Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

OPTIMIZING SMART RETAIL BY EXPERIMENT USING AN ONLINE AI MODEL EXPLORATION INTERFACE

Wenfei Huang[1], Matthias Melzer[1], Jan Dünnweber[1]

Faculty of Computer Science and Mathematics, OTH Regensburg,
Technical University of Applied Sciences, 93053 Regensburg, Bavaria, GERMANY[1]

ABSTRACT

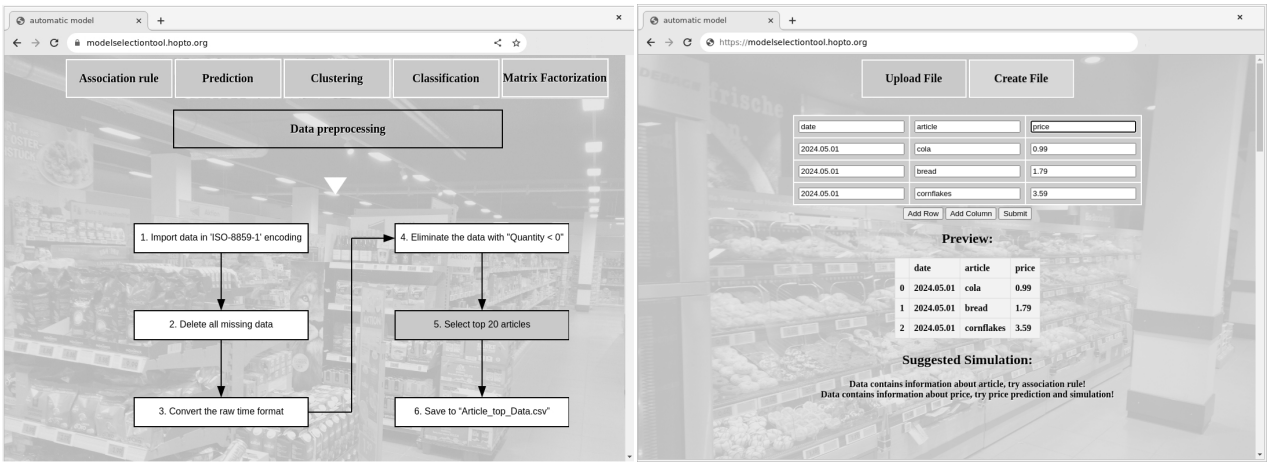
Smart retail technologies save grocery store operators a lot of work. At the same time, these technologies produce valuable data for building sustainable and economical inventory management strategies. AI models can be trained for sales forecasting using the data. The forecasts support the provisioning of fresh food over the whole week and help reducing food waste. In this paper, we present a Web portal which we developed to allow grocery store operators experiments with AI models revealing interrelations between observed and anticipated customer behavior. Clickable diagrams facilitate the exploration of data sets combining historical data and synthetically generated data. Pricing and ordering can be adapted accordingly to the simulated forecasts. By means of a case study, we show that our simulations are not only useful for predicting future sales but for other smart retail tasks as well.

1 INTRODUCTION

Business intelligence has gone through a remarkable revolution from early On-Line Analytical Processing (OLAP) systems (Codd and Salley 1993) to combinations of Internet-of-Things (IoT), computer vision and AI for customer tracking (Gonzales et al. 2024) over the last decades. However, the AI still cannot take the role of a business consultant who understands arbitrary sales data and can work out useful advice from them, independently of a particular goal specified by the user. Machine learning (ML) is a fixed-purpose technique, in the sense that one needs to know what learning algorithm can be fed with what training data before a model is created for classifying, grouping or extending the data gathered from product sales. Therefore, a notion of the necessary steps for training an ML model has become essential in modern trading. Nowadays, the techniques originating from e-commerce, such as data mining, are commonly used by retail store and grocery chain managers. Being able to understand interrelations in the data, the store management can adapt marketing and procurement strategies according to the patterns behind the data and increase their profit. The patterns that an ML model can discover are, e. g., association rules in transactions. These rules determine that certain commodities are often bought together. One logical action based on this observation is placing such item groups on the same shelf or binding them for sales promotion. Reordering strategies can benefit in the same manner, when store managers make sure that their groceries are sold before they exceed the expiry dates.

The particular ML model used for retail data analysis can be chosen by picking one from a machine learning library, which is theoretically favorable for time series and which has practically led to success in a different domain before. DBScan, e. g., has produced valuable insights in checking astronomical data for anomalies (Joncour et al. 2019). However, when the procedure is transferred to smart retail, one disregards that the contents of a shopping cart – contrary to interstellar clouds – are composed with human intention and the risk of neglecting relationships between the goods bought, which an intent-aware clustering procedure could have revealed (Forman, Nachlieli, and Keshet 2015), is high.

In this paper, we propose to base the selection of an AI model for smart retail on experiments with the data to be interpreted itself, rather than on theory.



