DOI: 10.15439/2021F129 ISSN 2300-5963 ACSIS, Vol. 26

Impact of time series clustering on fuel sales prediction results

Joanna Henzel*, Marek Sikora[‖]
Department of Computer Networks and System
Silesian University of Technology
ul. Akademicka 16, 44-100 Gliwice, Poland
Email: *joanna.henzel@polsl.pl, ‖marek.sikora@polsl.pl

Jakub Bularz
FuelPrime Department
AIUT Ltd.
Wyczółkowskiego 113, 44-109 Gliwice, Poland
Email: jakub.bularz@aiut.com

Abstract—**The purpose of the paper is to check the impact of data clustering in the process of predicting demand. We checked different ways of adding information about similar datasets to the forecasting process and we grouped the measurements in multiple ways. The experiments were executed on 50 time series describing fuels sales (gasoline and diesel sales) on 25 petrol stations from an international company. We described the data preparation process and feature extraction process. In the 9 presented experiments, we used the XGBoost algorithm and some typical time series forecasting methods (ARIMA, moving average). We showed a case study for two datasets and we discussed the practical usage of the tested solutions. The results showed that the solution which used XGBoost model utilising data gathered from all available petrol stations, in general, worked the best and it outperformed more advanced approaches as well as typical time series methods.**

I. Introduction

THE proper forecasting of sale is a crucial element in many sectors, also in the retail industry. This plays an important role in the competitiveness of any company. Accurate forecasts can help to create proper planning of demand and deliveries, which can lead to cost reductions by the company. In the paper, we focus on the petrol retail sector and the problem of forecasting fuel sales from multiple petrol stations.

A lot of research was done for forecasting sale in multiple retail industries with the use of Machine Learning methods, however not so much was done in this field regarding the fuel sales short-term predictions. Also, the usefulness of clustering similar petrol stations based on historical data and similarities of time series was not tested in this area.

The objective of this paper is to check the usefulness of data clustering in the process of predicting fuel sales. In our research, we inspected different ways of adding information about similar datasets to the forecasting process and we grouped the measurements in multiple ways. We also checked how adding the information about predicted sales for similar datasets can affect the results obtained for a single dataset. We hypothesised that using information about the historical sales from different petrol stations and the sales predictions for them can improve sales predictions results for the dataset.

The paper is organised as follows: the next section provides the review of the literature and related works, section III describes the data preparation process. Afterwards, the experiments and results are presented, followed by a more detailed case study. The paper ends with some conclusions and a discussion of the results.

II. Related Works

Traditionally, forecasting was made using statistical methods. These were exponential smoothing [1], moving average, the Auto-regressive Integrated Moving Average (ARIMA) model and the Seasonal Auto-regressive Integrated Moving Average (SARIMA) model.

More complex methods were also used for the task of sales forecasting. In [2] a comparison of various linear and non-linear models for the sales forecasting task was conducted and the results suggest that non-linear models should be highly considered when dealing with modelling retail sales. In the paper [3] a comparison of different Machine Learning Techniques was conducted regarding sales forecasting of retail stores. The authors concluded that boosting algorithms gave better results than the regular regression ones. For them, the best results were obtained for the GradientBoost algorithm and the XGBoost [4] implementation has been used in order to increase the accuracy. The successful usage of the XGBoost algorithm for sales forecasting was also presented in [5]. The authors in the paper extracted features based on the historical sales and then they trained one model using XGBoost. The proposed model performed extremely well for sales prediction with less computing time and memory resources. The paper [6] presented a framework based on Facebook's Prophet algorithm and backtesting strategy for forecasting future sales in the retail industry.

In the field of sales forecasting a lot of different neural network and Deep Learning approaches were tested e.g. evolutionary neural networks (ENN) [7], the Dynamic Artificial Neural Network [8] and the backward propagation neural network [9]. In the paper [10] a case study for using long short-term memory (LSTM) recurrent neural networks (RNN) was proposed. Deep Bi-Directional LSTM Networks was presented in [11]. The authors of the paper [12] proposed a novel forecasting method that combined the deep learning method – LSTM – and random forest (RF) for demand forecasting problems. They compared the results with neural networks, multiple regression, ARIMAX, LSTM networks, and RF method and their proposition was statistically significantly better.

Fuel sales forecasting was not considered in many scientific articles. Most of them were focused on the marketing research area and not machine learning models strategies. The paper [13] described the problem of forecasting fuels sales but from a perspective of long-term forecasts - for a few years ahead. The authors of the paper [14] focused more on the attributes describing the petrol stations and their surroundings and the models were developed for planning purposes in order to assess potential sales of new sites. The use of judgmental forecasting for predicting fuels sales on petrol stations was presented in the paper [15]. The authors used experts' knowledge to create forecasts. They used PROLOG in order to map the knowledge into production rules, thus enabling computer processing. The paper [16] showed the solution for real-time petrol sales forecasting using dilated causal convolutional neural network (CNNs). To the best of our knowledge, the tree boosting algorithm, especially the extreme gradient boosting (XGBoost) algorithm, has not been used to forecast fuels sales.

Clustering methods are widely used for forecasting purposes. In the paper [17] a k-means clustering algorithm was used along with decision trees, artificial neural networks and support vector machines methods. The author of the paper [18] described using k-means clustering and ARIMA model for forecasting electricity load. In [19], a hybrid sales forecasting system based on clustering and decision trees was presented. Clustering of time series data was broadly discussed in [20]. In this paper the clustering algorithms were divided into six groups: Partitioning, Hierarchical, Grid-based, Model-based and Density-based. Predicting based on similarities was also mentioned in [21]. The authors applied the fuzzy network of comparators (NoC) for the problem of ovulation date prediction.

