Regular Pattern and Anomaly Detection on Corporate Transaction Time Series

Francesca Soro, Marco Mellia
Politecnico di Torino
Torino, Italy
francesca.soro@polito.it
marco.mellia@polito.it

Nicolò Russo
CEE CIB Innovation, UniCredit S.p.A.
Vienna, Austria
Nicolo.Russo2@unicredit.eu

ABSTRACT

Business applications make extensive usage of time series analysis for the most diverse tasks. By analyzing the development of any phenomena over time we gain some useful insights on the stock market forecast, analyze the risk related to investments, understand the behavior of a company on the market and so on. More specifically, in a corporate investment banking environment, analyzing the transaction history of a customer over the years is crucial to establish a fruitful relationship and adapt to its behavioural changes. In this environment we recognize three macro-categories of phenomena of interest: cyclic events, sudden and significant changes in trend, and isolated anomalous points. In this paper we present a framework to automatically spot these behaviors by means of simple - yet effective - machine learning techniques. We observe that cyclic behaviors and sudden changes can be easily targeted by means of adaptive threshold algorithms, while unsupervised machine learning techniques are the most reliable in detecting isolated anomalies. We design and test our algorithms on actual transactions collected in the past two years from more than 2,000 customers of UniCredit Bank, showing the efficiency of our solution. This work is tested to serve as a decision aid tool for corporate investment banking employees to facilitate the inspection of years of transactions and ease the visualization of interesting events in the customer history.

1 INTRODUCTION

Time series are commonly defined as indexed points collected at regularly spaced points in time. Such representation makes them particularly suitable to represent a large amount of daily life and actual phenomena that vary over time. In the business field, for instance, they are extensively used in order to, among many other applications, evaluate risk, forecast future behaviors, predict stock prices changes, or detect anomalies in different types of transactions. In this work we focus on the latter application: we exploit, fine tune, evaluate and integrate into a comprehensive framework some well-known anomaly detection techniques to spot both typical and unusual behaviors in corporate banking transaction data [21]. Apart from raising awareness in presence of possible fraudulent events, detecting anomalies in this environment may rise the attention on a customer undergoing changes in ownership or management, choosing to operate in new markets, supplying new customers or adopting new suppliers, moving parts of its business relationships to a new bank, reducing or enlarging the number of employees, etc.. Bearing in mind these possibilities, we need to distinguish among three kinds of behaviors of interest:

- Cyclic phenomena: e.g., repeated peaks in the number of payments, representing salary or suppliers payments;
- Continuously increasing or decreasing trend in the incoming/outgoing amounts and counterparts, hinting to underlying policy changes;
- Single isolated anomalies that need more investigations.

In this paper we present different techniques to target each of these cases, spanning from simpler heuristics to a combination of Machine Learning (ML) models, to serve as a tool to compare the algorithms outcomes and advertise the most meaningful cases. The final aim of this work is to compare existing algorithms and techniques to detect anomalies in banking transactions data, and evaluate their performance, balancing the simplicity of the solution with its reliability. It should be clear that the final objective of the framework is not that of substituting the human supervision, but to serve as an instrument aiding the relationship managers to judge corporate clients behaviour and raise the attention on unusual movements. We apply these to a very large dataset of actual banking data. Given the impossibility to disclose actual banking data, it is not easy, to the best of our knowledge, to find a comprehensive work that provides an insight on the effectiveness of ML techniques in this environment and on anomalies showing different characteristics. As such, we are among the first to provide a practical solution in this area.

This paper is organized as follows: in Section 2 we discuss similar use cases or applications already present in literature. In Section 3 we describe the raw dataset and the preprocessing steps before feeding data to the algorithms. In Section 4 we provide a theoretical overview with the basic concepts underlying the applied methodologies. Section 5 contains a discussion and evaluation on the resulting outputs. Section 6 concludes the paper.

