УДК 004.4, 004.6, 004.942, 004.021: 519.6, 336.71 https://doi.org/10.17721/1812-5409.2022/4.7

Д.Ю.Круковець, аспірант

D.Krukovets, PhD student

Багатоетапний алгоритм з DTW та кластеризацією для прогнозування середньої депозитної ставки в Україні

Multi-stage approach with DTW and clustering for forecasting of average deposit rate in Ukraine

Київський національний університет імені Тараса Шевченка, Україна, 03022, м.Київ, просп. Академіка Глушкова, 4Д, 03022
e-mail: dkrukovets@kse.org.ua

Taras Shevchenko National University of Kyiv, 4D, Academician Glushkov ave., Kyiv, Ukraine, 03022
e-mail: dkrukovets@kse.org.ua

Уведіть текст тут

Стаття присвячена розробці багатоетапного методу прогнозування, котрий базується на методах Dynamic Time Warping, кластеризації та AutoARIMA, що порівнюється з кількома традиційними методами на унікальному наборі даних.

Мета полягає в тому, щоб спрогнозувати середню ставку за депозитами в Україні, використовуючи дані, зібрані з веб-сайтів банків про їхні індивідуальні ставки за депозитами на щоденній основі. З цього багатого набору даних, стаття зосереджена лише на 12-місячних депозитних ставках у гривні для кожного банку. Більшість проблем, що є традиційними для підходу веб-скреппінгу, не мають значення в нашому випадку через особливості набору даних.

Ці показники об'єднуються в групи за схожістю в динаміці, прогнозуються окремо за допомогою AutoARIMA моделі і, нарешті, об'єднуються в загальний прогноз з використанням вагових коефіцієнтів, отриманих за допомогою оцінки OLS.

У статті представлено результат і порівняння з кількома іншими моделями, починаючи від простого випадкового блукання, кількох специфікацій ARIMA та простого випадкового лісу. Багатоетапний підхід перевершує бенчмарки за оцінкою RMSE та графічним аналізом за останній період даних.

Ключові слова: Dynamic Time Warping, Кластеризація, ARIMA, Випадковий ліс, Депозитні ставки, Веб-скрепінг

The paper is dedicated to the development of the multi-stage forecasting method that is based on Dynamic Time Warping, Clustering and AutoARIMA techniques, which is compared with several traditional benchmarks on the unique dataset.

The goal is to forecast an average deposit rate in Ukraine using data that has been scrapped from banks' websites about their individual deposit rates on the daily basis. From this rich dataset the paper focuses only on 12-month deposits, UAH, for each bank. Most of the issues that are traditional for web-scraping approach are irrelevant in our case due to the dataset features.

These rates are aggregated into groups by similarity in dynamics, forecasted separately with an AutoARIMA routine and finally aggregated into the entire forecast using weights that have been obtained with an OLS estimation.

The paper presents the result and comparison with several benchmarks, starting from simple Random Walk, a few specifications of ARIMA and simple Random Forest. The multi-stage approach outperforms benchmarks by an RMSE and graphical analysis over the latter period of the data.

Keywords: Dynamic Time Warping, Clustering, ARIMA, Random Forest, Deposit rates, Web-scrapping

1. Introduction

The development of computer technologies opens a wide field for alternative datasets and seeks advanced methods for forecasting that can use such datasets. In the paper, we'll investigate a new dataset with deposit rates from all Ukrainian banks and the ability to forecast their average using the maximum of the available information. While traditional methods are concentrated on utilizing several aggregated time series, we'll have a look at more advanced methods that are able to squeeze more data from such a rich dataset.

In the second section, an overview of the data will be presented. A web-scrapping approach was used to extract the abovementioned dataset and it has many

advantages over other, more traditional methods of data collection. It provides the data faster than traditional methods, like reports from state offices that may publish the information with a significant delay.

However, there are some downfalls that may affect the quality of the data. Some of them may be solved only with the addition of data from other sources, such as the inability to aggregate rates by the deposit volumes as long as it's a type of data that is unable to be scrapped. Others are technical problems such as changes on banks' websites that lead to issues with data scrapping until the problem is fixed manually. This creates gaps in the data, however, in this particular case, such a problem is not that critical due to the features of the data.

In the third section, there will be a description of the models' pool that is taken as a benchmark for our key model. The pool is quite natural and standard, it includes a Random Walk model, a few specifications of the ARIMA model, and a Random Forest model. The first is the simplest possible model which just says that the next value will be equal to the last available. The second approach is more interesting and complex, it better explains the dynamics, however, it's unable to work with large datasets as long as it suffers from overfitting. The last approach is the simplest Data Science model that is more able to work with big datasets simultaneously and is less prone to overfitting.

Our key model is a multi-stage approach. The first stage is to take deposit rates from all banks and estimate distances between each pair using Dynamic Time Warping. It's an advanced method to find distances and count for any stretches, and reaction delays in data, which is a common problem for this dataset. Then the matrix with all distances is built and transformed into a two-dimensional graph. Every point in the graph corresponds to the bank and the distance between two points is an approximation of the distance from the matrix.