III. Data preparation

In the experiments, we used data from 25 petrol stations located in different parts of Poland. All of them belong to one international company and are part of a network of petrol stations. In the research, we focused on forecasting the sale of two types of regularly sold fuels: Gasoline and Diesel. We did not consider different types of fuels e.g. Premium Gasoline or Premium Diesel which are more expensive than the regular ones. We were making experiments for 2 types of fuels from 25 petrol stations, so finally, we were working with 50 tabular datasets, which described the time series of fuel sales. The datasets covered a 3-year period from 2017-01-01 to 2019-12-31.

The raw data derived from stations contained 5 columns:

- *meter* - ID of a fuel tank.
- *start_time* - Start of a considered period of time; timestamp from which fuel sale in a given period is counted.
- *end_time* - End of a considered period of time; timestamp to which fuel sale in a given period is counted.

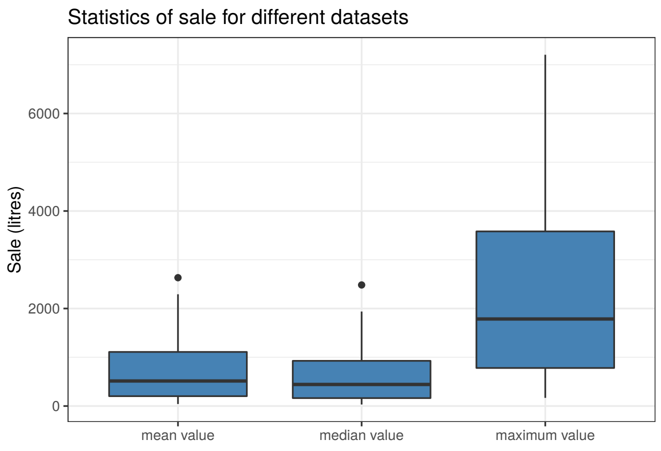

Fig. 1. A mean, median and maximum value of sales from datasets with 4-hour intervals.

- *sales* - Total fuel sales (in litres) in a given period of time.
- *temp* - Fuel temperature.

Most of the periods (*end_time* - *start_time*) last for about 15 minutes, but the periods were not exactly equal. The median time of periods (n = 5,062,511) was equal to 15.27 minutes, the mean value of periods was 16.11 minutes $\pm$ 10.42 minutes. The datasets also contained missing periods, i.e. the *end_time* and the following row *start_time* were not always equal.

In the first step of the data preparation process, we decided to prepare datasets in a way that they will contain equal periods of time. We decided to aggregate sales into 4-hour intervals, because experts gave us the information that forecasting for this amount of time is a reasonable approach. Forecasting sale for every 15 minutes does not have big practical usage but forecasting for bigger periods of time can be used e.g. for demand planning.

The sales value for a new interval was calculated as the sum of the sales from all periods that were entirely contained within a given 4-hour interval. To this was added the proportional value of sales from periods that only partially fell within the interval.

Basic statistics of sales from all of 50 datasets are presented in figure 1. They were calculated after creating modified datasets and cleaning the data. The minimum value was not presented because it is equal to 0 in each dataset.

A. Attributes

Each of the fifty time series of fuel sales was presented as a separate tabular dataset with 5 columns. In our experiments, we wanted to consider not only typical time series forecasting methods but also method used mostly for tabular datasets. Because of this, a crucial step was to create proper attributes which will describe the sale in each created interval. We decided to prepare attributes that could be divided into two categories:

- attributes based on time column and
- attributes based on previous values of sales.

1) Attributes from date: Each row of the datasets described one interval. After the first step of the data preparation process, all intervals were equal so in the next steps we used only one time column - in our case we used *end_time* column. Attributes derived from the time column were created in order to describe the time of sale – for example, different fuel sales can be between midnight and 4 am and different between 12 am and 4 pm. Also, weekday, public holidays, vacations etc. can have an impact on fuel sales at petrol stations.

The list of proposed time attributes created based on *end_time* column is as follow:

- basic attributes from time column: year number, quarter number, month number, day number, number of a day in the year, week number, hour, weekday, season, logical attribute describing the weekend;
- logical attributes describing the school holidays: information on whether any school holidays are in progress, information on whether summer holidays are in progress, information on whether winter holidays are in progress, information on whether spring holidays are in progress, information on whether Christmas brake is in progress;
- logical attributes describing the public holidays: information on whether any public holiday is in progress, information about New Year's Day, Easter, Easter Monday, All Saint's Day, All Souls Day, Christmas Eve, Christmas, New Year's Eve;
- cyclical attributes created based on basic attributes from time column - the problem of creating these attributes are described in detail in [22].
- numerical attributes describing a number of days before public holidays (any public holiday, Christmas and Easter) – these attributes have value from a range [0;7].

All considered petrol stations are located in Poland and because of this attributes for public and school holidays were prepared based on polish holidays.

2) Attributes from previous values of sales: Using forecasting methods typical for tabular datasets has some limitations and problems comparing with time series methods. In a tabular dataset, each row describes one measurement and they are not chronologically connected with each other. In a simple tabular dataset, we lose information about time series. Because of this, we proposed adding attributes describing the history of fuel sales:

- fuel sales observed in previous intervals – from one interval before to six intervals before;
- mean value from x previous intervals (x = (6, 12, 18, 24, 30, 36, 42));
- median value from x previous intervals (x = (6, 12, 18, 24, 30, 36, 42));
- fuel sales observed in previous days in the same hours as a considered measurement - from 2 days before to 7 days before;
- mean value from x previous days in the same hours as a considered measurement (x = (7, 10, 14, 21));
- fuel sales observed x weeks before and in the same hours as a considered measurement (x = (2, 3, 4));
- mean value from x previous weeks and in the same hours as a considered measurement (x = (2, 4, 6, 8));
- attributes describing trend: a preceding interval sales value compared with previous values, a preceding interval sales value compared with mean values from previous intervals, mean values of preceding intervals compared with previous mean values;

The example will be presented in order to better explain creating these attributes. Suppose we want to create attributes for the period where *end_time*= '2019-02-22 08:00:00' (Friday). Some attributes would be as follows:

