Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Department of Mathematics and Physics

Master of Data Science

An Anomaly Detection Process for a Business Solution

THESIS to obtain the **DEGREE** of
MASTER OF DATA SCIENCE

A thesis presented by:
Ana Victoria López Miranda

Thesis Advisors:
Dr. Juan Diego Sánchez Torres
Dr. Riemann Ruiz Cruz

Tlaquepaque, Jalisco, November, 2020

Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Department of Mathematics and Physics

Master of Data Science Approval Form

Thesis Title: **An Anomaly Detection Process for a Business Solution**

Author: **Ana Victoria López Miranda**

Thesis Approved to complete all degree requirements for the Master of Science Degree in Data Science.

Thesis Advisor, **Dr. Juan Diego Sánchez Torres**

Thesis Co-Advisor, **Dr. Riemann Ruiz Cruz**

Thesis Reader, **Dr. Rocío Carrasco Navarro**

Thesis Reader, **M. Juan Carlos Martínez Alvarado**

Academic Advisor, **M. Juan Carlos Martínez Alvarado**

Tlaquepaque, Jalisco, November, 2020

An Anomaly Detection Process for a Business Solution

Ana Victoria López Miranda

In this work, a business solution's implemented using machine learning algorithms. The solution consists of a particular realistic case of a corporation where the financial department has struggled to evaluate large quantities of information to capture cost irregularities from its external suppliers' billing process.

The design of this solution is to solve three critical problems from the business, a tool that detects the anomalies in an automated fashion that helps the increase savings reducing the number of experts who are currently needed only to detect the anomalies. Implement unsupervised machine learning methods that allow a massive tagging of the data due to the current lack of labels in the information.

Apply a method whose results are validated and reviewed by the business Subject Matter Experts, a.k.a. SMEs for use. A process that simulates the expert's classification is generated, using a method that allows us to tag the historical data and accelerate the SMEs' manual labeling.

The overall workflow consists of five different phases, where first gather the information from the organization into a single database from where the feature transformation and selection are applied. Once the characteristics are defined and ready to use, the process continues with the unsupervised training using a probabilistic method that provides us with the massive tagging of our binary classification. The labeled dataset is then shared with the business experts for a review and feedback process in which they provide us the correct classification for the observations that went into the model.

Finally, the data is inputted into a supervised algorithm selected through a fixed accuracy threshold and contamination rate. Using these parameters as conditions, the model that adjusts better than the probabilistic unsupervised approach is then selected. When the criteria are met, the model is deployed to a production environment for user consultation.

Contents

List of Figures

List of Tables

Dedicated to ITESO University, especially to the Mathematics and Physics department and all professors who contributed to my development and supported me in every stage of my studies during these two years to obtain my master's degree. To my thesis mentor and professor Juan Diego Sanchez, who provided me with the resources and knowledge required to keep moving forward in my thesis and studies. To my husband Omar, who also happens to be my classmate and team. Who has always supported me since the beginning and stood by my side on every step and obstacle. Thank you for always pushing me further and trusting in me. To my parents, my first teachers

and my unconditional support, all my love and appreciation to them and my siblings, thank you.

Introduction

1 Introduction

Contents

1.1 Motivation

NOWADAYS, we begin to experience the disruption of machine learning applications in almost every subject, particularly when it comes to large enterprises that thrive on seeking AI solutions for every business process. Most of these studies refer to financial areas, where data is often available, and day-to-day business is the problem of predicting trends and anomalies.

The cons that emerge with this type of implementation is the fact that the data is not always prepared to implement machine learning algorithms rapidly, requiring, in most cases, significant processing behind as well as inputs from experts to provide missing data or even labeled fields for a particular subject that needs to be addressed. In particular, this is one of the biggest challenges in any ML project; when the solution begins to be introduced, some of the first findings are that there are no labels available to fix this question. Data as independent variables are stored in the database. On the other hand, that is not available for the dependent variable that needs to be expected. This problem is getting bigger every time.

Therefore, a reason for this work is to satisfy a recurrent need of helping the business in labeling those required fields through unsupervised methods that are able to reach a good percentage of precision through small labeling efforts and spread that prediction over

to broader data sets, so these labels can start to be considered as input for supervised models.

1.2 Background

An international corporation has various business cases from which it wants to leverage business value; some of the concerns relate to identifying irregularities that must be documented in the invoicing process. An outlier is described as possible fraud on suppliers based on the invoice document's information. These fields are primarily linked to the company's internal data, such as addresses, locations, facilities, billing objects, specifics of those items, prices, taxes, descriptions, and other detailed information about its sub-client, the supplier itself.

The leading case concerns a category of prices categorized by the business in which we would like to detect a shift or variance based on regular rates. If the result has been registered, the information will be inputted into a classifier algorithm that can distinguish simple classes of potential VS typical behavior anomalies.

This project's main objective is to move from multiple unsupervised models to a single outlier method; this captures 'anomalies' for the company. The algorithms will be evaluated from an empirical to a theoretical point of view, describing the precision, variance, and other metrics that the model collects most of the information and produces the best results. These results will receive feedback from the SME(Subject Matter Expert) company, choosing the best algorithm for a posterior implementation in the business process.

1.3 Objectives

1.3.1 Main Objective

This work aims to approach multiple unsupervised models to two different and tailored mathematical methods to capture what a financial process would call anomalies in the day to day operational course.

1.3.2 Secondary Objectives

1. Identify the best unsupervised method that replicates the experts classification and identification of outliers.

2. Train a supervised model that uses the output from the unsupervised approach as input for its training process.

2 Mathematical Preliminaries

Contents

In this chapter, the mathematical preliminaries are presented as a complementary base for the work; it includes the definitions of various unsupervised and supervised methods such as PCA, MCD, clustering, K-Nearest Neighbors, and Isolation Forest. In addition, some key algebra concepts will be approached as a complementary explanation of the models.

2.0.1 *Linear Models*

2.0.2 *Principal Component Analysis (PCA)*

The main objective in PCA is finding projections x_n of data points that x_n are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality [1]. More precisely, an i.i.d. dataset is considered $X = \{x_1, ..., x_n\}, x_n \in R^D$, with mean 0 that possesses the data covariance matrix.

$$S = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T \tag{2.1}$$

[1] M.P. Deisenroth, A.A. Faisal, and C.S. Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020. ISBN 9781108470049

This conduct us to a low-dimensional reduced representation of x_n, from where the projection matrix's obtained:

$$z_n = B^T X_n \in R^M \tag{2.2}$$

$$\boldsymbol{B} := [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M] \in \mathbb{R}^{D \times M} \tag{2.3}$$

The aim is to find projections $\tilde{x}_n \in R^D$ so that they are as similar to the original data x_n and minimize the loss due to compression.

PCA has long been used for multivariate outlier detection. It considers the sample principal components, $y_1, y_2, \ldots, y_n$, of an observation x. The sum of the squares of the standardized principal component scores,

$$\sum_{i=1}^{n} \frac{y_i^2}{\lambda_i} = \frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} + \ldots + \frac{y_n^2}{\lambda_n}$$

is equivalent to the Mahalanobis distance of the observation x from the mean of the sample, where [2]

$$T^2 = \frac{n}{n+1}(\mathbf{X} - \overline{\mathbf{X}})' \mathbf{S}^{-1} (\mathbf{X} - \overline{\mathbf{X}})$$

at the same time, distributed as

$$\frac{(n-1)p}{n-p} F_{p,n-p}$$

Since the sample principal components are uncorrelated, under the normal assumption and assuming the sample size is large, it follows that

$$\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} = \frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} + \ldots + \frac{y_q^2}{\lambda_q}, \quad q \leq p$$

has a chi-square distribution with the degrees of freedom q. For this to be true, it must also be assumed that all eigenvalues are distinct and positive

$$\lambda_1 > \lambda_2 > \ldots > \lambda_p > 0.$$

Given a significance level α , the outlier detection criterion is given if

$$\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} > \chi_q^2(\alpha)$$

where $X_q^2(\alpha)$ is the upper α percentage point of the chi-square distribution with the degrees of freedom q.

To establish a detection algorithm using PCA, the model performs on top of the correlation matrix of the normal group. The correlation matrix is used because each feature is measured in different scales. In the proposed scheme, the principal component classifier (PCC) consists

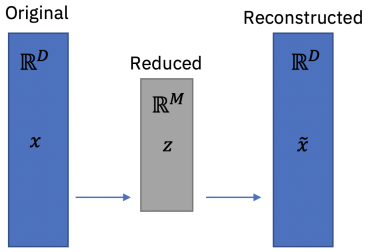


Figure 2.1: PCA Graphic representation.