(a) The clickable flow diagram

(b) Example preview and AI model suggestion

Figure 1: The AI model exploration portal: `modelselectiontool.hopto.org`

To allow for an investigative evaluation of the potentials of various AI models – ranging from traditional K-means to modern recommender techniques – we implemented a planning simulation with an online interface. Following the idea, that *simulation, experimentation, analytics and testing* (SEAT) should be based on expandable and loosely coupled components (Harrell et al. 2020), we built a serverless architecture and deployed it to the AWS cloud (Kumar, Pooja, and Kumar 2021). Running interactive simulations is possible on our online portal which is currently available on `https://modelselectiontool.hopto.org`. The starting page visualizes the essential steps in time series processing in the form of a clickable flow diagram (See Fig. 1a). The first step is the algorithm selection. Users can choose to either upload sales data from their computers or generate synthetic data for an experiment by clicking on the "`Create File`" button. Once the rows and columns of the sales data table are filled, users can submit it to run a simulation in a Jupyter notebook. In the example (Fig. 1b) above, the portal suggests to proceed with *association rule* and *price simulation* experiments (explained in detail in Section 3 and Section 4), since the data provided by the user contains the article name and price information. Once the users has selected the type of AI model to experiment with, the input is preprocessed and simulation results are computed. Our easily accessible portal is mainly targeted towards users who need to pick a suitable methodology for a smart retail application and who are overwhelmed by the large supply of possible strategies supported by modern AI libraries and frameworks. For future versions, we are planning to support altering the activation function and exploring other hyperparameter optimizations (Lars Hertel and Gillen 2022) to make the simulations also useful for AI methodology designers beyond the smart retail domain.

The rest of this work is structured as follows: In Section 2 of the paper we review related work. We explain association rule learning for grocery store data in Section 3. The use of prediction models for sales forecasting and price simulations are explained in Section 4. The clustering of stores is shown in Section 5. In Section 6 we describe how we label data in our smart retail simulation for using the classification model. In Section 7, we further discuss the potential of using a recommender algorithm for interpreting sales data. All the sections on the different AI models follow the same subsection-structure (overview – preprocessing – experiments), such that the models can be systematically compared and our experiments can be easily reproduced. We summarize and conclude our work in Section 8.

2 RELATED WORK

The use of ML in *promotional planning* has been studied before (O. Olayemi and C. Olayemi 2017). It has been shown that a promotional strategy that is useful for a single store can be found, based on an association rule analysis comprising no more than 3000 transaction records. In contrast, we cooperate with a grocery store chain that is concerned with the analysis of millions of data records from all their stores in Bavaria (Germany), where we identify general relationships between product sales, which are valid for all the analyzed sales locations.

Sales development *forecasting models* for grocery stores, such as the autoregressive integrated moving average, were studied previously using publicly available data from `kaggle.com` (Kumar Jha and Pande 2021). Our simulation interface allows data scientists to switch between multiple data sources, such that input comprising historical data, live data, synthetically generated data and combinations thereof are available for interactive experiments. Article prices can be raised or lowered for any date in the past and a comparison of the predicted sales development and the actually recorded sales quantities (including "live" data) can be depicted in a sales curves diagram. The forecasting model we employ takes account for multiple different impact factors: Besides prices, e. g., the store locations and weekdays are considered.

Another AI model provided by our simulation portal supports *data clustering*. Here, we benefit from the fact that in smart retail, every data entry in the input time series relates to a purchase event. If, otherwise, the measurements from a continuously running process, such as electricity flow, are clustered, a crucial step is the discretization of the input time series. This step can be accomplished via an elaborate process using vectors mapping between observations and events (Melzer, Dünnweber, and Baumann 2022). Contrary, in the simulation portal which we developed for smart retail and show in this paper, the clustering procedures can be applied to the raw input data without a preliminary disaggregation.

Unlabeled data can also be grouped in our simulation portal, allowing to explore interrelations between different stores via a *classification* model. When items are unlabeled, the grouping must be based on alternative information. In this work, we present simulations using time and position information for store classifications. Recognizing single products via a classification model has been analyzed before using photographs from grocery stores (Klasson, Zhang, and Kjellström 2020). However, for a sales analysis based on photographs, an image of every purchase would be required, which is typically not available.

In the matrix factorization (MF) section, our portal allows to experiment with single value decomposition (SVD), collaborative filtering (CF) and the K-Nearest Neighbour (KNN) algorithm. Combining these *recommender system* methods has been proven useful for rating movies according to viever preferences (Patra and Ganguly 2019). We use them to estimate how likely it is that a customer will buy a certain product.

Aside from research on data mining for sales data, more general work on simulating the behavior of AI models should also be mentioned in connection with the work presented in this paper: Instead of a clickable flow diagram, other researchers set up a platform that can run simulations of AI models which users describe using domain-specific languages and/or XML-based specifications (Gore, Diallo, and Padilla 2014). Observing an AI processing a task, such as solving a puzzle, helps with teaching data science students the working principles of the underlying AI model (Roberts, Mastorakis, Lazaruk, Franco, Stokes, and Bernardini 2021). For clinical applications, similar tools have been developed, which enable the visualization of an AI model and, at the same time, allow researchers to cooperate in exploring vast data quantities and enhancing the model (Gorre, Carranza, Fuhrman, Li, Madduri, Giger, and El Naqa 2023). When grocery store operators and data scientists access our portal via a Web browser for cooperating on sales data interpretation, they face the challenge that all experiments are based solely on information about the sold products and the shops. A lot of useful information, such as historical customer data, which online stores can collect to interpret over years, is not available. However, it is known that business investigations are possible starting with limited data using a *toy modeling* technique (Zellner, Milz, Lyons, Hoch, and Radinsky 2022). Similarly, our simulation interface allows users gearing up their experiments, when, e. g., a customer bonus card program is evaluated and more detailed customer information becomes available over time.