- fuel sales observed in previous intervals – these attributes would describe sales from intervals '2019-02-22 04:00:00', '2019-02-22 00:00:00', ..., '2019-02-21 08:00:00';
- mean value from 6 previous intervals - this attribute would be a mean value of sales from intervals '2019-02-22 04:00:00', '2019-02-22 00:00:00', ..., '2019-02-21 08:00:00';
- fuel sales observed 5 days before in the same hours as a considered measurement – this attribute would describe sale from interval '2019-02-17 08:00:00';
- mean value from 7 previous days in the same hours as a considered measurement – this attribute would be a mean value of sale from intervals '2019-02-21 08:00:00', '2019-02-20 08:00:00', ... , '2019-02-15 08:00:00';
- fuel sale observed 2 weeks before in the same hours as a considered measurement – this attribute would describe sale from interval '2019-02-08 08:00:00';
- mean value from 2 previous weeks in the same hours as considered measurement – this attribute would be a mean value of sale from intervals '2019-02-15 08:00:00' and '2019-02-08 08:00:00'.

After the data preparation process, each dataset had 157 columns and $3 \cdot 365 \cdot 6 = 6570$ rows (3 years of data; 365 days in a year; 6 4-hours intervals). It is worth mentioning that not all rows might be used in the experiments because some of them could have missing values of sales. Also, not all columns were used when training the model. For example columns *id_meter*, *start_time*, *end_time* were not used.

IV. Experiments

The purpose of this paper is to analyse the benefit of using clustering methods and information about similar records in the process of forecasting sales. In our experiments, we used datasets representing fuel sales on petrol stations. We assumed that there might be similarities among the different sales characteristics. However, some of them may also differ significantly from each other. For example, in some datasets sales can be strongly correlated with their previous values and in other datasets, we won't see a similarly strong connection

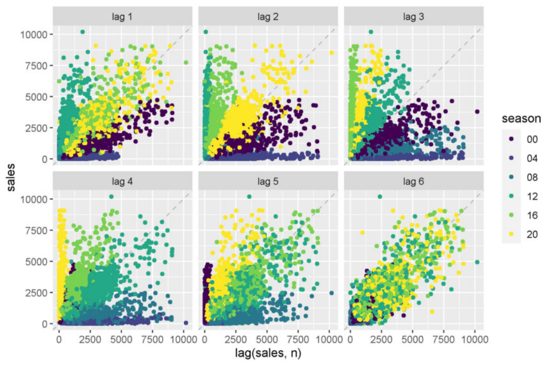

Fig. 2. Exemplary sales values plotted against lagged values of itself. It is an example of a dataset where the sales value is highly correlated with 6th lagged value.

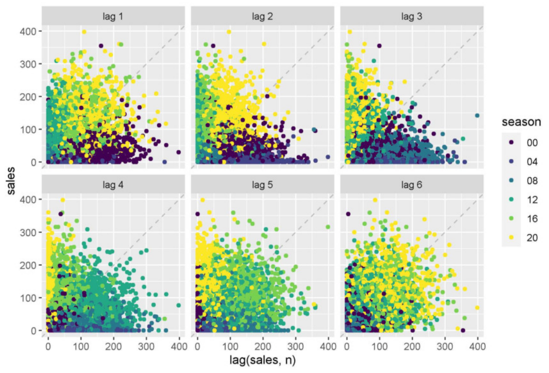

Fig. 3. Exemplary sales values plotted against lagged values of itself. It is an example of a dataset where the sales value is weakly correlated with previous sales values.

with historical data. It can be visible in the figure 2 and figure 3. Both illustrations show the time series of fuel sales against lags of itself. Each figure shows the results for a different dataset. In the figure 2 we can see that the sales value is highly correlated with lag number 6 of itself. As it was said before, the datasets described 4-hour intervals, so lag number 6 was an interval that occurred 24 hours before the interval under consideration. A different situation can be observed for the sales of the dataset presented in the figure 3 – here the correlation of historical values with the current value of the sales is not so prominent.

The differences between sales could be also visible when considering different attributes e.g. the attributes describing a time of a sale.

Taking into account the differences and similarities of sales characteristics, we wanted to find out how adding information about these similarities can improve sales forecasting.

A. *Description of experiments*

In the experiments, we used typical methods for time series and methods intended for tabular datasets. We forecasted one value ahead, so we always forecasted sales for the next 4 hours.

In all experiments in which we trained a predictive model using XGBoost, the first model was trained on data from 2017-01-01 to 2018-12-31, and then the model was re-trained every 7 days of data. Testing of a given model was done on data that fell between model recalculations.

We carried out following experiments:

1) Station-based forecast – experiments "*moving average*", "*arima predictions*" and "*separate groups predictions*".
2) Forecast after putting all datasets into one group – experiments "*one group – predict normalized sale*" and "*one group – add column with prediction*";
3) Forecast based on similar records – experiment "*regression knn*".
4) Forecast based on clustering time series – experiment "*partitional clustering – predict normalized sale*" and "*partitional clustering – add column with prediction*".
5) Forecast based on double clustering – experiment "*double clustering*".

Each experiment will be described below.

Method "*moving average*" (*MovAvg*) is a simple forecasting method based on moving average. The window for these calculations was 42 measurements (sales from the previous 7 days).

Experiment "*arima predictions*" (*ARIMA*) was based on ARIMA method. In this experiment, `auto.arima` function from the `forecast` R package was used. 84 previous measurements (2 weeks of data) was a training set (argument `learning_set_size = 84`).

In both experiments, "*moving average*" and "*arima predictions*", only the value of the sales was taken into consideration.

Experiment "*one group – predict normalized sale*" (*1G.PredNormSale*) is the experiment where each dataset in a first step was normalised. It means that the sales were normalised and all attributes created based on a sales history were generated again using normalised values of sale. Then all records from all datasets were connected into one dataset. The model was trained using XGBoost method on data from 2017-01-01 to 2018-12-31 and then recalculated every 7 days of data. The model forecasted the normalised value of the sales. For each recalculation predictions were made. Each prediction was then reversed from a normalised value to the regular one.

