AUTOMATIC DATA QUALITY MANAGEMENT INTERNSHIP 2018

TABLE OF CONTENTS

INTRODUCTION

ABOUT AVANADE

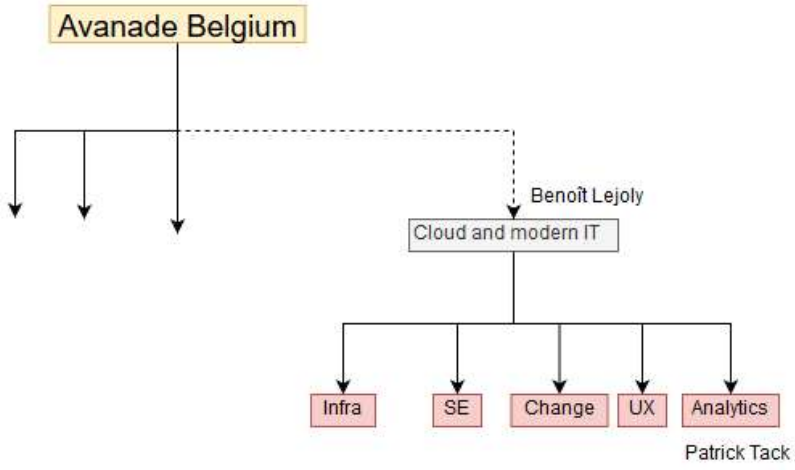
Before explaining the content of the internship that begins in section two of this report, a brief overview of what Avanade is and the choice of this company as a second internship is required.

Avanade was founded as a joint venture between Microsoft Corporation and Accenture LLP. This company was created in April 2000 but the company did not set up in Belgium until June 2008. Avanade is a global services company providing IT consulting based on the Microsoft platform on these domains: Software Engineering, Analytics, Infrastructure and Experience Design.

In the aspect of Software Engineering, the main activity of Avanade is to apply innovative solutions for clients with new Microsoft technologies (like cloud computing, etc…). It is a subsidiary of Microsoft and Accenture, that means that some projects are directed by those two companies (for example, when Accenture needs expertise on Microsoft technologies).

Having completed my first internship in the big company Euroclear in the field of Business Intelligence and Advanced Analytics, I wanted to deepen these areas by continuing on these roads while discovering another atmosphere, closer to a start-up, to have an idea on the two distinct environments.

Although it cannot be said that Avanade is a start-up, it is a growing company and organizes many internships to attract young graduates. Also, the projects are varied. Avanade being a consulting company, employees are invited to work at the client's premises. As a result, Avanade's offices are less centralized and the total number of employees is quite difficult to determine if we do not have activities to bring people together. 2 offices are nevertheless available to allow employees to work when they are not on a project. The first, in Merelbeke, where I had the opportunity to go there 3 times, and the second in Brussels, near the Porte de Namur metro station. It was at this second office that I was able to work on the project.



BI Analytics, the team I was affiliated with was led by Patrick Tack. It is very recent and mainly includes motivated young people with a sufficiently wide openness to allow everyone to develop and find interesting projects.

ORGANIZATION OF THE PROJECT

The project consists in automating data management and improving its content. Indeed, many companies today collect a lot of data through connected objects, databases, etc. and seek to extract information from this data through models, simplifications and compressions. However, in this data collection, small problems become big problems on a large scale and a tool to solve these problems as soon as they appear would be very interesting for everyone.

To find a way to develop this tool, I worked as a team with another South Korean intern. While I was an industrial computer engineer, she was a student in applied mathematics with a statistical orientation. All research and hypotheses were analyzed and then discussed to allow everyone at their level to understand the development of the data quality management automation algorithm.

The project can be divided into two sections: the research part and the implementation part. During the first period of the internship, the objective was to share the respective knowledge of each other to address a new issue, which is the automation of data quality management.

AUTOMATIC DATA QUALITY MANAGEMENT

DESCRIPTION OF DATA QUALITY PROBLEMS

Data collected in the business environment often contain errors, omissions and inconsistencies. The followings are the most common data quality problems:

- Typos: a typographical error in input.
- Mixed formats: several data formats are used for the same variable.
- Missing values: records with no corresponding values (e.g. location, age, year) or invalid values (e.g. a 300-year-old customer)
- Duplicated observations: replicated entries of the same real-world entity. The same instance was entered more than once.
- Outliers: values showing different trends from the majority of the data

The objective of the internship was therefore to do data cleaning. We define data cleaning as the process of identifying and correcting incomplete or incorrect aspects of the data. The correction can be performed by modifying or deleting the inaccurate part of the data. It is necessary to perform data cleansing before the analysis step, so that predictive models can be built based on the data of high quality. Otherwise, predictive analytics would lead us to inaccurate results and wrong decision makings. Once the errors are detected, data cleaning can be performed using scripts, human efforts, or a hybrid of both. Further description of data quality problems are described in the subsequent sub-sections.

Within the framework of the internship and the time allocated to us to carry out this task, we limited ourselves to the case of outliers and impossible data. Indeed, this was a very new field for both of them, and teamwork was also a challenge.

OUTLIER ALGORITHM

The idea behind it is an algorithm is really very simple: a first step would be to train a model to discern the classical data from outliers and errors, a second step would be to apply the model to new data to be able to classify them in their respective category, and finally a third and last step would consist in managing these data (by processing them and, if necessary, re-training the model if it is no longer relevant).

During the training phase, a data set is loaded and analyzed in its entirety. We analyze the frequencies and similarities between the data, determine the outliers, impossible data and relevant data and label these data without modifying them.

During the detection phase, however, any new data will be processed separately, labelled based on the above model.

Finally, during the management phase, the data deemed relevant will be kept, the data deemed impossible will be removed, and the data considered as outliers will be modified according to a specific rule.

TRAINING PART

The training phase of the model is probably the longest task we have implemented because it is the most complex to implement and yet the most important. A bad model can only lead to bad results and that is why the internship went through a long phase of research rather than implementation.

