Intelligent System for Processing and Forecasting Financial Assets and Risks

Nickolay Rudnichenko[1], Vladimir Vychuzhanin[1], Tetiana Otradskya[1] and Denys Shvedov[1]

[1] *Odessa Polytechnic National University, Shevchenko Avenue 1, Odessa, 65001, Ukraine*

Abstract

The article contains a description of the results of development and research of intelligent system for processing, accounting and forecasting financial assets and risks. An analysis of current problems and technical aspects related to the assessment of financial flows and risks of loss of assets in the context of modern market development in the field of accounting and forecasting of financial time series was carried out. A justification is given for the feasibility and effectiveness of using machine learning methods to automate the solution of operational and analytical problems in the field of accounting for financial assets. The proposed concept of using machine learning within the framework of the developed application software is described, the key aspects of its design and implementation of the software and modular structure are considered. The results of a comparative analysis of the created ACF and PACF models, assessment of the quality metrics of robotic models in different conditions when assessing financial flows and risks are presented, an analysis of the results obtained is carried out and further ways of developing the approach proposed in the article to improve its efficiency are considered.

Keywords

Data analysis, finance risks, financial time series, machine learning, software development, automatic correction of trend and seasonality

1. Introduction

1.1. Research relevance

In the modern world, where data volumes are constantly growing, the importance of accurate accounting and forecasting of financial indicators becomes particularly significant. The relevance in this field primarily lies in the increasing need for precise and efficient tools capable of automating operational and analytical actions on data from financial flows, taking into account the risks of incorrect investments [1].

This not only allows optimizing investors' budgets but also provides client companies with means for more effective management of material resources in terms of planning potential revenues and various expenses, ensuring the principles of overall financial stability.

In the context of formalizing the process of processing data from financial flows or indicators, it should be noted that their nature can vary. Specifically, data from the outset may be well structured, semi-structured, or unstructured [2].

Structured financial data have a clearly defined format, often adhering to a relational database model, characterized by a standard hierarchical order.

They are highly organized, easily accessible, and directly utilized for analysis. In well-defined schemas and collections, structured data is predominantly stored in tabular formats. For instance, names, dates, addresses, credit card numbers, stock information, geolocation, etc., serve as examples of structured financial data [3]. Semi-structured financial data are often not stored in relational databases like structured data but possess certain organizational properties that facilitate their analysis.

CMIS-2024: Seventh International Workshop on Computer Modeling and Intelligent Systems, May 3, 2024, Zaporizhzhia, Ukraine

nickolay.rud@gmail.com (N. Rudnichenko); icst_nuop@ukr.net (V.Vychuzhanin); tv_61@ukr.net (T. Otradskya); studylearnerstudy@gmail.com (D. Shvedov);
0000-0002-7343-8076 (N. Rudnichenko); 0000-0002-6302-1832 (V.Vychuzhanin); 0000-0002-5808-5647 (T. Otradskya); 0009-0002-4823-8782 (D. Shvedov)

Examples of such data include risk profiles, investment plans, and various financial accounting data. HTML, XML, JSON documents, NoSQL databases, etc., are the examples of semi-structured data types used in financial analysis.

For unstructured financial data, there is no predefined format or organization, significantly complicating their collection, processing, and analysis, as they mostly consist of textual and multimedia materials. Examples of unstructured financial data include dividend or investment reports, emails, blog posts, wikis, text documents, PDF files, images, presentations, web pages, and many other types of business documents containing valuable financial information [2, 4].

Within the framework of managing various types of data from financial assets of users or companies, the promising approach is the application of modern machine learning (ML) methods. This direction opens up a range of possibilities for automating the processing and analysis of large volumes of diverse financial data (executed transactions for the payment of goods, receipt of services, acquisition of securities, crypto assets, or other investment contributions). It allows uncovering hidden patterns and dependencies that may not be easily identified by humans in manual mode or through the use of specialized deterministic calculation programs [5].

The advantages of modern ML algorithms include their ability to automatically adapt to dynamic changes in financial profiles and investment risks of users, allowing an increase in the accuracy of data forecasting. ML algorithms can identify patterns and relationships in data that may be non-obvious to humans. This enables precise financial forecasts and informed investment decisions based on the analysis of diverse data.