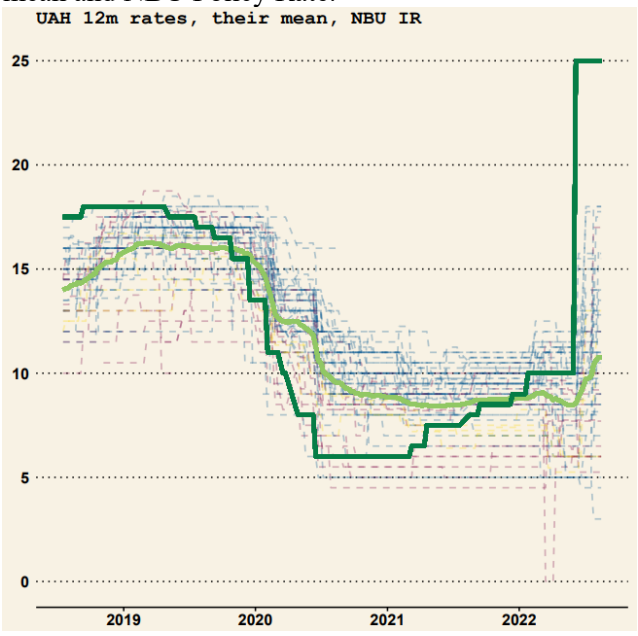
The second stage is to use a K-Means approach to divide banks into several groups by their dynamics and obtain an average deposit rate for each group. Then the third stage with an AutoARIMA approach is used to produce forecasts for each group's rate. An AutoARIMA is a method that iteratively finds the best ARIMA specification and forecasts with this ARIMA. The last stage is to aggregate forecasts using OLS estimated weights for each group.

Such an approach allows us to find more complex relations between banks and improve the quality of the forecast significantly, which can be seen from the section with results. The evaluation will be performed on the last month of the data while all other data is used to train the model. Results will be discussed based on the graphical analysis and Root Mean Squared Error value for each model.

2. Data

This paper presents a dataset of deposit rates from all banks in Ukraine, collected using web-scraping tools from the banks' websites. The methodology is described in the original paper by Khalil and Fakir, 2017. The data was collected from the beginning of 2018 and has no missing or inaccurate data due to the features of the methodology. The dataset is quite rich and consists of rates for most of the possible terms, from 1 to 36 months, and in all major currencies: UAH, USD, EUR and even CHF. For the sake of more consistent and simple-to-interpret results, we will concentrate on one of the most popular types of deposits – 12 months deposit rate in UAH. To investigate the overall dynamics of banks as long as the dynamics of mean and NBU Policy Rate (which is connected to the dynamics of deposit rates) the one may refer to Picture 1.

Picture 1. Deposit rates from different banks, their mean and NBU Policy Rate.



Web scraping tools can be useful for collecting data from websites, however, they are not always reliable. They may fail to capture certain data fields or fail to adapt to changes in the website structure if it was changed at some point in time. Regularly, in such a case, since the tool used to collect the data is not perfect, we cannot be completely sure of its reliability. This issue is discussed in the paper by Khder, 2021. However, we can assess its reliability because of the

data nature. Deposit rates are not changed on a regular basis which gives the ability to fill in the missing data if any problems arise in most cases.

One of the features of the tool is that it cannot collect data on the popularity of particular deposits or different rates for various types of customers. It dismisses the ability to weigh different deposit rates between each other based on volumes of those deposits thus making it hardly possible to obtain aggregated data from the scrapped one. The same story goes for the difference in volumes of deposits between banks. However, the assumption about the similar role of all banks as market players are able to be made thus the research will provide us with beneficial results, being open to further improvement and deeper analysis with the data from bank reports about their deposit volumes.

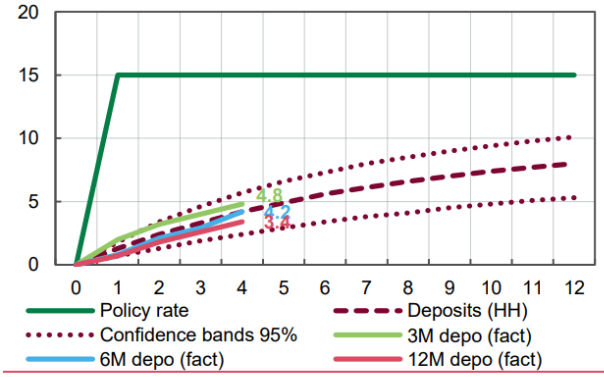
There is a key hypothesis that the dynamics of rates between banks in the same group by ownership (state-owned banks, banks with private capital, and banks with foreign capital), by client type (consumers-oriented, business-oriented) or by another feature are closer to each other than the dynamics between banks in different groups. Further analysis will be done to test this hypothesis and investigate the potential reasons for such similarities or differences. This opens a wide field for using combinations of time series distance measuring and clustering techniques, one of which will be used for forecasting in this paper.

Another interesting issue that will be discussed in the upcoming paper is that deposit rates may change infrequently, resulting in long periods with similar values, which can be problematic for similarity measures, such as the Dynamic Time Warping (DTW) algorithm, that is used to find similarities between time series. This point will be discussed to a more extent in the model section.

