represent the center of the cluster.

D. Increasing the accuracy

This section has discussed a general overview of machine learning algorithms, each one of them has its pros and cons. One method might be very accurate with classifying a particular subclass, but very inaccurate with the rest. The second method is opposite to the first, accurately classifying the rest, which was inaccurately classified by the first, but inaccurately classifies the particular subclass. Assuming this observation, we may ask if there is any general approach how to increase the accuracy? The answer is that there are several; the most commonly used are *bagging* and *boosting* [1].

1) Bagging: *Bagging* is an abbreviation of a bootstrap aggregation. The main idea behind bagging is a usage of k models of the same algorithm instead of one model. The labeled training data set is sampled by using bootstrap method into K subsets. Each subset is assigned to a single $M_1, ...M_k$ model to be trained in it. When classifying, each model classifies unlabelled data, and the label with the majority of votes is selected. On the outside, M_k models are conjunct into one $M*$ model.

The bootstrap method samples training data set and chooses training tuple with some probability, for example, 0.632 forming *0.632 bootstrap* method. The training set is tested d times, with replacement, resulting in a bootstrap sample or training set of d samples. Considering, it is very likely that some training tuples will be used more times than once and others, which are not used for the training, are moved to testing set [1].

2) Boosting: *Boosting* is based on very similar idea as begging but it includes a weight assignment mechanism to improve a learning accuracy. There are several boosting algorithms but for the sake of simplicity and because it is used in the state-of-the-art research, the *Adaboosting* algorithm will be used as an example.

We are given a D, a set of d labeled tuples. Each tuple is assigned a weight of $1/d$. The method generates k classifiers and iterates through k rounds. In each round, a training set D_i is created by sampling the D. Each tuple is sampled with a probability according to its weight. The classification model M_i is trained on D_i. The rest $D\backslash D_i$ forms a testing set, which is processed by classifier M_i. The classification is compared with the assigned label and weights of tuples are corrected. If the tuple were classified successfully, the weight would be decreased, otherwise, it would be increased [1].

This mechanism allows models learned in later iterations to focus more attention to previously erroneously classified tuples.

E. Semi-supervised learning

Semi-supervised learning algorithm class should be the most advanced because it takes benefits from supervised and unsupervised learning. Using a small amount of labeled data in conjunction with the vast amount of unlabelled can produce considerable improvement in learning accuracy.

Semi-supervised learning is very efficient when the labeling is very expensive and requires a human expert or a physical experiment. In contrast, an acquisition of unlabelled data is typically inexpensive.

Theoretically, these methods are based on *transductive* or *inductive learning/reasoning*. *Transductive learning* is trying to infer the correct label for given unlabelled data by training on specific cases alone —the specialization. The opposite is *inductive learning* that is understanding a global model from labeled training classes and classifying unlabelled, based on the global model knowledge —the generalization.

Practically, there are two commonly used methods —*Self training* and *Cotraining*.

[2] The outliers are objects that represent the noise. It might be an unknown, suspicious or otherwise unclassifiable user behaviour.

1) Self-training: Self-training uses a labeled data to train the classifier. Afterward, classifier tries to label the testing data and adds tuples with the most confident label to the training set and the process repeats. The simpleness of these methods is balanced by possible reinforcement which may occur [1].

2) Cotraining: Cotraining is more error resistant because there are more classifiers to train. Training data are split to x mutually exclusive and class-conditionally independent sets. Each of these sets is assigned to its respective $C_1, ..C_x$ classifier to be trained on. Then, classifiers are used to predict the class labels for testing data. Each classifier teaches other classifiers tuples with the most confident prediction; those tuples are then added to each classifier training set [1].

3) Active learning: Active learning is used when data are abundant, yet the class labels are only a few or expensive to obtain. The method is active in that it can query a user to provide labels. Therefore, the number of tuples needed for learning is much smaller than typically required.

Let D be all available data, then $D = L \cap U$, where L represents labelled and U unlabelled set, and $D \setminus U \subseteq L \bigwedge |L| \ll |U|$. The method uses one of the supervised learning algorithms to create a model trained on labeled tuples. Afterward, the method analyses unlabelled tuples and chooses one of more for human oracle to assign a label. Labeled tuples are used for increase accuracy of the model and the process repeats.

The method is reliant on algorithm selecting tuples for querying. Several methods have been presented, but the most commonly used is *uncertainty sampling*, which chooses tuples, with the least confident prediction of labels [1].