[2] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Miami Univ Coral Gables Fl Dept Of Electrical and Computer Engineering, 2003

of two functions of principal component scores, one from the major components which contain most of the variance, $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i}$ and another from the minor components $\sum_{i=p-r+1}^{p} \frac{y_i^2}{\lambda_i}$ where r indicates the minor components used in PCC whose variances or eigenvalues are less than 0.2 that indicates less correlation among the features.

The number of major components is calculated by knowing the amount of the variation in the data that is considered by these components. The classification method computes the principal component scores of each observation for which the class is to be determined between outliers and inliers.

- Classifies x as an outlier if $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} > c_1$ or $\sum_{i=p-r+1}^{p} \frac{y_i^2}{\lambda_i} > c_2$.
- Inlier $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} \leq c_1$ and $\sum_{i=p-r+1}^{p} \frac{y_i^2}{\lambda_i} \leq c_2$.

where c_1 and c_2 are outlier thresholds such that the classifier would produce a specified contamination rate.

2.0.3 *Minimum Covariance Determinant (MCD)*

The Minimum Covariance Determinant (MCD) method [Rousseeuw, 1984] is a highly robust estimator of multivariate location and scatter. Developed as a distributional fit to Mahalanobis distance which uses a robust shape and location estimate.

Given n data points, the MCD of that data is the mean and covariance matrix based on the sample of size $h (h \leq n)$ that minimizes the determinant of the covariance matrix [3].

[3] Peter Rousseeuw. Least median of squares regression. *Journal of The American Statistical Association - J AMER STATIST ASSN*, 79:871–880, 12 1984. DOI: 10.1080/01621459.1984.10477105

$$J = \left\{ \text{ set of } h\text{points} : \left|S_J^*\right| \leq |S_K^*| \, \forall \text{ sets } K \text{ s.t. } \#|K| = h \right\}$$

where $\#|\omega|$ defines the number of elements in set ω

$$\bar{X}_J^* = \frac{1}{h} \sum_{i \in J} x_i \tag{2.4}$$

$$S_J^* = \frac{1}{h} \sum_{i \in J} \left(x_i - \bar{X}_J^*\right) \left(x_i - \bar{X}_J^*\right)^t$$

$$MCD = \left(\bar{X}_J^*, S_J^*\right) \tag{2.5}$$

The value h can be thought of as the minimum number of points which must not be outlying. The MCD has its highest possible breakdown at $h = [\frac{(n+p+1)}{2}]$.

The raw Minimum Covariance Determinant (MCD) estimator with parameter $[\frac{(n+p+1)}{2}] \leq h \leq n$ defines the following location and dispersion estimates:

1. $\hat{\mu_0}$ is the mean of the h observations for which the determinant of the sample covariance matrix is minimal
2. $\hat{\Sigma_0}$ is the corresponding covariance matrix multiplied by a constant factor c_0.

The concept of the MCD can be modified easily to fit the multiple cluster setting. With a good initialization and a known number of clusters g,the MCD can be found separately for each of the clusters. The size of each cluster is determined by the number of points which are closer to that cluster center than to any other cluster center. The sizes of the clusters and the MCD samples will be n_i and $h_i = [\frac{(n+p+1)}{2}] i = 1, \ldots, g$, respectively.

As a side note, the MCD estimator can only be computed when $h > p$, otherwise the covariance matrix of any h-subset will be singular. Since $h \geq [(n+2)/2]$, this condition is certainly satisfied when $n \geq 2p$. To avoid the curse of dimensionality it is however recommended that $n > 5p$.

The outlier detection method described here is not dependent on any particular robust clustering algorithm. Any robust initialization would presumably give similar results. Random starts could be used if a condition is added to prevent the clusters from converging to a large dataset shape.

The core of the MCD estimation algorithm is as follows [4]:

- Let $H1$ be a subset of h points.
- Find $\overline{X}_{H_1}$ and S_{H_1} . (If $det(S_{H_1}) = 0$ then add points to the subset until $det(S_{H_1}) > 0$.)
- Compute the distances $d^2_{S_{H_1}}(x_i, \overline{X}_{H_1}) = d^2_{H_1}(i)$ and sort them for some permutation π such that,

$$d^2_{H_1}(\pi(1)) \leq d^2_{H_1}(\pi(2)) \leq \cdots \leq d^2_{H_1}(\pi(n)) \tag{2.6}$$

- $H_2 := \{\pi(1), \pi(2), \ldots, \pi(h)\}$

As a summary of all previous steps, the complete procedure for calculating the MCDs for each cluster is as follows:

1. Use a clustering algorithm) to find an initial robust clustering of the data.
2. From the initial clustering, calculate the mean and covariance of each of the clusters. (Each point belongs to at least one cluster, use the points belonging to a particular cluster to calculate its mean and covariance in the usual way.)

[4] Johanna Hardin and David Rocke. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Computational Statistics & Data Analysis*, 44:625–638, 01 2004. DOI: 10.1016/S0167-9473(02)00280-3

3. Calculate the MCD to each cluster, based on the most recently calculated mean and covariance, for each point in the dataset.
4. Assign each point to the cluster for which it has the smallest MCD,thereby determining a cluster size (nj) for each cluster based on the number of points that are closest to that cluster.
5. For each cluster, choose a "half sample" $(h_j = [(n_j + p + 1) = 2])$ of those points with the smallest Mahalanobis distance from step 4.
6. For each cluster, compute the mean and covariance of the current half sample.
7. Repeat steps 4–7 until the half sample no longer changes.

2.1 Proximity-Based method

2.1.1 Clustering-Based Local Outlier Factor

With the use of the outlier method *CBLOF*, the degree of a record's deviation can be determined, also known as anomalies. To compute the *CBLOF* approach, a clustering algorithm is needed. In this work, the *squeezer algorithm* is reproduced and introduced with the *CBLOF* explanation. The $|S|$ representation is utilized to denote the size of S, where S in general, is a set containing some elements.

Definition 1 *Let $A_1, ..., A_m$ be a set of attributes with domains $D_1, ..., D_m$, respectively. Let the dataset D be a set of records where each record $t : t \in D_1 \times \ldots \times D_m$. The results of a clustering method applied to D are denoted as $C = \{C_1, C_2, \ldots, C_k\}$ where $C_i \cup C_j = \emptyset$ and $C_1 \cup C_2 \cup, \ldots, \cup C_k = D$, where k represents the number of clusters*[5].

[5] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. Technical report, Harbin Institute of Technology, Harbin 150001, P.R. China Department Of Computer Science and Engineering, 2003

One of the major priorities while defining the cluster-based local outlier algorithm is how to identify whether a cluster is large or small. As discuss in the next definition:

Definition 2 *Suppose $C = \{C_1, C_2, \ldots, C_k\}$ is the set of clusters in the sequence that $|C_1| \geq |C_2| \geq, \ldots, \geq |C_k|$. Given two numeric parameters α and β, b is defined as the boundary of large and small clusters if one of the following formulas holds:*

$$\begin{cases} (|C_l| + |C_2| + \ldots + |C_b|) \geq |D| * \alpha \\ |C_b| \,/\, |C_{b+1}| \geq \beta \end{cases} \tag{2.7}$$

In that sense, the larger cluster is defined as: $LC = C_i, |i \leq b$ and the set of small clusters is defined as $SC = C_j | j > b$. From the first formula, most data points are identified in the data set as not outliers. Therefore, clusters that hold a large portion of data points should be considered

as large clusters. From the second formula is deducted that large and small clusters should have significant differences in size. For instance, if a k is added to β, the size of any cluster in LC is at least k times greater than the clusters in SC.

Definition 3 *Suppose $C = \{C_1, C_2, \ldots, C_k\}$ is the set of the clusters in the sequence $|C_1| \geq |C_2| \geq, \ldots, \geq |C_k|$ and the meanings of α, β, b, LC and, SC are the same as formalized in Definition 2. For any record t, the $CBLOF$ of t is defined as:*

$$CBLOF(t) = \begin{cases} |C_i|^* \text{ min (distance } (t, C_j)), \text{ where } t \in C_i, C_i \in SC \text{ and } C_j \in LC \text{ for } j = 1 \text{ to } b \\ |C_i|^* \text{ (distance } (t, C_i) \text{ where } t \in C_i \text{ and } C_i \in LC \end{cases} \tag{2.8}$$

The $CBLOF$ of a record is then determined by the size of its cluster, and the distance between the record and its closest cluster, if the observation lies in small clusters, it's determined by the distance between the record and the cluster it belongs to which provides importance to the local data behaviour.