Experiment "*one group – add column with prediction*" (*1G.AddPred*) is the experiment where, as before, data from all datasets were normalised and combined into one training set. A model that predicted the normalised sales value was created. Then, for each dataset describing one time series (to the train set and test set), a column was added with the forecast from the model (forecast of the normalised sales value). A separate XGBoost model was then created for each dataset (the decision column was the actual sales value, without normalisation). The model was also recalculated every 7 days.

In the experiment "*separate groups predictions*" each dataset was considered independently. A separate XGBoost model was trained for each dataset. A model was trained only on historical data from the specific time series.

Experiment "*regression knn*" (*RegKNN*) used a typical k-NN regression method. Firstly, all datasets were normalised and connected into one dataset. Then we found an optimal number of neighbours for this dataset (parameter `k` for k-NN algorithm) – repeated cross-validation was used on records from 2017-01-01 to 2018-12-31 in order to do this. In the next step, we performed k-nearest neighbour regression that was recalculated every 7 days. Predicted normalised values were reversed to the regular values of sales. k-NN does not work with missing data so in training datasets only rows with no missing values were used. In a test set, missing values were imputed using the last observed value.

In the experiment "*partitional clustering – predict normalized sale*" (*PC.PredNormSale*) the datasets were grouped by the nature of their normalised time series describing sales. The `dtwclust` R package and the partitional (or partitioning) clustering algorithm were used for clustering. Within each group, all examples from the datasets comprising the group were combined. An XGBoost model that predicted the normalised sales value was created for each group. The final prediction was made based on these models - the output was a normalised forecast, which then had to be inverted to a regular value. For example, assume that we have five datasets describing fuels sales - datasets *a, b, c, d, e*. In the first step of the experiment, each dataset was normalised and the value of the sale was normalised. Then only time series (not any additional created attributes) were compared and grouped using `dtwclust` package - assume that datasets *a* and *c* were classified to group *X* and datasets *b, d, e* were classified to group *Y*. Then 2 models were created - the first model that was trained on connected datasets *a* and *c* (group *X*) and a second model that was trained on connected datasets *b, d, e* (group *Y*). Then predictions were made based on an appropriate model for each data set - predictions from a test set from dataset *a* were made based on a model trained for group *X*, predictions from the test set from dataset *b* was made based on a model trained for group *Y* etc.

Experiment "*partitional clustering – add column with prediction*" (*PC.AddPred*) – as before, data from all time series were normalised and grouped by the nature of the sales time series. Within each group, a training set was created and a model was trained that predicted the normalised sales value. Then, for each datasets belonging to this group (to the train and test set), a column was added with the forecasts from this model (forecast of normalised sales value). A separate model was then created for each dataset (the decision column was the actual sales value, without normalisation). Showing it on the example: again, suppose that datasets *a* and *c* were classified to group *X* and datasets *b, d, e* were classified to group *Y* based on their sales characteristics. Then models were trained for group *X* and group *Y* - these models predicted the normalised value of the sales. Then for each non-normalised datasets (*a, b, c, d, e* before normalisation) new column with the prediction of normalised sale was added. So for the non-normalised dataset *a*, column with predictions from a model created for group *X* was added. Then for each non-normalised dataset *a, b, c, d, e* separate XGBoost model was trained.

In experiment "*double clustering*" (*2Clust*) data from all time series were also normalised and grouped by the characteristic of the sales time series. Then for each group, a new logical column was added to each normalised dataset. For example, if two groups were created, two new columns would be added to each dataset. The columns indicated whether the row belongs to the group represented by the column. At this point, all records from one dataset would have the same values in these columns – if the dataset was classified to group number 2 then in the second column all records would have a value "1", and in the first column all records would have a value "0". Clustering based on time series and adding columns with information about groups was the first clustering in this experiment. Then all datasets were combined into one dataset. Then an optimal number of clusters for algorithm k-means was calculated and then k-means clustering was performed on the created dataset. This was the second clustering in this experiment. Then for each cluster obtained from k-means, we created XGBoost model. It is worth mentioning, that the rows from one dataset could be added to the different clusters from k-means clustering. It means that rows classified into one k-means cluster could have different values in columns representing results from time series clustering.

The flowcharts of the more advanced experiments – *PC.PredNormSale*, *PC.AddPred* and *2Clust* – are presented in figures 4 and 5.

In clustering procedures always normalised sales and normalised datasets were used.

The important aspect of these experiments is the utility of the created models and using the proposed solution in a real-life scenario. One interesting problem is forecasting sales for new, unknown petrol station or new fuel on an existing petrol station. We do not have a history of sales for this fuel, so models from experiments *SepGr*, *ARIMA* and *MovAvg* couldn't be used. Because of the lack of historical data, we wouldn't also be able to assign this fuel into one of the previously created clusters that were created based on similarities of time series. It means that models created in experiments *PC.AddPred*, *PC.PredNormSale* and *2Clust* also couldn't be used. The only way to use these models would be to manually assign the fuel from the station to an existing cluster. Such a task could be performed by an expert, but this could create additional errors and bias. In the discussed problem, the models from experiment *RegKNN* could be possibly used however k-NN do not work with missing data so many attributes would have to be imputed. In the problem of forecasting sale for new petrol station, the only way would be to use models from experiments *1G.PredNormSale* and *1G.AddPred*, because XBGoost method handles missing data.