3 SIMULATION OF THE ASSOCIATION RULE MODEL

Association rules are used for static data analysis. Applying association rules on supermarket transactions in our simulation portal helps finding items that often appear together in transactions.

3.1 Algorithm overview

Association Rules infer whether a subset of all items will affect the presence of another subset. Formally, the implication $A \rightarrow B$ $(A, B \subseteq M)$ means that if A holds, then B holds as well.

Transferred to grocery stores $A \rightarrow B$ means, consumers who buy item A will also buy item B. *Support* and *confidence* are measures of association rules. Support illustrates the proportion of the number of transactions with the same items in the total number of transactions, while confidence is described as the frequency of the transactions with same items compared to transactions containing the preceding item.

$$Support(A,B) = \frac{No.\,of\,transactions\,containing\,both\,A\,and\,B}{Total\,No.\,of\,transactions}, \tag{1}$$

$$Confidence(\frac{B}{A}) = \frac{No.\,of\,transactions\,containing\,both\,A\,and\,B}{No.\,of\,transactions\,containing\,A}. \tag{2}$$

An example is the result from recent research on sales transactions (Santoso 2021):

$$support(diapers, soap) = 0.26 \quad \text{and} \quad confidence(\frac{soap}{diapers}) = 0.8.$$

That means, 26 % of all transactions contain both, diapers and soap, and in 80 % of all transactions containing diaper, soap is also bought. Using our simulation interface, such an a-posteriori analysis of the model is already possible after a portion of the data recorded in the grocery store has been processed.

3.2 Data preprocessing

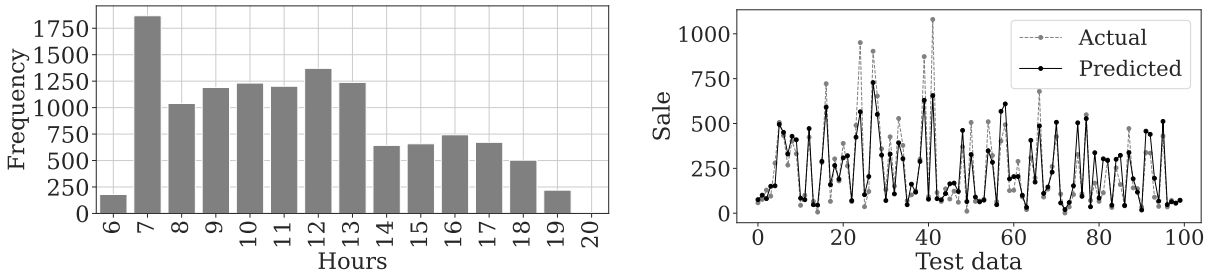
To obtain reliable results, we apply data cleansing methods to the sales data: We group products which have different names but are basically the same. For example, we group all different types of bread rolls (four in our example data) together. Moreover, we delete all deposit goods (yogurt jars and other returnable packagings) from the data, avoiding the need to deal with negative sales quantities.

In the middle (right) of the flow diagram in Figure 1a box 5 is grayed out, indicating that we modified the quantity selection parameter before taking the screenshot. By clicking on "top 20", one can adapt the value in intervals of ten, if, e. g., only the 10 best-selling articles are considered relevant for the next simulation.

An additional preprocessing step, which we apply to the input before we perform an association rule analysis, is breaking up two-item transactions into individual purchases. Thereby, we avoid results wherein random combinations of two items, such as bananas and cucumbers, appear as a shopping trend. Real customer behavior can be found by restricting the analysis of "multi-item purchases" to three- or four-item transactions, or by requiring diversity, i. e., considering only purchases combining different product categories. Using such selections rules, our simulation interface helps uncovering subgroups within the "multi-item purchases".

3.3 Experiments

Because of the repetitions in our data, we set the parameter *Support* to 0.25 and *Confidence* to 0.5, leading to a high reliability. The data is then processed using the *efficient apriori* algorithm (Delos Arcos and Hernandez 2019).



(a) Soft prezel sales by time period

(b) Random forest prediction on article soft prezels

Figure 2: Grocery store sales predictions

Using the association rule simulation, we found recurring types of particular "multi-item purchases": purchases containing diapers or baby food, which we label "family purchases" and purchases containing articles like tools and assembly lube, which we label "handyman purchases".

As it becomes transparent from Fig. 2a, in our association rule experiments, we observed that the sales of *soft pretzels* were concentrated around 7 am in our data. By facilitating such observations, our simulation interface can help reducing food waste, when more soft pretzels are prepared in the morning, accordingly.

4 SIMULATION OF SALES PREDICTION

Prediction is a machine learning technique used for forecasting a future state based on historical data. An interactive simulation of sales prediction can be run in our simulation interface (Fig. 1a) after clicking on the second box above the flow diagram. Contrary to association rules, which are observable in static data sets, prediction is a dynamic time series analysis technique and, therefore, of particular interest for smart retail applications.

