SOUTHEAST EUROPE JOURNAL OF SOFT COMPUTING
Available online at www.scjournal.com.ba

Out-of-stock problem: possible classification schemes

Haris Gavranović

[a] Faculty of Engineering and Natural Sciences, International University of Sarajevo, Hrasnička Cesta 15, 71210 Sarajevo, hgavranovic@ius.edu.ba

***Abstract*— An out-of-stock (OOS) event is referred as one of the biggest supply-chain management problem concerning retailers, distributors and consumers. We present available PCG data and discuss how to determine the importance of some features (fields), their interconnections and compare them with standard data fields used in other publicly accessible studies and recommendations from Efficient Consumer Response (ECR). We propose several models and algorithms to predict and solve Out of stock problem and at the end the computational results of these models are presented.**

Keywords—ECR, out-of-stock, supervised and semi-supervised machine learning, time series

1 INTRODUCTION

An out-of-stock (OOS) event is referred as one of the biggest supply-chain management problem concerning retailers, distributors and consumers. We say that an item is out-of-stock when a customer cannot find it on its usual place. The estimated rate of OOS is usually between 5 and 10 percent but for the fast moving items the rate could grow above that estimate easily. It is evident that the efficient detection of the OOS event presents great opportunity for improving overall retailers' business efficiency but also for distributors and manufacturers. The causes for OOS events are divided in the three categories and their estimated respective total shares are as follows:

- upstream causes 28 %
- in the store, not on the shelf 25 %
- store ordering and forecasting 47 %

This study concentrates on the items missing from the shelf but possibly (why possibly) available in the store. It could therefore potentially reduce more (why more) than 25 percent causes for OOS. Some authors call it out-of-shelf or shelf out-of-stock event. Consumer studies showed that the possible costumer reactions to the out-of-stock event ranged from "do not purchase an item" to "delay purchase" passing by "substitute-different brand", "buy item at another store" and "substitute-same brand". Typical retailer direct losses due to the OOS event are estimated to 4 percent of the sales (maybe, 4% sale decrease). Indirect losses are even bigger although the exact number is difficult to estimate.

Measuring out-of-stock problem is a problem on its own. Identifying OOS event through a physical audit is an expensive, time-consuming and finally unreliable exercise which cannot even help to measure the actual length of the OOS and therefore cannot estimate the total damage to the sales. On the other hand, physical count is the only fully reliable method to identify an OOS event without ambiguity. A large number of the different items in a typical store but also a sheer amount of the data about movements of these items in an observed store set a stage for automatic methods for identifying and predicting OOS events. Previous studies showed that these methods are often consistent with findings based on physical audit. We further examine these methods and propose several new models and assess their computational complexity as well as their precision and accuracy.

1.1 Previous studies

Standard demand modeling involves the demand and supply analysis and forecast, and predicts out-of-stock but cannot distinguish between an out-of-stock situation and the product that is simply not selling (explain this not clear). On the other hand, in the presence of promotions the product could easily goes out-of-stock while the demand model is again unable to predict it. However, this method is an acceptable choice for the fast moving consumer goods with relatively small volatility in the sales (Hausruckinger (2006) reports 41 such product). It would be helpful to implement and test this simple methodology and compare the obtained results with the results obtained by using more advanced algorithms.

Besides demand-supply models, several available public studies dealing with automatic identification of OOS events propose the use of the machine learning techniques. We continue along this path improving existing algorithms but also propose new approach with time series classification techniques.

1.2 Organization of the Paper

This paper is organized as follows. The second section present available PCG data and discuss how to determine the importance of some features (fields), their interconnections and compare them with standard data fields used in other publicly accessible studies and recommendations from Efficient Consumer Response (ECR). In the third section we propose several models and algorithms while the forth section presents computational results of these models on the Homeland data.