B. Results

The errors metrics for each experiment were presented in the table I. These are the standard error metrics that are frequently used in papers that describe creating forecasting models. In describing the results we used *Root Mean Square*

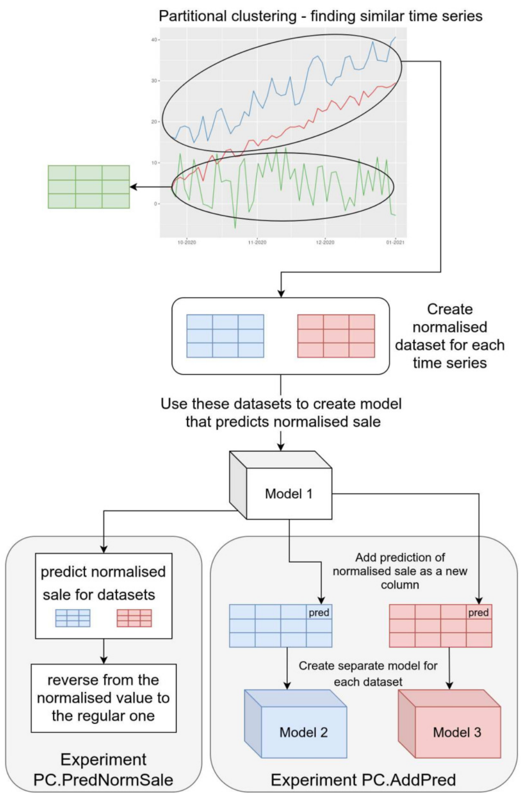
Partitional clustering - finding similar time series
Create normalised dataset for each time series
Use these datasets to create model that predicts normalised sale
Model 1
predict normalised sale for datasets
reverse from the normalised value to the regular one
Experiment PC.PredNormSale
Add prediction of normalised sale as a new column
pred
Create separate model for each dataset
Model 2
Model 3
Experiment PC.AddPred

Fig. 4. Flowchart of the experiments *PC.PredNormSale* and *PC.AddPred*.

Error (RMSE), *Mean Absolute Error (MAE)*, *Coefficient of determination (R2)* and *Weighted Mean Absolute Percentage Error (WMAPE)*. In our opinion, the most intuitive errors are *MAE* and *WMAPE* and these metrics will be briefly explained in order to understand the results correctly.

The MAE express what error, on average, the model makes. It does not explain if it was an overestimation or underestimation error. In the case of described experiments, it says what was the average error in forecasting sales for a 4-hour period.

The WMAPE is an advanced version of a metric *mean absolute percentage error (MAPE)*. The MAPE is meaningful only when the values are large. If the actual value is close to 0, the value of MAPE gives uninterpreted results. In order to bypass this problem, a similar measure – WMAPE – was developed. It is the sum of absolute errors divided by the sum of the actual values and it works well with smaller numbers. It is widely used in the retail sector. The sale of fuel on some petrol stations was often equal to 0, so it was important to use a modified version of the MAPE. The WMAPE, multiplied by 100, can be interpreted as the average percentage by which the model is wrong.

The results presented in the table I show that the best solution among the tested approaches is for experiment *1G.PredNormSale*.

In figure 6 the histogram of errors for the experiment *1G.PredNormSale* is presented, where $error = prediction - real\_value$. We can see that the distribution of errors is balanced for this approach.

We also wanted to compare how these errors differ in real-life situations. In order to do this, for each dataset and each experiment we compared the absolute errors of predictions with the maximum value of sale from a test dataset. The calculation of this error can be presented as follows $\frac{|pred-real|}{maxReal} \cdot 100$, where $pred$ is the predicted value, $real$ is the actual value and

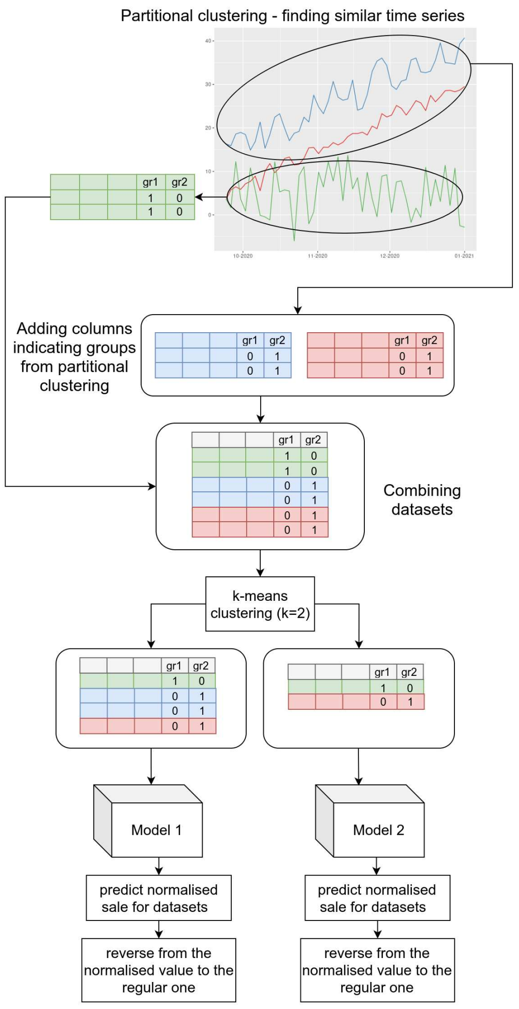
Partitional clustering - finding similar time series
gr1 gr2
Adding columns indicating groups from partitional clustering
Combining datasets
k-means clustering (k=2)
Model 1
Model 2
predict normalised sale for datasets
reverse from the normalised value to the regular one

Fig. 5. Flowchart of the experiment *2Clust*.

TABLE I
ERROR METRICS FOR THE EXPERIMENTS.

model	RMSE	MAE	R2	WMAPE
1G.PredNormSale	347.97	207.13	0.65	0.32
PC.PredNormSale	349.55	208.95	0.64	0.33
SepGr	365.17	222.46	0.60	0.36
1G.AddPred	372.86	225.61	0.58	0.36
PC.AddPred	374.83	228.69	0.58	0.36
RegKNN	413.61	262.83	0.50	0.42
2Clust	459.69	343.93	0.17	0.54
ARIMA	505.40	351.03	0.23	0.57
MovAvg	583.09	435.50	0.07	0.67

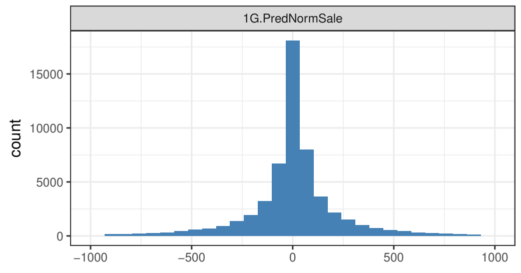

Fig. 6. Histograms of errors for the experiment *1G.PredNormSale*.