4) Transfer learning: Traditional learning algorithms assume that the training data and test data are drawn from the same distribution and the same feature space. Thus, if the distribution changes, such methods need to rebuild the models from scratch [1].

Transfer learning may even deal with continuous slight changes to the user behavior itself as long as the classifier observes data in chronological order. This would be very beneficent in an elimination of false positive warnings, but it might also introduce false negatives as well.

There are several approaches how to apply transfer learning; the most common is *instance-based transfer*. This method reweighs some of the data from the source identifier and uses it to learn the target identifier.

The *TrAdaBoost*, i.e., Transfer AdaBoost is an example of *instance-based transfer* learning approach. The algorithm requires a labeling only of a small part of the new data. It uses a considerable portion of the old data and filters out the influence of any that are very different from the new data by automatically adjusting weights assigned to the training tuples.

F. Measurement of performance

There are several statistical metrics used to measure performance learning methods. If we abstract and consider only binary classifiers, then we can define terms:

- *P* (positive) identified occurance *belonging* to classification class,
- *N* (negative) identified occurance *not belonging* to classification class,
- *TP* (true positives) correctly identified occurance *belonging* to classification class,
- *TN* (true negatives) correctly identified occurance *not belonging* to classification class,
- *FP* (false positive) incorrectly identified occurance as *not belonging* to classification class,
- *FN* (false negatives) incorrectly identified occurance as *not belonging* to classification class,

Based on these terms, the *confusion matrix* (see Table II) could be created for analyzing how well the classifier performs.

Classes	**Positive**	**Negative**	**Total**
Positive	TP	FN	P
Negative	FP	TN	N
Total	P'	N'	$P+N$

TABLE II. THE CONFUSION MATRIX FOR BINARY CLASSIFIER.

By using information from confusion matrix, other statistical indicators could be computed. The essential and the most important is the accuracy (see Equation 1). The accuracy of a classifier M on given test set is the percentage of test set tuples that are correctly classified.

$$accuracy(M) = \frac{TP + TN}{P + N} \tag{1}$$

The error rate or misclassification rate is a supplement of the accuracy (see Equation 2). It describes a percent coverage of tuples that were incorrectly classified e.g false positives and false negatives in regard to all data.

$$errorrate(M) = \frac{FP + FN}{P + N} = 1 - accurasy(M) \tag{2}$$

The accuracy and the error rate seems to describe the performance of a classifier reliably but only in the case that $|P| \cong |N|$, in other words, when none of the sets cardinalities dominates. To better describe performance on more realistic scenarios when one of bases dominates, we define a *sensitivity*, see Equation 3, and a *specificity*, see Equation 4. The sensitivity, i.e., the true positive recognition rate, is a proportion of *positive* tuples that are correctly identified. The specificity, i.e. the true negative rate, is the proportion of *negative* tuples that are correctly identified.

$$sensitivity(M) = \frac{TP}{P} \tag{3}$$

$$specificity(M) = \frac{TN}{N} \tag{4}$$

Combining all equations mention above, it could be shown that accuracy is a function of sensitivity and specificity (see Equation 5).

$$\begin{aligned} accuracy(M) = sensitivity(M)\frac{P}{P+N} + \\ specificity(M)\frac{N}{P+N} \end{aligned} \tag{5}$$

Another useful indicators that are commonly used are the *precision*, see Equation 6, and the *recall*, see Equation 7. The precision, i.e., exactness, is measuring a percentage of tuples labeled as positive that are positive. Quite similar to this is the recall, i.e., completeness, which is measuring a percentage of positive tuples that are labeled as such. Both have a meaning only when they are presented together because there tends to be an inverse relationship between them where it is possible to increase one at the cost of reducing the other. Typically, the recall is set to a constant value, let's say 0.75, and the precision is used as a metrics.

$$precision(M) = \frac{TP}{TP + FP} \tag{6}$$

$$recall(M) = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{7}$$

This combination of the precision and the recall might be difficult to work with; therefore, they can be combined into *F measure*, i.e., F_1 or F-score, see Equation 8, which is a harmonic mean of the precision and the recall and is commonly used to compare the performance of various classification methods.

$$F(M) = \frac{2 \times precision \times recall}{precision + recall} \tag{8}$$

If there were a preference to put more meaning to one of the precision or the recall, generalized F_β measure, see Equation 9, could be used. Commonly used F_β measures are F_2, which weights the recall twice as much as the precision, and $F_0.5$, which weights the precision twice as much as the recall [1].