Let $A_1, ..., A_m$ be a set of attributes with domains $D_1, ..., D_m$, respectively. Let the dataset D be a set of tuples where each tuple $t : t \in D_1 \times \ldots \times D_m$. Let TID be the set of unique ID of every tuple. For each $tid \in TID$, the attribute value for A_i of corresponding tuple is represented as $tidA_i$. Given a Cluster C and a tuple t with $tid \in TID$, the similarity between C and tid is defined as:

$$\text{Sim}(C, tid) = \sum_{i=1}^{m} \left(\frac{\text{Sup}\,(a_i)}{\sum_{a_j \in VAL_i(C)} \text{Sup}\,(a_j)} \right) \text{ where } tid.A_i = a_i \tag{2.9}$$

From the equation above is deducted that the larger the similarity is between a tuple and an existing cluster, the bigger the probability that this tuple belongs to it. This algorithm has n tuples as input that are read in, and a Cluster Structure (CS) is constructed with the first tuple.

The similarity index is computed for all tuples in all existing clusters, the largest value of similarity is then found, if the value is larger than the given threshold, defined by s, the tuple will be put into the cluster that has the largest value of similarity. The CS is also updated with the new tuple. If the above condition does not hold, a new cluster must be created with this tuple.

2.1.2 *k-Nearest Neighbors*

To denote the distance of point p, the next representation is used $D^k(p)$ from its k^th nearest neighbor. The points are ranked on the basis of their $D^k(p)$ distance, leading to the following definition for $D^k(n)$ outliers:

Definition 4 [6] *Given an input data set with N points parameters n and k, a point p is a D_n^k outlier if there are no more than $n-1$ other points p' such that $D^k(p') > D^k(p)$.*

[6] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. volume 29, pages 427–438, 06 2000. DOI: 10.1145/335191.335437

In other words, if the points are ranked according to their $D^k(p)$ distance, the top n points in this ranking are considered outliers. Many metrics can be used such as the ("Manhattan") or ("euclidean") distances for measuring the length between a pair of points. With the previous definition4 for outliers, it is possible to rank outliers based on their $D^k(p)$ distances, outliers with larger $D^k(p)$ distances have fewer points close to them and are thus intuitively stronger outliers.

For this specific approach, the square of the Euclidean distance is applied (instead of the Euclidean distance itself) as the distance metric involves fewer and less expensive computations. The distance is denoted between two points p and q by $dist(p,q)$. A point p in δ dimensional space is included by $[p_1, p_2, \ldots, p_\delta]$ and a δ dimensional rectangle R by the two endpoints of its major diagonal: $r = [r_1, r_2, \ldots, r_\delta]$ and $r' = [r'_1, r'_2, \ldots, r'_\delta]$ such that $r_i \leq r'_i$ for $1 \leq i \leq n$. Let us denote the minimum distance between the point p and rectangle R by MINDINST(p,R), where,

$$\text{MINDIST}(p,R) = \sum_{i=1}^{\delta} x_i^2, \text{ where } x_i = \begin{cases} r_i - p_i & \text{if } p_i < r_i \\ p_i - r'_i & \text{if } r'_i < p_i \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

Every point in R is at a distance of at least MINDINST(p,R) from p. The maximum distance between the point p and rectangle R is denoted by MAXDIST(p,R). That is, no point in R is at a distance that exceeds MAXDIST(p,R) from point p. MAXDIST(p,R) is calculated as follows:

$$\text{MAXDIST}(p,R) = \textstyle\sum_{i=1}^{\delta} x_i^2, \text{ where}$$
$$x_i = \begin{cases} r'_i - p_i & \text{if } p_i < \frac{r_i + r'_i}{2} \\ p_i - r_i & \text{otherwise} \end{cases} \tag{2.11}$$

The minimum and maximum distance are defined between two MBRs. Let and be two MBRs defined by the endpoints of their major diagonal (and respectively) as before. The minimum distance is defined between R and S by MINDIST(R,S). Every point in R is at a distance of at least MINDIST(R,S) from any point in S (and vice versa). Similarly, the maximum distance between R and S, denoted by MAXDIST(R,S) is defined. The distances can be calculated using the following two formulas:

MINDIST $(R,S) = \sum_{i=1}^{\delta} x_i^2$, where

$$x_i = \begin{cases} r_i - s'_i & \text{if } s'_i < r_i \\ s_i - r'_i & \text{if } r'_i < s_i \\ 0 & \text{otherwise} \end{cases}$$

$$\text{MAXDIST}\ (R,S) = \sum_{i=1}^{\delta} x_i^2, \text{ where } x_i = \max\left\{\left|s_i' - r_i\right|, \left|r_i' - s_i\right|\right\}$$

2.2 *Angle-Based Outlier Detection*

This algorithm is proposed to not only use distance but primarily the directions of distance vectors. Comparing the angles between pairs of distance vectors to other points helps to discern similar data points between outliers.

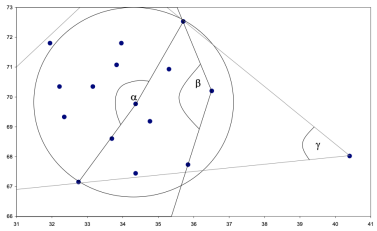


Figure 2.2: intuition of ABOD.

This idea is motivated by the following intuition. Consider a simple data set as illustrated in Figure 2.2. For a point within a cluster, the angles between the difference vectors of pairs of other points differ widely. The variance of the angles will become smaller for points at the border of a cluster. [7]However, even here the variance is still relatively high compared to the variance of angles for real outliers. Here, the angles of most pairs of points will be small since most points are clustered in some direction.

[7] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452, 2008

As a result of these considerations, an angle-based outlier factor (ABOF) can describe the divergence in the directions of objects relatively to one another. If the spectrum of observed angles for a point is broad, the point will be surrounded by other points in all possible directions, meaning the point is positioned inside a cluster. If the spectrum of observed angles for a point is rather small, other points will be positioned only in certain directions. This means the point is positioned outside of some set of points that are grouped together. Thus, rather small angles for a point $\overrightarrow{P}$ that are rather similar to one another imply that $\overrightarrow{P}$ is an outlier.

As an approach to assigning the ABOF value of any object in the database D, a calculation of the scalar product of the difference vectors of any triple of points (i.e., a query point $\overrightarrow{A} \in D$ and all pairs$(\overrightarrow{B}, \overrightarrow{C})$ of all remaining points in $D\{\overrightarrow{A}\}$) normalized by the quadratic product of the length of the difference vectors is computed, i.e. the angle is weighted less if the corresponding points are far from the query point. By this weighting factor, the distance influences the value after all, but only to a minor part. Nevertheless, this weighting of the variance is important since the angle of a pair of points varies naturally stronger for a bigger distance. The variance of this value over all pairs for the query point $\overrightarrow{A}$ constitutes the angle-based outlier factor of $\overrightarrow{A}$. Formally:

Definition 5 *Given a database $D \subseteq R^d$, a point $\overrightarrow{A} \in D$, and a norm*

$\cdot : R^d \longrightarrow R_o^+$. *The scalar product is denoted by*

$$
\begin{aligned}
ABOF(\vec{A}) =& VAR_{\vec{B},\vec{C}\in\mathcal{D}} \left(\frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot \|\overline{AC}\|^2} \right) \\
=& \frac{\sum_{\vec{B}\in\mathcal{D}} \sum_{\vec{C}\in\mathcal{D}} \left(\frac{1}{\|\overline{AB}\| \cdot \|\overline{AC}\|} \cdot \frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot \|\overline{AC}\|^2} \right)^2}{\sum_{\vec{B}\in\mathcal{D}} \sum_{\vec{C}\in\mathcal{D}} \frac{1}{\|\overline{AB}\| \cdot \|\overline{AC}\|}} \\
& - \left(\frac{\sum_{\vec{B}\in\mathcal{D}} \sum_{\vec{C}\in\mathcal{D}} \frac{1}{\|\overline{AB}\| \cdot \|\overline{AC}\|} \cdot \frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot \|\overline{AC}\|^2}}{\sum_{\vec{B}\in\mathcal{D}} \sum_{\vec{C}\in\mathcal{D}} \frac{1}{\|\overline{AB}\| \cdot \|\overline{AC}\|}} \right)^2
\end{aligned}
\tag{2.12}
$$

The algorithm ABOD assigns the angle-based outlier factor ABOF to each point in the database and returns as a result the list of points sorted according to their ABOF. Consider again the sample data set in Figure 2.2. The ranking of these points as provided by ABOD is denoted in Figure2.3 . In this toy example, the top-ranked point (rank 1) is clearly the utmost outlier. The next ranks are taken by border points of the cluster. The lowest ranks are assigned to the inner points of the cluster.

