Study of Supervised Algorithms for Solve the Forecasting Retail Dynamics Problem

Ohrimuk Ekaterina S.
Faculty of Computer Science and Technologies
Saint Petersburg Electrotechnical University "LETI"
Saint Petersburg, Russia
cat.oxrimuck@yandex.ru

Mikhailov Yury I.
Faculty of Economics and Management
Saint Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
yimikhaylov@etu.ru

Natalya V. Razmochaeva
Faculty of Computer Science and Technologies
Saint Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
nvrazmochaeva@etu.ru

Bezrukov Artem A.
Faculty of Economics and Management
Saint Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
aabezrukov@etu.ru

***Abstract*—In this article, we examine the use of machine learning models to predict retail dynamics. The main goal of this article is to analyze existing approaches and draw a conclusion about the advantages and disadvantages of each method. Main approaches and practical examples of using machine learning to solve the problem of forecasting retail dynamics are considered in practice. The advantages of using various machine learning methods, such as learnability, dynamism, nonlinearity and others are shown. The quality of the results of the applied methods is estimated using various quality metrics. A hierarchy of algorithms was proposed that showed better results than individual algorithms. This allows us to conclude that the application of the proposed hierarchy of algorithms improves the quality for predicting the dynamics of retail trade.**

Keywords—machine learning; forecasting; sales, retail; prediction; supervised algorithms.

I. Introduction

Retail is one of the most significant sectors of the economy for society. The effective operation of a retailer depends on many circumstances, including the assessment of reasonable quantities of goods purchased for subsequent sale, which is impossible without forecasting the demand for goods, without taking into account the expected intensity of sales.

There are many methods for predicting the dynamics of retail trade, including in the field of learning algorithms with a teacher. Each of them has its own advantages and disadvantages and has a specific field of application. A technique is needed that would allow us to confidently predict the expected sales volumes with fluctuations in the dynamics of sales of goods. The subject of the study is the teaching algorithms with the teacher to solve the problem of forecasting the dynamics of retail trade. Object of study - teacher-training algorithms used to analyze data in retail.

The goal of this work is to find the advantages and disadvantages of algorithms by comparative analysis of existing approaches.

To achieve the goal, it is necessary to solve a number of the following tasks:

- To analyze existing methods for forecasting retail dynamics.
- To highlight the criteria by which the comparison will be conducted.
- To identify the disadvantages and advantages of each method.

II. Analogues Review

Since the present work is related to retail forecasting, the search for analogues was carried out among scientific studies in the field of machine learning (a set of methods by which a computer can find initially unknown relationships and patterns in arrays) applied to the analysis of retail data.

After studying the proposed algorithms from table 1, table 2 was formed, which contains a comparison of the algorithms presented above according to the following selected criteria:

- Variability of parameters

The same algorithm used to solve the problem of forecasting retail dynamics can show different efficiencies with different data, therefore it is important to be able to adjust the model parameters to increase the forecast accuracy of the algorithm used.

TABLE 1. CONSIDERED ANALOGUES AND THEIR MAIN COMPONENTS

Article title	Used algorithms	Used quality metrics
Sales Forecast in E-commerce using Convolutional Neural Network [1].	Convolutional Neural Network (CNN) , ARIMA, FE+GBRT, DNN, CNN, CNN+WD, CNN+WD+TL, Single-CNN	Mean Squared Error (MSE)
Machine-Learning Models for Sales Time Series Forecasting [2].	ExtraTree, Linear Model, ARIMA, RandomForest Lasso, Neural Network Stacking	Mean absolute error (MAE)
Sales forecasting using WaveNet within the framework of the Kaggle competition [3].	Convolutional Neural Network	Normalized weighted root mean squared logarithmic error (NWRMSLE)
Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology [4].	Globally training a Long Short-Term Memory network (LSTM)	Mean absolute percentage error (mMAPE)
Prediction of investment indicators based on the decision tree method [5]	Decision trees using the CART algorithm	MSE

- Accuracy

It is important that the obtained values of the forecast of the target variable are close to the true values of the target variable, since the efficiency of the algorithm directly depends on this. Designations used to assess accuracy: high - accuracy indicators in the considered article are not lower than 80%, medium - in the range from 60% to 80%, low - below 60%.