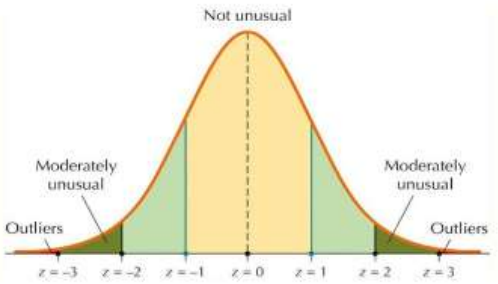
There are a variety of definitions of an outlier. Outliers are data points that deviate from the other data points in the dataset. Frequently, outliers are confused with extreme values, however, extreme values are not necessarily outliers if the data have a skewed distribution. Outliers should be handled carefully in data science applications. When outliers are unknowingly included in predictive analysis, this may introduce a bias into the model. It is very important to detect outliers properly and handle them.

Outlier detection can be performed in the univariate and multivariate manner. In univariate outlier detection, a method is applied to only one variable. For example, when one examines a dataset for heights and weights, unusual values for heights and weights are detected separately, therefore, one observation may have an outlying height but normal weight. Observations are declared as outliers, if they lie outside of the normal range. However, variables in the same dataset often have correlations among themselves and these relations are crucial for predictive models which are constructed after data cleaning. Univariate outlier detection does not account for such correlations between variables. In the example above where a linear correlation between heights and weights exist within the data, a patient with high heights and heavy weights would be identified as an outlier in the univariate manner. However, since the patient is following the same linear trend of the rest of data points, he or she may not be a normal data point. In order to resolve such problems of univariate outlier detection, multivariate outlier detection is required. Especially, if predictive analysis is performed in the multivariate setting, multivariate outlier detection is recommended.

UNIVARIATE METHODS

For univariate methods, the following 4 methods are used: Z-scores, the inter-quartile range, boxplot and winsorization.

In order to use Z-scores, the data should follow a normality distribution or at least have a similar distribution (e.g. T-distribution), so that some observations deviating from the mean of the distribution are correctly identified as outliers. On the other hand, one can use the median to find outliers, not the mean. The crucial problem with the mean is that the mean value itself is affected by outliers in the calculation process, and thus, is not robust to outliers.



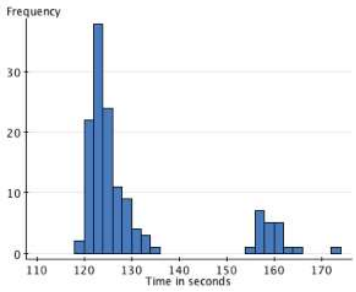
The inter-quartile range uses the median value in order to measure variability in the dataset. Quartiles divide the dataset into four halves based on the minimum, 1st quartile, 2nd quartile (median), 3rd quartile and maximum. The inter-quartile range is the difference between the 3rd quartile and the 1st

quartile (Q3-Q1). It is possible to define observations smaller than Q1-IQR*1.5 or larger than Q3-IQR*1.5. The value 1.5 indicates approximately 1% of measurements are outliers and this is consistent with the 3 sigma principal. However, the inter-quartile range method also requires a centrality tendency within the data.

For a right–skewed distribution, the boxplot detects too many high value outliers and too few small value outliers.

Lastly winsorization replaces outliers with 'normal' values. For winsorization method, users can specify the percentage to cut on each side of the data array with respect to the number of unmasked data, when the data array is sorted. The percentage has a numeric value between 0 and 1 (limit=[L1,L2]). When n is the number of unmasked data before trimming, the (n*L1)th smallest data and the (n*L2)th largest data are masked, and the total number of unmasked data after trimming is n*(1-sum(limits)). For example, a 90% winsorization would see all data below the 5th percentile set to the 5th percentile, and data above the 95th percentile are set to the 95th percentile. In order to make use of winsorization method correctly, it is crucial to define an appropriate cut-off value for outliers. It is recommended to refer to published literature to see if the data at land are commonly winsorized and what percentage is usually taken in the field. It means that literature search and descriptive statistics are to be performed before data pre-processing. In the automatic data cleaning context, such careful specification of the cut-off value is not feasible. Therefore, one should keep that in mind that automatic cleaning may result in some errors.

However, it should be remembered that these methods work very well for normal distributions, but pose problems when the latter are a little more exotic as below:



The univariate methods are simple and allow the detection of many outliers that are obvious at first sight. Indeed, by this method, any value considered as approvable because it is too different from the others will be considered as to be corrected. However, univariate methods were very quickly abandoned and our research was more oriented towards multivariate methods. Indeed, the detection of outlier was not complete. In a table of individuals where each observation had several numerical data, these data were very often correlated with each other. However, for some data, this relationship was no longer respected, which could lead to the model being distorted, and yet univariate methods do not allow the detection of such outliers.

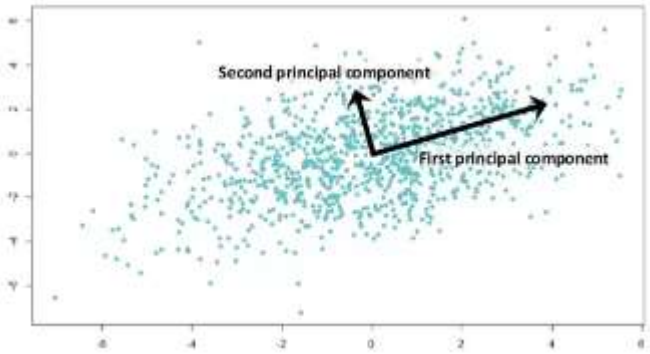
Let's take an example: we have a population of individuals with 2 characteristics: height and weight. The most elementary law would tell us that the taller an individual is, the heavier he weighs. However, in the survey population, this is not always the case. The height is not necessarily incredibly large but what is peculiar is that the weight seems to be more fabulous than the average. Only a method that observes all the variables of an individual would be able to recognize the outlier. Similarly, let us

imagine that we have a list of outliers found by univariate methods. How can we change the excess value of the data without looking at its relationship to the other columns?