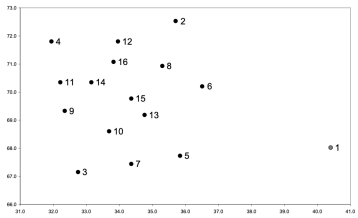


Figure 2.3: Ranking assigned by ABOD.

Since the distance is accounted for only as a weight for the main criterion, the variance of angles, ABOD is able to concisely detect outliers even in high-dimensional data where LOF and other purely distance-based approaches deteriorate in accuracy.

2.3 *Isolation Forest*

The Isolation forest method takes advantage of to anomalies' quantitative properties: I) they are the minority consisting of fewer instances and II) they have attribute values that are very different from those of normal instances. In other words, anomalies are 'few and different', which make them more susceptible to isolation than normal points. A tree structure can be constructed effectively to isolate every single instance. Because of their susceptibility to isolation, anomalies are isolated closer to the root of the tree; whereas normal points are isolated at the deeper end of the tree. This isolation characteristic of trees forms the basis of our method to detect anomalies, and this is called the 'Tree Isolation' or 'iTree'. [8]

The proposed method, called Isolation Forest or iForest, builds an ensemble of iTrees for a given data set, then anomalies are those instances which have short average path lengths on the iTrees. There are only two variables in this method: the number of trees to build and

[8] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 413–422, USA, 2008. IEEE Computer Society. ISBN 9780769535029. DOI: 10.1109/ICDM.2008.17. URL https://doi.org/10.1109/ICDM.2008.17

the sub-sampling size. The iForest's detection performance converges quickly with a very small number of trees, and it only requires a small sub-sampling size to achieve high detection performance with high efficiency. Apart from the key difference of isolation versus profiling, iForest is distinguished from existing model-based, distance-based, and density-based methods in the following ways:

- The isolation characteristic of iTrees, which enables them to build partial models and exploit subsampling to an extent that is not feasible in existing methods. Since a large part of an iTree that isolates normal points is not needed for anomaly detection; it does not need to be constructed. A small sample size produces better iTrees because the swamping and masking effects are reduced.
- iForest utilizes no distance or density measures to detect anomalies. This eliminates the major computational cost of distance calculation in all distance-based methods and density-based methods.
- iForest has a linear time complexity with a low constant, and low memory requirement. To our best knowledge, the best-performing existing method achieves only approximate linear time complexity with high memory usage.
- iForest has the capacity to scale up to handle extremely large data sizes and high-dimensional problems with a large number of irrelevant attributes.

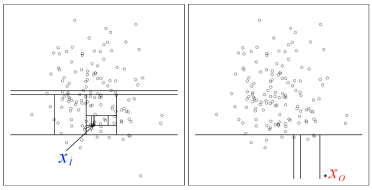


Figure 2.4: Isolation Visual Process.

The term isolation means 'separating an instance from the rest of the instances'. Since anomalies are 'few and different', therefore they are more susceptible to isolation. In a data-induced random tree, the partitioning of instances are repeated recursively until all instances are isolated. To demonstrate the idea that anomalies are more susceptible to isolation, we observe in Figure 2.4 that a normal point, x_i, generally requires more partitions to be isolated. The opposite is also true for the anomaly point, x_o, which generally requires fewer partitions to be isolated.

Since each partition is randomly generated, individual trees are generated with different sets of partitions. The path lengths are averaged over a number of trees to find the expected path length.

Definition 6 *Let T be a node of an isolation tree. T is either an external node with no child, or an internal node with one test and exactly two daughter nodes (T_l, T_r). A test consists of an attribute q and a split value p such that the test $q < p$ divides data points into T_l and T_r.*

Given a sample of data $X = \{x_1, \ldots, x_n\}$ of n instances from a $d-$variate distribution, to build an isolation tree, a recursively division

over X by randomly selecting an attribute q and a split value p is applied, until either:

- the tree reaches a height limit,
- $|X| = 1$ or
- all data in X have the same value.

An iTree is a proper binary tree, where each node in the tree has exactly zero or two daughter nodes. Assuming all instances are distinct, each instance is isolated to an external node when an iTree is fully grown, in which case the number of external nodes is n and the number of internal nodes is $n - 1$; the total number of nodes of an iTrees is $2n - 1$; and thus the memory requirement is bounded and only grows linearly with n.

The task of anomaly detection is to provide a ranking that reflects the degree of anomaly. An anomaly score is required for any anomaly detection method. The difficulty in deriving such a score from $h(x)$ is that while the maximum possible height of iTree grows in the order of n, the average height grows in the order of $\log n$. [9]

[9] D.E. Knuth. *The Art of Computer Programming: Volume 3: Sorting and Searching*. Pearson Education, 1998. ISBN 9780321635785

Since iTrees have an equivalent structure to Binary Search Trees or BST, the same analysis is used to estimate the average path length of the iTree. Given a data set of n instances, the average path length of an unsuccessful search in BST as:

$$c(n) = 2H(n-1) - (2(n-1)/n) \tag{2.13}$$

where $H(i)$ is the harmonic number and it can be estimated by $\ln(i) + e$. As $c(n)$ is the average of $h(x)$ given n, it's then used to normalise $h(x)$. The anomaly score s of an instance x is defined as:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{2.14}$$

where $E(h(x))$ is the average of $h(x)$ from a collection of isolation trees. In the previous equation (2.14):

- when $E(h(x)) \to c(n), s \to 0.5$
- when $E(h(x)) \to 0, s \to 1$
- and when $E(h(x)) \to n - 1, s \to 0$

Using the anomaly score s, the following assessment is concluded:

- if instances return s very close to 1, then they are definitely anomalies.
- if instances have s much smaller than 0.5, then they are quite safe to be regarded as normal instances.

- if all instances return $s \approx 0.5$, then the entire sample does not really have any distinct anomaly.

2.4 *Supervised Method - Support Vector Machine (SVM)*

The SVM view starts by designing a particular function that is to be optimized during training, based on geometric intuition; starting by designing a loss function that is to be minimized on the training data.

Intuitively, and imagining a binary classification data set, which can be separated by a hyperplane where in Figure2.5 every example x_n (a vector of dimension 2) is a two-dimensional location $(x^(1)_n$ and $x^(2)_n)$, and the corresponding binary label y_n is one of two different symbols.

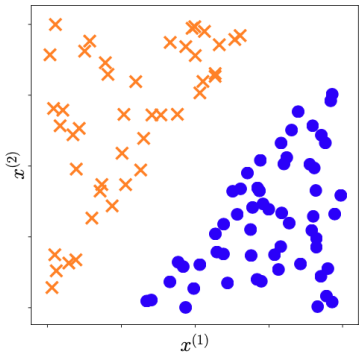


Figure 2.5: Representation of a linear separation of two categories, orange and blue dots.

Given two examples represented as vectors x_i and x_j , one way to compute the similarity between them is using an inner product (x_i, x_j), so that the inner products are closely related to the angle between two vectors. The value of the inner product between two vectors depends on the length of each vector. Furthermore, the inner product allows to rigorously define the orthogonality and projections which are extensively used in this process. A linear partition is first considered, splitting the space into two halves using a hyperplane. Let an example $x \in R^D$ be an element of the data space. Consider a function [10]

$$\begin{aligned} f &: \mathbb{R}^D \to \mathbb{R} \\ \boldsymbol{x} &\mapsto f(\boldsymbol{x}) := \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b \end{aligned} \tag{2.15}$$

[10] M.P. Deisenroth, A.A. Faisal, and C.S. Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020. ISBN 9781108470049

The hyperplane that separates the two classes in our binary classification problem is defined as:

$$\left\{ \boldsymbol{x} \in \mathbb{R}^D : f(\boldsymbol{x}) = 0 \right\} \tag{2.16}$$

The vector w is then specified as a vector normal to the hyperplane and b the intercept. It can be derived that w is a normal vector to the hyperplane by choosing any two examples x_a and x_b on the hyperplane and showing that the vector between them is orthogonal to w as both points are within the hyperplane creating a 90° angle to the w vector.

$$\begin{aligned} f\left(\boldsymbol{x}_a\right) - f\left(\boldsymbol{x}_b\right) &= \langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b - \left(\langle \boldsymbol{w}, \boldsymbol{x}_b \rangle + b \right) \\ &= \langle \boldsymbol{w}, \boldsymbol{x}_a - \boldsymbol{x}_b \rangle \end{aligned} \tag{2.17}$$

Since x_a and x_b are chosen to be on the hyperplane, this implies that $f(x_a) = 0$ and $f(x_b) = 0$ and hence $(w, x_a - x_b) = 0$. Knowing that two vectors are orthogonal when their inner product is zero. Therefore, it's deducted that w is orthogonal to any vector on the hyperplane.