Consequently, the application of ML in the field of financial risk assessment holds significant practical interest. Specifically, accounting and forecasting of financial indicators can be utilized in various sectors such as banking, financial consulting, asset management, and personal finance. The accuracy of such forecasts can greatly impact decision-making in these areas, particularly in risk management, budget planning, and investment planning [6].

Among the features of applying supervised ML algorithms, it is important to note that the models created upon these algorithms form a solution basis for tasks, relying mostly on information obtained from the input dataset. Another important characteristic of ML models is their ability to self-improve, meaning the formalization and utilization of acquired experience based on a data-driven approach. In fact, the more diverse and balanced data models analyze, the more accurate their predictions become. This is particularly relevant in sectors where high accuracy is required, such as finance, medicine, or security systems [7].

The primary scientific challenge in this context is achieving and instilling in the model a high level of generalization ability to effectively solve tasks on new, previously unknown data not used during training, with a minimal level of errors. This is why, when using ML in the field of financial analysis, it is necessary to integrate approaches from other applied and theoretical scientific directions, including mathematical statistics, optimization methods, and traditional mathematical domains. It is important to take into account the unique characteristics of model-building algorithms related to both non-functional aspects (computational efficiency or speed) and functional aspects (overfitting issues, sample balance, hyperparameter tuning) of ML utilization [8].

Various methodologies are used for forecasting financial data and creating client profiles, typically based on the analysis of historical data and influencing factors. This involves establishing statistical relationships between different characteristics and developing predictive models. Previously, the single-factor forecasting methods were dominant, relying on time series and regression analyses. However, not all of these approaches are sufficiently effective for assessing financial indicators and investment potential [8]. All of this makes the research topic relevant within the scope of this work.

1.2. Aspects of financial data processing issues

It is necessary to note that the specificity of processing and accounting financial data for their preliminary preparation to address forecasting tasks involves several substantive factors.

Specifically, forecasting in the financial domain is a non-trivial task as it entails analyzing and predicting changes in various financial flows, such as income, expenses, savings, and investments. Particularly in the risk analysis process, it is essential to consider the fact that personal finances have distinctive characteristics that vary in each specific case for different users. This is because they often rely on individual behavioral factors, which can be less predictable than market indicators. This process is crucial for individual financial planning as it helps individuals determine how best to allocate their resources to achieve financial goals [5].

In general, the process of accounting and forecasting finances is accompanied by a series of challenges, among which the following can be highlighted [9]:

- High unpredictability of the external environment. Chosen strategies of investment behavior and risk assessment can be extremely unpredictable due to various unforeseen events, such as sudden fluctuations in exchange rates, seasonal declines or surges in demand and supply, accidents and technological disasters, changes in population employment, political decisions and sanctions, or other types of crises. These events can significantly impact an individual's risk profile and complicate the creation of accurate long-term financial forecasts.
- Income variability. For many investors, income is not constant and may vary from month to month, especially for professionals or experts whose activities are largely commission-based. This variability complicates forecasting and requires a more flexible approach to budgeting and financial planning.
- Behavioral factors. Financial decisions often depend on behavioral factors such as personal beliefs, habits, emotions, and cognitive biases. For example, psychological barriers may hinder effective saving or investing, while impulsive purchases may lead to unplanned expenses.
- Economic and market conditions. External economic factors, such as inflation, interest rates, market fluctuations, and economic crises, can have a significant impact on personal finances. Predicting these conditions is challenging, but they must be taken into account when planning finances [10].

Thus, the specificity of forming and managing data structures regarding financial flows should be considered from the perspective of systematic intelligent analysis. Inaccuracies, data errors, outliers, and data incompleteness can significantly impact the quality of forecasts. In financial time series, there is a considerable amount of "noise" — random fluctuations that do not have an obvious explanation. Distinguishing the "signal" from the "noise" is one of the main challenges in financial modeling and forecasting, leading to various types of risks that are not always easily predictable and quantifiable with a high level of certainty. All of this underscores the relevance of this research and the fulfillment of the stated objective, which involves developing software for the accounting and forecasting of financial assets and risks based on the application of modern ML algorithms to automate the data analysis process.