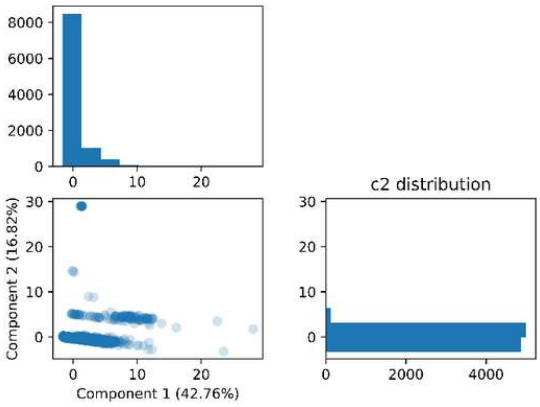
Those reasons have led us to take an interest in multivariate methods.

PRINCIPAL COMPONENT ANALYSIS

Principal component analysis (PCA) is a dimensionality reduction algorithm which is useful for data visualization and outlier detection. PCA algorithm detects the relationship between features and quantifies this relationship by finding multiple principal axes in the data. These principal axes are used to describe the dataset.



For the first step of principal component analysis, the features are standardized, provided that all the columns possess the same importance for the analysis. For example, a dataset can have the number of passengers in a taxi and the total trip distance of a taxi. The scales of these two features (the number of people and kilometers) are fairly different. Since the variance of total distances is much higher than the variance of the number of passengers in the taxi, without standardization, the value of information from total distances would be over-rated. After standardization (normalization), principal components are extracted and PCA scores for all rows are computed, by projecting each row to the eigenvectors. The longer principal axes contains more information on the distribution of the data than the shorter principal axes, and thus it contributes to the description of the data more. The projection of each data point onto the principal axes is equivalent to the 'principal components' of the data. In other words, PCA algorithm looks for correlations among multiple variables and combines the variables in order to best capture differences within the data. These combined feature values which are used to create a more compact feature space called the principal components. For anomaly detection, each new input is analyzed, and the anomaly detection algorithm computes its projection on the eigenvectors, together with a normalized reconstruction error. The normalized error is used as the anomaly score. The higher the error, the more anomalous the instance is.

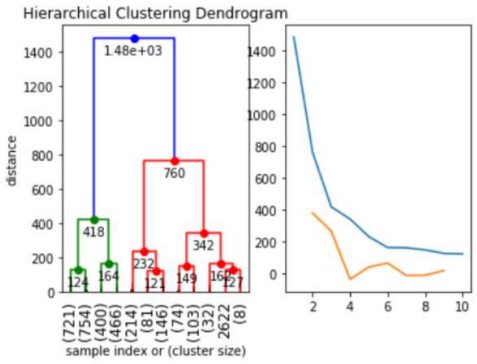


The main component analysis therefore seemed ideal for outlier detection. Indeed, from a table of individuals with many variables, these variables were limited to the different observable behaviors, and any individuals with abnormal behavior (too marked or even missing) were considered as outliers. The modification was also very easy, since it was simply adding or removing a behavior that is independent of other observable behaviors.

In practice, unfortunately, it didn't work. Indeed, when we talk about analysis in terms of main components, we are talking about projection and this is always accompanied by a loss of information (margin of error). This can be very small if the data are very similar, but very large if the data are highly variable. And in our case, the data was too varied and to keep an adequate amount of information, the subspace had to remain large. Thus, many individuals saw several behaviors modified because the same variable was contained in several different behaviors. A data considered as outlier because one of its variables was outlier and was completely modified

CLUSTERING METHODS

Hierarchical clustering creates a hierarchical structure among clusters. It is visualized with a tree diagram (a dendrogram) where small clusters are located at the bottom of the structure and large clusters are at the top. There are two approaches to hierarchical clustering: agglomerative (from the bottom up, grouping small clusters into larger ones) and divisive clustering (from the top down, splitting big clusters into small ones). Divisive clustering is often computationally expensive and not efficient. Therefore, the focus is on agglomerative clustering where small clusters are all merged into one cluster at the end. Especially, Ward's method merges clusters in a way that it maximizes the between-cluster distances, while it minimizes the within-cluster distances.

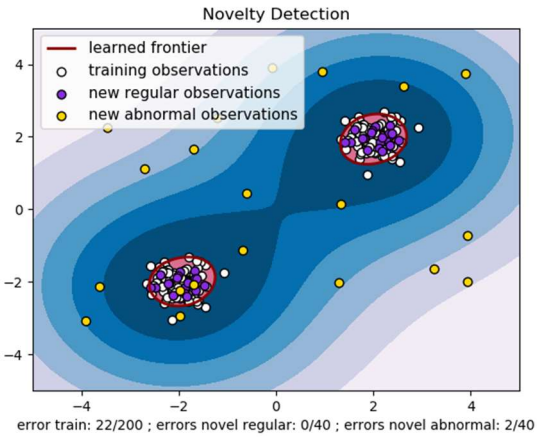


Ward's method calculates the distance between cluster members and the centroid. The centroid of a cluster is defined as the center of a cluster (the point at which the sum of squared Euclidean distances between the point and each other data point is minimized).

ONE-CLASS SVM

One-class SVM is an unsupervised algorithm that learns a decision function for novelty detection: classifying new data as similar or different to the training set.

The core SVMs are implemented in scikit-learn in the classes sklearn.svm.SVC for classification and sklearn.svm.SVR for regression. In these two classes, it is possible to specify a kernel using the "kernel" parameter. This nucleus can be one of the great classics (linear, polynomial, RBF), but it is also possible to define other nuclei.