When presented with a test example, the example's classified as positive or negative depending on the side of the hyperplane on which it occurs. Therefore, to classify a test example x_test, the value of the function $f(x_test)$ is calculated and classify the example as $+1$ if $f(x_test) \geq 0$ and -1 otherwise. Thinking geometrically, the positive examples lie "above" the hyperplane and the negative examples "below" the hyperplane.

When training the classifier, it has to be ensured that the examples with positive labels are on the positive side of the hyperplane,

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \geq 0 \quad \text{when} \quad y_n = +1 \tag{2.18}$$

and the negative labels are on the other side represented by:

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b < 0 \quad \text{when} \quad y_n = -1 \tag{2.19}$$

Both conditions can be simplified to a single equation,

$$y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geq 0 \tag{2.20}$$

For a dataset $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ that is linearly separable, there are infinitely many candidate hyperplanes, that solve our classification problem without any (training) errors. To find a unique solution, one idea is to choose the separating hyperplane that maximizes the margin between the positive and negative examples. Figure2.6

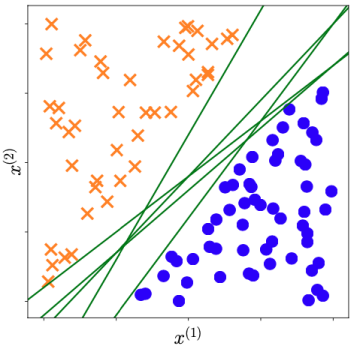


Figure 2.6: Possible scenarios for hyperplane separation.

The margin is represented by the distance of the separating hyperplane to the closest example in the dataset, assuming that the dataset is linearly separable.

Considering a hyperplane $\langle w, x \rangle + b$, and an example x_a as shown in Figure 2.7 . Without loss of generality, the example x_a is considered to be on the positive side of the hyperplane, $\langle w, x_a \rangle + b > 0$. The distance $r > 0$ of x_a is then computed from the hyperplane by considering the orthogonal projection of x_a onto the hyperplane, which is denoted by x'_a. Since w is orthogonal to the hyperplane, the distance r is just a scaling of this vector w. If the length of w is known, the scaling factor r can be used to work out the absolute distance between x'_a and x_a. For a vector of unit length (norm = 1) which is obtained by dividing w by its norm, $\frac{w}{\|w\|}$. Using vector addition, the next equation is calculated

$$\boldsymbol{x}_a = \boldsymbol{x}'_a + r \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \tag{2.21}$$

The positive example is preferred to be further than r from the hyperplane, and the negative examples to be further than distance r (in

the negative direction) from the hyperplane. Analogously, the objective formulation is,

$$y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geq r \tag{2.22}$$

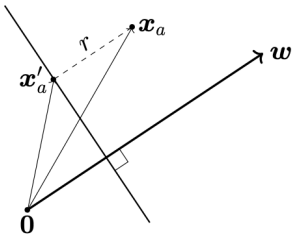


Figure 2.7: Primal Support Vector Machine

Adding the assumption that the parameter vector w is of unit length to our model, $\|\boldsymbol{w}\| = 1$, where the Euclidean norm is used $\|\boldsymbol{w}\| = \sqrt{w^\top w}$, allows a more intuitive interpretation of the distance r since it is the scaling factor of a vector of length 1.

The objective is represented by,

$$\begin{aligned} &\max_{\boldsymbol{w},b,r} \underbrace{r}_{\text{margin}} \\ &\text{subject to } \underbrace{y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geq r}_{\text{data fitting}}, \ \underbrace{\|\boldsymbol{w}\| = 1}_{\text{normalization}} \ , \quad r > 0, \end{aligned} \tag{2.23}$$

which says that the margin r maximization is prioritized while ensuring that the data lies on the correct side of the hyperplane.

Derivating (2.23), and by observing that the only interest lies in the direction of w and not its length, leads to the assumption that $\|\boldsymbol{w}\| = 1$.

This scale is chosen ensuring that the value of the predictor $\langle w, x \rangle + b$ is 1 for the closest example. It also denotes the example in the dataset that is closest to the hyperplane by x_a. Since x'_a is the orthogonal projection of x_a onto the hyperplane, it must by definition lie on the hyperplane,

$$\left\langle \boldsymbol{w}, \boldsymbol{x}'_a \right\rangle + b = 0 \tag{2.24}$$

By substituting (2.21) into (2.24), it's obtained

$$\left\langle \boldsymbol{w}, \boldsymbol{x}_a - r \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \right\rangle + b = 0 \tag{2.25}$$

Exploiting the bi-linearity of the inner product,

$$\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b - r \frac{\langle \boldsymbol{w}, \boldsymbol{w} \rangle}{\|\boldsymbol{w}\|} = 0 \tag{2.26}$$

Observe that the first term is 1, by our assumption of scale, $\langle w, x_a \rangle + b = 1$ knowing that $\langle \boldsymbol{w}, \boldsymbol{w} \rangle = \|\boldsymbol{w}\|^2$. Hence, the second term reduces to $r\|\boldsymbol{w}\|$. Finally obtaining,

$$r = \frac{1}{\|\boldsymbol{w}\|} \tag{2.27}$$

Combining the margin maximization with the fact that examples need to be on the correct side of the hyperplane (based on their labels)

gives us

$$\begin{aligned} &\max_{w,b} \frac{1}{\|\boldsymbol{w}\|} \\ &\text{subject to } y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1 \quad \text{for all} \quad n = 1, \ldots, N \end{aligned} \tag{2.28}$$

The squared norm is often minimized, and it also includes a constant $\frac{1}{2}$ that does not affect the optimal w, but yields a tidier form when the gradient's computed. Then, the objective becomes

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{subject to } y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1 \quad \text{for all} \quad n = 1, \ldots, N \end{aligned} \tag{2.29}$$

This last equation is known as the hard margin SVM. The reason for the expression "hard" is because the formulation does not allow for any violation of the margin condition.

2.4.1 *Linear Kernel*

In the binary classification setting, let $((x_1, y_1)...(x_n, y_n))$ be the training dataset where x_i are the feature vectors representing the instances (i.e. observations) and $y_i \in -1, +1$ be the labels of the instances. Support vector learning is the problem of finding a separating hyperplane that separates the positive examples (labeled $+1$) from the negative examples (labeled -1) with the largest margin.

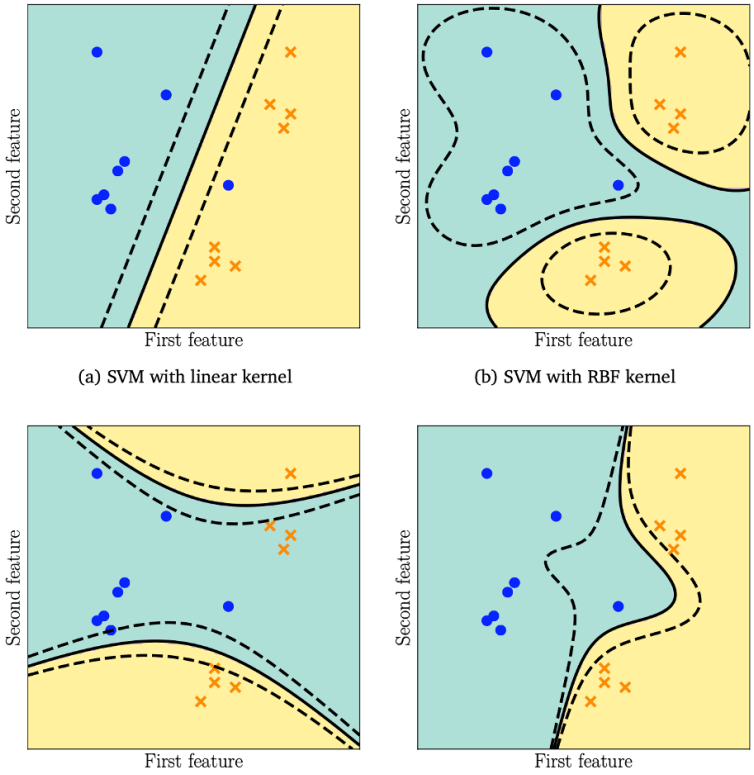
The margin of the hyperplane is defined as the shortest distance between the positive and negative instances that are closest to the hyperplane. The intuition behind searching for the hyperplane with a large margin is that a hyperplane with the largest margin should be more resistant to noise than a hyperplane with a smaller margin. [11]

[11] Vasileios Apostolidis-Afentoulis. Svm classification with linear and rbf kernels. 07 2015. DOI: 10.13140/RG.2.1.3351.4083

Figure 2.8 represents the decision function and boundaries for each kernel. In this work the linear kernel is discussed, as it represents the method used in our practical case.