2 RELATED WORK

Financial modelling and business use cases make extensive use of time series analysis techniques. Authors of [22] enumerate some of the most relevant applications: interest rates, growth rate of the gross domestic product, inflation, index of consumer confidence, unemployment rate, trade imbalance, corporate earnings, book-to-market ratio, etc. Authors of [6] present a visualization tool that aggregates transactions recorded by Bank of America, to serve as an aid in spotting the first signs of money laundering activities. [11] reports a survey of the most used machine learning techniques in stock forecasting, providing some outlines on how to build an extensive set of input features. Another common field of application is risk in investment evaluation. This aspect is addressed in [24], where authors focus on noise prediction give information about its influence on trading behavior. Authors of [21] propose the usage of a regression classifier operating on

This work has been supported by the SmartData@PoliTO center on Big Data and Data Science and was developed in collaboration with UniCredit S.p.A. Zweigniederlassung, Wien.

Published in the Workshop Proceedings of the EDBT/ICDT 2020 Joint Conference (March 30-April 2, 2020, Copenhagen, Denmark) on CEUR-WS.org.

the tensor representation of High-Frequency Trading data. The analysis of such transactions and the modelling of their waiting time is also addressed by the usage of random variables in [18]. Time series can be enriched by other quantitative data: the work in [27] exploits a wider set of company-related financial indices, cash flow information and industry specific variables, to perform bankruptcy prediction. The above mentioned solutions are developed to model single specific cases, and often fail in capturing the similarities among phenomena. Moreover, detecting anomalies by using time series prediction outcomes implies a very strong assumption, i.e., the fact that the data used to train the model do not contain anomalies. Since this is not always the case in a real world scenario, unsupervised algorithms can represent a suitable solution to group together and spot similar patterns, isolating the most interesting anomalous ones. The work in [13] provides a useful partition of the time series clustering approaches: raw-data-based, feature-based, model-based (i.e., respectively giving in input to the model the raw dataset, some feature extracted from it or modelling the coefficients or the residual). The same work summarizes some of the most common clustering evaluation metrics. Authors in [12] provide an example of model-based clustering of time series, which models residuals by means of a Gaussian distribution. Many of the mentioned solutions are very specifically tailored to the problem they target, and are hence difficult to generalize to a wider set of use cases. In this paper we propose a comparative analysis of common off-the-shelf algorithms for anomaly detection applied to a large dataset of actual corporate investment banking transactions. The final aim is to provide techniques to tackle such problem efficiently and from different perspectives, according to the operator needs, sticking to easy to implement and understand algorithms commonly known by data science practitioners. The output of our analysis aims at facilitating the business relationships between the bank and the customer, allowing the relationship managers employees, who are not ML experts, to know in advance and adapt to eventual changes in the customer activities.

3 DATASET

The case study reported in this paper takes advantage of a large dataset recording years of corporate customers payment transactions. We define a transaction as a single payment directed to or operated by a given customer entity. The original dataset consists of more than 50,000,000 transactions, characterized by 35 different fields. All the transactions are reported from the point of view of the customer: it is either a beneficiary, i.e., a payment recipient, or a source, i.e., it is carrying the payment out. For this, we take into account the direction of the transaction and separate "incoming from "outgoing" transactions. In total, we count about 500,000 customers, many of them recorded only a small number of transactions, and hence not so relevant. To filter them, we keep only those overcoming a monthly threshold $T = 100$ of minimum incoming or outgoing payments, depending on the targeted application. Given the transactions involving each client, we group them by time interval, computing (i) the sum of amounts, and (ii) counting the number of unique counterparts. We set a different time granularity (i.e., daily, monthly, weekly, quarterly) according to the required task and need of the analysts. To provide an overview of the data, we report an example of the periodic payments phenomena we are looking for in Figure 1. In this case we are able to distinguish the payments of salaries from those directed to suppliers thanks to an algorithm internally developed by the company, called *Company or Physical Person*, or *CoPP*. By means of a Random Forest algorithm, it labels the transaction recipients based on a set of input features and flags such as the presence of company-related stop words (e.g., GmbH, S.r.l., S.p.a., etc.), the number of characters in the name, the volume of the transactions or the presence of a numerical value in the name of the beneficiary. In output, it classifies the payment transactions in those directed to suppliers and those directed to employees, respectively in blue and green in Figure 1, with an average accuracy of 96.3%. As it is visible in the plot, salaries constitute the largest number of transactions, and look periodical. Suppliers payment transactions tend to be more spread over the considered time period and generally lower in number.