4.1 Algorithm overview

A regression model determines the strength of the relationship between a dependent variable and a series of independent variables, which are the factors impacting the future values. Our online simulation interface allows to process smart retail data using the five most popular regression models.

1. Lasso regression is a modified linear regression model. In our simulation, we evaluate the use of Lasso for sales predication and also for predicting the customers' length of stay.
2. A multi-layer Perceptron (MLP) is a fully connected feed-forward artificial neural network with at least three layers (input, output, and at least one hidden layer). The input layer has several neurons $\{x_i|x_1,x_2...x_m\}$, corresponding to the m independent variables in the Lasso model. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1 + w_2x_2 + ... + w_mx_m$. The output layer receives the values from the last hidden layer.
3. A Random Forest is one of the averaging algorithm based on randomized decision tree in ensemble methods which combines the prediction results of several base estimators. A decision tree is a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation. All trees in a Random Forest are built from samples drawn with replacement from the training set and have the same weight.
4. AdaBoost is a boosting algorithm in ensemble methods. It is also usually generated based on decision tree, just like Random Forest. The difference is that in AdaBoost, the weight of the trees are not equal. They are adjusted according to the error from the previous round.
5. Extra Tree is a variant of Random Forest. It introduces more randomness on the basis of Random Forest to construct decision trees: The Extra Trees, wherein each tree has randomly chosen split

points for each node. Thus, it will not only evaluate on a random subset of features, it will also randomly select a feature and then randomly select a segmentation point within the range of the eigenvalues of that feature.

4.2 Data preprocessing

For any prediction model simulation, we flatten the input data by default, such that the "day of week" column only indicates whether a line of data was recorded on a week day or on a weekend. Similarly, we subdivide the "hours of the day" into four intervals for every four consecutive shop opening hours.

4.3 Experiments with different regression methods

The abovely mentioned five regression models were implemented using the *scikit-learn library* (Hao and Ho 2019) for our simulation portal.

Table 1: Prediction evaluation on training and test set

Model	training score	test RMSE	test R^2
Lasso	0.4548	246.71	0.4699
MLP	0.2136	377.75	0.2016
AdaBoost	0.7180	211.81	0.6796
Random Forest	0.9827	115.51	0.8909
Extra Tree	0.9999	122.87	0.8598

Any prediction simulation starts with the segmentation of the input into training data and test data. Using the grocery store data, with about $1,000$ lines per article after preprocessing, we selected 70 % of the input for training and the remaining 30 % for testing. The training took us less than 12 seconds on our dual-GPU server (NVIDIA A100). The *root-mean-square error* (RMSE) and *R Square* (R^2) in Table 1 reflect the average score of all the considered commodities.

In our experiments, the Random Forest and the Extra Tree performed best in sales prediction. Therefore, our online portal displays the Random Forest curves (See Fig. 2b). Random Forest fits a little less accurately than Extra Tree on the training set, but it performs better on the test set. The reason is overfitting: It randomly splits all features and is more susceptible to noise and abnormal samples. In the simulation with the Random Forest model, one can see that the final result averages the votes of multiple decision trees, making the generalization stronger. When we focus on article soft pretzel again and we assume that we are close to October, we can use the January-September data. The actual number of sales in 2022 was 74379 and our software predicts 67656 reaching 91 %, i. e., the prediction is in an acceptable range.

4.4 Price simulation experiments

Price simulations are another type of prediction experiment which can be conducted using our online interface. Users can enter a threshold for the acceptable sales loss in a text field and find the price for increasing the profit as much as possible. After prices for the most frequently sold articles were entered, our software calculates the price elasticity of an example article and allows for experiments with the *price change threshold*. We assume that the price and sales quantity fit the following equation:

$$log(salesquantity) = elasticity * log(price) + intercept. \tag{3}$$

We initialize *elasticity* and *intercept* with 1 and -1. The squared difference between the result of equation 3 and the actual quantity are to be minimized. Coming back to the soft pretzel example, we computed a price elasticity of -0.66. That means, a 1 % price raise leads to a 0.66 % sales decrease.

For multi-item transactions like "family purchases", cross-product effects are also present: We observed effects of changing the "bread"-price on the "diaper"-sales. Thus, the price of one item can affect the sales quantity of other items. This happens, once the price for one article exceeds a tolerance maximum, which causes that complete customer groups, e. g., the families, decide buying multiple articles elsewhere.