2.5 *Performance Metrics*

For the evaluation of the different methods used in this work, the main focus will be on three standard performance indicators, precision, recall, and F-Score.



(a) SVM with linear kernel

(b) SVM with RBF kernel

(c) SVM with polynomial (degree 2) kernel

(d) SVM with polynomial (degree 3) kernel

Figure 2.8: SVM and the decision boundary drawn by the different kernels.

For illustration purposes, the following simple setting is considered: each observation is associated with a binary label which accounts for the correctness of the object. In addition, the model produces a result or a prediction indicating whether it believes the object to be correct or not. The experimental outcome is conveniently summarised in a confusion table that classifies the results in four different groups.

		Prediction	
		+	-
Label	+	TP	FN
	-	FP	TN

Figure 2.9: Confusion Matrix

In Figure 2.9 the $+$ and $-$ symbols represent the correct and incorrect classification of the predictions, where TP stands for True Positive, FN False Negative, FP False Positive and TN True Negative.

From the previous definitions, one can compute the precision p and recall r:

$$p = \frac{TP}{TP + FP} \quad r = \frac{TP}{TP + FN} \tag{2.30}$$

Taking the (weighted) harmonic average of precision and recall leads to the F-Score:

$$F_\beta = \left(1 + \beta^2\right) \frac{pr}{r + \beta^2 p} = \frac{\left(1 + \beta^2\right) TP}{\left(1 + \beta^2\right) TP + \beta^2 FN + FP}. \tag{2.31}$$

Both precision and recall have a natural interpretation in terms of probability. Indeed, precision may be defined as the probability that an object is relevant given that it is returned by the system, while the recall is the probability that a relevant object is returned:

$$p = P(\text{label} = + \mid \text{pred} = +) \quad r = P(\text{pred} = + \mid \text{label} = +) \tag{2.32}$$

This may seem like a trivial reformulation. However, there is a big semantic difference: in the original formulation, p and r are just formulas calculated from the observed data; in the probabilistic framework, the data $D = (TP, FP, FN, TN)$ actually arises from p and r, which are parameters of a (primitive) generative model. Thus, the usual expressions (2.30) arise only as estimates of these unknown parameters.

3 Feature Engineering

Contents

3.1 Feature Selection

The feature selection process was based on different statistical methods, that allowed to understand which variables are more relevant compared to the response feature.

The first subset of data consists of a list of filters applied to the original data set of 200 columns, obtaining 51 main characteristics. From these fields, the dataset is pre-filtered by three key categories, service type, rate type, and category type. As a result, a cluster of hardware information is obtained that gathers all prices of various asset models from where the standard cost is pulled as a basis for classifying outliers. The subsequent data sets get divided into groups of models and service subcategories from each type of hardware.

The first process started by comparing the business expertise on selecting the features used by the SMEs (Subject Matter Experts) to detect the anomalies for this particular phenomenon. Out of 9 variables of the original data set were extracted from the database for this application's purpose. Each variable is described in the following list:

- Model: Represents the type of machine/hardware that is being used for providing network services.
- Charges: Total cost in dollars, that is paid in a specific recurrent period of time.
- Country.

- Geography: The company divides the different regions into five different geographies, they represent a higher hierarchy of country division.
- Service Type: The type of service that is being required by the client is within the network scope.
- Account: Client name or unique code that identifies a client in the system.
- Price upper boundary: Highest limit of price for any specific asset.
- Price lower boundary: Lowest limit of price for an specific asset.
- Item Reference: Short description of an asset in terms of characteristics, machine type, service, composition, etc.

Our variables are divided into two wider groups, 3 continuous variables which are closely related to the price and its behaviours, and 6 discrete features that categorize each of the observations in distinct classes depending its value.

Finally, the discrete response feature, which is not available in the original dataset is then defined by the unsupervised approach, later explained in the next chapter 4. It is essential to mention that the response feature consists of a binary classification of an outlier versus an inlier. An outlier behavior is mostly driven by the cost assigned to its observation; when an average normal range is defined, the outliers are caught as they distance from the average limit range.

3.1.1 *Statistical Measures*

The Pearson Correlation was initially used as a metric to define which variables showed a greater relationship towards the price (charge) variable. In addition, any type of collinearity between them needs to be captured, so those features would be removed from the input variables.

The Pearson correlation function is applied in the original data set containing the nine features to see the percentage of relationship between them. Five Variables showed a strong collinearity, meaning that those can be linearly predicted from the others with a substantial degree of accuracy. Therefore, those are not included as input variables into the model.

The first group showing this phenomenon was the charge feature versus the lower and upper ranges, each having more than 85% of

the relationship. (88% - Lower range , 92% - Upper range). The only variable considered in the model was the charge one, representing a better standard cost behavior.

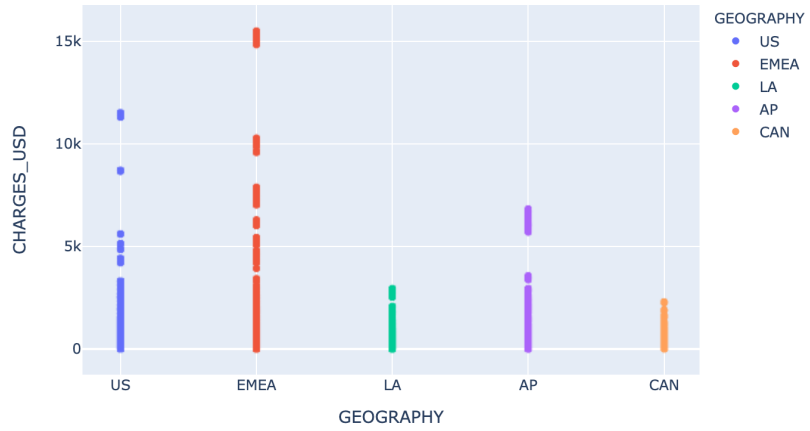


Figure 3.1: Geography categories

The next group is different due to their nature, geography, country, and account are each discrete variables, a business statement is applied for the selection process, where it states that most of the prices variate depending on its geography and not necessarily on its country. Neither accounts, as the client does not determine if they are more prone to receive outliers, those factors are mainly given to the type of service, items and geography where the client is located. Figure 3.1

	Charges	Model	Geography	Service Type
Charges	1.00	0.30	0.01	−0.43
Model	0.30	1.00	−0.03	−0.78
Geography	0.01	−0.03	1.00	0.07
Service Type	−0.43	−0.78	0.07	1.00

Table 3.1: Pearson Correlation

The subsequent data sets get divided into groups of models and service subcategories, as well as geographies from each type of hardware. The input for the algorithm becomes the rate of the model group, including the categorical features previously mentioned, which in this phase would be transformed into numeric variables depending on each group.

3.2 *Variable Transformation*

The only transformation applied to the subset due to the fact that most of them are discrete features, which is a *dummy treatment* which converts

a categorical variable into a dummy/indicator feature by supplying a binary category $(0, 1)$ depending on its frequency of appearance.

A dummy variable is often used to distinguish different treatment groups. For this particular implementation, it's applied as a binary feature $(0, 1)$ where an invoice for the geography category is given a value of 1 if they are in the "US" group or a 0 if they belong to any other geography. The Dummy treatment is very useful as it enables to use a single regression equation to represent multiple groups. Doing so, the separation of equation models for each subgroup is not needed anymore, as the dummy variables act like 'switches' that turn various parameters on and off in the model equation. Another advantage of a binary-coded variable is that even though it is a nominal-level variable it can be treated statistically as an interval-level variable.

Dummy variables are incorporated in the same way as quantitative variables are included in the regression model. For example, considering on predicting the price of an invoice based on our discrete variables:

- Geography
- Service Type
- Model

The model with the original features would be:

$$\text{Price} = \beta_0 + \beta_{Geography} + \beta_{ServiceType} + \beta_{Model} \tag{3.1}$$

Applying the treatment to the discrete variables, the model is:

$$\text{Price} = \beta_0 + \beta_{US} + \beta_{Europe} + \beta_{LatinAmerica} + \beta_{Canada} + \beta_{Asia}\beta_{Model_1} + \ldots + \beta_{Model_n} \tag{3.2}$$

In the model, $US = 1$ when an invoice belongs to that region and $US = 0$ when the the invoice has a different location from US. β_{US} can be interpreted as the difference in price between the distinct regions, holding all other variables constant.

Let us remember that our objective of using the unsupervised approach, later described in the next chapter, is to generate our response variable, which determines when an invoice is considered an anomaly versus those that are not. As stated before, most of this behavior is driven by the price in cooperation with the three discrete factors. The dummy transformation has modified the model so far in increasing the number of input features ingested to the model. However, it increases the discrete features' explanation, understanding which category from each class is the most significant based on the data. The first approach

contained four different variables, while the transformed approach contains 57 features. Despite the increase of variables, this did not show a significant impact of performance in the model.