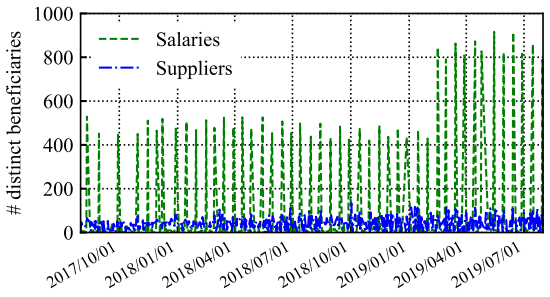


Figure 1: Output of CoPP algorithm classification

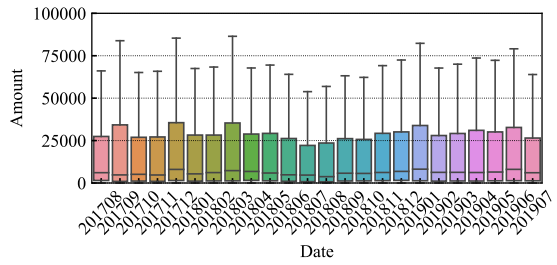
If we take a look at the distributions of the outgoing amounts per month, reported in Figure 2 for the same customer, we can see that despite being less in number, the transactions to suppliers (top plot) constitute the largest part of the overall outgoing amount, outperforming the salary payments (bottom plot) of two orders of magnitude. Moreover, from the spikes in Figure 2(b), we could understand that the customer in this examples may pay the 13th salary to its employees.

Figure 3 reports, on the other hand, an example of client showing a significantly decreasing trend. Such behavior should raise the attention of the relationship manager and lead to the investigation of the business relationships and events that originated this outcome.

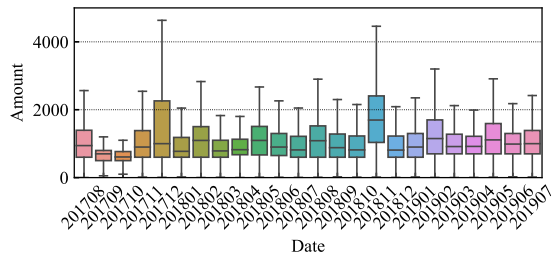
To serve as input to the proposed supervised and unsupervised models we further enrich the time series data to build a complete set of features useful for inspection. We report all the features and their brief description in Table 1. Please note that for confidentiality and privacy reasons we omit some of the details regrading the original dataset, and we cannot report complete examples.

Table 1: Lags dataset

Field	Value
y	Target variable
date	Date, daily granularity
y_t-D	Value of y at day d-D (D={1,..,6})
monthly_avg	Average value of y per month
weekly_avg	Average value of y per week
is_<(dayofweek)>	Flag, 1 if date is <(dayofweek)>
prev_W_weeks	Value of y at week w-W (W={1,..,3})
prev_M_months	Value of y at month m-M (M={1,..,3})
quarter_avg	Average value in quarter
neigh_4_days	Average value of y at +/-2 days



(a) Company amount distribution



(b) Physical person amount distribution

Figure 2: Amount distribution towards physical persons and companies

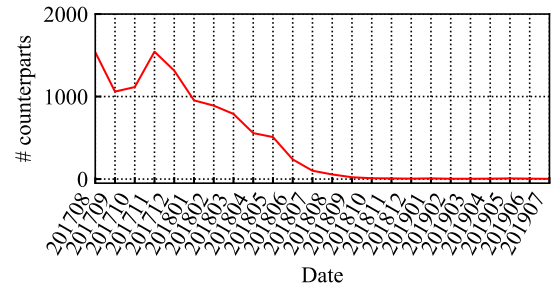


Figure 3: Example of client showing decreasing trend in incoming counterparts count

4 OBJECTIVES AND METHODOLOGIES

In this section we provide a brief description of the adopted algorithms. The heuristic techniques in Section 4.1 allow to recognize customers showing periodical repeated phenomena or sudden trend changes. Later we compare a set of supervised and unsupervised techniques to spot isolated single-point anomalies. We assume the reader is familiar with ML algorithms.

4.1 Periodicity and trend detection

We use heuristic techniques to detect two types of clients: the ones who periodically pay salaries or suppliers, and the ones who show a steep trend (either ascending or descending). As previously said, spotting these kind of customers is relevant as it allows the bank to highlight changes, e.g., in the company business operations or in the personnel composition. For these applications, we use the per-customer time series as input.