There should be a connection between the deposit rates and the monetary policy of the central bank, with a lag between changes in the policy rate and changes in deposit rates. This is because the central bank policy rate affects the cost of funds for banks and, thus, the rates they offer on deposits. Therefore, analyzing the data in the context of the central bank policy rate could provide insight into the behaviour of banks and the broader macroeconomic environment. Counting for such an effect will help to adapt the dynamics of each particular bank and extract the similarity based on this particular factor which will open a field to study similarities that are based on other reasons, for example, the bank type.

This paragraph enhances yet another possibility of potential additions to the data and the pool of models respectively. Such a relationship was investigated by many institutions and researchers, including National Bank of Ukraine economists using the ARDL model by Pesaran and Shin (1995). The results can be seen in Picture 2, which is taken from the Inflation Report, October 2022 by the National Bank of Ukraine.

Picture 2. Change of deposit rates as a response to the shock in NBU Policy Rate



* NBU estimates for the period 2015–2021 are based on the ARDL model of Pesaran and Shin (1995);
** 5-day (working days) moving average UIRD.
Source: NBU.

3. Model

In this research, we analyze five forecasting models for deposit rates:

- Random Walk,
- ARIMA(1,1,1)
- ARIMA(7,0,7)
- Random Forest
- Multi-stage key model

The first four models are, to some extent, traditional benchmarks to compare with the final and the most interesting model. The idea is to show that the DTW/Clustering routine is superior and gives a lot of additional information for both analysis and forecasting purposes. This will also support the initiative for further development of the methodology to fit such problematic data in the upcoming research.

3.1. Random Walk

The Random Walk model (RW) supposes that the potential value of a series will be the same as its most recent value. This approach is frequently used as a benchmark for other prediction models because of its simple nature of use and lack of assumptions about the underlying data generation process. Mathematically, RW can be expressed as:

$$y_{t+1} = y_t$$

where y_t is observed variable at the time t and y_{t+1} is the predicted at the time t+1. A prediction for the further horizon can be built iteratively.

The Random Walk model ignores any other variables that can have an impact on the series, such as some seasonality, shocks from other variables, or dynamics that are not linear since it assumes that the future value of the series will be identical to the last known value of the series. As a result, this model is frequently used as a primitive forecasting model or as a benchmark against which to measure how well more sophisticated models perform. However, there is also a great field to study the model deeper with different additions like in the paper by Dshalalow, 2021.

3.2. ARIMA

The ARIMA (Autoregressive Integrated Moving Average) model is a time-series statistical model that is frequently applied. It is a framework with several orders of autoregressive (AR) and moving average (MA) components, as well as differencing terms. The model's autoregressive term captures the linear connection between an observation and its previous values, while the moving average term captures the linear relationship between an observation and its prior errors. The differencing term is used to eliminate any patterns or seasonality in the series, which is required for the model to be valid.

Mathematically, the ARIMA(m,0,n) model can be described with the following formula:

$$y_t = \beta_1 y_{t-1} + \cdots + \beta_m y_{t-m} + \gamma_1 \varepsilon_{t-1} + \cdots + \gamma_n \varepsilon_{t-n} + \varepsilon_t$$

where y_t is a variable at the time t, ε_t is an error term at time t, β_k and γ_k are estimated coefficients. A middle, differencing term describes whether we're using y_t or its difference of the corresponding degree.

There are some limitations to the ARIMA model. One of the most significant constrictions is the inability to seize dependencies with other variables. The model only considers the series' historical values and neglects any other variables that may influence the series. If other variables have a strong relationship with the series but are not included in the model, this can be a major drawback.

Another limitation of the ARIMA model is its incapability to detect non-linear data patterns. The model presumes a linear connection between the series and its previous values and errors. If there are non-linear patterns in the data, such as exponential growth or decay, the model may not be able to capture these patterns accurately. More advanced issues that arise with such type of a model and corresponding tests are discussed in the article by Gandhi, 2020.

Two specifications have been chosen for this research based on the results of a wider set of models for such a type. The ARIMA(1,1,1) model is the most straightforward choice and a great benchmark which is not far away by its properties and capabilities from the previous model. It's relatively simple to interpret and gives better results than the previous approach. The ARIMA(7,0,7) model is another ARIMA-based model, which is more complex than the ARIMA(1,1,1) and is capable of capturing more complex patterns in the data, based on the dynamics during the last week. This may be important as long as deposit rates may change due to some shock, but with a minor delay which can't be captured well with one lag only.

The ARIMA model is traditional for econometrics and time series analysis which leads to its usage on yearly, quarterly or monthly data. Usage of ARIMA on daily data may lead to many additional problems such as the inability to work with yearly seasonality. However, when there is a necessity for short-term forecasts, it performs relatively well according to a number of researches, especially in the area of stock price forecasting like in the paper by Almasarweh and Al Wadi, 2018.

With such a methodology we cannot directly dig out all of the available information from all the banks and are limited to working with an average rate only. This means that there is a huge field for further improvement that will be utilized by the following two models that will give an additional advantage and possible improvement of the result.

3.3. Random Forest

The Random Forest is a powerful Machine Learning algorithm that is often used for regression and classification tasks. It is an ensemble learning method that combines the predictions of multiple decision trees to improve the accuracy and robustness of the model. The spike in the popularity of Data Science techniques has led to many papers with Random Forest as a main or secondary (benchmark) model for financial and even macroeconomic series. For example, Yoon studied Random Forest for Real GDP forecasting in 2021.