1.3. Discussion with methods

Currently, there are various approaches to the analysis of heterogeneous data of time financial series of different orders and states (both stationary and non-stationary) in the context of integration with ML models. According to solving problem we should characterize: vector autoregression (VAR), generalized autoregressive conditional heteroskedasticity (GARCH) and automatic correction of trend and seasonality (ARIMA) [8,10].

VAR is a statistical method that allows us to model relationships between several time series simultaneously. A feature of the method is the expression of input variables in the form of linear combinations of past values that they took in the system. Var advantages are:

- taking into account the relationship between several time series, which can be used when analyzing heterogeneous financial data, for example, in the case when the prices of various assets or market indicators depend and influence each other stochastically;
- allows us to formalize the modeling of various emerging effects and delays in relationships in the data, which helps to identify key criteria for the financial series over time;

- there is potential application for assessing the impact of various non-obvious factors on financial risks.

Disadvantages:

- choosing the optimal number of lags and parameters is a labor-intensive and non-trivial process, requiring a large number of experiments on heterogeneous data;
- difficulty in interpreting results in the case of non-stationary processes;
- difficulties in software implementation for risk assessments in the context of their ranking into groups and formalization of the probabilistic factor.

GARCH is used to model and forecast the volatility of financial time series. GARCH allows us to take into account the nature of the occurrence of variability in data, which helps to predict future levels flows volatility, which is used in the securities valuation [9]. The volatility of a time series can change over time and depend on past values of this series; modeling is implemented through conditional heteroscedasticity based on previously obtained values (conditional variance depending on past squared errors). Advantages:

- taking into account variability in data to formalize the data structure, which can increase the accuracy of the forecast;
- adaptability for stationary time series by supporting modeling of dynamic volatility effects, including by clustering data according to internal financial market conditions.

Weakness:

- the limitation of the method in the context of use for big data due to the need to conduct a large number of heterogeneous experiments to obtain reliable results;
- high sensitivity to initial conditions, which can lead to unstable results if the parameters or model specification are incorrectly selected;
- are not effective when the nature of the dependencies of financial time series is initially uncertain (probabilistic).

ARIMA is a statistical method for analyzing and forecasting time series that allows us to model a financial time series as a combination of autoregressive, integrated and moving average components. The process of building an ARIMA model includes the following steps: identifying the necessary parameters based on the analysis of autocorrelations and partial autocorrelations of the time series, estimating the model parameters using the least squares or maximum likelihood method, checking the adequacy of the model, and analyzing the residuals [11]. ARIMA advantages:

- adaptability by supporting automatic correction of trend and seasonality, as well as searching for anomalies and patterns in the data.
- flexibility in accounting for integrated and moving averages through the use of complex components;
- support for forecasting horizon management capabilities, including short-term and long-term time periods, which allows expanding models in composite forecasting scenarios;
- automatic identification of model parameters based on data;
- extensibility of the model through the possibility of aggregated accounting of external factors, which is useful to increase forecasts accuracy in risks assessing.

Disadvantages:

- difficulty of application for short and noisy financial time series with a decrease in forecasting accuracy;
- reduced efficiency when used for high-frequency data (intra-day or online fed in multi-stream collection systems) due to the need for a large number of observations and a high degree of autocorrelation.

As a result of the comparative analysis of the considered methods within the framework of our task, the most appropriate is the use of ARIMA due to the greater degree of adaptability and software implementation possibility in the separate module form.

2. Project concept proposal

Within this research, in addition to developing the software logic for a web application for financial data accounting, the creation of hybrid polynomial regression ML models for data forecasting will be conducted, particularly based on ARIMA. ARIMA represents a unique combination of autoregressive models and moving average models. The advantage of this approach is the ability to use such models for automating the forecasting of time series values (in our case, the values of financial assets) based on their past values, taking into account the obtained error estimates from previous forecasts, providing the models with a higher degree of adaptability [11].

During the data analysis process, it is proposed to perform several preliminary stages. This includes data import and preliminary processing, their structuring, determination of statistical indicators, and assessment of data normality. Thus, it is possible to better understand the nature of the considered data and identify the characteristics of financial risks [12].