4.1.1 Salaries and suppliers payments detection. For this case study, we focus only on the number of unique counterparts towards the customer performs transactions. An (almost) regular pattern in the count of such operations is a clear sign that the customer is paying employees salaries, or suppliers. Normally such transactions appear as visible periodical spikes whose height does not show significant changes over time. This characteristic makes such behavior easy to spot by means of a simple adaptive threshold algorithms. Given the existing time series, we compute the maximum number of counterparts registered every year, namley $max\_y\_counterpart$. We then define the threshold $\tau = 0.8 \cdot max\_y\_counterpart$. We then test every row against this threshold. We get as output 1 if the number of counterparts for that day overcomes the threshold, 0 otherwise. To decide that a customer shows a cyclic behavior, we check that the threshold is overcome at least once per month for at least the 85% of the considered months (i.e., 20 months out of 24). Not reported here due to space limitations, the choice of parameters results robust, and the suggested values have been selected as best candidates.

4.1.2 Trend detection. We run the trend detection procedure on the count of both outgoing and incoming counterparts. Since a daily aggregation is not suited to detect a trend change, we aggregate the data considering their average per quarter of the year, as typically done in economy fields. We then compute the variations that interested the same quarters across all the available years (namely δ_{Q1}, δ_{Q2}, δ_{Q3}, δ_{Q4}). As a following step, we further aggregate the counterparts count by month, and we then fit to such time series a simple univariate linear regression model as:

$$y_i = \alpha + \beta \cdot x_i + \epsilon_i \tag{1}$$

We take into account the values of β and the *p-value*. The former gives information on the slope of the intercept line, the latter on the correlation of the target with the given regressor. We only consider as statistically significant those customers having $p\text{-}value \leq 0.05$. We then define two subset of customers: those having a relevant *increase* in the trend for at least a quarter (i.e., $\delta_{Qi} \geq 30\%$), and those showing a relevant *decrease* (i.e., $\delta_{Qi} \leq -30\%$). This combination of thresholds set on the δ_{Qi} and on the *p-value* allows us to filter customers with clear trends. At the end of this stage, we notify the relationship manager with a list of customers to carefully monitor.

4.2 Isolated anomalous points detection

Heuristics and simple linear regression models do not target isolated anomalies equally well. For this purpose we realize a framework including standard algorithms for time series forecasting, supervised and unsupervised techniques. We describe the specifications about our implementation below. A detailed discussion on the techniques is out of the scope of this work.

4.2.1 ARIMA models. Time Series Forecasting is an extensively used technique to tackle the anomaly detection problem ([17], [7], [23], [9]). Its final objective is to provide a prediction of the future values of a time series based on its past values. By defining a reasonable confidence interval for the predicted values, we identify as anomalies the points that fall outside such predicted interval. Here we focus on Auto Regressive Integrated Moving Average Models (ARIMA). Every ARIMA model requires as input a stationary time series. Wand three fundamental parameters: p, the number of autoregressive terms, d, the order of the differencing term and q, the number of moving average

terms. As a general best practice, we should keep in mind that the values of p, d and q are usually kept below 3. We hence run a grid search with parameters ranging from 0 to 3. We instantiate an ARIMA model per each combination and we chose the best model by calculating the Mean Squared Error between the actual value and the prediction yielded by every (p, d, q) triplet. Given the best model, we compute the upper and lower boundary of the confidence interval for each prediction, and we flag as anomalous every point falling outside such boundaries.

4.2.2 Supervised techniques. All the following techniques use as input the dataset described in Table 1, properly standardized. Similarly as before, we consider well-accepted ML algorithms that we train to predict the next value $\hat{y}$. We run hyperparameter selection, and compare the prediction $\hat{y}$ with the actual value y. Differently from ARIMA models, we do not have a standard way to compute confidence intervals, thus we rely on domain knowledge driven heuristics to flag outliers. In details we define a set of threshold-based control criteria to label a point as anomalous: the anomaly is advertised if at least one criterion is triggered. We list all the criteria below:

- Criterion 1 (multiplicative):

$$\begin{cases} \text{KO : if } (y > \tau\hat{y}) \ \wedge (\mid y - \hat{y} \mid >= \sigma_{min}) \\ \text{OK : else} \end{cases}$$