4 Modeling

Contents

The model selection requires a broader scope of outlier algorithms to be considered. This group was separated into four higher groups that were explained previously in chapter 2

From the wide variety of models being provided by each category, one of them was finally selected after extensive testing by comparing the contamination rate that simulates the same percentage provided by the business experts. It was also considered to compare a small subset of data labeled by the SMEs in which four different performance metrics known as accuracy, recall, precision, and F1 metrics where considered.

In the next subsections, the process of modeling is defined for the unsupervised approach that was first selected from the previous unsupervised categories mentioned above. The labeling method and training of the second supervised approach will be then specified.

4.1 Mathematical Description

Traditionally, as for any machine learning implementation. There's always a tendency to train a model based on historical information to calculate the best parameters, using a set of labels provided by the experts. This process is mostly utilized for supervised methods, which require the dependent variable predefined, so the model understands and learns from it.

When applied to real data, some cases do not contain enough data labeled or even an endogenous feature for the model to train. In such cases, the first method to use is an unsupervised algorithm that allows the user to label the data based on its behavior, also depending on the prediction problem.

In this work, An unsupervised method is first used allowing to reproduce enough labels as the prediction problem relies on outlier detection, which can also be determined as a binary classification with a particular and well-studied phenomenon. Once the unsupervised method is applied and reveals the outcome, the binary labels representing $y_0 = inlier$ and $y_1 = outlier$; are then reviewed by a group of business experts who provide their input. This feedback in hand with the algorithm's labeling process allows us to recreate enough labeled information to train a supervised method.

4.1.1 *Unsupervised Method: Minimum Covariance Determinant (MCD)*

The first assumption is that the data's stored in an $n \times p$ data matrix $X = (x_1, \ldots, x_n)^t$ with $x_i = (x_{i1}, \ldots, x_{ip})^t$ the ith observation. Hence n stands for the number of observations and p for the number of variables.

To illustrate, the first step is defining the dataset of 113 features (once transformed into dummy variables)[3]. hence $p = 113$. In figure 4.1 two of the variables from the original dataset (invoice cost and geography) are illustrated before the transformation process, together with the classic and the robust 97.5% tolerance ellipse.

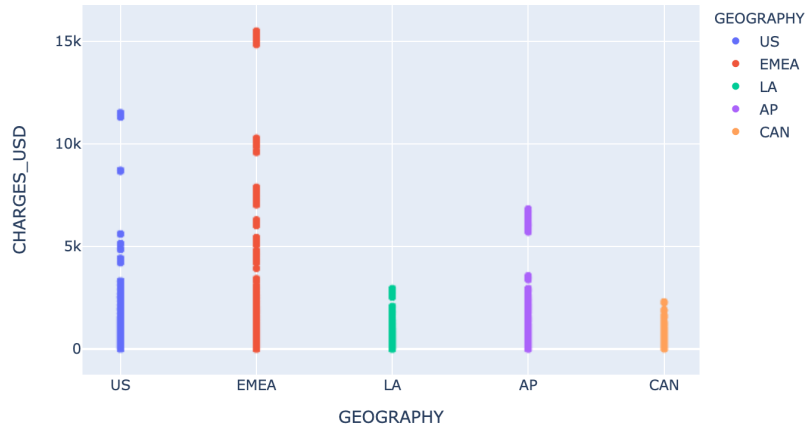


Figure 4.1: Invoice cost and Geography variables with classical and robust tolerance ellipse

The classical tolerance ellipse is defined as the set of p-dimensional

points x whose Mahalanobis distance is defined as follows:

$$\mathrm{MD}(x) = \sqrt{(x - \bar{x})^t S^{-1} (x - \bar{x})} \tag{4.1}$$

The Mahalanobis distance MD(xi) should tell us how far away xi is from the center of the cloud, relative to the size of the data points. On the other hand, the robust tolerance ellipse in Figure 1 which is based on the robust distance

$$\mathrm{RD}(x) = \sqrt{(x - \hat{\mu}_{MCD})^t \hat{\Sigma}_{MCD}^{-1} (x - \hat{\mu}_{MCD})} \tag{4.2}$$

that is much smaller and encloses the regular data points. Here, $\hat{\boldsymbol{\mu}}_{MCD}$ is the MCD estimate, and $\hat{\boldsymbol{\Sigma}}_{MCD}$ the MCD covariance estimate. The robust distances exposed in Figure 4.2 (b) now clearly spot 3 outliers while chart (a) spots only 1 outlier.

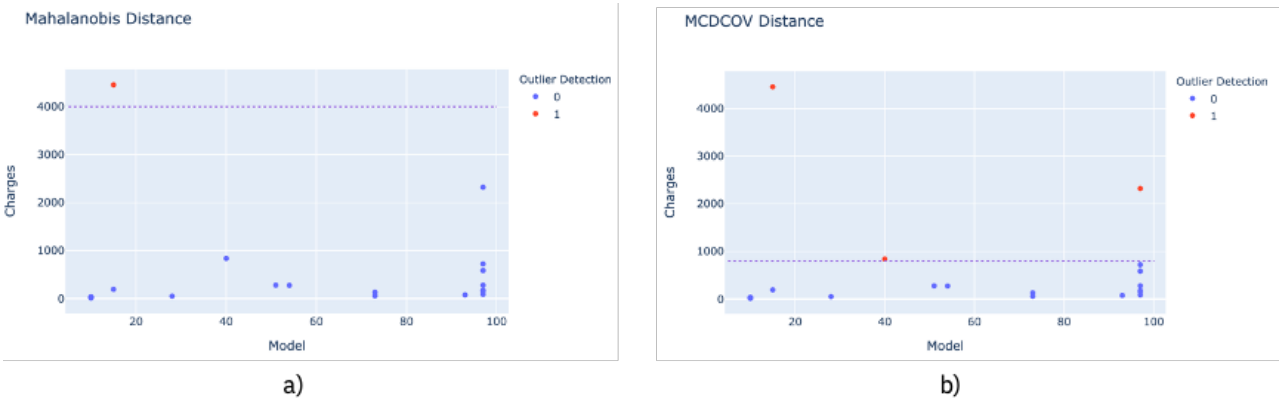


Figure 4.2: (a) Mahalanobis distance VS (b) MCD robust distance applied to same sample dataset.

This illustrates the masking effect: classical estimates can be positively affected by outlying values that diagnostic tools such as the Mahalanobis distance can no longer detect. Robust estimators are required to get a more reliable detection of such potential outliers. The MCD estimator is considered one of those.

4.1.2 *Supervised Method: Support Vector Machine (SVM)*

As mentioned previously, the minimum covariance determinant method allows us to identify the observations that are considered as anomalies, a.k.a outliers. One step of the process that does not involve a mathematical approach is the expert's feedback, where the data points reviewed by them manually are then retrieved. The data is prepared and displayed to the business into the different categories adding the labels predicted by the MCD model.

Once the feedback and selection process are completed, the next phase is the Support Vector Classifier's training.

The same initial four variables described before in chapter 3 are considered as a starting point. Once transformed, the features are ingested in the Support Vector classifier using a linear kernel. In figure 4.2 the separation between the two categories is shown, outliers and inliers depend on a linear separation mainly due to the cost assigned to the invoice in contrast to the discrete variables. This behavior allows to specify a linear kernel as the best method for classification within the SVM.

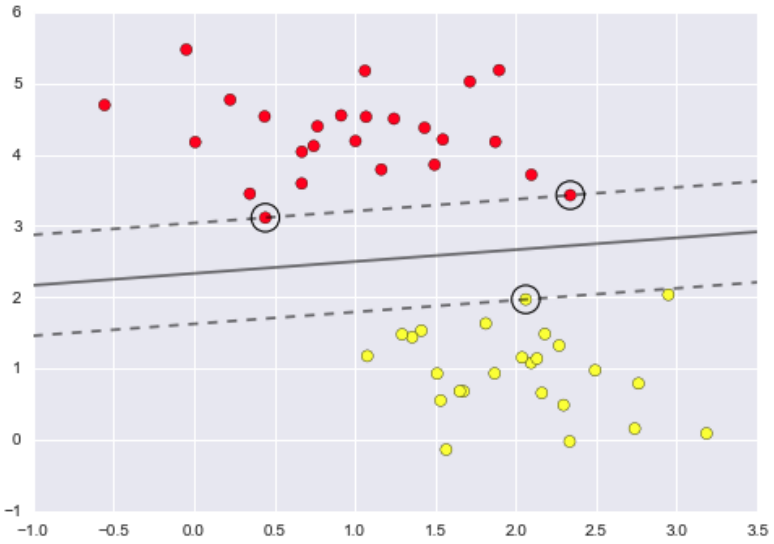


Figure 4.3: Support Vectors from Linear Kernel applied to sample data

The next process leads to the testing phase using 20% of the previously separated data through the cross-validation process (80 - 20), where the model is evaluated and analyzes the support vectors calculated as the boundaries to separate both classes 4.3. The algorithm is saved once it reaches the same or higher precision and performance than the MCD algorithm.