2 DEFINITION OF THE PROBLEM AND AVAILABLE DATA

Let us first formally define the problem at hand. An instance s is a set of data at the end of the day D for a given product P in a given store S. We are given a chronologically ordered list of instances $s_1, s_2,\ldots, s_N$ such that the set of features $f_1, f_2,\ldots, f_M$ is associated with each instance. The computational method should give an answer to the question: "Is the instance s_{N+1} an OOS or not?". Or we can rephrase it and ask: "Is the instance s_N at the end of the day an OOS event or not?". Whether an instance (on a particular day for the product P in the store S) is out-of-stock or not is not known in advance. Actually, in this framework, all along this study, there will be no physical audit to assess whether an item is in-stock or not. All information should be derived from the available database. We are giving one reduced example for the product SARA LEE CINN MINI BAGEL in the Homeland store number 238 in Edmond from a real world database.

Table 1 describes the set of instances used in our investigation.

We refer to Table 1 as an input table and the day following the last day in Table 1 as a target day. In our example in Table 1 the target day is 29th of January 2011. It seems natural to collect and investigate the data for a given product in a given store. However, it could be more appropriate for this study to consider a similar table containing the data for one product across all (or several similar stores) or a table comprises the data for a set of products from one category across one or several stores. Then, we would have more data in the input table and the computation would become more accurate One has to have in mind that empirical studies showed that OOS event patterns also depend on the day of the week, on the product category, on type of retail store etc. Nevertheless, the question will be always the same: "Is the instance s_{N+1} an OOS or not?".

We make several useful remarks here. It is clear that instances s_1, s_2 and s_5 are not out-of-stock as there was some sold items or (as in instance s5) there is returned item in the

store. On the other hand we cannot be sure if the instances s_3 and/or s_4 are out-of-stock or not. Although for some instances existing algorithms (Zero Scan Days and others) can determine with a high confidence (say above 90 percent) that they are out-of-stock. The set of instances for which it is known for sure or with high confidence whether they are OOS or not will be used as a train set in proposed supervised machine learning algorithms (C4.5 and Naive Bayes) or as a seminal set of positive instances for semi-supervised algorithms dealing with time series. At the same time, this set will allow to identify some of the false positive and some of the false negative results and assess the accuracy and precision of the proposed models.

Probably the most important problem when one tries to apply machine learning is selection of the pertinent features (or attributes) of the data. Often, the choice of the algorithm is of the secondary importance comparing with the selection of a suitable set of attributes. This important task will be carried out using Principal Component Analysis, computational results, guided with previous studies and in-house specific knowledge about the data.

Table 1 probably exhibits and explains all important features for the out-of-stock problem found in the database (neobična rečenica). We remark that this set of features is in great detail similar to the set of features used in (Papakiriakopoulos et al., 2009) studies on out-of-stock problems.

3 Models and Algorithms

In this paper, we propose several different models based on the machine learning techniques for detection an out-of-stock event from the data in the point-of-sale (POS) database. One can divide these models in two categories:

- when the instance to be classified is a single day with associated features
- when the instance to be classified is a sequence of consecutive days with associated features.

In both cases, our aim is to determine if the instance is in the class "next day presents an out-of-stock event" or in the class "next day does not present an out-of-stock event". A classifier in both cases is a programmable function that can decide to which class a given instance belongs.

3.1 Single Day is an Instance

In this framework an instance of the problem represent the data associated to a product in a store for a single day. Every feature is a single string or more often a single number. Some of these features, such as average sales, are derived data already present in the database or that could be easily calculated as it is case for the average duration between two consecutive sales. The principle that more available data for method training implies better classifier does not have to be true here. It is useful to define the sliding window for data and simply drop the oldest data. The optimal length of sliding window should be experimentally defined. This is not surprising because sales often have a seasonality and actuality.

We can consider this approach as a classical one already used, the author proposed the use of two classification algorithms, C4.5 and Naive Bayes, and found that C4.5 showed better overall performance. However, it is reported that this approach have two important shortfalls. The first one is a sensitivity of this approach to the class imbalance for the OOS events. It means that we should expect more accurate results in the stores with bigger OOS rates as the class imbalance decreases. The second shortfall is requirement for the good quality and completeness of the data used to represent or calculate features. This problem is even bigger in this approach as the use of the manual audit to train the classifier is not available. For obtaining the training set, we propose the use of the already known algorithms, such as zero scan, that classify only instances having high confidence indicator for either in or out the OOS class.