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall} \tag{9}$$

IV. Outlier Detection

An *outlier detection* is a task of finding anomalous patterns that manifest abnormal behavior. The definition what is an abnormal behavior is defined by a context of given task. An *outlier* is a data sample that expresses abnormal patterns. An *outlier detection technique* is an outlier detection problem solver. A *normal pattern* is a pattern not representing anomalous behavior. The *output* of an outlier detection method is typically labeled model, classifying a pattern as a normal data or outlier. Besides, some outlier detection techniques can also provide an *outlier score* describing a probability with which a sample is classified as an outlier [6].

In routine machine learning task, especially in data mining, outliers are considered to be noise, errors or damaged data patterns. In other disciplines, outliers are very useful for detection of abnormal behavior of studied subject(s). Various researchers used anomaly detection to identify credit card frauds, network intrusions or malware activities and doing so, developed different detection mechanisms that are applicable in their particular application domain or are generically usable in multiple areas.

Outliers as patterns that do not express normal behavior are therefore considered as anomalies. On Figure 1 are present two clusters, i.e., $G1$ and $G2$, and two outliers, i.e., $O1$ and $O2$. Outliers might be in the form of single occurrence, e.g., $O1$, or a set of multiple occurrences with a low carnality, e.g. $O2$.

A. Outlier Types

In pursue of outliers detection, firstly, we need to review outlier types to be able to choose best-suited algorithms and approach. Outliers detection methods behave differently and are not generic. Outliers are classified into classes based on their composition and relation to rest of the data [6].

1) Type I: This kind of outliers is the most thoroughly studied and the detection is the most straightforward one. The majority of available researches have been done on *Type I* schemes detection. This type is typically defined as an individual outlying occurrence without any additional context. On Figure 1, outliers $O1$ and $O2$ are instance of a *Type I.* These outliers are identifiable by data internal relation analysis, i.e., analysis of the relationship of an individual instance concerning the rest of the data.

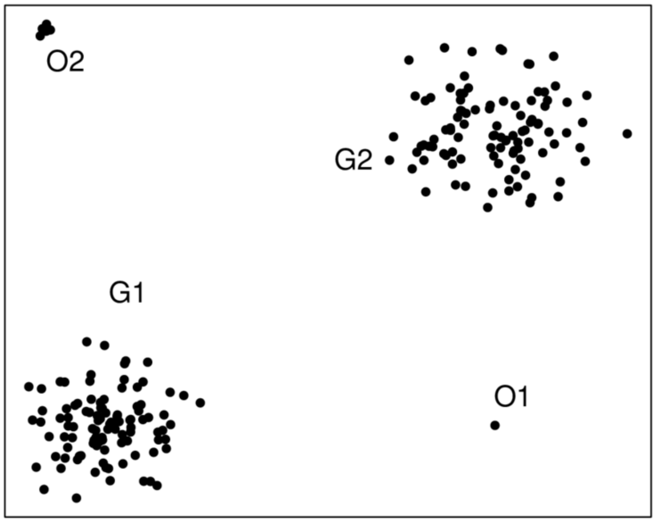

Fig. 1. Example of two-dimensional outliers. Point labeled O1 and points labeled O2 deviate significantly from groups of points labeled G1 and G2 [7].

2) Type II: This type is very similar to the *Type I*, i.e., instances are also individual occurrences, but with an exception that *Type II* outliers are contextual. *Type II* outlier is considered to be outlier only in a particular context. The context is defined by a data structure as a part of problem formulation. For the outlier to belong to *Type II* class, it needs to satisfy two properties: data has to have a *spatial/sequential nature* and the *behavior* is determined by values of behavioral attributes withing a context. The *spatial or sequential attributes* define a position of occurrence in a while data set; therefore, they are creating a context. The *behavior attributes* do not relate to the position in a data set; therefore, they are not contextual.

Type II outliers are commonly connected with timeline series or spatial data, but only, when time or space is considered to be a spatial or sequential attribute, i.e., introducing a context to data, not a behavior. On Figure 2 is a timeline series and points $t1$ and $t2$. Both points are in a range of values, but concerning data series, the point $t1$ fits, but $t2$ is obviously the outlier.