- Criterion 2 (additive):

$$\begin{cases} \text{KO : if } (y - \hat{y}) > k\sigma_{rolling} \\ \text{OK : else} \end{cases}$$

- Criterion 3 (multiplicative positive):

$$\begin{cases} \text{KO : if } (y > \hat{y}) \ \& \text{ Criterion 1} \\ \text{OK : else} \end{cases}$$

- Criterion 4 (additive absolute):

$$\begin{cases} \text{KO : if } \mid y - \hat{y} \mid >= k\sigma_{rolling} \\ \text{OK : else} \end{cases}$$

Where τ and k are multiplicative thresholds we manually tune, σ_{min} is the minimum monthly standard deviation of the label variable, and $\sigma_{rolling}$ is the rolling standard deviation of the label computed month by month.
For the regression, we consider state of the art algorithms. We briefly report the chosen configurations below. We manually tune all the algorithms parameter, and we hereby report only the resulting best configurations for the sake of space. For the Support Vector Regressor ([25], [4]) we exploit three different kernel functions: the linear, the polynomial and the RBF one. All kernels require a regularization parameter $C = 100$, an $\epsilon = 0.1$, while the polynomial kernel takes also $degree = 3$. The Stochastic Gradient Descent Regressor [10] exploits the standard concept of stochastic gradient descent to fit linear regression models. We choose to fix the number of maximum iterations to $I = 100,000,000$, and a stopping criterion on the validation score improvement $tol = e^{-10}$. We then exploit a set of Decision-Tree based regressors: we first instantiate a simple Decision Tree, which we then use as a building block for an AdaBoost regressor [20] and a Random Forest regressor [1]. For the former we specify that we want to terminate the boosting at $n\_estimators = 300$; for the latter we require a number of trees $N = 100$ and we set $max\_depth = 20$.

4.2.3 Unsupervised techniques. All the implemented unsupervised techniques require the specification of different parameters and outlier identfication criteria. The former is algorithm-specific: We combine three cluster quality measures: the *Silhouette* [19], the *Davies-Boulding index* [3] and the *Calinski-Harabasz index* [2] to choose the best clustering configuration. More details for each algorithm below. The latter is generic and based on the definition of a threshold p that defines the maximum number of points a cluster should contain to be labeled as anomalous. We found $p = 5$ to provide best results for our case study.

We consider four clustering algorithms: k-Means [15], DBScan [5], Hierarchical clustering [16] and Isolation Forest [14]. More in detail, for k-Means we are required to specify the parameter k, whose evaluation is commonly pointed out as a critical aspect of the algorithm itself ([8], [26]). We target this problem by restricting the possible range of k to a reasonable set of values defined according to our domain knowledge: we let k range from 2 to 7. For each k we compute the number of anomalous clusters (i.e., the clusters containing $N_j \leq p$ points), and among those, we identify the most common number of anomalous clusters by taking the mode of such column. We chose the best k according to each score. If the algorithm never identifies any anomalous cluster, it returns 0. The advantage of DBScan is that it does not require to define the number of clusters a priori and it isolates noise points without using the aforementioned threshold p. We automatize the choice of the parameters ϵ and $minPoints$ by first evaluating the distribution of the nearest neighbour distances. Once we define the average most common value of distances as our ϵ, we calculate the distribution of the number of points according to this $\epsilon - neighbourhood$, and we choose our $minPoints$ value in the same way. We finally run the algorithm with the selected parameters, and we consider anomalous all the points marked as noise. When using Hierarchical Clustering we again face the problem of defining the correct number of clusters. This depends on the setting of the cutoff level on the obtained output dendrogram. We limit the set of reasonable cutoff levels up to $C = 5$, and we evaluate the best C by considering the number of anomalous clusters according to the threshold p combined with the best scores. If we are unable to identify small anomalous clusters, we yield 0 as a result. Isolation Forest algorithm shows as its main strength the fact that it does not require an a-priori definition of the number of clusters, but only on the selection of a reasonable number of base trees, as required by some of the previously described supervised algorithms. The main underlying idea of this technique is that anomalies are often isolated points whose identification requires just a few partitions of the feature space to separate them from the more concentrated sets of points. We do not rely on the output of a single tree, but on the average output generated by a set of $N = 100$ trees.