RF is particularly useful for capturing overall dynamics and non-linearity in the data. Unlike linear models such as ARIMA, the Random Forest model can capture complex non-linear relationships between the features and the target variable which might be the case for the current dataset. This is due to the model's use of decision trees, which can simulate non-linear relationships and feature interactions. Because RF can

process data in parallel and does not require the same level of preprocessing as ARIMA, it can handle larger datasets more efficiently. Our goal is to forecast the average rate across all banks in Ukraine, but the dataset includes rates from individual banks. While ARIMA is limited to a single time series and may not be able to capture the overall dynamics of the dataset, RF models can use that information to make more accurate predictions of the average rate.

Each decision tree in the RF model is trained on a randomly selected subset of the attributes and a stochastic subset of the training data. This diminishes overfitting and raises model diversity, improving generalization ability. Averaging the predictions from each of the decision trees in the forest yields the final prediction.

A random subset of the training data is chosen for each tree in the forest at each stage of the training process. This helps to avoid overfitting and improves tree variety. For each tree in the forest, a random subset of the features is selected to use for splitting the nodes of the tree. This helps to reduce the correlation between the trees and improve the generalization ability of the model. Each tree in the forest is built using a recursive process of binary splits, where each node of the tree represents a test on one of the selected features. The tree is grown until some stopping criteria are met, such as a minimum number of samples per leaf node. The final prediction of the model is made by aggregating the predictions of all the decision trees in the forest by taking the mean in our case.

The RF is trained using an ensemble of decision trees, each of which is trained to minimize the variance of the target variable. The variance of the target variable y is defined as:

$$Var(y) = E\left(\left(y - E(y)\right)^2\right)$$

where $E(y)$ is the expected value of y. The goal of the RF is to minimize the variance of the target variable by creating an ensemble of decision trees that have low correlation and high individual predictive power.

The splitting criteria used for each node of the decision tree is based on the Gini impurity, which quantifies the homogeneity of the samples at each node and is used to select the feature and split point that maximizes the information gain of the split.

The Gini impurity index measures the degree or probability of a particular data point being misclassified when it is randomly labelled according to the class distributions in the dataset. The Gini impurity index ranges from 0 to 1, where a value of 0 indicates a perfectly pure node (all samples belong to the same class), and a value of 1 indicates a completely impure node (samples are evenly distributed among all classes). It is given with the formula:

$$Gini = 1 - \sum(p_i^2)$$

where p_i is the probability of the i-th class.

The Gini impurity index is used to assess the quality of each potential split when building a decision tree. The Gini impurity is computed for each possible split, and the split with the lowest Gini impurity is chosen. This process is repeated for each successive split until a stopping criterion is met (e.g., a maximum depth is reached, or the number of samples in a node falls below a specified threshold).

When making a prediction for a new sample, each tree in the forest returns a prediction, and the final prediction is determined by a majority vote. This approach helps to reduce overfitting and improve the generalization performance of the model. The formula can be written as:

$$\hat{y} = \frac{1}{N} * \sum(\hat{y_i})$$

where $\hat{y_i}$ is the predicted value of the i-th decision tree in the forest, and N is the total number of trees in the forest.

Key parts of the RF algorithm are focused on making a generalized and robust forecast, capturing potential non-linearities by aggregating many models with different responses to the same shocks. Also, it works well with huge datasets without overfitting which is crucial for the particular work. For more discussion about the properties, advantages and drawbacks of the algorithm the one can refer to the paper by Biau, 2012.

3.4. Multi-stage key model

For the final model, there is a multi-stage approach that takes all rates, aggregates them into groups with Dynamic Time Warping (DTW) and clustering, then forecasts each group separately with ARIMA and, finally, aggregates it with weights, obtained using OLS estimation. Such an approach allows squeezing the maximum of the information from the dataset and opens a wide field for analysis both from the economic and data analysis points of view.

3.4.1. DTW

Let's cover the algorithm step-by-step. The first part is to find a similarity in dynamics between deposit rates. There are several main approaches for this stage that are described in the paper by Krukovets, 2020. In this particular paper, we'll focus on the DTW only, the most advanced and the one, that is suitable

for signals that vary in speed because they do not consider the time-axis. The last point is important because rates in different banks may react to shocks with a significant delay.

We can compute the distance matrix between the two signals, D, where $D(i,j)$ represents the distance between x_i and y_j. We can calculate a distance between two series with a variety of methods, including a Euclidean distance, Manhattan distance, or Minkowski distance. The idea of DTW is to find a path, P, through this distance matrix that minimizes the total distance between the two signals. The path is constrained to start at (1,1) and end at (n,m). It can only move to adjacent cells in the matrix or by diagonal, which doesn't allow any jumps or gaps in correspondence.

To find the optimal path, we use a dynamic programming approach. We define a cost matrix, C, where $C(i,j)$ represents the minimum cost of the path that ends at (i,j). We can compute $C(i,j)$ recursively using the following equation:

$$C(i,j) \leftarrow D(i,j) + \min\big(C(i-1,j), C(i,j-1), C(i-1,j-1)\big)$$

The intuition behind this equation is that the cost of the path that ends at (i,j) is equal to the distance between x_i and y_j, plus the minimum costs of the three possible paths that could have led to (i,j). The proof that the total distance is minimized is trivial using mathematical induction.