This non-supervised algorithm allows to have not only the information on whether the data is an inlier or an outlier but also a confidence measure on this decision (expressed in probability) and is very easily implemented in machine learning studio. It is finally this methodology that will have been used in the deployment

DETECTION PART

Alternatively, imputation can be performed using the K Nearest Neighbor (KNN) approach. The assumption behind using KNN for missing values is that a point value can be approximated by the values of the points that are closest to it, based on other variables. The KNN algorithm identifies 'k' closest observations based on the euclidean distance and computes the weighted average (weighted based on the distance) of these 'k' observations.

The k-nearest neighbors algorithm (KNN) is a non-parametric method for classification and regression, which does not make assumptions for the data distribution. Therefore, not the assumptions but the data determine the model structure. For classification, the target feature is a categorical variable, whereas for regression, the target is a continuous variable. For regression, the algorithm uses a weighted average of the k nearest neighbors, weighted by the inverse of their distance.

The input is the k closest training examples and the output is the property value for the object. To be specific, this value is the average of the values of its k nearest neighbors. KNN is a type of instance-based learning (lazy learning) where the function is only approximated locally and all computation is deferred until final classification. That is, the algorithm does not learn from the data or perform generalization based on the dataset, but rather retains all the data points in the training set. The retained data points are used as 'knowledge' for the test phase. All the data points are needed in the test phase. The KNN algorithm is one of the simplest machine learning algorithms.

Similarity is defined by a distance metric between two data points and often Euclidean distance is used.

MANAGEMENT PART

Anomalies can be treated in the following ways:

- Removal: remove the entire row of the data.
- Mean imputation: replace outliers with the mean. Mean imputation makes the data more centralized around the mean.
- Winsorization: replace data with the nearest 'normal' values. Winsorization reduces the effect of possibly spurious outliers to the model.
- Imputation by the nearest inlier

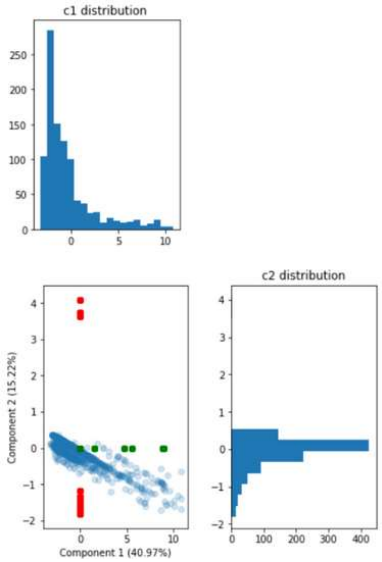
REMOVE

This very simple method is indeed very often used because it limits the margin of error. Indeed, any modification can sometimes go against what logic would want. This can lead to changes that would change the character of the observed data, remove any useful information from it or even create new information.

However, removing data involves significant risks. First, the orientation of outliers would be lost. Indeed, if all outliers were on the same side of the distribution, removing these data would reduce the distribution to more symmetric relative to its average. In addition, it is impossible for a model to learn from its mistakes and evolve if the incorrect data is removed from the dataset.

In the univariate setting, outliers are rather easily modified. One can define a range of normal data and modify outliers so that they fall into the normal range. However, in the multivariate setting, it is already hard to define a range of normal data since multiple columns are taken into consideration. Replacing outliers with new values is complicated, since one should decide which variables should be changed and to which extent the change would be. Therefore, if the proportion of outliers is acceptably low, then removing outliers reduce the risks of data modification.

MEAN IMPUTATION



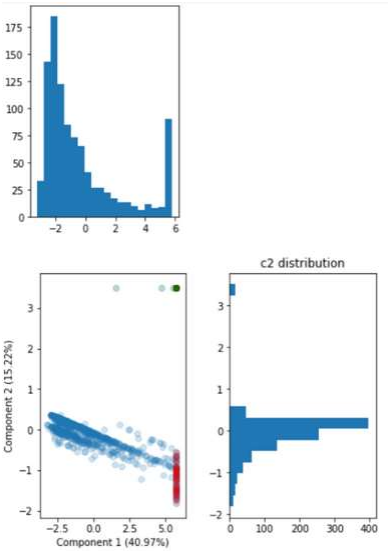
The replacement of an outlier by the average of the observed data is a second possibility for the management of outliers.

However, even if the number of data is retained, the amount of information is lost as much as if the data had been removed from the dataset. Indeed, the substitution of values by their mean does not allow to locate the orientation of the data in relation to the data cluster and the mean cannot be corrected by a trend of outliers.

This methodology only offers the possibility to keep the same number of data in a dataset without adding extravagant information. However, it should be noted that a non-centralized distribution (two normal distributions on either side of the centre of gravity for example) would see this modification as an error made by the user.

In the example opposite, we applied this method for data represented in the subspace of the main component analysis. For each component, the data undergoes univariate analysis (because each component is necessarily independent of the others) and the behaviour outside the normal range is then reduced to zero (since the distribution is based on reduced centred data)

WINSORIZATION



The winsorization principle is similar to that of mean imputation, noting that the latter takes into account the orientation of the data. Thus outliers to the right of the average will undergo truncation to the right of the average. This methodology may seem good, however, grouping all the data into the few points (those defined by winsorization) can be problematic because they are very subjective. Where to place the window? Another problem also remains that if the data do not follow a normal distribution, then they will tend to see a truncation that may be different from the one expected.

In the example opposite, we applied this method for data represented in the subspace of the main component analysis.