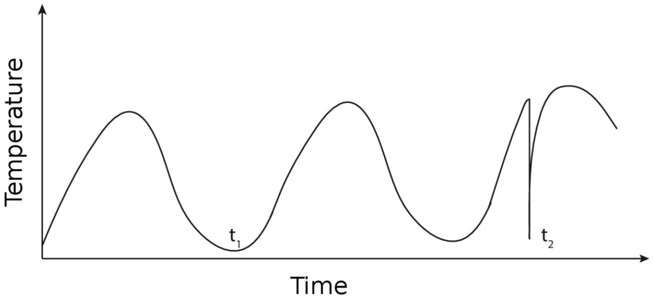

Fig. 2. An example of a contextual outlier. Points labeled t1 and t2 represent the same temperature value. Though, point t2 is considered an outlier, while point t1 is not [7].

3) Type III: In comparison with outliers of *Type I and II*, *Type III* outliers are not only single occurrences but anomalous subsequences or subgraphs that express abnormal behavior in sequential context. The individual instances of *Type III* outliers are not outliers by themselves, but they are considered to be outliers only in a subsequence context. The same property condition restricting the presence of *spatial/sequential data nature* and the *behavior attribute* in data has to apply for an outlier to belong to *Type III* and more strongly related to data sequence and not an only individual instance.

On Figure 3 is EKG (electrocardiograph) that shows a periodical heart rhythm. On x axis at a point of 6000, you can observe that there was an anomaly when the heart did not made a pulse. This outlier is not detectable as *Type I*, because the point 6000 is in a value range, and also not as *Type II* because the pulse consists of multiple points and not only single occurrence.

B. Detection Approaches

The result of outlier detection can be dual. Detection method might only label occurrences as outliers without any additional information, i.e., to behave like binary classifiers, those are called *Labeling Techniques* or to provide the *Outlier score* to emphasize how strongly is an occurrence considered to be an outlier. *Outlier score* based methods are more powerful compared to *Labeling Techniques* because the decision threshold with which is outlier proclaimed as outlier may be variably defined.

Outliers detection have been broadly studied in past decades. A large number of detection methods have been constructed, a majority of them were based on particular discipline and are derived from concepts previously known, applied and best suited for the related problem. Disciplines like Statistics, Information Theory, Classification, and Clustering are a popular knowledge base for methods to be constructed from.

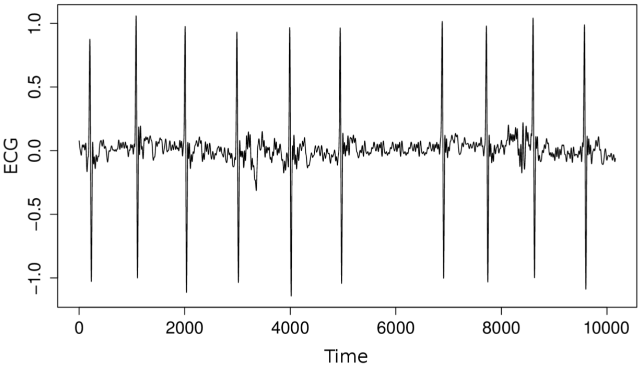

Fig. 3. An example of the collective outlier in the human electrocardiogram [7].

The most exhaustively studied were outliers of *Type I*. Therefore, a majority of methods can detect them, and some of them are modifiable to also detect outliers of *Types II and III*. Some techniques have been constructed solely for complex *Types II and III*.

For *Type I* outliers, we group existing algorithms according to a discipline and principles that they are based on following categories. For the sake of brevity, only pros [+], cons [-] and a brief description of methods [M], types [T] and algorithms [A] is provided.

- **Classification Based** [6, pp. 20-23]
  - - Majority requires expert knowledge, i.e., labels, because of their supervised nature.
  - ○ *Neural Network Based* [6, pp. 23-27]
    - A Multi Layered Preceptrons, Self Organing Maps, Habituation Based, Neural Trees, Autoassociative Networks, Adaptive Resonance Theory Based, Radial Basic Funtion Based, Hopfield Networks, Oscillatory Networks
    - M Backpropagation
    - T Supervised, (Unsupervised)
    - + Robust method,
    - + easily paralellized,
    - + fast classification,
    - + small model size,
    - + multilabel classification,
    - + well known,
    - + attributes are weighted.
    - + Self Organizing Maps are unsupervised.
    - - Slow in learning phase,
    - - tends to overfit.
  - ○ *Bayesian Network Based* [6, pp. 27]
    - A Naive Bayesian Network, Pseudo-Bayes Estimators
    - M Bayes Theorem
    - T Supervised
    - + Easily paralellized,
    - + well known,