On the other hand, it would be computationally inexpensive to try and test other classification algorithms and to further develop ensemble type of classifier. Two studied classifiers have weaknesses and we should expect even weaker results due to the lack of manual audit. The study shows the presence of good instances for the case of increased out-of-stock rate and limited data sources.

3.2 Several Consecutive Days Represent an Instance

To our knowledge, there is no publicly available research that uses time series classification approach to tackle out-of-stock problem. On the other hand, the data such as replenishments, sales, amount of goods in stock are temporal discrete processes by their nature and OOS happens in the particular context and within dynamics of the given market, or retail. Considering longer time period than a single day can illuminate some of the OOS events unrecognizable on a daily basis but can also help us to discover new causes of OOS which would be even more important for the store management.

This idea develops further the approach that uses sales patterns to recognize an unexpected sales event (Hausruckinger, 2006). As it is already mentioned in this study some of the patterns appears only when specific days are analyzed, or just days with the promotions or some other more elaborated average than a simple one over a period of two weeks. While author in this study determines the OOS events based on the expected values for the product, such as amount of sales, special offer or not, registered stock etc., we propose to classify these pattern in automatic way. Classifying time series or sequence classification is relatively new but challenging approach used to tackle the problem such as genomic analysis, information retrieval, finance, abnormal detection (the credit card fraud detection or money laundering) and others. Another very important property of time series is that they can be plotted and events can be recognized and studied further.

Generally speaking, a time series is an ordered set of vectors. The vectors usually have a timestamp and the vectors are ordered with respect to the timestamp. In general, vectors can be numerical and/or nominal and can have more than one dimension in which case we speak about multivariate time series. The length of a time series can be constant, variable or even infinite. In our case, one vector represents a row in the

input table while an associated timestamp is a date in the row. An instance is a sequence of these vectors (days) which do not have to be consecutive (we can consider only Tuesdays in a given table or only the days with promotions). The optimal length of the time series for this study is yet to be defined and subject to further experimental studies but natural choices are a week, or month, or the time series comprises all days after the last replenishment etc. We would like to classify the instances according to the following question: "Is the last day an OOS event or not?" (or according to the question: "Is the first day after the end of the sequence an OOS event or not?").

Labeling instances and initial training set always a problem in the same way it was the case for the single day instances. Again, initial set of labeled instances would be produced using existing algorithms labeling only almost sure instances.

We need a distance to employ known methods to classify and cluster these instances. One of the standard, and the best, choice is Euclidean distance defined between two one-dimensional time series of length n as a square root of the sum of the squared differences between each corresponding pair of data points.

This distance has a meaning only if each time series is normalized to have mean zero and a standard deviation one. It is possible also to define and test other standard distances such as Manhattan, L-, etc. To deal with several dimensions we have three possible choices:

1. throw away all but one dimension
2. add the dimensions (only if they are the same units)
3. use Principal Component Analysis (PCA) to combine dimensions

Once we have defined distance we can use some existing classification methods.

3.2.1 Semi-Supervised Learning

It is obvious that in the store with 100000 items, even with manual audit help, there will be much more unlabeled than labeled data. Some unlabeled data can share common features with labeled data and it would be advantageous to use these additional data to build a better classifier. The whole procedure would be divided in several phases:

1. Define (using Zero Scan Days or other) initial, possible small set of labeled instances
2. Label additional easy instances in the past and add them to the training set
3. Classify the last day

3.2.2 Motif Finding

Instead of looking for the two classes, one can also try to find irregularities in the data as a signal for out-of-stock (could finish also with overstock signal which is also irregularity).