The Dynamic Time Warping (DTW) algorithm is sensitive to the length and magnitude of similarities between the time series data. In the case of long periods with similar deposit rate values, this may result in misleading similarity measurements. One way to address this issue is by aggregating the data on a weekly or monthly basis, instead of daily. Such an approach will reduce the impact of long periods with similar values and increase the sensitivity of the DTW algorithm to shorter-term variations in the data. Alternatively, a modified version of the DTW algorithm that takes into account long stretches of similar values can be used. However, this topic will be studied and clarified in the upcoming paper.

An important point is that the adaptation of the DTW algorithm called FastDTW will be used. It is quite consequentially that the algorithm is focused on adaptations that allow working faster than the original one. While both algorithms find the optimal warping path between two time series by calculating the distance between each pair of points in the series, FastDTW achieves this goal by using a lower-resolution version of the series and then iteratively refining the warping path using increasingly higher resolutions. This allows FastDTW to reduce the number of comparisons required to find the optimal warping path, making it faster than DTW but sacrificing some accuracy. FastDTW also employs a technique called "masking" to further reduce the number of comparisons required to find the optimal warping path. Masking works by dividing the time series into subseries and computing the distance between only certain points in each subseries, ignoring others. This is achieved by creating a binary mask that specifies which points in each subseries should be considered when computing the distance. The mask is created by finding the median point in each subseries and selecting a range of points around it, with the size of the range depending on a user-defined parameter called the "radius." By using masks, FastDTW is able to limit the number of comparisons required to find the optimal warping path, making it even faster than the standard FastDTW algorithm. However, the trade-off is that the accuracy of the warping path may be reduced, especially if the radius parameter is set too large. In more detail, it is discussed in the original paper by Salvador and Chan, 2004.

The used DTW version is made in a package for R, which is described by the library author in the paper Giorgino, 2009. It consists built-in FastDTW algorithm and several additional features that make work with an algorithm simpler.

After finding distances between all deposit rates in the previous exercise, we're building a matrix that will be translated into a 2D graph with approximately the same distances using affine transformations. The explanation and proof of the minimization for approximation error is quite cumbersome and can be accessed from chapter 5 of the book by Datorro, 2019.

3.4.2. Clustering with K-Means

The next part of the routine is a clustering exercise. The methodology was chosen using a graphical analysis of the graph based on the distance matrix from the previous exercise, which will be described in the next chapter. A simple K-means algorithm was chosen. It is an unsupervised machine learning algorithm used to cluster data points into K distinct groups or clusters. The algorithm works by iteratively assigning data points to the nearest cluster center, and then updating the cluster centers based on the newly assigned data points. The algorithm converges when the assignments and the cluster centers no longer change or a maximum number of iterations is reached.

Given a set of N data points $X = \{x_1, x_2, \dots, x_t\}$ and a number of clusters K, the algorithm seeks to minimize the sum of squared distances between each data point and its assigned cluster center. This is known as the within-cluster sum of squares (WCSS) and can be expressed as:

$$WCSS = \sum_{i=1}^{K} \sum_{j=1}^{n_i} \left|\left|x_j - c_i\right|\right|^2$$

where n_i is the number of data points assigned to the i-th cluster, c_i is the center of the i-th cluster, and ||.||^2 denotes the Euclidean distance.

To find the optimal cluster centers that minimize the WCSS, the K-Means algorithm uses an iterative approach known as Lloyd's algorithm. In each iteration, the algorithm first assigns each data point to the nearest cluster center using the following formula:

$$argmin \left|\left|x_j - c_i\right|\right|$$

where argmin is the function that returns the index of the cluster center that minimizes the distance to the data point x_j.

Once all data points have been assigned to a cluster, the algorithm updates the cluster centers by computing the mean of all data points assigned to that cluster using the following formula:

$$c_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x_j$$

where n_i is the number of data points assigned to the i-th cluster, and x_j is the j-th data point assigned to the i-th cluster.

The algorithm repeats these steps until the assignments and the cluster centers no longer change or a maximum number of iterations is reached.

In more detail, the explanation and discussion may be accessed from the paper by Jie et al., 2020.

Instead of the exercise above, with DTW and clustering, we can assume that the dynamics in groups are divided by some features (bank ownership, such as state, private or foreign; client type orientation such as consumers, small or large business). Such an approach has many downfalls, such as a necessity for a deeper understanding of economic strategies and the relationship of individual banks. As long as the focus of the paper is on the usage of advanced forecasting techniques, the algorithm-based solution for grouping seems to be a more viable option.

In the end, we've got several groups of banks. Their rates are aggregated as an average in each group which will give multiple series that will be used in the following stages.

3.4.3. AutoARIMA

To forecast average rates for each group of banks, we'll use an advanced version of the traditional ARIMA. It's called an AutoARIMA and, as it comes from the name, the algorithm automatically selects the best ARIMA model for a given time series data. It estimates the parameters of the ARIMA model by maximizing the Akaike Information Criterion (AIC).

AIC is a statistical measure that evaluates the goodness of fit of a model while taking into account the complexity of the model. The AIC is calculated as a combination of two terms: the first term measures the goodness of fit of the model to the data, while the second term penalizes the model for its complexity. The formula for AIC is:

$$AIC = -2 * \log(L) + 2 * k$$

where L is the maximum likelihood of the model; k is the number of parameters in the model.