Finally the last prediction is performed as the model gets deployed into a production environment for business use.

4.2 *Process Diagram*

In the next diagram 4.4 each of the steps previously explained in the mathematical description are summarised, including the data engineering process of collecting and ingesting the data and deployment, until production.

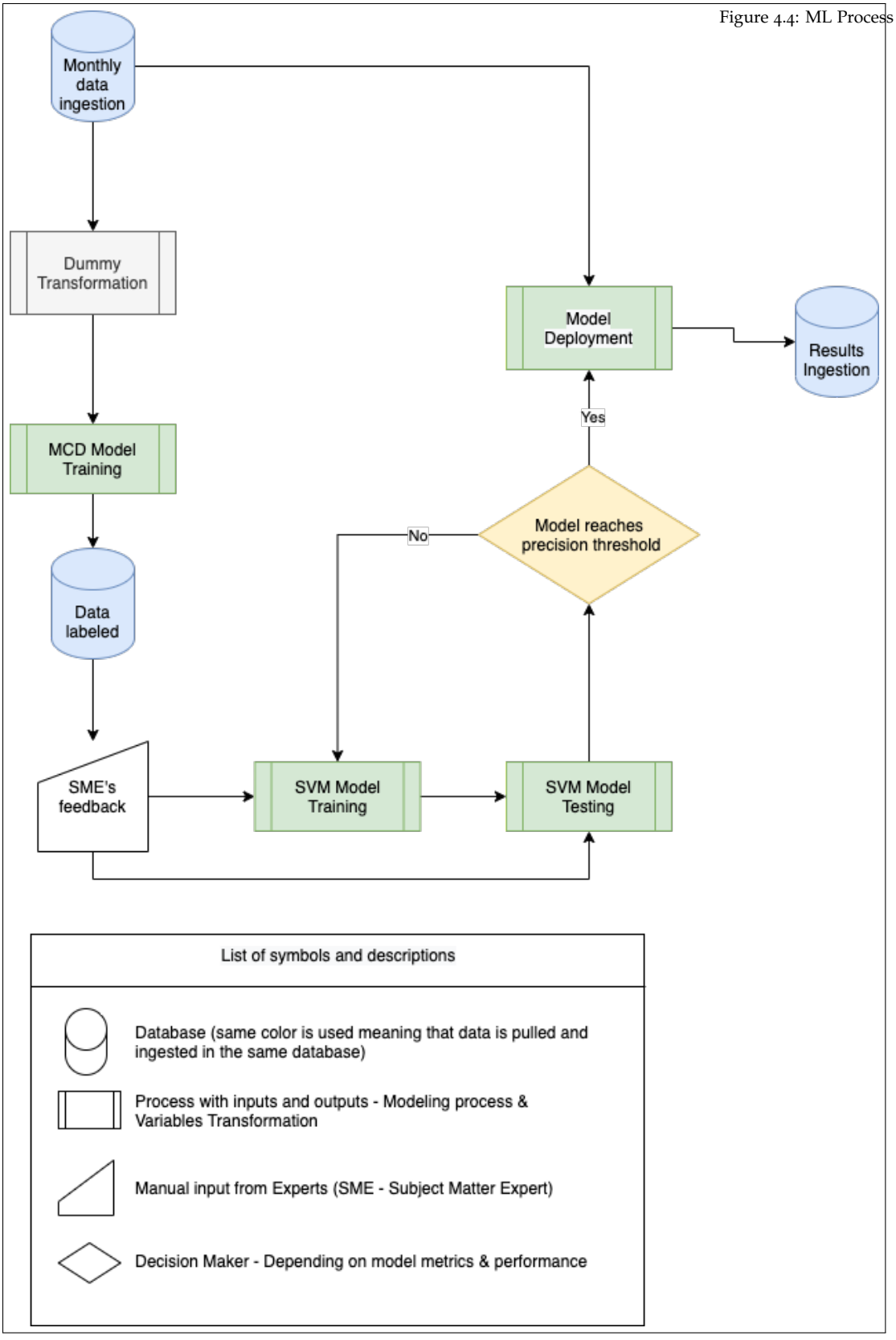


Figure 4.4: ML Process Workflow

5 Results

As it was stated in the modeling chapter 4 , the MCD model had the best results in terms of accuracy as well as adjusting to the contamination rate provided by the experts, once compared to the other models. This results are shown as a reference in the tables below, where the MCD metrics are compared to the results of the second best approach, Isolation Forest, which precision drops significantly.

	Precision	Recall	F1-Score	Support
Inlier	0.95	1	0.97	7581
Outlier	0.90	0.49	0.63	369

Table 5.1: MCD Metrics

	Precision	Recall	F1-Score	Support
Inlier	0.94	0.98	0.96	7266
Outlier	0.67	0.37	0.48	684

Table 5.2: Isolation Forest Metrics

From the hypothesis previously stated in the work, it's concluded that the model and data process were able to show a good relationship in the variables versus the outlier's behavior getting as a result, good precision and performance of the process.

In addition, regarding the second hypothesis, which relates to having a supervised method that could use as input the labels from the unsupervised approach. It's shown that this fact was proven once the experts reviewed the labels; these became the input characteristics to our support vector machine increasing the percentage of accuracy from the previous MCD model, as shown in the next table.

	Precision	Recall	F1-Score	Support
Inlier	1.0	1.0	1.0	7568
Outlier	0.94	0.97	0.96	382

Table 5.3: Support Vector Machine

6 Conclusions and Future Work

Contents

6.1 Conclusions

Since the historical data provided from the business did not contain labels identifying the inliers VS outliers, it was decided to select an unsupervised model that was able to simulate the expertise of an SME, replicating the selection process. The MCD model has proven to replicate this selection process with the highest accuracy from the wide variety of unsupervised approach that were described in chapter 2.

Even though a small subset of labels was received initially to test the unsupervised methods, this subset did not represent our universe as it was only representing a unique group of models due to the sampling selection. However, after implementing the MCD method and the expert's feedback on the model's output, it was decided to train a supervised model representing the universe of the different categories contained in the discrete features and the continuous variable.

Some of the outcomes to consider while applying a machine learning method to an unlabeled approach are unsupervised algorithms that help provide a massive tagging with less manual effort due to its implementation process. Unsupervised methods are pretty good at finding patterns within the data but are not as consistent as supervised approaches. One suggestion that came out of this work is that data scientists can leverage value using these approaches, mainly on the massive tagging process, which is one of the most common problems encountered in real data.

6.2 *Future Work*

- Evaluate the MCD model, dividing the dataset on categories and run parallel algorithms for each specific model category and cost.
- Compare results versus the iterative parallel approach and the current transformation method applied to the Minimum Covariance Determinant.
- Augment the descriptive analysis to apply feature engineering that allows increasing the predictive capability of the model.
- Create process documentation that allows the scalability of the method and ease solving problems such as the amount of data, categories, and the odds of having labeled data.

Bibliography

Vasileios Apostolidis-Afentoulis. Svm classification with linear and rbf kernels. 07 2015. DOI: 10.13140/RG.2.1.3351.4083.

M.P. Deisenroth, A.A. Faisal, and C.S. Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020. ISBN 9781108470049.

Johanna Hardin and David Rocke. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Computational Statistics & Data Analysis*, 44:625–638, 01 2004. DOI: 10.1016/S0167-9473(02)00280-3.

Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. Technical report, Harbin Institute of Technology, Harbin 150001, P.R. China Department Of Computer Science and Engineering, 2003.

D.E. Knuth. *The Art of Computer Programming: Volume 3: Sorting and Searching*. Pearson Education, 1998. ISBN 9780321635785.

Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452, 2008.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 413–422, USA, 2008. IEEE Computer Society. ISBN 9780769535029. DOI: 10.1109/ICDM.2008.17. URL https://doi.org/10.1109/ICDM.2008.17.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. volume 29, pages 427–438, 06 2000. DOI: 10.1145/335191.335437.

Peter Rousseeuw. Least median of squares regression. *Journal of The American Statistical Association - J AMER STATIST ASSN*, 79:871–880, 12 1984. DOI: 10.1080/01621459.1984.10477105.

Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Miami Univ Coral Gables Fl Dept Of Electrical and Computer Engineering, 2003.