4 COMPUTATIONAL RESULTS

4.1 Description of Available Test Data

The computational study uses two different types of data. These data differs by their origins (store and local market) and the way they are labeled (OOS or EXISTS). The first type of data is referred to as Data2004 are obtained from (Papakiriakopoulos et al 2006) and was used in their computational studies. The exceptional value of this data is the fact that the labels are obtained through painful and time-consuming process of physical audit. On the other hand, the labels obtained through this process are the most reliable ones. For the sake of this study we reduce the set of attributes in Data2004 to the most pertinent, available attributes in the existing, target database (Homeland): day $\{Mon, Tues, Wedn, Thur, Frid, Sat, Sund\}$, pos, average total, StD total, daily average, daily StD, lastzeroseq, zeros avg, zeros StD and the class label OOS, EXISTS. The names of these attributes are self-explanatory and the period for which the average and the standard deviation are calculated needs to be precisely defined. The beginning of .arff file associated to Data2004 for Retail1 is given in the appendix. (gdje je appendix)

The data present in Homeland database allows us to calculate all mentioned attributes. Although the size of the whole database is considerable and somewhat complex the data necessary for this study can be found in the table *sum product store daily*. There are 881 different products and 72 different stores where these products are sold. For this initial study, the products are independent as well as the stores (the OOS for one type of milk does not influence OOS event for another type of milk). There are 24055 different product-store combinations with some sort of data. Some of these combinations (or views) are useless for this study (just few recorded sales, the most of recorded sales are zero, etc.). We continue only with product-store combination that has more than 30 non-zero sales and that has more than 100 recorded sales (zero and others). There are 4159 such product-store combinations. The data extracted from the database, for a given product and a given store, are the transaction date, sold quantity and inventory quantity while all others attributes are derived in more or less obvious way. There are several types of average (and daily average) and therefore several types of standard deviation

1. total average so far zero sales excluded
2. total average so far zero sales included
3. total average for last four weeks zero sales included
4. total average for last thirteen weeks zero sales included
5. total average for last twenty six weeks zero sales included

The corresponding standard deviation and daily average are also calculated and we have 5 different types of data with respect to the type of the average.

The OOS label for the homeland data is one of the obstacles for the direct application of the machine learning methods for out-of-stock problem. Three different OOS indicators are calculated with different strength and combined to decide about the OOS label for a given product in a given store on a given day. All these indicators are based on the assumption that the quantities are regular with some probability distribution and when some quantity differs more than a standard deviation (or twice the standard deviation) from corresponding average than with high probability we can conclude that the product is in irregular situation (for example

out-of-stock event). For example, the simple OOS indicator for the a given product on a given day is given by the following formula (with respect to 13 weeks average and the associated standard deviation)

$$sold\_qty[1] > avg\_13[i] - timesStD \cdot StD\_avg\_13[i] \quad (1)$$

$$sold\_qty[1] < avg\_13[i] - timesStD \cdot StD\_avg\_13[i] \quad (2)$$

$$indicator = \begin{cases} true, & if\ (1) \\ false, & if\ (2) \end{cases}$$

The variable *timesStD* is a parameter that determines the quality of estimate and reasonable values to consider are 1 and 2. The similar formula is used for daily quantities with associated averages and standard deviation. The attribute zero scan number of days (consecutive days without sold with associated average etc.) gives rise to the third type of the indicator. This indicator is calculated only for whole sequence average and standard deviation.

These three basic indicators can be combined in 7 different ways and will produce seven different (more or less strong) composite indicators for OOS event (the strongest indicator would be the one that is confirmed by all three basic ones). The 7 composite indicators can have in this study two different strengths, when $timesStD = 1$ and when $timesStD = 2$. Here is the list of 7 composite OOS indicators:

1. *BelowAvgDOWStD*
2. *ZeroScanStD*
3. *BelowAvgDOWStD* and *ZeroScanStD*
4. *BelowAvgStD*
5. *BelowAvgStD* and *BelowAvgDOWStD*
6. *BelowAvgStD* and *ZeroScanStD*
7. *BelowAvgDOWStD* and *ZeroScanStD* and *BelowAvgStD*

Table 1 and Table 2 give the number of OOS event found using these 7 indicators for five types of statistics. Table 1 presents results when the parameter *timesStD* takes value 1 while the second table presents results when the parameter *timesStD* takes value 2.