In AIC, a lower value is better, meaning the model with the lowest AIC value is preferred. A difference in AIC values of two or more between models suggests strong evidence for the model with the lower AIC value.

The AutoARIMA algorithm uses a stepwise approach to select the optimal ARIMA model. First, it selects the best ARIMA model with no differencing (i.e., d=0) going through all possible parameters (grid search) and evaluating AIC. Then, it evaluates the AIC for ARIMA models with differencing (i.e., d=1 or d=2) and selects the best model. Finally, it selects the best seasonal ARIMA (SARIMA) model if the time series data exhibit seasonality. For more details and how they're made in R, the one may read the paper by Hyndman and Khandakar, 2008.

At the end of this stage, there will be a single ARIMA model for each group that will be able to produce a forecast.

3.4.4. OLS-based aggregation

The final stage is a model, that will help to aggregate forecasts from the previous stage into a single average rate. For this, we'll use a simple OLS approach that is evaluated on the historical aggregated rates. Such an approach, as many other and more difficult ones, is overviewed in the paper by McAndrew et al., 2020. The formula is simple:

$$y_t = \beta_1 y_{t,1} + \dots + \beta_k y_{t,k} + \varepsilon_t$$

where y_t is an average rate for all banks at the moment t; $y_{t,k}$ is an average rate for all banks from group k.

This allows us to check the importance of the dynamics of a particular group in the overall dynamics, because groups may have only a few banks

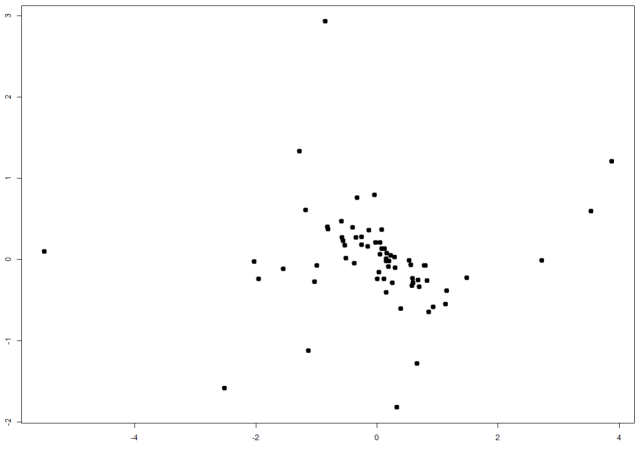
(outliers) or a majority of banks. Such an approach will be even more beneficial for further investigation in the case when the data of bank size is used, thus some particular banks may have way more weight on the overall dynamics than others which will be reflected in the coefficients of the linear regression.

Results

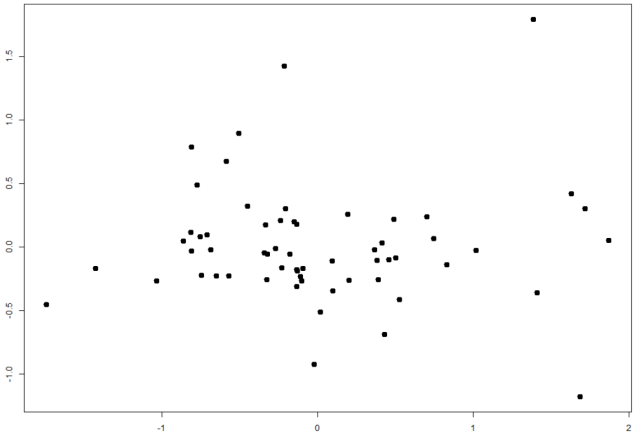
This section will show the results of the five models described above based on the data, that is described in the data section. While for benchmark models the forecasting process is relatively straightforward, our key method requires multiple stages to produce the final result.

The result of the DTW approach is given in Picture 3. Each point represents a particular bank and distances between points are approximated to the ones from the distance matrix based on DTW distances. The result shows a number of outliers that will decrease the quality of the next stage. Moreover, such a difference in dynamics can be explained by the economic conditions and strategies, some of the banks are frauds that were removed from the market at some point in time. Thus, in further analysis, we've removed those banks and re-run the exercise. The result is given in Picture 4 and will be passed further on to the next stage.

Picture 3. Banks, distances between which are calculated with DTW.

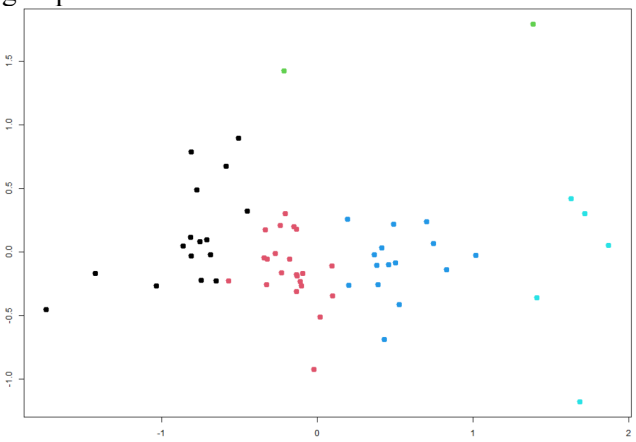


Picture 4. Cleaned from outliers pool of banks, distances between which are calculated with DTW.