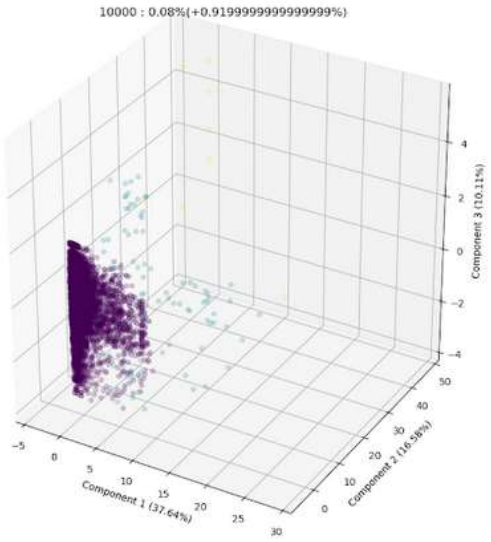
IMPUTATION BY THE NEAREST INLIER

this last method consists in replacing the different outliers detected by the values of the nearest inlier. This therefore requires having a data table composed only of inlier (which is provided by the model training) and allows us to solve all the problems mentioned above. Indeed, since the inlier has already been obeserved, we are sure to impound the outlier with observable data, we do not create a new cluster following the grouping of outliers at a specific point and finally, the problem of non-normal distribution is fixed.

DEPLOYING

PYTHON

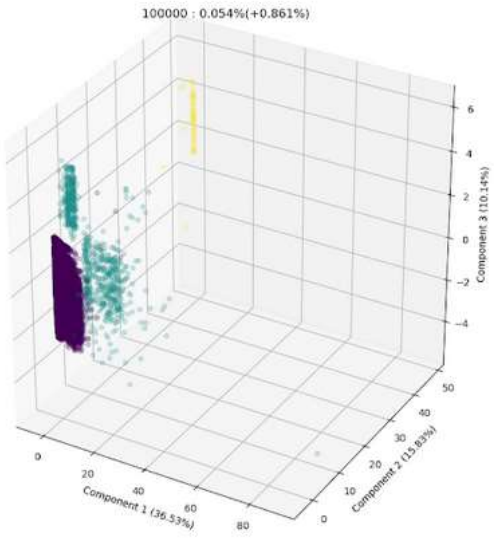
The first deployment was on the jupiter Notebook environment. A data set was loaded from the 2018 NY taxi Data data. To train the model we had opted for the DBSCAN algorithm (Density-based spatial clustering of applications with noise) which allowed us to determine the number of clusters and the data belonging to them on the basis of a few parameters. Initially, the choice was made for the K-means algorithm, but this choice was very much refuted because this algorithm, although fast, required a number of clusters, which we could provide.



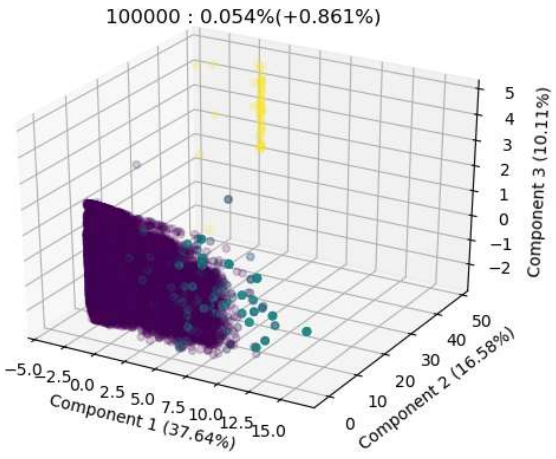
The dataset is thus initially driven by a rather limited number of data (in this example 10000). By playing on the parameters (specific to each data set) in order to limit the number of outliers, we have managed to define 3 classes: definitively inlier data, definitively erroneous data, and data whose character varies according to the parameter.

These 3 categories are therefore considered as inliers, errors and outliers.

The data set once sorted, we have a model on which we apply the k nearest neighbours to label the new data to be analyzed. Since this last algorithm is much faster than the previous one we have selected a much larger number of data to analyze (100000). Since the percentage of data considered as outliers and anomalies is similar to that of the initial model, we can assume that this is an effective model for determining outliers.



The last step is therefore to substitute the outlier data with the data from the nearest inliers and not to take into account errors. What is achieved in this last graph. The data representation was done under the PCA components in order to have the best visualization of the data in 3 dimensions.



PORTAL AZURE

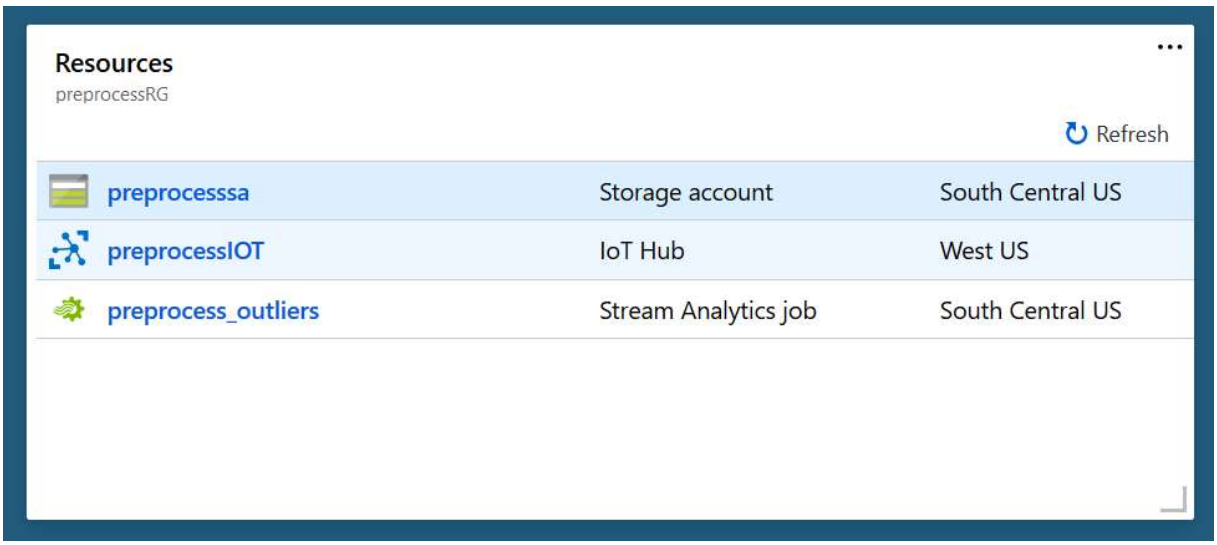
INTRODUCTION

Microsoft Azure is an open, flexible, enterprise-grade cloud computing platform. It offers a large set of tools which allows the user many different jobs.