- Entry threshold

Since our models will be used by ordinary people who do not have deep knowledge in the field of machine learning, it is important for them to be easy to use algorithms, it is important to use those algorithms that are very simple and convenient to use. The symbols used to estimate the threshold of occurrence are: 1 – very simple to apply, 2 – more complicated, 3 – very difficult to apply.

TABLE 2. THE COMPARISON TABLE ON THE CRITERIA

	Criteria		
Algorithms	***Variability of parameters***	***Accuracy***	***Entry treshold***
Convolutional neural network (CNN)	+	high	3
ARIMA	-	low	2
ExtraTree	-	high	2
Linear Model	-	medium	1
Globally training a Long Short-Term Memory network (LSTM)	+	medium	3
RandomForest	+	high	2
Lasso	-	medium	3
Neural Network Stacking	-	high	3
Decision Trees	+	high	1

Based on the review and analysis of analogues, algorithms that show the best results according to the selected criteria and good results for use in systems of algorithms were selected. According to Table 2, they are: Decision Tree, Random Forest and XGBRegressor.

A large set of considered methods can be explained by the fact that the result of the study may be not one method, but combinations of different methods.

III. DESCRIPTION OF ALGORITHMS

- *Linear Regression*

Linear regression in Python 3.6 is implemented using sklearn.linear_model.LinearRegression (v0.21.3 version) [6].

From a mathematical point of view, the linear algorithm in regression tasks is as follows:

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x^j,$$

where w_0 — free ratio, x^j— features, w^j — weights.

If we add the (d + 1) -th feature, which on each object takes the value 1, the linear algorithm can be written as:

$$a(x) = \sum_{j=1}^{d+1} w_j x^j = \langle w, x \rangle,$$

where the notation $\langle w, x \rangle$ is used for the scalar product of two vectors [7].

- Decision Tree

Decision Tree in Python 3.6 is implemented using sklearn.tree.DecisionTreeRegressor (v0.21.3 version) [8].

In general, a decision tree can be called visual instructions on what to do in what situation. In terms of machine learning, we can say that this is a model that predicts the value of the target variable by several attributes (features).

The structure of the tree consists of "leaves" and "branches". The edges ("branches") of the decision tree contain the attributes that the objective function depends on, the values of the target variable are written in the "leaves", and the attributes based on which cases could be distinguished are written in the other nodes. To classify a new case, it is necessary to go down the tree to the leaf and issue the corresponding value. Это жадный алгоритм - строится от корня к листьям [10].

- *Random Forest*

Random forest in Python 3.6 is implemented using sklearn.ensemble.RandomForestRegressor (v0.21.3 version) [9].

A random forest is a composition of trees that are built independently of each other. A composition is a union of N algorithms $b_1(x),\dots,b_n(x)$ into one. The general idea is to train the algorithms$b_1(x),\dots,b_n(x)$, and then average the responses received from them:

$$a(x)=\frac{1}{N}\sum_{n=1}^{N}b_n(x),$$

where $a(x)$ - an algorithm that returns the mean or sign of the mean, which is called a composition of N algorithms, $b_1(x),\dots,b_n(x)$ – basic algorithms [7].

- *XGBRegressor*

Gradient boosting in Python 3.6 is implemented using sklearn.ensemble.GradientBoostingClassifier (v0.21.3 version) [10].

Boosting is an approach to building compositions, within which:

- Basic algorithms are built sequentially, one after another.
- Each subsequent algorithm is constructed in such a way as to correct the errors of an already constructed composition.

Because the construction of compositions in boosting is directional, it is enough to use simple basic algorithms, for example, shallow trees.