The clustering stage contains a K-Means approach based on the result from Picture 4. It will divide these points into several groups. The amount of groups is defined by the graphical analysis of the results and common logic, which states that a number of groups can't be too small or too large. It was found that the optimal number of groups is five and the resulting picture is given in Picture 5.

Picture 5. A cleaned pool of banks divided into groups with K-Means.



Then the AutoARIMA finds optimal coefficients that will minimize the AIC. Thus, we may be sure that we'll obtain a forecast with good quality for each individual group. Then these groups are aggregated with an OLS method. Key statistics, including the number of banks in a group, ARIMA specification and weight according to OLS may be seen in Table 1.

Table 1. Specifications for each group.

Group number	1	2	3	4	5
Number of banks	16	22	2	15	5

ARIMA specification	(1,1,2)	(1,1,1)	(0,1,0)	(2,1,3)	(0,1,0)
OLS weight	0.267	0.367	0.033	0.250	0.083

To evaluate the quality of the forecasts, according to the amount of available data, we've decided to simply forecast the last month of rates and compare it with the real values both graphically and with Root Mean Squared Error (RMSE), a traditional way to evaluate the quality of the forecast. RMSE is given with the formula:

$$RMSE = \sqrt{\frac{1}{n} * \sum((y_i - \hat{y}_i)^2)}$$

where n is the total number of observations; y_i is the actual value of the i-th observation; $\hat{y}_i$ is the predicted value of the i-th observation.

The graph with the historical average rate and forecasts for each model can be seen in Picture 6. Corresponding RMSEs are provided in Table 2.

Picture 6. Average deposit rate and its forecast according to the estimated models.

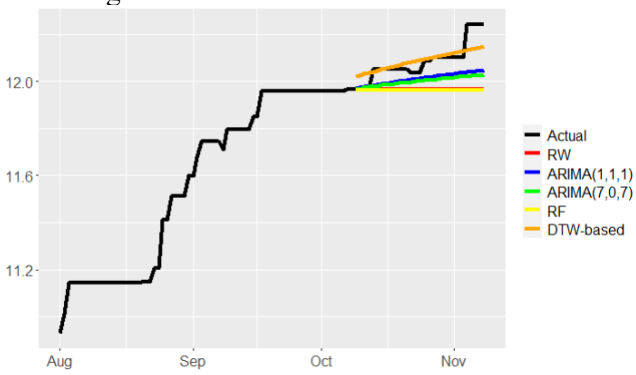


Table 2. RMSE for all models, calculated on the last available month.

Model	RW	ARIMA (1,1,1)	ARIMA (7,0,7)	RF	DTW-based
RMSE	0.143	0.095	0.107	0.146	0.050

Several key results can be dug out from the graph and table. Our key algorithm, which is based on the DTW routine, outperforms all other methods. ARIMA models work pretty well too. Random Forest was unable to outperform Random Walk and gives bad results.

Graphical analysis shows that on the current dataset, all models were able to produce lines with relatively small deviations. Such a result is acceptable for models that do not have information about the future (exogenous variables, assumptions). For even better and more robust results another approach for model quality evaluation such as rolling pseudo-out-of-sample forecasts may be used in the following research. Despite poor results in this exercise, Random Forest is able to give good results and find non-linearities still. However, it needs more fitting (because in this paper it was a simple benchmark model) and will also benefit from another method of evaluation.

Conclusion

The paper shows how a novel multi-stage approach outperforms several traditional benchmarks on the unique dataset, which consists of deposit rates of Ukrainian banks that were scrapped from the abovementioned banks' websites. Despite the mostly technical focus of the paper and model design, it opens a wide field for further improvements with area-specific adjustments to the model, and additional data that may explain some part of the variation due to the economic interconnection. Moreover, such an approach provides us with the tool for the economics-oriented paper that will study new insights that have been caught with a DTW and clustering approach.

The work highlights a promising way to improve the model by solving one of the problems that the model faces. The data consists of series that are given on the daily basis but vary rather rarely creating long sequences with zero variance. It may be harmful to the DTW algorithm due to its design, thus an upcoming work dedicated to the issue will be favourable.

Also, there are multiple ways to improve and evolve the current results of the paper with the usage of other mechanisms to evaluate the model quality and by spending more time to fit the RF model, which wasn't a key point of interest in the current paper.