The last columns in these two tables report the number of OOS events calculated using the sum of the numerical counterparts of the three indicators. For example, the numerical indicator associated to the simple sold quantity average is calculated using the following formula:

$$num\_ind = \frac{avg\_13[i] - sold\_qty[i]}{StD\_avg\_13[i]} \quad (3)$$

If the sum of the three numerical indicators is bigger than a given thresholds than the item in the corresponding day is considered as an OOS item. We use the values of 1 and 2 as possible thresholds.

4.2 Evaluation of Standard Classifiers

In Table 4, 5, 6 and Table 7, we report the results obtained using three standard classifiers: C45, Naive Bayes, Neural Network and Bayes AODE. Table 8 shows the chosen results from Tables 4, 5, 6 and 7. We test these classifiers with different possible scenarios and report the number of false positive OOS and false negative OOS rates with respect to the total number of OOS events. The results indicate the importance to continue with the tests with other classifiers and the necessity to fine tune the algorithm parameters in order to obtain even better results.

All tests and data manipulation are developed from scratch using Java and Perl programming language. The whole computation, starting with database Homeland, would take approximately 24 hours on the standard i7 processors with 6 GB of RAM. The computation may become an issue with greater number of products and stores. There is still a lot of room for computational improvements and this was not our priority in this study. The classification was effectuated using Weka open source toolkit and therefore it would be possible to produce better and faster results if the need shows up.

A simple improvement would be to allow the classifiers to be trained also while they are tested against test instances. As the number of test instances is not small and these instances are the closest one to the last instances it is safe bet that the results would be slightly better than those obtained with simple train-and-test method.

5 CONCLUSIONS

In this study, we gave an overview of existing literature and computation approaches to tackle the OOS problem. We also propose several possible new approaches and set a basis for the beginning of more complete computational study. We effectively implemented and tested several standard classifiers on the Homeland database. The challenge would be to test other existing classifiers as well, find out their weaknesses and strengths, define the best parameters and eventually combine them in one superior classifier. The obtained results are promising but only the practice can give definite answer to the questions about the quality of the approach.

6 REFERENCES

Anupindi, R., Dada, M. and Gupta,S. (1998) Estimation of Consumer Demand with Stock out based Substitution: An Application to Vending Machine Products, Marketing Science, 17: 406-423.

Campo,K. Gijsbrechts,E. and Nisol,P. (2000) Towards understanding consumer response to stock-outs. Journal of Retailing, 76.2: 219-242.

Clark, T., and Lee, H. (2000) Performance interdependence and coordination in business-to business electronic commerce and supply chain management, Information Technology and Management, 1.1: 85-105.

Colacchio, F., Tikhonova, O., Kisis, J. (2003) Consumer Response to Out-Of-Stock: Decision making process and influencing factors. ECR Europe Conference, Berlin.

Corstjens, J., and Corstjens,M. (1999) Store wars: the battle for mindspace and shelfspace, Wiley, New York

Desmet, P., Renaudin.V (1998). Estimation of product category sales responsiveness to allocated shelf space. International Journal of. Marketing Research, 15: 443-457.

Emmelhainz, M., Emmelhainz, L., Stock, J. (1991) Consumer Responses to Retail Stock-outs, Journal of Retailing, 67.2: 138–147.

Gruen, T.W., Corsten, D.S., Bharadwaj, S. (2002) Retail Out-of-Stocks: A Worldwide examination of Extent Causes and Consumer Responses. The Food Institute Forum.

Hausruckinger, G., (2006) Approaches to measuring on-shelf availability at the point of sale, ECR Europe.

Papakiriakopoulos Dimitrios, Pramatari Katerin, Doukidis Georgios (2009) A decision support system for detecting products missing from the shelf based on heuristic rules. Decision Support Systems, 44: 685–694.

Roland Berger (2002) Full-Shelf Satisfaction. Reducing out-of-stocks in the grocery channel. Grocery Manufactuers of America (GMA).

Roland Berger (2003) ECR-Optimal Shelf Availability. ECR Europe.

Vuyk, C. (2003) Out-of-Stocks: A Nightmare for Retailer and Supplier. Beverage World.