$maxReal$ is the maximum value of the actual sales in the test set. Then, for each experiment, the average for each set was calculated and then the average values from all datasets were obtained. The average real-world prediction errors are given in table II. This error allows us to understand the scale of the error we are dealing with and allows us to answer the question of how significant these errors are in a relation to the domain. Here, the *1G.PredNormSale* approach was also the best one.

TABLE II
RELATIVE ERRORS FOR THE EXPERIMENTS.

	mean_realtive_error [%]
1G.PredNormSale	6.59
PC.PredNormSale	6.65
SepGr	7.11
1G.AddPred	7.16
PC.AddPred	7.25
RegKNN	8.38
2Clust	10.99
ARIMA	11.61
MovAvg	14.44

In most industries where the demand for products needs to be forecast, overestimation and underestimation errors mostly are not equal to each other. We can imagine that in a problem of forecasting fuel demand on petrol station, underestimation errors are much more harmful to the industry. If these forecasts would be used for deliveries planning then underestimation could create a situation where there wouldn't be fuel on a petrol station. It could lead to losing clients. Because of this, we performed an analysis of underestimation and overestimation errors. We considered prediction as an overestimation when the prediction was at least 20 litres larger than the real sales during the 4-hour period. The prediction was an underestimation when it was at least 20 litres smaller than the real sales. The numbers of examples that were overestimated or underestimated in each dataset and in each experiment are presented in figure 7. We can see that for each experiment the average number of underestimation is bigger than the number of overestimated examples. We can also see from the plot that for experiments *RegKNN* and *ARIMA* the average number of examples overestimated and underestimated is higher than for other experiments. It means that these methods made more errors with an absolute value higher or equal to 20 litres, so they made more significant errors. This conclusion

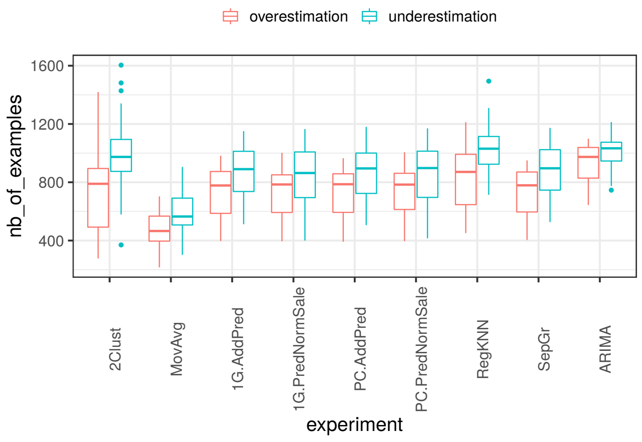

Fig. 7. Box plots representing number of examples that were overestimated or underestimated in each dataset for each experiment.

is consistent with the results obtained from the RMSE metric, which penalises large mistakes.

V. CASE STUDY

In this section, we would like to present a case study for two different datasets and discuss the utility of created methods. Both of the datasets represent the sale of gasoline on two different petrol stations.

In the previous section, we discussed what are the errors and the error metrics. Each error was calculated for every 4-hour period independently. Suppose that the errors in 6 consecutive 4-hour periods were (-250, 250, -300, 100, 300, -100). Each of these errors is significant, but they add up to 0. In this case, we can say that the model at the end of the day did not make a mistake, because ultimately as much fuel was sold in one day as the model predicted. This is an important aspect for demand forecasting.

In the following case study, we present the moving sums of errors. The window for moving sum was equal to 7 days (42 measurements). We presented what was the average moving sum of errors for the test dataset and we compared it with an average moving sum of real sales that have taken place within 7 days.

Results for the first dataset (dataset A) are presented in table III (*mean* – the average value of the moving errors, *stder* – the standard deviation of the moving errors, *%mean* - the ratio of the average moving error to the moving average of sales expressed as a percentage, *%stder* - the ratio of the standard deviation of the moving errors to the moving average of sales expressed as a percentage). The smallest average value of a moving error within 7 days was obtained for the experiment *1G.PredNormSale*. This is in line with the results presented in the table I and II where this solution had the best results for error metrics. The average value of moving error was equal to 0.3% of a moving sum of sales within 7 days for this dataset. The standard deviation of moving error was equal to 6.3% of a moving sum of sales within 7 days for this dataset.

We can observe that the average moving error for most of the experiments is greater than zero – it means that mostly we will deal with overestimation and as it was discussed before this is the better kind of error in the retail sector.

The moving sum of the actual sale in the 7-days window for dataset A was around 23,000 litres.

TABLE III
THE ANALYSIS OF THE MOVING ERROR CALCULATED IN WINDOW EQUAL TO 7 DAYS FOR THE DATASET A

experiment	mean	stder	%mean	%stder
1G.PredNormSale	67.57	1472.71	0.29	6.25
PC.PredNormSale	93.93	1440.51	0.40	6.11
1G.AddPred	193.71	1167.56	0.82	4.96
PC.AddPred	246.76	1273.60	1.05	5.41
MovAvg	303.66	1301.34	1.29	5.52
ARIMA	311.59	2312.32	1.32	9.81
SepGr	360.86	1022.90	1.53	4.34
RegKNN	-587.24	1400.77	2.49	5.95
2Clust	-2912.15	4487.68	12.36	19.05

As a contrast, we present the results for the second dataset (dataset B). The results are presented in IV. For this dataset the method *1G.PredNormSale* got worse results comparing with other methods. The value of the average moving error was also less than zero so it can lead more frequently to underestimation. The best average value of moving error was obtained for the experiment *PC.AddPred* and it was equal to 0.06% of a moving sum of sales within 7 days for this dataset.