References

1. KHALIL, S., FAKIR, M. (2017): *RCrawler: An R Package for Parallel Web Crawling and Scraping.* "SoftwareX", Volume 6, pp.98-106.
2. KHDER, M. (2021): *Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application.* International Journal of Advances in Soft Computing and its Applications, 13(3), pp.145-168.
3. PESARAN, H., SHIN, Y. (1995): *An Autoregressive Distributed Lag Modeling Approach to Co-integration Analysis.* "In S. Strøm (Ed.), Econometrics and Economic Theory in the 20th Century: The Ragnar Frisch Centennial Symposium (Econometric Society Monographs), Cambridge: Cambridge University Press", pp. 371-413.
4. NATIONAL BANK OF UKRAINE (2022): *Inflation Report, October 2022.*
5. DSHALALOW, J., WHITE, R. (2021): *Current Trends in Random Walks on Random Lattices.* "Mathematics", 9(10), pp. 11-48.
6. GANDHI, P. (2020): *7 Statistical Tests to validate and help to fit ARIMA model*.
7. ALMASARWEH, M., WADI, S. (2018): *ARIMA Model in Predicting Banking Stock Market Data.* "Modern Applied Science", Vol. 12, No. 11.
8. YOON, J. (2020): *Forecasting of Real GDP Growth Using Machine Learning Models: Gradient Boosting and Random Forest Approach.* "Computational Economics", volume 57, pp. 247–265.
9. BIAU, G. (2012): *Analysis of a Random Forests Model*. "Journal of Machine Learning", Volume 13, pp. 1063-1095.
10. KRUKOVETS, D. (2020): *Analysis of similarity between artificially simulated time series with Dynamic Time Warping.* "Proceedings of Workshop on Intelligent Information Systems WIIS2020", pp.97-108.
11. SALVADOR, S., CHAN, P. (2004): *FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space.* "Intelligent Data Analysis", 11(5), pp.70-80.

Список використаних джерел

1. Khalil, S. RCrawler: An R Package for Parallel Web Crawling and Scraping. / S. Khalil, M. Fakir, // SoftwareX. – 2017. – Volume 6 – P. 98-106.
2. Khder, M. Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application. / M. Khder // International Journal of Advances in Soft Computing and its Applications. – 2021. – Volume 13 – Issue 3 – P. 145-168.
3. Pesaran, H. An Autoregressive Distributed Lag Modeling Approach to Co-integration Analysis. / H. Pesaran, Y. Shin // In S. Strøm (Ed.), Econometrics and Economic Theory in the 20th Century: The Ragnar Frisch Centennial Symposium (Econometric Society Monographs). – Cambridge University Press – 1995 – P. 371-413.
4. National Bank of Ukraine. Inflation Report, October 2022. – 2022
5. Dshalalow, J. Current Trends in Random Walks on Random Lattices. / J. Dshalalow, R. White // Mathematics. – 2021. – Volume 9 – Issue 10 – P. 11-48.
6. Gandhi, P. 7 Statistical Tests to validate and help to fit ARIMA model. / P. Gandhi // 2020
7. Almasarweh, M. ARIMA Model in Predicting Banking Stock Market Data. / M. Almasarweh, S. Wadi // Modern Applied Science. – 2018. – Volume 12 – Issue 11
8. Yoon, J. Forecasting of Real GDP Growth Using Machine Learning Models: Gradient Boosting and Random Forest Approach. / J. Yoon // Computational Economics. – 2020. – Volume 57 – P. 247–265.
9. Biau, G. Analysis of a Random Forests Model. / G. Biau // Journal of Machine Learning. – 2012. – Volume 13 – P. 1063-1095.
10. Krukovets, D. Analysis of similarity between artificially simulated time series with Dynamic Time Warping. / D. Krukovets // Proceedings of Workshop on Intelligent Information Systems WIIS2020. – 2020. – P. 97-108.
11. Salvador, S., Chan, P. FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space. / S. Salvador, P. Chan // Intelligent Data Analysis. – 2004. – Volume 11 – Issue 5 – P. 70-80.

12. GIORGINO, T. (2009): *Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package.* "Journal of Statistical Software", 31(7), pp. 1–24.
13. DATORRO, J. (2019): *Convex optimization and Euclidean distance geometry*.
14. JIE, C., JIYUE, Z., JUNHUI, W., YUSHENG, W., HUIPING, S., KAIYAN, L. (2020): *Review on the Research of K-means Clustering Algorithm in Big Data.* "2020 IEEE 3rd International Conference on Electronics and Communication Engineering (ICECE)", Xi'An, China, 107-111.
15. HYNDMAN, R., KHANDAKAR, Y. (2008): *Automatic Time Series Forecasting: The forecast Package for R.* "Journal of Statistical Software", 27(3), 1-22.
16. MCANDREW, T., WATTANACHIT, N., GIBSON, GC., REICH, NG. (2021): *Aggregating predictions from experts: a review of statistical methods, experiments, and applications.* Wiley Interdiscip Rev Comput Stat, 13(2), e1514.

12. Giorgino, T. Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package. / T. Giorgino // Journal of Statistical Software. – 2009. – Volume 31 – Issue 7 – P. 1-24.
13. Datorro, J. Convex optimization and Euclidean distance geometry. / J. Datorro .
14. Jie, C. Review on the Research of K-means Clustering Algorithm in Big Data. / C. Jie, Z. Jiyue, W. Junhui, W. Yusheng, S. Huiping, L. Kaiyan // 2020 IEEE 3rd International Conference on Electronics and Communication Engineering (ICECE), Xi'An, China. – 2020. – P. 107-111.
15. Hyndman, R. Automatic Time Series Forecasting: The forecast Package for R. / R. Hyndman, Y. Khandakar // Journal of Statistical Software. – 2008. – Volume 27 – Issue 3 – P. 1-22.
16. McAndrew, T. Aggregating predictions from experts: a review of statistical methods, experiments, and applications. / T. McAndrew, N. Wattanachit, GC. Gibson, NG. Reich // Wiley Interdiscip Rev Comput Stat. – 2021. – Volume 13 – Issue 2

Надійшла до друку 30.09.2022