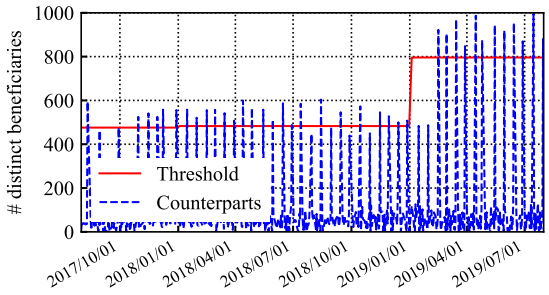
5 CASE STUDY AND RESULTS

In this section we proceed with the description and comment of the results obtained with the different automatic detection techniques illustrated in Section 4. Please note that, for data confidentiality reasons we report an indicative range for all the numeric results. We use Python 3.7[1] as a programming language, together with Pandas [2] and scikit-learn [3]libraries. All the examples use as input the transactions of a subset of 2,000 customers.

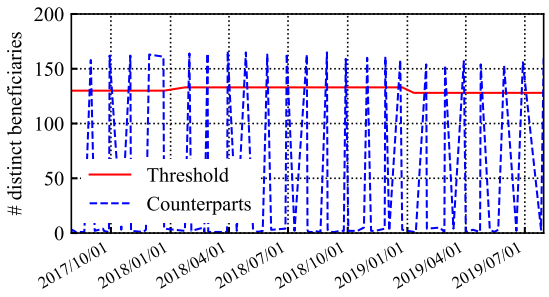
[1] https://www.python.org/downloads/release/python-370/
[2] https://pandas.pydata.org/pandas-docs/stable/index.html
[3] https://scikit-learn.org/stable/

5.1 Periodicity and trend detection

5.1.1 Salaries and suppliers payment detection. As already reported in Section 4.1.1, we define salaries and suppliers payment those outgoing transactions showing a regular spikes over time. The height of the spikes is generally almost constant but it may be subject to changes from time to time. The output of the adaptive threshold heuristic points out that slightly less than the 25% of the customers under analysis performs periodical payment transactions. We manually verified about 100 cases and found no evident sign of miscalssification. For instance, we report the output for two customers, namely *CLI1* and *CLI2*, in Figure 4. In the two figures we can see clearly how a large number of transactions are concentrated in certain days of the month, and repeated periodically. CLI1, in Figure 4(a), shows an increase in the number of distinct transactions after the beginning of 2019: this may suggest, for instance, a change in the relationships with the suppliers, or in the composition of the workforce with newly hired employees. Our algorithm automatically adapts the threshold τ, helping the relationship manager to detect the change. For instance, we can suppose that the company is growing or trying to enlarge its business. On the other hand, CLI2 in Figure 4(b) shows an almost regular pattern over the analyzed time period, with a variation of ±3 counterparts over the years. Identified peaks are consistent with personal payments as identified by the CoPP algorithm. The output of the salaries detection procedure is meant to be read together with CoPP, to allow the relationship manager to have a clear overview on the customer business choices and structures. Notice that our solution does not require labelled dataset, a often time-consuming process.



(a) Counterparts, CLI1



(b) Counterparts, CLI2

Figure 4: Output of the salary detection process

5.1.2 Trend detection. For the sake of space, we discuss the output of the heuristic reported in Section 4.1.2 for the first and last quarter variations, namely δ_{Q1} and δ_{Q4}. We consider incoming and outgoing counterparts separately. Out of the 2, 000 clients originally in scope, 20% shows an increasing trend in the incoming counterparts and 7.5% shows a decreasing trend. Considering outgoing counterparts, 20% of customers shows increasing trend, 6% shows decreasing trend. Figure 5 reports some examples of the automatically detected behaviors for 3 different customers. As visible, all of them show a clear decreasing trend, correctly identified by the algorithm. Also in this case, we manually verified the results, showing no errors in the classification.