In this context, an important factor is the identification of the characteristic trend of seasonality in transactions and investments because it can negatively impact the accuracy of forecasting within a defined horizon. Therefore, it is necessary to exclude this aspect before the forecasting process begins. It is also essential to analyze the variance to determine whether the data changes over time or if the variance is constant, which serves as a criterion for the balance of the data sample.

As it is impossible to empirically check the normality of a large volume of data visually, it is advisable to implement in the created web application a check for data normality using the Jarque-Bera test with the following expression

$$JB = n\frac{s^2}{6} + \frac{(k-3)^2}{24} \quad , \tag{1}$$

where n is the sample size, s is the skewness coefficient, and k is the kurtosis coefficient. For normally distributed data, $s = 0$ and $k = 3$. In this case, the value of JB is equal to 0.

According to the null hypothesis that the data is normally distributed, the Jarque-Bera test has shown that the JB test asymptotically follows the Chi-square distribution in this equation [13].

The definition of the autocorrelation function (ACF) and the partial autocorrelation function (PACF) is also an important concept for determining the models we create. In time series analysis, the ACF shows the degree of linear statistical relationship between the values of a time series. Numerically, the ACF is a sequence of correlation coefficients between the original series and its copy, shifted by a given number of series intervals (this number is called the lag ACF). In general view proposed ACF is defined as follows

$$f(L) = \sum_{L=0}^{m} r_{t,t-L} \ , \tag{2}$$

where m - time series members number, r - correlation coefficient.

The autocorrelation function (ACF) measures the linear predictability of time series at time t, as it only uses lagged values. The partial autocorrelation function allows a stationary financial time series to have partial correlation with its own values. This is convenient in our case due to the support for regressing data values at short lags.

Since most our financial data is stationary, applying ARIMA models requires transforming variables. Therefore, ACF and PACF must be analyzed in combination to determine the order of the model, specifically to minimize forecast errors in model creation.

The complete definition of the ARIMA model involves choosing values for the parameters p, d, and q, which is a complex task. Parameter p represents the order of the autoregressive part, d is the degree of differencing the series, and q is the order of the moving average part. It's worth noting that within the proposed concept of financial data forecasting, a necessary condition is $p > 1$ [14].

The key idea within this concept, reflecting its scientific novelty, is the development and application of the designated ARIMA approach using ML within the framework of the analysis of stationary data on financial transactions based on the probabilistic approach (probability is ranked according to the Harrington desirability function). To reduce the degree of uncertainty when analyzing financial risks, a fuzzy logic model based on the Sugeno algorithm is used. All this forms a hybrid model with significant generalizing capabilities for predicting financial risks over various planning horizons, reducing (formalizing) the overall level of uncertainty in the impact of external factors, which is implemented in software system.

3. Intelligent system implementation

Let's describe the process of implementing the software project based on creating diagrams that formalize its logic and structure. When working with the web application, the user has the option to choose from several functions, some of which are related and have a certain dependency on each other:

- User transactions import. This functionality allows the user to import bank statements (currently supporting statements from monobank in CSV format). The system transforms these statements into internal entities called Transactions. These data are directly used for forecasting the user's income and expenses.
- Verification of imported transactions in a list with the ability to filter and sort.
- Forecasting. The system provides access to the forecasting module, where we can configure certain parameters such as: the type of forecasting (expenses or profits), the forecasting period, or the forecasting horizon, the start date of forecasting, i.e., the date of the last transaction on which the model will be trained and from which the forecast will be generated for the specified number of months.
- Visualization. The system provides the ability to view forecasting results for each type of forecast. The display is presented in the form of cumulative forecast sums and transactions for the forecast period. This will allow for the observation of trends in data changes and correlation between the forecast and actual data.
- Financial risk assessment. Implements functionality to obtain optimistic and pessimistic risk values from increasing expense levels and decreasing profit levels within the forecasting horizon based on the scenario method.

The developed scheme for detailing the stages of operation of the web application for accounting, processing, and forecasting user financial data is shown in Fig. 1. An important detail of the architecture of the created web application is the support for the re-creation of the model on different data fragments, its subsequent training, and use for generating user-created forecasts.