In gradient boosting, the composition under construction is the sum, not the averaging, of the underlying algorithms $b_1(x),\dots,b_n(x)$:

$$a_N(x)=\sum_{n=1}^{N}b_n(x),$$

Gradient boosting algorithm description:

1. Initialization: composition initialization$a_0(x)=b_0(x)$, that is construction of the simple algorithm b_0.
2. Iteration step:
   a) Calculation of the shift vector:
   $$s=-\blacktriangledown F=\begin{pmatrix}-L_z'(y_1,a_{n-1}(x_1)),\dots\\-L_z'(y_l,a_{n-1}(x_l))\end{pmatrix}.$$
   b) Construction of the algorithm:
   $$b_n=argmin_b\frac{1}{l}\sum_{i=1}^{l}(b(x_i)-s_i)^2,$$
   whose parameters are selected in such a way that its values on the elements of the training sample were as close as possible to the calculated optimal shift vector s.
   c) Adding of the $b_n(x)$ algorithm to the composition:
   $$a_n(x)=\sum_{m=1}^{n}b_m(x).$$
3. If the stop criterion is not met (the stop criterion is a way to combat retraining), then perform another iteration step. If the stop criterion is met, stop the iterative process [7].

- *KNN Regressor*

Nearest neighbor method in Python 3.6 is implemented using sklearn.neighbors.KNeighborsRegressor (v0.21.3 version) [11].

The essence of the nearest neighbor method is that the new point belongs to the same class as the nearest point from the training sample.

The weighted kNN for the regression problem is defined as:

$$a(x)=\frac{\sum_{i=1}^{k}w(x_{(i)})x_{(i)}}{\sum_{i=1}^{k}w(x_{(i)})},$$

where x — new object that is need to be classified, a $x_{(i)}$ — i-th nearest neighbor from the training set [7].

IV. Error Measure

The following metrics are used to evaluate forecasting:

- *MAE – Mean Absolute Error*

MAE of loss regression in Python 3.6 is implemented using sklearn.metrics.mean_absolute_error (v0.21.3 version) [12]

MAE is calculated as:

$$MAE=\frac{1}{n}\sum_{i=1}^{n}|y_i-\hat{y}_i|,$$

where y_i - target variable forecast, $\hat{y}_i$- true value of the target variable, n – total number of target variables.

- *MSE – Mean Squared Error*

MSE of loss regression in Python 3.6 is implemented using sklearn.metrics.mean_squared_error (v0.21.3 version) [13]

To calculate MSE, all individual regression residuals are squared, added up, the sum is divided by the total number of errors:

$$MSE=\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{n},$$

(see explanations for the designations of this formula above).

- *RMSE – Root Mean Squared Error*

RMSE of loss regression in Python 3.6 is implemented using numpy.sqrt(sklearn.metrics.mean_squared_error()) (v0.21.3 version) [13].

RMSE is calculated as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}},$$

(see explanations for the designations of this formula above).

These quality metrics allow you to objectively evaluate the accuracy of models on a test set and compare the methods used.

V. EXPERIMENTS

Dataset Description

The selected algorithms were tested on a large real data set from a task on the Kaggle website [14], which are divided into sets:

- stores.csv – contains anonymous information about 45 stores, indicating the type and size of the store.
- train.csv – historical training data that spans the period from 2010-02-05 to 2012-11-01. This file contains the following fields:

➢ Store – store number.

➢ Dept - department number.

➢ Date – week.

➢ Weekly_Sales sales for the concrete department in the specified store.

➢ IsHoliday - determines whether a week is a special holiday week.

- test.csv – identical to train.csv, except for the absence of the target variable - Weekly_Sales.
- feature.csv – contains additional data related to the activities of the store, department and region for the specified dates. It contains the following fields:

➢ Store - store number.

➢ Date – week.

➢ Temperature – average temperature in the region.

➢ Fuel_Price - fuel cost in the region.

➢ MarkDown1-5 – anonymous data related to Wal mart promotional discounts. MarkDown data is only available after November 2011 and is not available to all stores permanently. Any missing value is marked with NA.

➢ CPI - consumer price index.

➢ Unemployment - unemployment rate.

➢ IsHoliday - determines whether a week is a special holiday week.

Dataset preparing

Model training is carried out on 75% of the data, testing - 25%. First it is necessary to replace the missing values in the signs MarkDown1-5 with 0, convert the categorical variable 'Type' into a numerical variable for A = 1, B = 2, C = 3 and the categorical variable IsHoliday into a numerical variable: if there is an extra day off this week, then 1 (= Yes), otherwise 2 (=No).

Quality accessment

In addition to the considered algorithms from the articles in the table 1 there are also such as: Linear Regression and KNN Regressor, which were also added to the comparison.