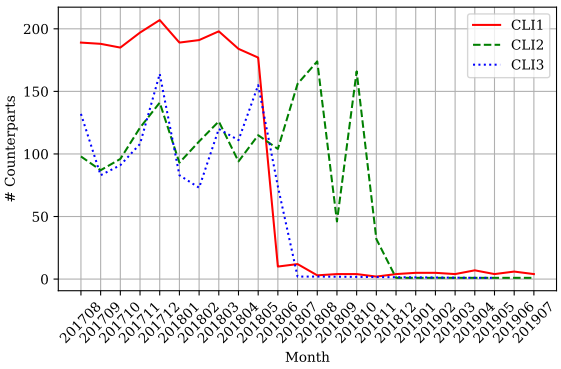


Figure 5: Customers with relevant changes in trend for outgoing counterparts count

In case of trend detection, we also present to the relationship manager a set of aggregated statistics on the quarter variations. Figure 6 shows, for instance, the δ_{Q1} (in red) and δ_{Q4} (in blue) distributions per turnover buckets. In the figure we observe a positive growth from one year to the following in most of the buckets. The exception to be highlighted is the case of small companies (i.e., the ones having turnover between 0 and 500, 000 RON), which show a very significant growth in $Q1$. The Relationship Manager should pay attention to the bucket 500k - 1MLN, showing a significant decrease.

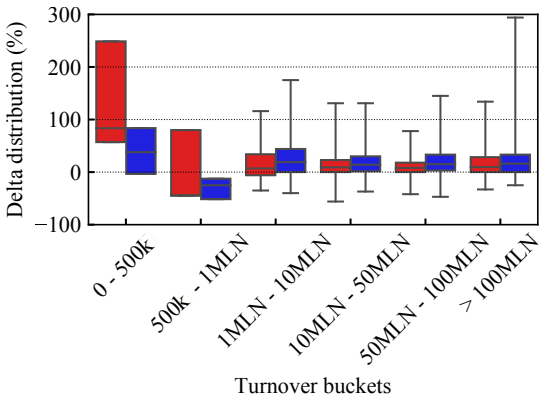


Figure 6: Delta Q1 and Q4 distribution for outgoing counterparts per turnover bucket

Table 2: Arima and Supervised algorithms scores

	ARIMA	ADA	DT	RF	SGDR	Lin	Poly	RBF
Accuracy	0.75	0.88	0.89	0.73	0.51	0.61	0.28	0.54
Precision	0.06	0.19	0.18	0.14	0.08	0.11	0.01	0.03
Recall	1	0.89	0.89	0.89	0.4	0.58	0.79	0.88

Table 3: Unsupervised algorithms scores

	DBScan	Agglomerative	Isolation	K-means
Accuracy	0.99	0.98	0.97	0.99
Precision	0.52	0.51	0.19	0.77
Recall	0.63	0.62	0.48	0.8

5.2 Isolated anomalous points

For the sake of space, in this Section we discuss the results for a subset of 30 clients whose time series passed the stationarity test. We run all of the considered algorithms using the 80% of the original dataset for the training phase, and the remaining 20% for testing (i.e., roughly the last three months of transactions). Since we do not know if there are anomalies or not, we evaluate the reliability of the predictors through the insertion of artificial anomalies. In particular, we add anomalies on the testing set by extracting a random subsample of $L = 10$ events from such set (i.e., about 10% of the instances of the testing set). We iterate over the whole time series adding one anomaly at a time. We want the anomaly to be clearly out of the standard range of the time series, therefore we randomly choose a random entry e, and we modify it as:

$$e^* = e \cdot N(7, 2.5) + k \cdot max(ts) \tag{2}$$

where N is a normal distribution with mean $\mu = 7$ and standard deviation $\sigma = 2.5$, and k is a multiplicative coefficient we set equal to 2.

We now evaluate the obtained outputs. Recall that a point is considered anomalous by ARIMA if it falls out of the confidence interval boundaries; for the supervised models if it falls out of the defined threshold boundaries; and for the unsupervised models if it belongs to a small cluster or it is recognized as a noise point. We should further point out that we retrain every model from scratch for each client. Tables 2 and 3 report the performance metrics for all the algorithms. Such metrics allow us to surely consider unreliable the Support Vector-based algorithms and the Stochastic Gradient Descent Regressor, as they are not able to recognize the true anomalies (small recall), and they wrongly tend to mark a very large set of points as anomalies (small precision). This second problem is common to all supervised approaches, since all them present a very low precision. In practice, the "noisy" time series does not allow the regressor to correctly predict the next value, which too often results as an outlier (raising false alarms). The unsupervised models show instead a better average behavior, coming from the fact that they tend to advertise anomalies only if their classification is very sure. This leads them to be more precise: a simple k-means would indeed identify 80% of anomalies, with 77% of recall. These results are consistent also for different values of (μ, σ, k) - not reported here for the sake of brevity.