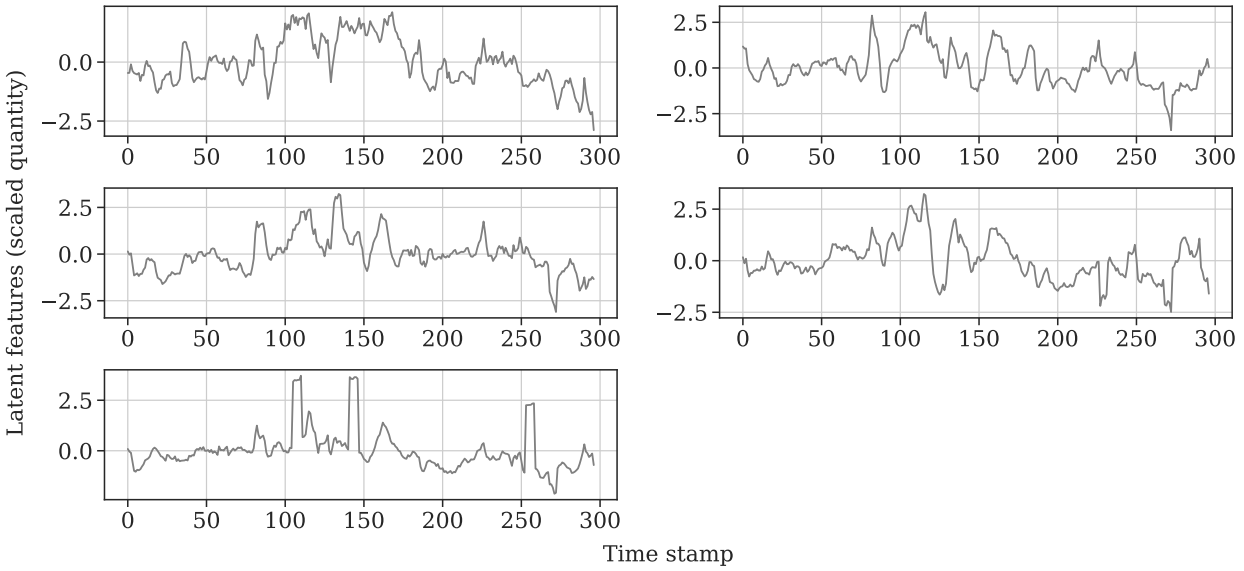


Figure 3: Store features overserved in clustering experiments

5 SIMULATION OF CLUSTERING

In data mining, *clustering* means separating different groups. We are interested in groups of stores wherein the stores exhibit significant similarities among each other. Thus, the clustering simulations we perform in our portal do not relate to single commodities but to information about the stores.

5.1 Algorithm overview

For clustering experiments, our portal provides an implementation of the classic *K-Means* algorithm (Lloyd 1982) in the clustering section, which still is a commonly used procedure. By default, data from 14 stores is used as the input. We take the total sales quantity of these stores as feature values, and the total sales of each store on different dates are treated as the values of each dimension, when the distances to the imaginary centroids (the arithmetic means of all values in the clusters) are computed.

Before the clustering starts, a value for K, i. e., the number of desired clusters must be determined. For that purpose, we use a combination of the popular elbow and the silhouette method to find a suitable value for K (Oğul and Zeybek 2023). We start with the elbow method and look for a bend in the curve we yield when we plot a variation score for an increasing number of clusters. If such bend can be detected, its cutoff point is the optimal value for k. If not, the silhouette method is used: We measure the distances of the points in a cluster to the points in the neighboring clusters. The better the separation between the points is, the better is the K-value we found. We iteratively adapted the K-value according to an evaluation of all K-values in the $1 \ldots 10$-interval. After fitting our experimental data, we plotted elbow and silhouette curves and found 3 as the optimal number of clusters for our grocery store clustering experiments.

5.2 Data preprocessing

To prepare the grocery store data for clustering, the total sales quantities per day for each store are computed. To avoid noise, we smooth the data by averaging the sales via the "`rolling(6).mean()`" method supported by *Series objects* in the Pandas library (McKinney 2011). The window size 6 is used for highlighting the weekly trend, since the stores are open for 6 days a week.

The *K-Means*-implementation that runs when one clicks on *K-Means* in our simulation interface is the "`TimeSeriesKMeans`" model provided by the *tslearn*-library (Tavenard, R. et al. 2020). Time series data is characterized by its chronological order, i. e. the ordering of data points it considered relevant in this implementation. Many other implementations of *K*-Means treat each data point independently, ignoring the temporal ordering, which would lead to poor results when grocery stores are divided into groups and one of the groups exhibits, e. g., characteristic late-shopping patterns.

5.3 Experiments

Our results for the first of the three clusters we analyzed are illustrated in Figure 3. This cluster contains five stores and we see that the sales curve of these stores show a similar pattern, starting out steadily, then having a trend at the middle period, rising sharply at first, then stabilizing for a while, and then falling sharply again. Such knowledge refers to total sales numbers instead of single commodities. Consequently, clustering is less often used in inventory and supply chain management, but it is rather helpful in accounting, procurement, risk management and compliance tasks.

Store operators can plan to empty the waste bins more often in the most busy hours or adapt the automatic air conditioner and shop lighting settings, avoiding the waste of energy over the less busy hours.

6 SIMULATION OF CLASSIFICATION

Clustering identifies stores with similar sales patterns helping store managers to understand which stores have similar inventory needs. A store classification, in contrast, identifies specific stores, such as urban or rural ones, and therefore takes, instead of the characteristics of the sold goods, the characteristics of the stores such as their geographic locations, opening hours etc. into account. Using our online portal, a *store classification* helps retrieving valuable customer behavior information from these characteristics.