Yang, M. (2001) An efficient algorithm to allocate shelf space. European Journal of Operational Research, 131: 107-118.

Ian H. Witten, Frank Eibe, Hall A. Mark (2011) Data Mining Practical Machine Learning Tools and Techniques. Elsevier.

Instance	Store id	Trans date	Sold qty	Returned qty	Received qty	Inventory qty
s_1	238	2011-01-24	2	0	5	3%
s_2	238	2011-01-24	1	0	0	2%
s_3	238	2011-01-24	0	0	0	2%
s_4	238	2011-01-24	0	0	1	3%
s_5	238	2011-01-24	0	1	0	2%

Table 1 The table describes the set of instances we used for the experimentation: their names

Avg type	1	2	3	4	5	6	7	Fuzzy (1,0)
ZeroExcl	709014	46934	15440	719783	689497	20145	14664	480384
ZeroIncl	709014	46934	15440	36385	33932	28	28	334120
avg_4	52167	46934	569	46047	19602	311	103	42031
avg_13	44673	46934	32	31311	20489	2	1	32994
avg_26	24230	46934	0	17536	12632	0	0	30320

Table 2 Number of OOS events for different indicators when timesStD takes value 1: The last column represents total number product-store-day instances. The first column represents number of OOS events based on Below Avg minus StD on the DOWcriteria, the second column represents number of OOS events based on Zero Scan statistics while the third one is a combination of thirst two. The forth column represents number of OOS events based on the Below Avg minus StD criteria. The fifth, sixth and seventh columns are binary combination of previous columns.

Avg type	1	2	3	4	5	6	7	Fuzzy (2,0)
ZeroExcl	5107	0	0	4458	725	0	0	0
ZeroIncl	6106	0	0	2186	342	0	0	0
avg_4	0	0	0	502	0	0	0	0
avg_13	508	0	0	348	131	0	0	0
avg_26	290	0	0	141	87	0	0	0

Table 3 Number of OOS events for different indicators when timesStD takes value 2: The last column represents total number product-store-day instances. The first column represents number of OOS events based on Below Avg minus StD on the DOWcriteria, the second column represents number of OOS events based on Zero Scan statistics while the third one is a combination of thirst two. The forth column represents number of OOS events based on the Below Avg minus StD criteria. The fifth, sixth and seventh columns are binary combination of previous columns.

Avg type	1		2		3		4		5		6		7		Fuzzy (2,0)	
ZeroExcl	0.120	0.019	0.074	0.617	0.272	0.772	0.044	0.024	0.057	0.741	0.164	0.000	0.213	0.000	0.045	0.212
ZeroIncl	0.119	0.020	0.076	0.583	0.248	0.731	0.632	0.100	0.651	0.110	1.000	0.000	1.000	0.000	0.209	0.344
avg_4	0.740	0.719	0.062	0.744	0.885	0.365	0.515	0.391	0.673	0.000	0.989	0.000	1.000	0.000	0.588	0.482
avg_13	0.593	0.630	0.066	0.619	1.000	0.053	0.437	0.253	0.539	0.028	1.000	0.000	1.000	0.000	0.489	0.367
avg_26	0.514	0.456	0.094	0.598	0.000	0.000	0.309	0.216	0.406	0.019	0.000	0.000	0.000	0.000	0.574	0.378

Table 4 Table represents false positive and false negative percentage with respect to the total number of OOS events when classifier **C45** was used and trained for specific product and specific store on the specific day: The explanation for the columns is the same as in all other tables. The total number of OOS events is considerably smaller than the total number of instances. Calculating the percentage of false positive and false negative with respect to total number of instances would always give only few percent as a result.