6 CONCLUSIONS

In this paper we presented a case study on anomaly detection in corporate investment banking transaction data. The anomalies have been divided in three different categories, according to their general characteristics, and targeted with the most appropriate set of techniques spanning from simple adaptive threshold heuristics, to several types of machine learning algorithms. We demonstrated that phenomena such as salaries and periodic suppliers payments can be reliably spotted by means of an adaptive threshold algorithm, while a standard linear regression comes in handy when major changes in trend need to be detected. We further provided a comparative analysis of the performance of well-known machine learning algorithms in spotting isolated anomalies, whose result make us lean towards the usage of unsupervised algorithms. All the provided results are presented in a way that they can serve as a decision-aid tool for the bank employees that need easy to read and understand results when dealing with corporate customers.

REFERENCES

[1] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[2] Tadeusz Caliński and Jerzy Harabasz. 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods* 3, 1 (1974), 1–27.
[3] D. L. Davies and D. W. Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), 224–227.
[4] Harris Drucker et al. 1997. Support vector regression machines. In *Advances in neural information processing systems*. 155–161.
[5] Martin Ester et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *Kdd*, Vol. 96. 226–231.
[6] Chang et al. 2007. Wirevis: Visualization of categorical, time-varying data from financial transactions. In *2007 IEEE Symposium on Visual Analytics Science and Technology*. IEEE, 155–162.
[7] Durdu Ömer Faruk. 2010. A hybrid neural network and ARIMA model for water quality time series prediction. *Engineering Applications of Artificial Intelligence* 23, 4 (2010), 586–594.
[8] Greg Hamerly and Charles Elkan. 2004. Learning the k in k-means. In *Advances in neural information processing systems*. 281–288.
[9] Karin Kandananond. 2019. Electricity demand forecasting in buildings based on ARIMA and ARX models. In *Proceedings of the 8th International Conference on Informatics, Environment, Energy and Applications*. ACM, 268–271.
[10] Jack Kiefer et al. 1952. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics* 23, 3 (1952), 462–466.
[11] Bjoern Krollner et al. 2010. Financial time series forecasting with machine learning techniques: a survey.. In *ESANN*.
[12] Mahesh Kumar et al. 2002. Clustering seasonality patterns in the presence of errors. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 557–563.
[13] T Warren Liao. 2005. Clustering of time series data - a survey. *Pattern recognition* 38, 11 (2005), 1857–1874.
[14] Fei Tony Liu et al. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 413–422.
[15] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA, 281–297.
[16] Frank Nielsen. 2016. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*. Springer, 195–211.
[17] Ping-Feng Pai and Chih-Sheng Lin. 2005. A hybrid ARIMA and support vector machines model in stock price forecasting. *Omega* 33, 6 (2005), 497–505.
[18] Marco Raberto et al. 2002. Waiting-times and returns in high-frequency financial data: an empirical study. *Physica A: Statistical Mechanics and its Applications* 314, 1-4 (2002), 749–755.
[19] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.
[20] R. E. Schapire. 2013. Explaining adaboost. In *Empirical inference*. Springer, 37–52.
[21] Dat Thanh Tran et al. 2017. Tensor representation in high-frequency financial data for price change prediction. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 1–7.
[22] Ruey S Tsay. 2014. Financial Time Series. *Wiley StatsRef: Statistics Reference Online* (2014), 1–23.
[23] Fang-Mei Tseng et al. 2001. Fuzzy ARIMA model for forecasting the foreign exchange market. *Fuzzy sets and systems* 118, 1 (2001), 9–19.
[24] Tony Van Gestel et al. 2001. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on neural networks* 12, 4 (2001), 809–821.
[25] Vladimir Vapnik et al. 1997. Support vector method for function approximation, regression estimation and signal processing. In *Advances in neural information processing systems*. 281–287.
[26] Kiri Wagstaff et al. 2001. Constrained k-means clustering with background knowledge. In *Icml*, Vol. 1. 577–584.
[27] Qi Yu et al. 2014. Bankruptcy prediction using extreme learning machine and financial expertise. *Neurocomputing* 128 (2014), 296–302.