Linear Regression and KNN Regressor are far from the true value of the target variable.

In Fig. 1-5 you can see scatterplots that show the values that our models predicted and the values of the target variable in the form of points on the Cartesian plane. It could be seen that the Linear Regression and KNN algorithms show a low level of quality, while the XGBRegressor shows a very high.

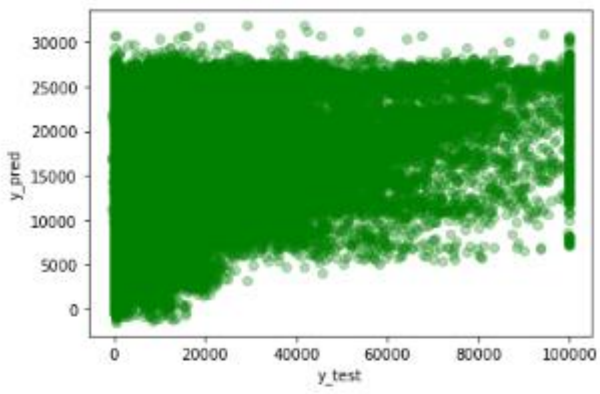

Fig. 1. Scatterplot for Linear Regression

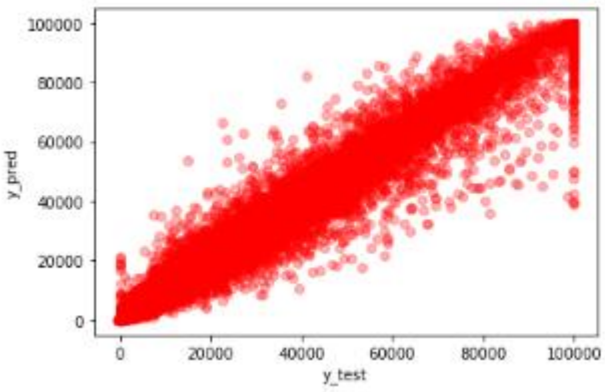

Fig. 2. Scatterplot for Random Forest

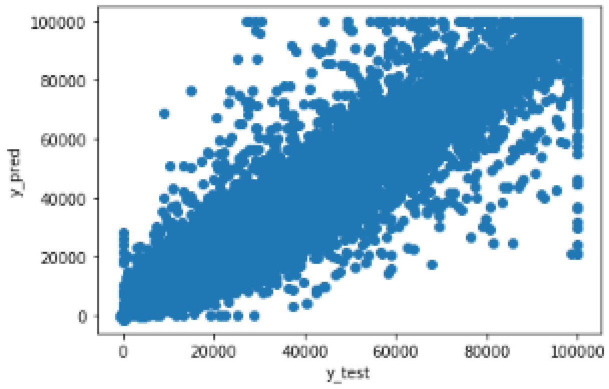

Fig. 3. Scatterplot for Decision Tree

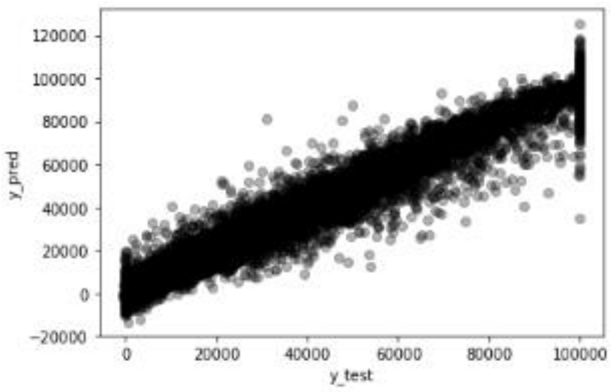

Fig. 4. Scatterplot for XGBRegressor

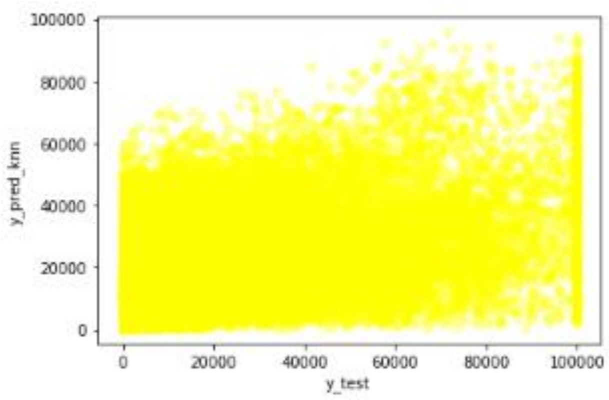

Fig. 5. Scatterplot for KNN Regressor

Table 3 presents the obtained results of quality assessments for the selected metrics after training the models and applying them to the test sample.