To simulate characteristic customer behavior, we start experimenting without any particular expectation: We suppose, each transaction is independent and, thus, unrelated. Finally, all successive transactions are carried out by different, ordinarily unrelated customers. A customer who buys one liter of milk does not directly increase the probability that the next customer will also buy one liter of milk. However, buying milk in beverage cartons might turn out as a typical urban behavior, less often seen in rural areas, where people prefer farm shops and milk filling stations.

6.1 Algorithm overview

Similarly to clustering, classification is aimed at a model that can assign labels to data. However, classification represents an unsupervised machine learning model. The classification section of our online portal provides an implementation of random forest, which we have already seen – in a supervised version – in our experiments with alternating product prices in Section 3.3. For classification, the model is trained with a combination of real and synthetic data and the proximity matrix which we use for the store classification is extracted from the resulting forest, which is known to work well on previously unprocessed data (Zhu 2020).

The implementation of random forest used by our portal is a module from the latest version of Weka, a data mining library that is already popular for 30 years (Holmes, Donkin, and Witten 1994). The random forest module can not be exchanged directly in our portal. However, if the data is prepared using the standard Weka administration tool (the Weka *workbench*) simulations based on modules implementing alternative classification models can also be selected interactively. Such alternatives include any model implemented in

Weka, e. g., *logistic regression* and *dynamic time warping* (DTWSearch (Senin 2008)). Moreover, a module that implements a model for time series processing using a different library, such as the implementation of *long short term memory*(LSTM (Hochreiter and Schmidhuber 1997)) in the Keras library (Ketkar 2017), can also be plugged into our classification simulations, as we show in our experiments (Section 6.3).

6.2 Data Preprocessing

We take the daily sales quantity of certain frequently sold articles as one feature that is relevant for the store classification. Each column in the input lists row-by-row how many items the store sold on a given day. This information is formatted in *attribute relationship format* (`arff`) which is the standard form of data representation used in Weka. The script that we wrote for pre-processing raw transaction data and formatting them either into `arff`, or into the representation used by Keras, is included in our *github*-repository: `https://github.com/KaterinaVV/A-Case-Study-On-Supermarket-Sales-Data`.

6.3 Experiments

After importing the data into the Weka *workbench*, in the "`Classify`" option panel, we choose a classifier module, e. g., "`Random Forest`" from the "`trees`" directory. For the training/testing-split we choose 70 % in the "`test option`" panel. Finally, we change the classified default target into "`StoreNumber`" and retrieve the results.

Experiments with logistic regression, DWTSearch or LSTM classification can be done by clicking on the respective classifier module. The accuracy of the classification of the grocery stores using all mentioned models is listed in Table 2. In order to improve the accuracy, we grouped the similar stores according to the results of our clustering experiments, described in Section 5. The results are in the "`Accuracy with store clustering`" list (Table 2). The accuracy of Random Forest is more than 90 %, and the Logistic Regression model reaches 85 %. These results are also shown in the classification section of our online portal. That much reliability is only possible, when enough training data is available which can easily be confirmed in our portal: After reducing the input to the "top 10 articles" by clicking on the corresponding box in the flow diagram, the accuracy of random forest goes down to almost 50%.

Table 2: Classification accuracy

Model	Specific for time series data?	Accuracy without store clustering	Accuracy with store clustering
Random Forest	No	78.5377 %	92.3077 %
Logistic Regression	No	74.9214 %	85.7143 %
DTWSearch	Yes	52.9088 %	77.6557 %
LSTM	Yes	72.7415 %	81.7518 %

The experiments confirm our hypothesis that without detailed customer information (collected e. g. in a bonus card program) the best classification is possible using using random forest.

Notably, not only the accuracy but the classification results themselves, i. e., the number of detected classes decreased considerably, if the time when the products were sold was omitted. This observation confirms our assumption that different stores have different peak times. We are currently investigating if there is a relation between the peak times and the geographic position, e. g., typical "rush hour" for an urban supermarket.

7 SIMULATION OF A RECOMMENDER SYSTEM

Our online portal discloses factors influencing shopping decisions in transactional data. Such factors include, whether a person belongs to a particular customer group (students, pensioners etc.), her residential area and the time limits within those she is shopping. However, most crucial are undoubtedly the customer preferences: Somebody who refuses meat will never buy a sausage roll. To detect customer aversions and inclinations, our portal provides a simulation section for *recommender system* experiments.

7.1 Data Preprocessing

While the users of some streaming services rate can rate media according to a multigradual scoring system, in a grocery store the rating is binary: For our application customers either like or dislike a product. When a customer buys a product, we suppose that she likes the product and the rating is 1, while the rating of any product that is not purchased is 0. We do not consider the quantities, as, for groceries, a higher quantity rather indicates a higher demand for a product than an actual affection to it. In our experiments with the Surprise library, we only consider transactions of at least four different articles and we store user-item-ratings in a file with the columns ["User", "Item", "Rating"].

7.2 Algorithm overview

This section is implemented using the Python Surprise library, which is frequently used in media rating applications (Hug 2020). From this library, we use a *confusion matrix* which relates between the customers and the items they buy.

Since most ratings are 0, the input data is extensive and very sparse. Thus, we run the `SVD()` function for performing a *single value decomposition* which produces multiple component matrices exposing the useful properties of the input.