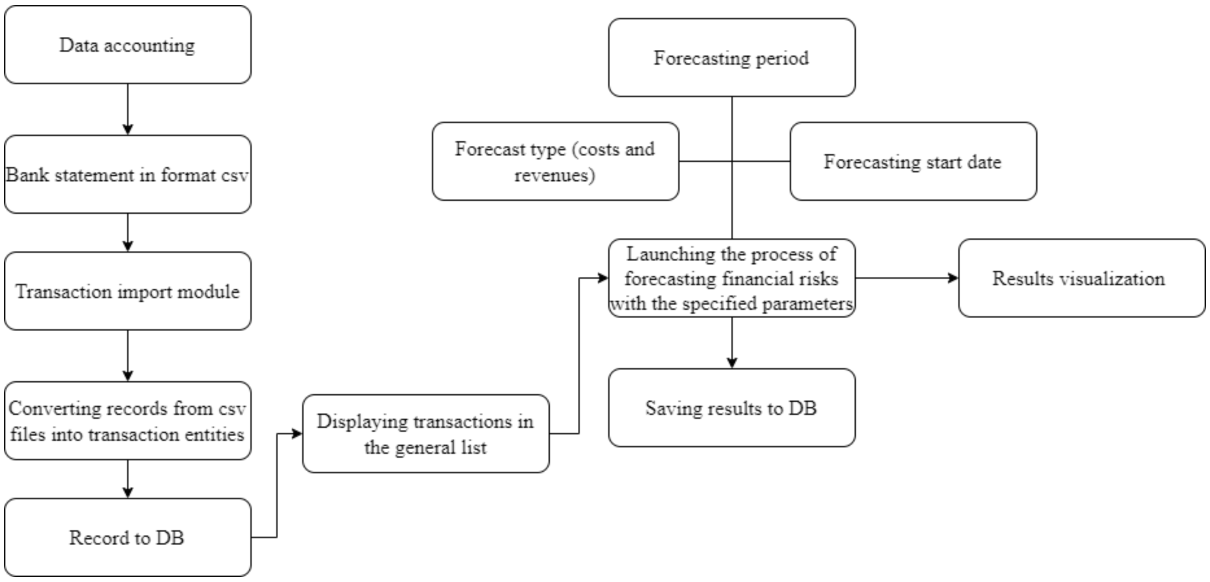

Data accounting
Bank statement in format csv
Transaction import module
Converting records from csv files into transaction entities
Record to DB
Displaying transactions in the general list
Forecasting period
Forecast type (costs and revenues)
Forecasting start date
Launching the process of forecasting financial risks with the specified parameters
Results visualization
Saving results to DB

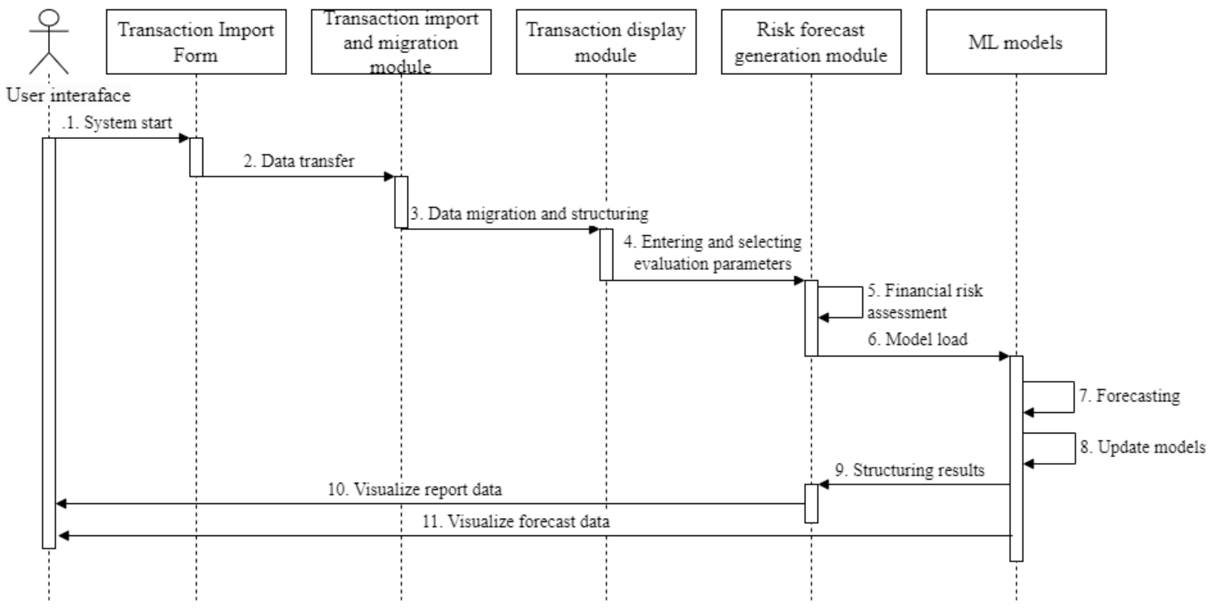
Figure 1: The detailed scheme of the web application's operational stages