In our case, we will only use a storage account to save the data, an IoT Hub to collect stream data, and Stream Analytics job to manipulate and detect outliers.

For this part, I used this tutorial: https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-quick-create-portal

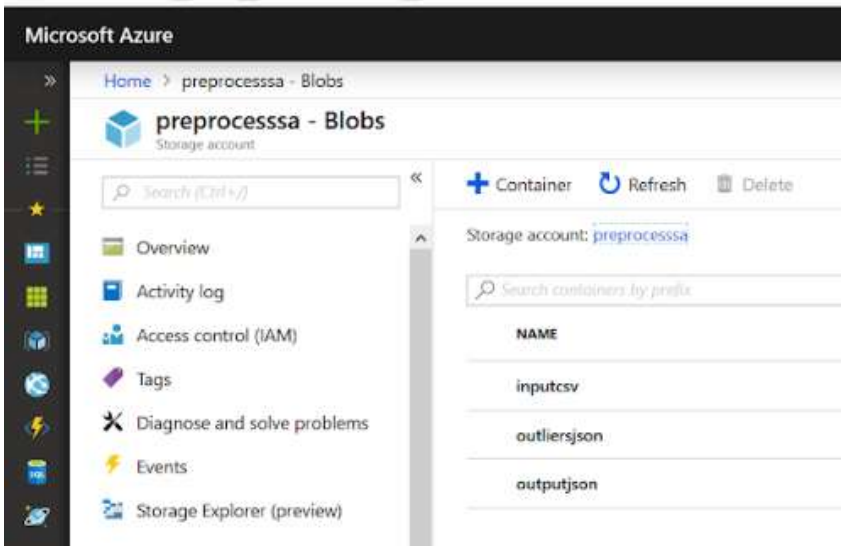
The final objective is to be able to continuously process incoming data and to save only the processed data. As a result, we needed a storage account to store the processed data, an IoT Hub to collect Stream data from different devices, and stream analytics job to work immediately as soon as data arrived in the IoT Hub



Storage Account

The storage account allows to save the data in various formats, blob, tables, queues, queues. For data coming out of a stream analytics job, two formats are possible: csv and json. These two formats can be saved in the blob storage which is only a text file backup. It is also the cheapest type of backup but has the disadvantage of being quite heavy to read.

The organization of the data storage was as follows: from the training phase a set of inliers data saved in the inputcsv, the data detected as outliers or anomalies during the stream analytics job are saved in the outliersjson (in case of error committed by the algorithm), and the outputjson includes the detected inliers as well as the modified outliers.




```
1 ,VendorID,tpep_pickup_datetime,tpep_dropoff_datetime,passenger_count,trip_distance,RatecodeID,store_and_fwd_flag,PULocationID,D
2 8699946,2,2017-01-28 20:55:27,2017-01-28 21:11:40,1,2.85,1,N,239,161,1,13.0,0.5,0.5,2.86,0.0,0.3,17.16
3 1039413,2,2017-01-17 17:26:40,2017-01-17 17:36:22,1,1.81,1,N,239,163,2,8.5,1.0,0.5,0.0,0.0,0.3,10.3
4 5630156,2,2017-01-12 08:42:20,2017-01-12 08:50:25,2,1.13,1,N,234,68,1,7.0,0.0,0.5,1.56,0.0,0.3,9.36
5 783635,1,2017-01-16 19:32:51,2017-01-16 19:38:52,1,1.4,1,N,164,113,2,6.5,0.0,0.5,0.0,0.0,0.3,7.3
6 7285489,1,2017-01-25 07:52:56,2017-01-25 07:55:08,1,0.3,1,N,161,161,2,3.5,0.0,0.5,0.0,0.0,0.3,4.3
7 4687264,2,2017-01-22 09:46:20,2017-01-22 09:50:14,1,0.95,1,N,162,237,2,5.5,0.0,0.5,0.0,0.0,0.3,6.3
8 9689581,1,2017-01-31 22:26:12,2017-01-31 22:39:43,1,1.7,1,N,186,161,1,10.5,0.5,0.5,2.35,0.0,0.3,14.15
9 5019797,1,2017-01-10 09:42:19,2017-01-10 09:53:59,1,1.4,1,Y,234,170,2,9.0,0.0,0.5,0.0,0.0,0.3,9.8
```

IoT Hub

The IoT hub is only there to have an address where to deliver the data. So after setting up the IoT Hub and recording the different devices that will send data to it, we are able to use the IoT hub as an input for the stream analytics job

Stream Analytics Job

The stream analytics job is actually a simple querry that calls a service deployed by Microsoft Machine Learning services and sends and receives data from the same service. In our case, the training of the model is done directly on the Machine Learning services platform (which we will discuss below). Detection is done via the first query written above where only ordinal digital data are included as a parameter for the service. The second querry allows to directly record the analyzed data as inliers in the outputjson. The 3rd querry saves the data considered as anomalies before they are

modified/removed from the dataset and the last querry selects only outliers and calls the outlier correction management service to then save the modified outliers in the outputjson.

```
WITH detection AS (
SELECT  VendorID,
        tpep_pickup_datetime,
        tpep_dropoff_datetime,
        RatecodeID,
        store_and_fwd_flag,
        PULocationID,
        DOLocationID,
        payment_type,
        detect(
            passenger_count,
            trip_distance,
            fare_amount,
            extra,mta_tax,
            tip_amount,
            tolls_amount,
            improvement_surcharge,
            total_amount
            ) AS copy
FROM inputcsv
)

SELECT *
INTO outputjson
FROM detection
WHERE copy.[Scored Labels] = '0';
```