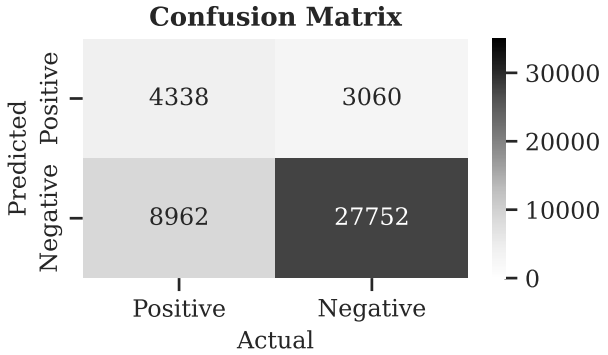


Figure 4: Confusion matrix for predictive recommender simulation.

7.3 Experiments

For any grocery store data file uploaded to the portal, the result is a static confusion matrix, such as the one shown in Figure 4. In the depicted experiment, we selected 80 % of the rating data for training. The accuracy reaches more than 0.7 , indicating that our model can reliably predict whether a customer will like (and buy) a particular item.

8 CONCLUSION

To demonstrate our online interface for machine learning models we conducted five simulation experiments with supermarket sales data.

Independently of the ML model used, there is a flow of steps which we display in a flow diagram on the bottom of all experimentation sections in our portal. By clicking on the boxes, users can modify corresponding steps in different AI models, which helps comparing the models. By exploring real data from grocery stores, we found grouping rules which are specific for this application domain of machine learning. Namely, these groups are the "individual purchases", "multi-item purchases" and more specific ones, such as "family purchases" and "handyman purchases". Such grouping by means of simulation helps developing customer-group targeted marketing strategies.

To explain the functionality of predictions in smart retail, we simulated the flow of the data processing steps used for regression. The displayed graphical output in the form of one curve for the predicted and one curve for the actual values, one can be intuitively see the connection. Thus, our portal is also useful for educational purposes. Besides regression-based sales forecasting, other classical ML models, namely, Random Forest and K-means, can be illustrated using our portal.

In a simulation with varying prices, we have shown how a retail store can calculate the price elasticity of all top-seller articles to find the optimal price and maximize the revenue. Finally, we have shown that methods from the *recommender system* domain can help predicting whether someone might buy a certain item or not.

The presented smart retail strategies are not only relevant for grocery stores. Especially the last experiments with matrix factorization are a basis for a more in-depth exploration of the long-term sales of high-priced products. Product associations and store classifications help focusing on offers targeted at specific customer groups such as hobbyists. Thus, e. g., outdoor equipment stores, HiFi electronics dealers, toy stores, florists or jewelers may also benefit from using our simulation software. However, grocery stores heavily depend on the day-by-day planning. In this context, our simulations, can avoid wrong decisions and are, therefore, particularly useful.

ACKNOWLEDGMENTS

We are grateful to Netto Marken-Discount for their support.

REFERENCES

Codd, E. F. and Salley, C. T. 1993. "Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate". E. F. Codd and Associates.

Delos Arcos, J. R. and A. A. Hernandez. 2019. "Efficient Apriori Algorithm using Enhanced Transaction Reduction Approach". In *2019 IEEE 13th International Conference on Telecommunication Systems, Services, and Applications (TSSA)* https://doi.org/10.1109/TSSA48701.2019.8985482.

Forman, G., H. Nachlieli, and R. Keshet. 2015. "Clustering by intent: a semi-supervised method to discover relevant clusters incrementally". 20–36: Springer.

Gonzales et al., E. C. 2024. "Smart Shelf System for Customer Behavior Tracking in Supermarkets". *Sensors* 24(2).

Gore, R., S. Diallo, and J. Padilla. 2014, February. "ConceVE: Conceptual modeling and formal validation for everyone". *ACM Transactions on Modeling and Computer Simulation* 24(2):1–17 https://doi.org/10.1145/2567897.

Gorre, N., E. Carranza, J. Fuhrman, H. Li, R. K. Madduri, M. Giger *et al.* 2023, April. "MIDRC CRP10 AI interface—an integrated tool for exploring, testing and visualization of AI models". *Physics in Medicine & Biology* 68(7):074002 https://doi.org/10.1088/1361-6560/acb754.

Hao, J. and T. K. Ho. 2019, June. "Machine Learning Made Easy: A Review of *Scikit-learn* Package in Python Programming Language". *Journal of Educational and Behavioral Statistics* 44(3):348–361 https://doi.org/10.3102/1076998619832248.

Hochreiter, S. and J. Schmidhuber. 1997, November. "Long Short-Term Memory". *Neural Computation* 9(8):1735–1780 https://doi.org/10.1162/neco.1997.9.8.1735.

Holmes, G., A. Donkin, and I. Witten. 1994. "WEKA: a machine learning workbench". In *Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference*, 357–361. Brisbane, Qld., Australia: IEEE https://doi.org/10.1109/ANZIIS.1994.396988.

Hug, N. 2020. "Surprise: A Python library for recommender systems". *Journal of Open Source Software* 5 https://doi.org/10.21105/joss.02174.

Ketkar, N. 2017. "Introduction to Keras". In *Deep Learning with Python*, 97–111. Berkeley, CA: Apress https://doi.org/10.1007/978-1-4842-2766-4_7.