The models used in the web application optionally have the ability for dynamic learning, which is necessitated by the need to use new portions of financial data for more detailed analysis and the formation of a more generalized forecast each time a model is generated. To overcome this problem, the "Forecast" entity has been created, which contains the forecasting results and the parameters used in forecasting.

From the perspective of object-oriented design efficiency, this allows for a reduction in computational resource costs. If the dataset and configurations remain unchanged, the forecast result will remain the same.

Therefore, the module will retrieve results from the database without performing additional operations.

To describe the usage process of all modules in the web application during its design, we will use the UML (Unified Modeling Language). The developed general sequence diagram of the application's operation is presented in Fig, 2.



User interaface
Transaction Import Form
Transaction import and migration module
Transaction display module
Risk forecast generation module
ML models
.1. System start
2. Data transfer
3. Data migration and structuring
4. Entering and selecting evaluation parameters
5. Financial risk assessment
6. Model load
7. Forecasting
8. Update models
9. Structuring results
10. Visualize report data
11. Visualize forecast data

Figure 2: The sequence diagram of user actions in the software

After logging into the web application with their account, the user has the option to import transactions into the system through the transaction import form. The user can add a file or files in CSV format, generated by a bank or another financial institution with which they have relationships.

After migrating transactions into the system, the user can download a forecasting model to assess potential profits, expenses, and aggregated financial risks using the forecasting functionality. This module is programmatically linked to a form where the user can configure parameters for forecasting financial risks.

After confirming the parameters, they are passed to the ML control block, which deserializes the models, uses them, compares the parameters with the generated forecasts, and, if the data is correctly specified, forms a set of forecast values for each model.

The result is then sent to the user interface on the web page in the form of a report and graphical dependencies. In case the forecast is not reliable (error exceeds the set value) or the data is invalid, an informational message is displayed to the user in the form of a pop-up window.

From a software perspective, the application is a distributed microservices SPA based on JavaScript programming language, Nodejs platform, and React frontend framework, providing interactive user interaction capabilities. Due to the client-server architecture, the server-side backend and DB are separately launched and deployed.

The backend performs all functional tasks related to data accounting and processing, interacts with the DB for searching, filtering, creating, editing, reading, or deleting performed records.

Additionally, the Python programming language and a set of functional libraries for creating structures to manipulate data, including pandas and numpy, are employed.

Interaction between the backend and frontend parts of the web application is achieved through REST API. The backend interacts with MongoDB using mongoose. The library provides built-in validation for schemas to automate checking data compliance with specified rules and formats before they are saved or updated. The overall architecture of the software is shown in Fig. 3.

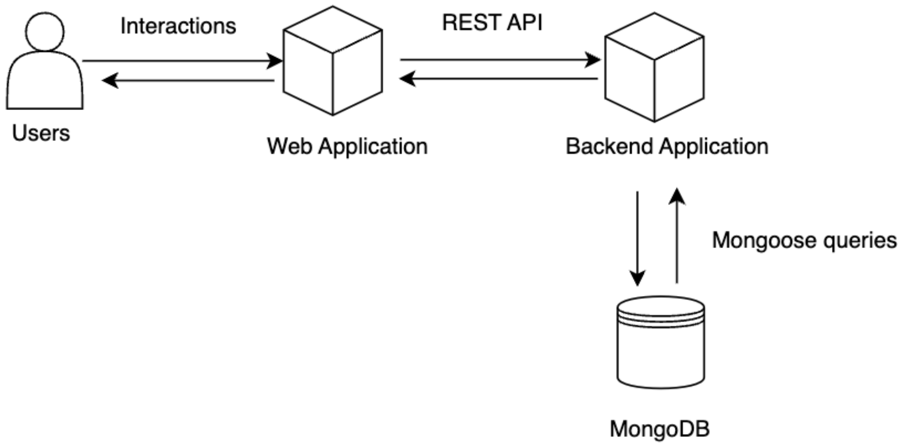


Figure 3: Architectural diagram of the components of the system in a general view