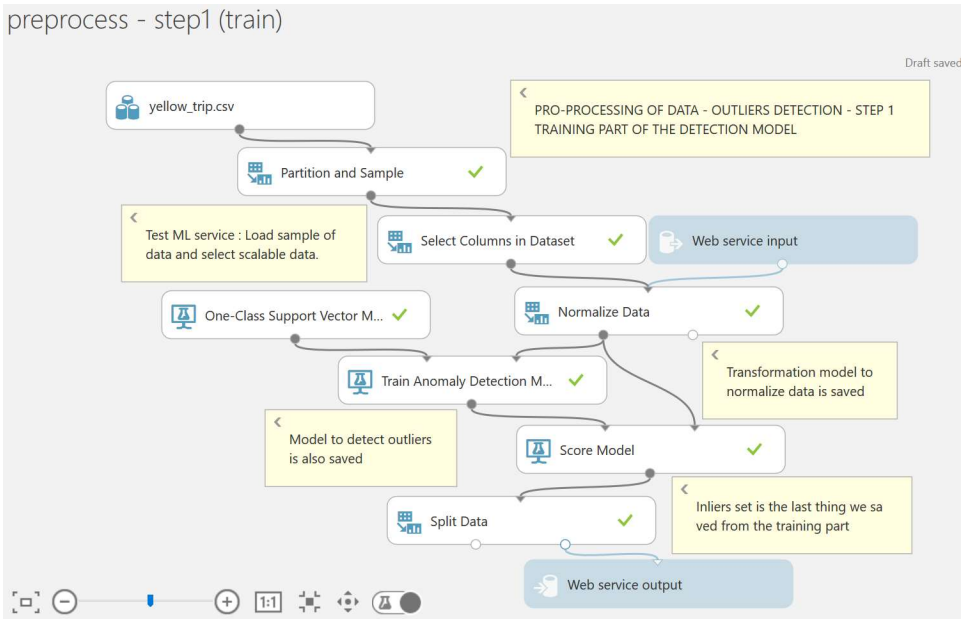
```
SELECT *
INTO outliersjson
FROM detection
WHERE copy.[Scored Labels] = '1';


SELECT  VendorID,
        tpep_pickup_datetime,
        tpep_dropoff_datetime,
        RatecodeID,
        store_and_fwd_flag,
        PULocationID,
        DOLocationID,
        payment_type,
        manage(
            copy.passenger_count,
            copy.trip_distance,
            copy.fare_amount,
            copy.extra,
            copy.mta_tax,
            copy.tip_amount,
            copy.tolls_amount,
            copy.improvement_surcharge,
            copy.total_amount,
            copy.[Score Label],
            copy.[Score Probabilities]
            ) AS modified
INTO modifiedjson
FROM detection
WHERE copy.[Scored Labels] = '1'
AND CAST(copy.[Scored Probabilities] AS FLOAT) < 30
```

MACHINE LEARNING SERVICES

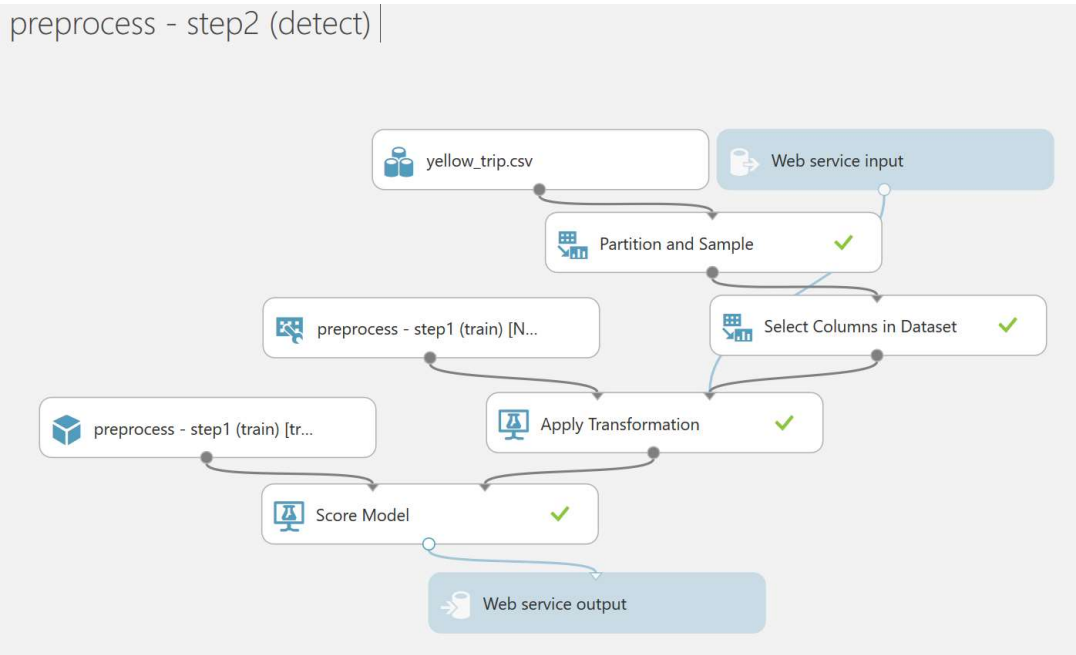
TRAIN

The data input is done via a csv file that is loaded and partitioned (ideally bagging is applied). The relevant columns are then selected, data standardization is applied to give equal weight to each column. Then, we train the model (here 1-class SVM) and apply the model to the dataset to obtain a set of inliers. This learning phase thus makes it possible to recover 3 important elements: the model allowing the detection of outliers, inliers and anomalies, a model to normalize the data (indeed, it is necessary to be able to normalize the data before applying the model) and finally, a set of inliers for the modification of outliers