TABLE 3. QUALITY ASSESSMENT RESULT FOR ALGORITHMS FOR DIFFERENT QUALITY METRICS

Model	**Metrics**		
	MAE	***MSE***	***RMSE***
Linear Regression	13911	380118945	19496
Random Forest	1429	9413595	3068
XGBRegressor	1879	17866064	4226
KNN Regressor	1972	10960605	3310
Decision Tree	1879	17866064	4226

According to the table 3, the Random Forest algorithm showed the best accuracy from the considered algorithms, Decision Tree and XGBRegressor algorithms showed acceptable and rather high accuracy respectively.

VI. CONCLUSION

According to the results of a comparative analysis of teaching algorithms with a teacher to solve the problem of forecasting retail dynamics, the best of all the algorithms considered was the Random Forest algorithm, which showed the highest accuracy of forecasting the target variable for all considered quality metrics on the test set. The Decision Tree and XGB Regressor algorithms showed high accuracy, while the Linear Regression and KNN Regressor algorithms have a big mistake when examining them in practice. This may be due to the fact that they work well only in a system with other algorithms and special conditions are required for their use.

Future research plans include the implementation of a system of algorithms that will be able to predict the target variable more accurately than the separately considered algorithms.

REFERENCES

[1] Kui Zhao, Can Wang, "Sales Forecast in E-commerce using Convolutional Neural Network", 8 p., 2017. Available at: https://arxiv.org/pdf/1708.07946.pdf

[2] Bohdan M. Pavlyshenko, "Machine-Learning Models for Sales TimeSeries Forecasting, 11 p., 2018. Available at: https://ru.scribd.com/document/427249237/data-04-00015-v2-pdf. (accessed: 2019-11-28)

[3] Glib Kechyn, Lucius Yu, Yangguang Zang, Svyatoslav Kechyn, "Sales forecasting using WaveNet within the framework of the Kaggle competition," 6 p., 2018. Available at: https://arxiv.org/pdf/1803.04037.pdf. (accessed: 2019-11-28)

[4] Kasun Bandara, Peibei Shi, Christoph Bergmeir, Hansika Hewamalage, Quoc Tran, Brian Seaman, "Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology", 16 p., 2019. Available at: https://arxiv.org/pdf/1901.04028.pdf. (accessed: 2019-11-28)

[5] Китова О.В., Колмаков И.Б., Пеньков И.А., "Прогнозирование показателей инвестиций на основе метода деревьев решений (in Russian)," Вестник РЭУ им. Г.В Плеханова №6 (90), 10 p., 2016. Available at: https://cyberleninka.ru/article/n/prognozirovanie-pokazateley-investitsiy-na-osnove-metoda-dereviev-resheniy/viewer

[6] class sklearn.linear_model.LinearRegression. Available at:https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (accessed: 2019-11-28)

[7] Московский физико-технический институт, Яндекс, "Обучение на размеченных данных". Available at: https://www.coursera.org/learn/supervised-learning/home/welcome.

[8] class sklearn.tree.DecisionTreeRegressor. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html. (accessed: 2019-11-28)

[9] class sklearn.ensemble.RandomForestRegressor. Available at https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (accessed: 2019-11-28)

[10] class sklearn.ensemble.GradientBoostingClassifier. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html (accessed: 2019-11-28)

[11] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html. (accessed: 2019-11-28)

[12] sklearn.metrics.mean_absolute_error. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html (accessed: 2019-11-28)

[13] sklearn.metrics.mean_squared_error. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html (accessed: 2019-11-28)

[14] Walmart Recruiting - Store Sales Forecasting. Use historical markdown data to predict store sales. Available at: https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data (accessed: 2019-11-28)