The moving sum of the actual sale in 7-days window for dataset B was around 13,000 litres.

This case study shows that the error metrics presented in the table I can indicate the general best solution taking into account the measurements independently but it does not mean that it will always be the best solution in each case (e.g. for the demand planning). Also, the average best solution does not need to be the best for each considered dataset.

TABLE IV
THE ANALYSIS OF THE MOVING ERROR CALCULATED IN WINDOW EQUAL TO 7 DAYS FOR THE DATASET B

experiment	mean	stder	%mean	%stder
PC.AddPred	8.29	880.62	0.06	6.54
1G.AddPred	30.26	839.20	0.22	6.24
SepGr	54.63	902.84	0.41	6.71
ARIMA	-64.55	1425.59	0.48	10.59
MovAvg	73.78	1019.11	0.55	7.57
PC.PredNormSale	-87.06	794.90	0.65	5.91
1G.PredNormSale	-122.13	775.17	0.91	5.76
RegKNN	189.07	603.95	1.41	4.49
2Clust	-489.86	4107.20	3.64	30.52

VI. CONCLUSIONS

The accurate forecasts are an important aspect in all companies from a retail sector, also from the petrol retail sector. This study checked the usefulness and impact of data clustering in the process of predicting fuel sales. The research showed that the best results, in general, were obtained for the experiment where for each time-series the new features were extracted and then all obtained tabular datasets were connected into one dataset. This dataset was used to train one predictive model which predicted the normalised sale for each fuel and each petrol station. This solution outperformed the typical simple time series forecasting methods (ARIMA, moving average), but also it was better compared with the results where advanced clustering methods were used. It also performed better than creating a predictive model for each time series separately based only on its historical data with extracted features. In general, adding the predictions obtained from the model generated on a bigger number of datasets as a new column did not improve the results of creating models separately for each dataset.

The worst results were obtained for time series methods, so it proves that in some cases the use of methods typical for tabular datasets can lead to better results for time series predictions. Also, it shows that the process of developing proper features from historical data, as well as using information about the time of the measurements (information derived from the date and time and information about holidays), can lead to obtaining better results. However, this hypothesis should be checked in further studies where our results will be compared with results obtained for more advanced methods used in time series forecasting problems.

The additional asset of the presented best performing approach (*1G.PredNormSale*) is also the fact that it can be quickly and straightforward used when predicting sale for a new petrol station or a new fuel on the petrol station because the sale of this fuel wouldn't need to be compared and assigned to one of the previously created cluster based on its historical data. In this proposed solution, one model is created for all petrol stations and it can be used with new data. Also, the XGBoost method, which is used in the proposed solution, would deal with missing data. However, if we would like to have the real values for all of the extracted features then only 8 weeks of historical sales would be needed to do this. The 4-week historical data would leave only 2 from 157 attributes with empty values. The solution with the best results is also practical from the perspective of time constraints because only one model is trained for all datasets. In the proposed solution, the model was recalculated weekly, which minimises the chances of a concept drift problem.

In the presented case study, we showed a different aspect of the predictions. We presented cumulative errors for the 7-day long periods. The results showed that for each presented dataset, the average moving sum of errors obtained for the best approach from the considered case study was relatively low. However, the case study also proved that the method which is generally the best do not give the best results in every case. In practical applications, a method would need to be developed to select the appropriate model (e.g. by tracking recent errors). This is a challenge for future research.

Other aspects that we will investigate in further work will be making similar experiment with the use of Deep Learning and ensemble methods. The experiments will be also performed on different datasets from another retail sector e.g. on data describing the sale of fast-moving consumer goods (FMCG).

The forecasting of sale during the promotion in FMCG sector and the efficiency of promotions were conducted in our previous works ([22], [23]), but the analysis was not yet performed on the data describing regular sales and the clustering methods were not tested by us in this field.

In conclusion, the results showed that the use of clustering methods did not produce the best results among the considered approaches, however, our study showed that it is useful to use historical data from other stations when making predictions and it should be considered to create one predictive model for many datasets and not for each dataset separately.

ACKNOWLEDGMENT

This research was realised in co-operation with FuelPrime Department in AIUT Ltd. and was partially supported by the European Union through the European Social Fund (grant POWR.03.05.00-00-Z305).

REFERENCES

[1] E. S. Gardner Jr., "Exponential smoothing: The state of the art," vol. 4, no. October 1983, pp. 1–28, 1985.

[2] C. W. Chu and G. P. Zhang, "A comparative study of linear and nonlinear models for aggregate retail sales forecasting," *International Journal of Production Economics*, vol. 86, no. 3, pp. 217–231, dec 2003. doi: 10.1016/S0925-5273(03)00068-9

[3] A. Krishna, V. Akhilesh, A. Aich, and C. Hegde, "Sales-forecasting of retail stores using machine learning techniques," in *Sales-forecasting of Retail Stores using Machine Learning Techniques*. IEEE, 2018. doi: 10.1109/CSITSS.2018.8768765. ISBN 9781538660782 pp. 160–166.

[4] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. KDD '16. ACM, 2016. doi: 10.1145/2939672.2939785. ISBN 9781450342322 pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[5] X. Dairu and Z. Shilong, "Machine Learning Model for Sales Forecasting by Using XGBoost," in *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*. Institute of Electrical and Electronics Engineers Inc., jan 2021. doi: 10.1109/ICCECE51280.2021.9342304. ISBN 9781728183190 pp. 480–483.