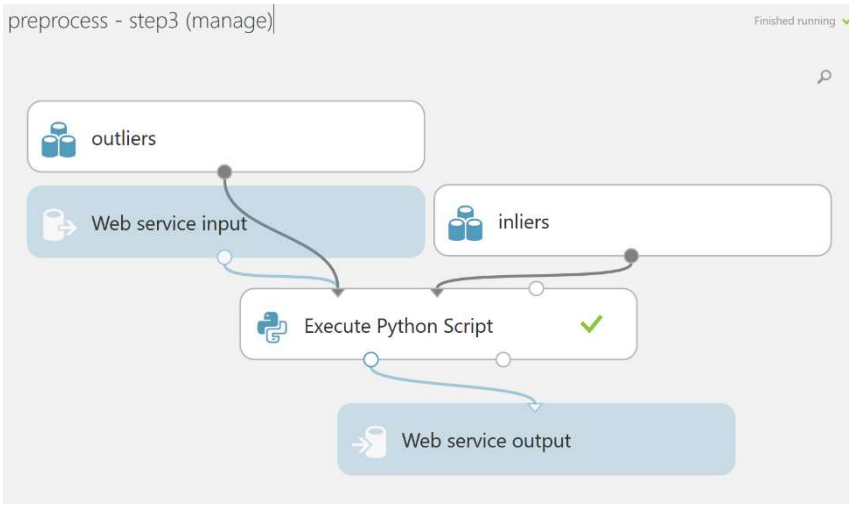
DETECT

For detection, the data are loaded, normalized, compared and finally evaluated by the model. This model thus adds a label 0 or 1 depending on whether the given data is considered as belonging to the class, or out of the normal and adds a probabilistic score giving an indication of its rate of certainty about the decision made.



MANAGE

Finally, the outlier set is loaded as well as the inlier set of the training set, a python code is executed to find the nearest inlier for each outlier, and the outlier data is substituted by the selected inlier data



Python script

```python
# The script MUST contain a function named azureml_main
# which is the entry point for this module.

# imports up here can be used to
import pandas as pd
import numpy as np

# The entry point function can contain up to two input arguments:
#   Param<dataframe1>: a pandas.DataFrame
#   Param<dataframe2>: a pandas.DataFrame
def azureml_main(dataframe1 = None, dataframe2 = None):
    for i, o in dataframe1.iterrows():
        d = np.sum((dataframe2 - o) ** 2, axis=1)
        dataframe1.iloc[i] = dataframe2.iloc[np.argmin(d)]
    # Return value must be of a sequence of pandas.DataFrame
    return dataframe1,

```

CONCLUSION

This 3-month internship allowed me to develop in various fields.

The first and, in my opinion, the most important, teamwork within a company. I had the opportunity to work in a team with another trainee with another background. Conversations and arguments were often informative, and the need to understand what you were doing was essential to get support from the rest of the team. Also every week we had the opportunity to have a moment with our manager to present him the evolution of our work. This moment was not only with the manager but with the entire analytics team in order to be able to have everyone's opinions and suggestions.

A second was to know the atmosphere within a company smaller than Euroclear's and also in full growth. from regular team meals to kick-off days bringing together all Avanade employees in one place for a full day and evening, many activities are organized to create team cohesion and try to create a family spirit within the company. In this way, even if the project can lead us to work alone at the client's premises without his colleagues next to him, we know that we can always count on office colleagues, a professional skype has even been set up to make it easy to find a colleague.

Finally, from a developer point of view, I have invested a lot of time and energy in machine learning and deployment in machine learning studio and Portal Azure. It was something very new for me and this internship was my first experience in this field. An internship, an objective, a solution.

Although the client's request (here Patrick Tack) has been met and the objective of the internship will have been fully met, some minor reservations can be noted, particularly regarding the personal objectives I had set for myself.

First of all, when I started learning by doing internet research and talking to Nina, many leads were started before they were abandoned because we were finding irrelevant solutions and our expertise in the field was too weak to predict the outcome in advance. Added to this are sometimes problems that you have to have encountered once in your life to never commit the fault again, and again, it took time to find the solution.

Then there is always that look between what you wanted to do and what you finally did. During our research, magnificent examples of machine learning services were presented, including bagging, competitive algorithm and paradox processing such as simpson's, and yet our solution remains relatively simple.

Finally, the complexity of Portal Azure once mastered, I also very quickly found its limits as well, probably because I hadn't found the backdoor way to get what I wanted. I am thinking in particular of the unnormalization of data that is not available in machine learning services, the obligation to use SQL queries when stream analytics jobs, etc.

But I remain with a good impression of the internship, and a great motivation to come back to the forefront after the studies, because certainly the objectives have not been up to my expectations, but I remain a young man difficult to fulfill and I am convinced that tomorrow cannot be worse than yesterday when it comes to programming

I would like to thank Patrick Tack for his availability throughout the internship. Living in Holland, he regularly travelled to Brussels to work in the offices with me and keep me company while I was struggling with technical problems.

I would also like to thank Nina, my internship partner with whom I was able to spend most of the internship discussing, arguing, learning, and finding the fault in what I thought was right. I think that this union has been more than beneficial and that a team of two will always be more effective than two teams of one.

Finally, I would like to thank Mr. Combéfis for his availability and the supervision of my internship.

SOURCES

- https://gunnarpeipman.com/iot/beer-iot-stream-analytics-sql/
- https://azure-samples.github.io/raspberry-pi-web-simulator/#Getstarted
- https://docs.microsoft.com/en-us/azure/iot-hub/quickstart-send-telemetry-python
- https://www.youtube.com/watch?v=mG4ZpEhRKHA (Detecting outliers and anomalies in realtime at Datadog - Homin Lee)
- https://www.youtube.com/watch?v=Z35tGNHmlD0 (Algorithms for Outlier Selection and One-Class Classification by Jeroen Janssens)
- https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/load-trained-model
- https://blog.aptitive.com/query-streamed-data-in-real-time-with-azure-stream-analytics-e88513494195
- https://medium.com/@ace139/azure-function-apps-with-python-a4621f944ba3