Klasson, M., C. Zhang, and H. Kjellström. 2020, November. "Using Variational Multi-view Learning for Classification of Grocery Items". *Patterns* 1(8):100143 https://doi.org/10.1016/j.patter.2020.100143.

Kumar, L., Pooja, and P. Kumar. 2021. "Amazon EC2: (Elastic Compute Cloud) Overview". In *Proceedings of Integrated Intelligence Enable Networks and Computing*, 543–552: Springer Singapore.

Kumar Jha, B. and S. Pande. 2021, April. "Time Series Forecasting Model for Supermarket Sales using FB-Prophet". In *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, 547–554. Erode, India: IEEE https://doi.org/10.1109/ICCMC51019.2021.9418033.

Lars Hertel, P. B. and D. L. Gillen. 2022. "Reproducible Hyperparameter Optimization". *Journal of Computational and Graphical Statistics* 31(1):84–99 https://doi.org/10.1080/10618600.2021.1950004.

Lloyd, S. 1982, March. "Least squares quantization in PCM". *IEEE Transactions on Information Theory* 28(2):129–137 https://doi.org/10.1109/TIT.1982.1056489.

Harrell et al. 2020. "Autonomous and Composable M&S System of Systems with the Simulation, Experimentation, Analytics and Testing (Seat) Framework". In *2020 Winter Simulation Conference (WSC)* https://doi.org/10.1109/WSC48552.2020.9384040.

Joncour et al. 2019. "Multiscale Spatial Analysis of Young Stars Complex using the DBScan Clustering Algorithm". *Astronomical Data Analysis Software and Systems XXVII* 523:87.

O. Olayemi and C. Olayemi 2017, April. "Analysis of Sales Data for Decision Making Using Association Rule Mining". In *International Conference on Statistical Research and Applications*.

Tavenard, R. et al. 2020. "Tslearn, a machine learning toolkit for time series data". *The Journal of Machine Learning Research* 21(1):4686–4691. Publisher: JMLRORG.

McKinney, W. 2011, November. "pandas: a Foundational Python Library for Data Analysis and Statistics". In *Python for high performance and scientific computing*, Volume 14, 1–9. Seattle, USA: IEEE.

Melzer, M., J. Dünnweber, and T. Baumann. 2022, August. "Towards Smart Home Data Interpretation Using Analogies to Natural Language Processing". In *2022 IEEE International Conference on Smart Internet of Things (SmartIoT)*, 65–71. Suzhou, China: IEEE https://doi.org/10.1109/SmartIoT55134.2022.00020.

Oğul, İ. and Ö. Zeybek. 2023. "Store Segmentation Using Machine Learning Methods: An Organized Supermarket Chain Case". In *Intelligent and Fuzzy Systems*. Springer https://doi.org/10.1007/978-3-031-39777-6_18.

Patra, S. and B. Ganguly. 2019. "Improvising Singular Value Decomposition by KNN for Use in Movie Recommender Systems". *Journal of Operations and Strategic Planning* 2(1):22–34 https://doi.org/10.1177/2516600X19848956.

Roberts, J. O., G. Mastorakis, B. Lazaruk, S. Franco, A. A. Stokes and S. Bernardini. 2021, May. "vPlanSim: An Open Source Graphical Interface for the Visualisation and Simulation of AI Systems". *Proceedings of the International Conference on Automated Planning and Scheduling* 31(1):486–490.

Santoso, M. H. 2021, December. "Application of Association Rule Method Using Apriori Algorithm to Find Sales Patterns Case Study of Indomaret Tanjung Anom". *Brilliance: Research of Artificial Intelligence* 1(2):54–66 https://doi.org/10.47709/brilliance.v1i2.1228.

Senin, P. 2008. "Dynamic time warping algorithm review". In *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*. University of Hawaii at ManoaHonolulu, USA.

Zellner, M., D. Milz, L. Lyons, C. Hoch and J. Radinsky. 2022, November. "Finding the Balance Between Simplicity and Realism in Participatory Modeling for Environmental Planning". *Environmental Modelling & Software* 157:105481 https://doi.org/10.1016/j.envsoft.2022.105481.

Zhu, T. 2020, August. "Analysis on the Applicability of the Random Forest". *Journal of Physics: Conference Series* 1607(1):012123 https://doi.org/10.1088/1742-6596/1607/1/012123.

AUTHOR BIOGRAPHIES

WENFEI HUANG is a master student at the Faculty of Computer Science and Mathematics at OTH Regensburg. His master thesis is focused on sales data analysis using AI tools. His email address is wenfei.huang@st.oth-regensburg.de.

MATTHIAS MELZER is a Research Assistant at the Faculty of Computer Science and Mathematics at OTH Regensburg. His research interests include Energy Efficiency and Sufficiency, Data Science and Machine Learning. His email address is matthias.melzer@oth-regensburg.

JAN DÜNNWEBER is a Professor at the Faculty of Computer Science and Mathematics at OTH Regensburg. His research interests include Parallel and Distributed Computing, Smart Environments and Artificial Intelligence. His email address is jan.duennweber@oth-regensburg.de.