[6] E. Žunić, K. Korjenić, K. Hodžić, and D. Đonko, "Application of Facebook's Prophet Algorithm for Successful Sales Forecasting Based on Real-world Data," *International Journal of Computer Science and Information Technology*, vol. 12, no. 2, pp. 23–36, apr 2020. doi: 10.5121/ijcsit.2020.12203

[7] K.-F. Au, T.-M. Choi, and Y. Yu, "Fashion retail forecasting by evolutionary neural networks," *International Journal of Production Economics*, vol. 114, no. 2, pp. 615 – 630, 2008. doi: 10.1016/j.ijpe.2007.06.013

[8] V. Adithya Ganesan, S. Divi, N. B. Moudhgalya, U. Sriharsha, and V. Vijayaraghavan, "Forecasting food sales in a multiplex using dynamic artificial neural networks," in *Advances in Intelligent Systems and Computing*, vol. 944. Springer Verlag, 2020. doi: 10.1007/978-3-030-17798-0_8. ISBN 9783030177973. ISSN 21945365 pp. 69–80.

[9] C. Giri, S. Thomassey, J. Balkow, and X. Zeng, "Forecasting New Apparel Sales Using Deep Learning and Nonlinear Neural Network Regression," in *2019 International Conference on Engineering, Science, and Industrial Applications (ICESI)*. Institute of Electrical and Electronics Engineers Inc., aug 2019. doi: 10.1109/ICESI.2019.8863024. ISBN 9781728121741 pp. 1–6.

[10] Q. Yu, K. Wang, J. O. Strandhagen, and Y. Wang, "Application of Long Short-Term Memory Neural Network to Sales Forecasting in Retail—A Case Study," in *Advanced Manufacturing and Automation VII*. Springer Singapore, 2018. doi: 10.1007/978-981-10-5768-7_2. ISBN 978-981-10-5768-7. ISSN 18761119 pp. 11–17.

[11] D. Ruta, L. Cen, and Q. H. Vu, "Deep Bi-Directional LSTM Networks for Device Workload Forecasting," in *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems, FedCSIS 2020*, 2020. doi: 10.15439/2020F213. ISBN 9788395541674 pp. 115–118.

[12] S. Punia, K. Nikolopoulos, S. P. Singh, J. K. Madaan, and K. Litsiou, "Deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail," *International Journal of Production Research*, vol. 58, no. 16, pp. 4964–4979, aug 2020. doi: 10.1080/00207543.2020.1735666

[13] J. Chai, S. Wang, and S. Wang, "Demand Forecast of Petroleum Product Consumption in the Chinese Transportation Industry," *Energies*, vol. 5, no. 3, pp. 577–598, 2012. doi: 10.3390/en5030577. [Online]. Available: www.mdpi.com/journal/energiesArticle

[14] I. Themido, A. Quintino, and J. Leitao, "Modelling the Retail Sales of Gasoline in a Portuguese Metropolitan Area," *International Transactions in Operational Research*, vol. 5, no. 2, pp. 89–102, mar 1998. doi: 10.1111/j.1475-3995.1998.tb00106.x. [Online]. Available: http://doi.wiley.com/10.1111/j.1475-3995.1998.tb00106.x

[15] K. Kalid, J. Ahmad, S. Yong, and K. H. Yew, "Petronas Petrol Station Fuel Consumption Forecast System," in *Proceedings of the Second International Conference on Artificial Intelligence in Engineering & Technology*, 2004. doi: 10.13140/2.1.4493.8568. ISBN 10.13140/2.1.4. [Online]. Available: https://www.researchgate.net/publication/271637545

[16] S. M. Rizvi, T. Syed, and J. Qureshi, "Real-time forecasting of petrol retail using dilated causal CNNs," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, p. 3, feb 2021. doi: 10.1007/s12652-021-02941-3. [Online]. Available: https://doi.org/10.1007/s12652-021-02941-3

[17] P. F. Jiménez-Pérez and L. Mora-López, "Modeling and forecasting hourly global solar radiation using clustering and classification techniques," *Solar Energy*, vol. 135, pp. 682–691, oct 2016. doi: 10.1016/j.solener.2016.06.039

[18] B. Nepal, M. Yamaha, A. Yokoe, and T. Yamaji, "Electricity load forecasting using clustering and ARIMA model for energy management in buildings," *Japan Architectural Review*, vol. 3, no. 1, pp. 62–76, jan 2020. doi: 10.1002/2475-8876.12135. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/2475-8876.12135

[19] S. Thomassey and A. Fiordaliso, "A hybrid sales forecasting system based on clustering and decision trees," *Decision Support Systems*, vol. 42, no. 1, pp. 408–421, oct 2006. doi: 10.1016/j.dss.2005.01.008

[20] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-series clustering - A decade review," *Information Systems*, vol. 53, pp. 16–38, may 2015. doi: 10.1016/j.is.2015.04.007

[21] Ł. Sosnowski, I. Szymusik, and T. Penza, "Network of Fuzzy Comparators for Ovulation Window Prediction," in *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, M.-J. Lesot, S. Vieira, M. Z. Reformat, J. P. Carvalho, A. Wilbik, B. Bouchon-Meunier, and R. R. Yager, Eds. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-50153-2_59. ISBN 978-3-030-50153-2 pp. 800–813.

[22] M. Blachnik and J. Henzel, "Estimating the Performance Indicators of Promotion Efficiency in FMCG Retail," in *Neural Information Processing. ICONIP 2020. Lecture Notes in Computer Science*, vol. 12533. Springer, Cham, 2020. doi: 10.1007/978-3-030-63833-7_27. ISBN 9783030638320. ISSN 16113349 pp. 320–332.

[23] J. Henzel and M. Sikora, "Gradient Boosting and Deep Learning Models Approach to Forecasting Promotions Efficiency in FMCG Retail," in *Artificial Intelligence and Soft Computing. ICAISC 2020. Lecture Notes in Computer Science*, vol. 12416. Springer, Cham, 2020. doi: 10.1007/978-3-030-61534-5_30. ISBN 978-3-030-61533-8. ISSN 16113349 pp. 336–345.