Avg type	1		2		3		4		5		6		7		Fuzzy (2,0)	
ZeroExcl	0.099	0.007	0.058	0.293	0.214	0.386	0.015	0.006	0.029	0.741	0.074	0.000	0.138	0.000	0.045	0.212
ZeroIncl	0.099	0.007	0.053	0.289	0.221	0.363	0.533	0.031	0.558	0.110	1.000	0.000	1.000	0.000	0.069	0.352
avg_4	0.818	0.238	0.056	0.348	0.962	0.183	0.400	0.429	0.760	0.000	0.898	0.000	1.000	0.000	0.517	0.323
avg_13	0.612	0.245	0.042	0.312	1.000	0.211	0.299	0.171	0.468	0.028	1.000	0.000	1.000	0.000	0.408	0.204
avg_26	0.446	0.229	0.046	0.315	0.000	0.000	0.137	0.119	0.239	0.019	1.000	0.000	0.000	0.000	0.414	0.205

Table 5 Table represents false positive and false negative percentage with respect to the total number of OOS events when classifier **Naive Bayes** was used and trained for specific product and specific store on the specific day: The explanation for the columns is the same as in all other tables. The total number of OOS events is considerably smaller than the total number of instances. Calculating the percentage of false positive and false negative with respect to total number of instances would always give only few percent as a result.

Avg type	1		2		3		4		5		6		7		Fuzzy (2,0)	
ZeroExcl	0.099	0.007	0.058	0.293	0.214	0.386	0.015	0.006	0.029	0.741	0.074	0.000	0.138	0.000	0.045	0.212
ZeroIncl	0.099	0.007	0.053	0.289	0.221	0.363	0.533	0.031	0.558	0.110	1.000	0.000	1.000	0.000	0.069	0.352
avg_4	0.818	0.238	0.056	0.348	0.962	0.183	0.400	0.429	0.760	0.000	0.898	0.000	1.000	0.000	0.517	0.323
avg_13	0.612	0.245	0.042	0.312	1.000	0.211	0.299	0.171	0.468	0.028	1.000	0.000	1.000	0.000	0.408	0.204
avg_26	0.446	0.229	0.046	0.315	0.000	0.000	0.137	0.119	0.239	0.019	1.000	0.000	0.000	0.000	0.414	0.205

Table 6 Table represents false positive and false negative percentage with respect to the total number of OOS events when classifier **Neural Networks** was used and trained for specific product and specific store on the specific day: The explanation for the columns is the same as in all other tables. The total number of OOS events is considerably smaller than the total number of instances. Calculating the percentage of false positive and false negative with respect to total number of instances would always give only few percent as a result.

Avg type	1		2		3		4		5		6		7		Fuzzy (2,0)	
ZeroExcl	0.160	0.061	0.163	0.446	0.230	0.571	0.102	0.071	0.105	0.741	0.134	0.000	0.178	0.000	0.113	0.346
ZeroIncl	0.156	0.061	0.163	0.433	0.236	0.573	0.765	0.105	0.774	0.110	1.000	0.000	1.000	0.000	0.185	0.627
avg_4	0.787	0.471	0.068	0.592	0.865	0.101	0.541	0.443	0.770	0.000	0.886	0.000	0.950	0.000	0.563	0.493
avg_13	0.642	0.364	0.110	0.446	0.947	0.421	0.552	0.265	0.656	0.028	1.000	0.000	1.000	0.000	0.507	0.406
avg_26	0.590	0.318	0.129	0.405	0.000	0.170	0.528	0.251	0.613	0.019	0.000	0.000	0.000	0.000	0.578	0.391

Table 7 Table represents false positive and false negative percentage with respect to the total number of OOS events when classifier **Bayes AODE** was used and trained for specific product and specific store on the specific day: The explanation for the columns is the same as in all other tables. The total number of OOS events is considerably smaller than the total number of instances. Calculating the percentage of false positive and false negative with respect to total number of instances would always give only few percent as a result.

Avg type	Avg_4(5)		Avg_26(5)		Fuzzy (2.)(13)	
C45	0.760	0.000	0.239	0.019	0.211	0.299
Naïve Bayes	0.673	0.000	0.406	0.019	0.053	0.437
Neural Networks	0.666	0.000	0.351	0.019	0.421	0.629
Bayes AODE	0.770	0.000	0.613	0.019	0.507	0.406

Table 8 In this table we exctract some of the results from the previous tables to compare numerically different classifiers. For the moment, this could be the performance order for these classifiers.