    - + yields outlier score.
    - - Not as accurate,
    - - binary classification.
    - ■ Training creates a tree like structure where all child nodes are variables which feed the value to one root node for aggregation and classification of the event as normal or outlier.
  - ○ *Support Vector Machine Based* [6, pp. 28]
    - A Support Vector Machine, Robust Support Vector Machine
    - M Kernel functions for hyperplane exportation.
    - T Supervised, unsupervised based on high and low-density regions), semi-supervised for single class classification
    - + Fast classification,
    - + reasonably accurate,
    - - binary classification.
    - - It is hard to find fitting hyperplane exportation function, kernel functions usually used to approximate dot product between mapped vectors.
    - - Overfits when outliers exist in training set.
    - ■ Separates data belonging to different classes by fitting a hyperplane between them with maximized separation.
  - ○ *Rule Based Models* [6, pp. 28-29]
    - A IREP, RIPPER, robust C4.5, ADAM, LERAD
    - M Frequent Pattern Outlier Factor, Frequent Episodes
    - T Supervised, semi-supervised for single-class classification.
    - + Fast classification,
    - + reasonably accurate,
    - + multilabel classification.
    - + Does not overfit when outliers exist in training set.
    - ■ Captures normal behavior of data, any instance not covered by model is considered as an outlier.
- **Clustering Based** [6, pp. 30-31]
  - A Self-Organizing Maps, K-means Clustering, Expectation Maximization
  - M Semantic outlier factor,
  - T Unsupervised, semi-supervised
  - + Detects relation between data.
  - + Does not require expert knowledge.
  - + Utilized when incremental model creation is needed.
  - - Majority is not primarily developed for outlier detection which is a byproduct of clusterization.
  - - Usually very computationally expensive.
  - ○ Based on the assumption that clusters contain normal data and anything not belonging to the cluster, or belongs to a small cluster is considered an outlier.
  - ○ Clustering partitions data into groups which contain similar objects.
- **Nearest Neighbour Based** [6, pp. 32-36]
  - A Probabilistic Suffix Trees, Outlier Detection using In-degree Number, Multi-granularity Deviation Factor Recursive Binning, and Re-Projection
  - M Relative Density Factor, Distribution of pair-wise distance, Local Outlier Factor, Connectivity-based Outlier Factor, Set Based Nearest path
  - T Unsupervised
  - + Analyzes each data object with respects to its neighbors.
  - + Does not require expert knowledge.
  - - Requires well defined metric to measure proximity, defined as a distance or similarity.
  - - Sensitive when data density varies except *Local Outlier Factor* aware methods.
  - - Competitively expensive $O(n^2)$.
  - ○ Based on a knowledge of distance, or proximity between data. Compared to clustering which is analyzing data from the outer point of view, NNB analyzes with respect to local neighbors.
- **Statistical** [6, pp. 36-45]
  - + Usually competitively very efficient.
  - + Incorporates also methods that do not require expert knowledge.
  - + Usually yields outlier score.
  - ○ Assumes that outlier is not generated by a stochastic model. Thus statistical analysis can determine a probability that data were generated by a statistical model.
  - ○ Parametric
    - + Singlevariate or multivariate support.
    - - Assumes that data are generated with a known distribution which not always captures real word scenarios.
    - A Gaussian Models
      - M Grubbs, Rosner, Dixon, t-test tests, Mahalanobis distance, Maximum Lekhood Estimates,
      - T Supervised
      - ∗ Assumes normal distribution, computes variance, mean.
      - ∗ Visualizable using *box-plot.*
      - ∗ Variety of statistical tests *Grubbs, Rosner, Dixon, t-test tests*
      - ∗ Uses metric, like *Euclidean, Mahalanobis*, to calculate data similarity, closeness.
    - A Regression Models
      - M Maximal likelihood estimation, projection pursuit, Kurtosis coefficient, Studentized residuals, Akaike Information Content, Robust regression, ARIMA
      - T Supervised
      - ∗ Extensively investigated for time-series data.
      - ∗ Tries to fit regression model on data.
      - ∗ Quarrel about completeness among statisticians.
      - ∗ Complexity increase with multivariate data.

            - ∗ Recognizes *Observational* and *Innovations* outliers, i.e., outliers that affects subsequent observations.
        - A Mixture of Parametric Models
            - M Expectation Maximization, extreme value statistics, Hierarchical shrinkage
            - T Both supervised and semi-supervised
            - ∗ Methods tries to model normal instances and outliers as separate parametric distributions in supervised mode. During verification, the method determines to which distribution sample belongs to.
            - ∗ In semi-supervised, methods tries to models normal instances as a mixture of models. A test instance that does not belong to any distribution is determined to be an outlier.
        - A Markov and Hidden Markov Models
            - M Maximum entropy models, Conditional Random Fields, mixture of HMM
            - T Supervised
            - ∗ Methods used to model sequential data.
            - ∗ Methods are also able to detect *Type II* outliers.
    - ○ Non-parametric
        - + Does not assume knowledge of data distribution.
        - + Very competitively efficient.
        - ▪ Histograms
            - A Packet Header Anomaly, Application Layer Anomaly Detection
            - M Q, S, IS statistic
            - T semi-supervised, unsupervised
            - ∗ Also called *Frequency, counting based*.
            - ∗ Typically defines a distance between test data and learned histogram profile and classify outlier based on defined threshold.
        - A Final State Machines
            - T Semi-supervised
            - ∗ FSA represents normal data behavior that has temporal or sequential nature.
            - ∗ Everything that is not identified as normal is proclaimed to be an outlier.
            - ∗ Mostly focused on *Type II and III* outlier detection.
        - ▪ Kernel Functions
            - A Parazen Windows Estimation
            - T Supervised, Semi-supervised
            - ∗ Uses *kernel functions* to approximate density distribution, similarly to parametric methods with the only difference that the distribution function is used.

- **Information Theory Based** [6, pp. 45-46]
    - A Local Search Algorithm
    - M Komogorov Complexity
    - ○ Utilizing information theoretic measures such as *entropy, relative entropy, relative conditional entropy, information gain, and information cost*
    - ○ Measures *regularity* and any point that introduces *irregularity* is proclaimed to be an outlier.
- **Spectral Decomposition Based** [6, pp. 46-47]
    - A Principal Component Analysis (PCA)
    - M Dimensionality reduction
    - T Unsupervised, semi-supervised
    - ○ Based on assumption that top few PC capture bulk of variability and expect that smallest PC result in constant values. Therefore, data that violates smallest PC are outliers.
- **Visualization Based** [6, pp. 47]
    - A Grand tour
    - M Concentration ellipse
    - T Unsupervised
    - ○ Data inside concentration ellipse are normal occurrences, outside are outliers. The number of times an instance is outlier in all projection reflects its outlier score.

Type II outliers differ of the previous category by their contextuality. Every *Type II* outlier is recognize on universum that contains *contextual attributes* that defines context, and *behavioral attributes* to detect outliers withing a context. The *contextual attributes* provide a structure to the data of following types: *spatial, sequential, profile*.

Spatial attributes define a location of data instances in given context, i.e., neighborhood. Typically used to detect outliers for geographical information systems [6, pp. 47].

Sequential attributes define a position of data in a given sequence. Typically used for time-series, event data with timestamps defined for each instance, such as operating system calls The difference between time-series and event sequence is that consecutive inter-arrival time for event data is uneven [6, pp. 48].

Profile is used when data do not have explicit spatial or sequential structure but are segmentable or clusterable into components using a set of contextual attributes [6, pp. 48].

For outliers belonging to the *Type II*, we do similarly recognize these categories:

- Reduction to *Type I* outliers Detection Problem [6, pp. 48-49]
    - A Per group analysis
    - M Class outlier detection, Spatial Local Outlier Measure,
    - T Unsupervised, semi-supervised
    - ○ Methods capable of semantic context understanding and outlier detection within.
- Utilizing the Structure in Data [6, pp. 49-51]
    - A Robust regression, auto-regession models, ARMA models, ARIMA models, Markov Chain Monte Carlo
    - M Kolmogorov Complexity, Frequent Items Mining, FSA, Markov Models
    - ○ Usable in cases where context is not straightforward, e.g., time-series, event sequences.

Type III outliers are a subset of instances that occurs together as a substructure and whose occurrence is not normal with respect to a normal behavior [6, pp. 51]. The individual substance is not an outlier by is self, but only might be in a given context. It is the most challenging type of outliers to be detected. It requires that data have a structure and recognizes two types: *sequential* and *spatial*.

Sequential Type III methods analyze sub-sequences in sequential data and look for sequences that do not fit, i.e., outliers. A typical application is to detect call frauds in sequential call data or numerical time-series sequences.

Spatial Type III methods look for connected subgraphs or subregions that are not common and detected as outliers. The application is typical for multi-spectral imagery data and graph data.

For outliers belonging to the *Type III*, we do similarly recognize these categories:

- Handling sequential outlier [6, pp. 51-61]
  - ○ Reduction to *Type I* outliers detection problem
    - A Adaptive Resonance Theory (NN), Clustering, Box Modeling, RIPPER
    - T Supervised, unsupervised
    - ▪ Transformation of sequence into a feature space and application of any *Type I* method.
    - ▪ Used when sequences are properly aligned.
  - ○ Modeling Sequences
    - A Hidden Markov Models, Finite State Automatons, Probabilistic Suffix Trees, Sparse Markov Trees
    - M Extracting Short Sequence from a Long Sequence, Detecting Outlier Sequences in a Long Sequence
    - T Supervised, semi-supervised
    - ▪ Used when sequences are not properly aligned.
  - ○ Used when assumption about sequence alignment becomes too prohibitive.
- Handling spatial outliers [6, pp. 61]
  - A Multivariate Gaussian Random Markov Fields,
  - M Enthropy, bottom-up subgraph enumeration.
  - T Unsupervised
  - ○ Searches for a subgraph or subcomponent that is anomalous in a data.
  - ○ Very scarcely researched.

V. Conclusion

This paper briefly introduced machine learning as a concept used for classification and clustering. Basic statistical features and recommendation how to evaluate performance in order to compare results of several ML methods are also briefly discussed.

The contribution of this paper is the analysis of Safetica's DPL platform and identification of candidate features that are to be used in the following proof-of-concept solution implementation. Also, based on the requirement that outlier detection is to be run on end-points, a comparison of outlining detection methods for each type of outlier was summarized.

Based on an acquired knowledge, it is recommended to proceed with an implementation of *statistical based* outlier detector that will be computationally very efficient to run on end-point devices. Learned models, used for classification of user behavior, and its byproduct outlier detection, are generally very space efficient, thus easy to store and fast to transfer.

Statistical methods provide an excellent scalability and future extensibility. Because statistical methods can incorporate weighted features, and even weighted features per model, whenever there is a new user behavior observed, a simple feature detecting it can be created and it gives meaning only for this particular situation without a consequential noise introduction like in a case of conventional ML methods. Because statistical methods are *model based*, i.e., each event that is classified by them need to have a model, a fingerprint, to be compared with. Statistical methods do provide an *outlier score* which signifies how strongly is the event considered to be an outlier.

The author believes that because of extensibility of statistical methods and conventional approaches for outliers detection of *Type II and III*, it would be best to implement non-contextual statistical based outlier detecting proof-of-concept solution that will be able to create behavioral user models. Consequential user behavior will be compared to those models and assumptions will be made, whether the new behavior corresponds to regular user activities or is to be considered as the anomaly.

VI. Acknowledgement

This research was supported by EUROPEAN UNION, European Regional Development Found Operational Programme Enterprise and Innovations for Competitiveness and Czech Ministry of Industry and Trade.

EVROPSKÁ UNIE
Evropský fond pro regionální rozvoj
Operační program Podnikání
a inovace pro konkurenceschopnost

MINISTERSTVO
PRŮMYSLU A OBCHODU

References

[1] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[2] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2012.

[3] Y. Wang, Y. Xiang, and S. Yu, "Internet Traffic Classification Using Machine Learning: A Token-based Approach," *2011 14th IEEE International Conference on Computational Science and Engineering*, pp. 285–289, 2011.

[4] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948. [Online]. Available: http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf

[5] G. La Mantia, D. Rossi, A. Finamore, M. Mellia, and M. Meo, "Stochastic packet inspection for TCP traffic," in *IEEE International Conference on Communications*, 2010.

[6] V. Chandola, A. Banerjee, and V. Kumar, "Outlier detection: A survey," *ACM Computing Surveys*, 2007.

[7] Wikipedia, "Outlier — Wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Outlier&oldid=784274201, 2017, [Online; accessed 13-June-2017].