To perform a forecast for the selected model, data is required. In our case, this involves a test dataset of transactions over 3 years for the customer of monobank. Each transaction is described by the Transaction schema, and all transactions are stored in the system for further use in other modules. The Transaction entity retains information about each transaction in a structured format. The main modules that utilize the Transaction entity are the statistical processing and forecasting modules. Overall, the forecasting module performs only a few roles:

- formation of forecasted values for profits, expenses, and financial risks;
- a set of tools for interaction between the web application and a Python script, which is used for deserialization, description, training, and utilization of ML models;
- connection to the statistical module for aggregating transactions used to build graphical visualizations of the reporting results. With this data, the user can correlate real data with forecasted data, evaluate results with specified parameters, conduct additional computational experiments, and perform a consolidated analysis of financial risks, with the further use of data for developing their investment strategy.

The forecasting module uses the Forecast schema to record data about generated forecasts and the parameters used in the process. ForecastResult is a distinct software type, essentially a collection of data that describes a set of financial time series resulting from the model's forecast and has the following structure:

- dateTime [string], the date of one forecasted value of a financial asset, described in the form of an ISO string;
- amount [number], the amount of one forecasted value of a financial asset;
- ForecastOptions, a specialized object data type that describes the parameters used for generating a forecast or calculated for assessing the relevance of the forecast;
- startTime [string], the date from which the forecast creation will commence. It is also used for aggregating transactions; the model should not consider data beyond this date, as otherwise, the results would be inaccurate;
- period [1M, 2M, 3M], the period or forecast horizon. Currently, the software supports forecasting values for 1, 2, or 3 months, due to increasing errors and decreasing forecast reliability with longer periods;
- nTransactions [number], the number of transactions in the system, a parameter used to assess the relevance of the forecast;
- modelVersion [string], the version of the model, an internal parameter that essentially serves as metadata;
- forecastType [income, expenses], the type of financial asset for forecasting.

The user is provided with an authentication and registration system in the web application, so each user has their own personal record, and the User schema is used for this purpose.

The module responsible for importing transactions from the bank statement is the Mono module, which depends on the Transaction schema. Its main functionality involves parsing the bank statement structure in CSV format and migrating the data to the system. The

MonoController, which is an element responsible for managing endpoints for this module in the web application architecture, has only one API endpoint, /api/v1/mono/import. It is designed to receive the input configuration set for importing all transactions, including accountID, date (import start date), and the CSV file itself.

The imported transactions are utilized in the Analytics module, which is responsible for executing forecasts and aggregating transactions for correlation with the forecast. The module operates with the Forecast schema, used to store the forecast results in the DB. The module consists of AnalyticsController and AnalyticsService. AnalyticsController is responsible for the API and interaction with the main functionality in the service.

Below is a description of the code execution sequence for creating the ML model for forecasting:

1. Reading parameters from sys.argv, which are passed as requests when launching the Python process, namely pediod (forecast period) and forecast_type (forecast type). These two parameters are necessary for using factors in constructing the dataframe and determining parameters for the model.
2. Parsing transaction data from data.json using pandas.
3. Sorting and preparing data.
4. Determining model parameters based on the type of forecasting.
5. Normalizing data and creating additional factors.
6. Removing outliers for expense forecasting by introducing errors through increasing model indicators. This is part of the normalization process, where records in the dataframe with abnormally high or low values are removed.
7. Splitting data into test and training sets.
8. Creating an ML model.
9. Generating predictions for test data and determining model metrics.
10. Creating a forecast based on the period parameter.
11. Preparing output data and saving them to results.json.

The text indicates that illustrates the diagram of the software user interface (Fig. 4).

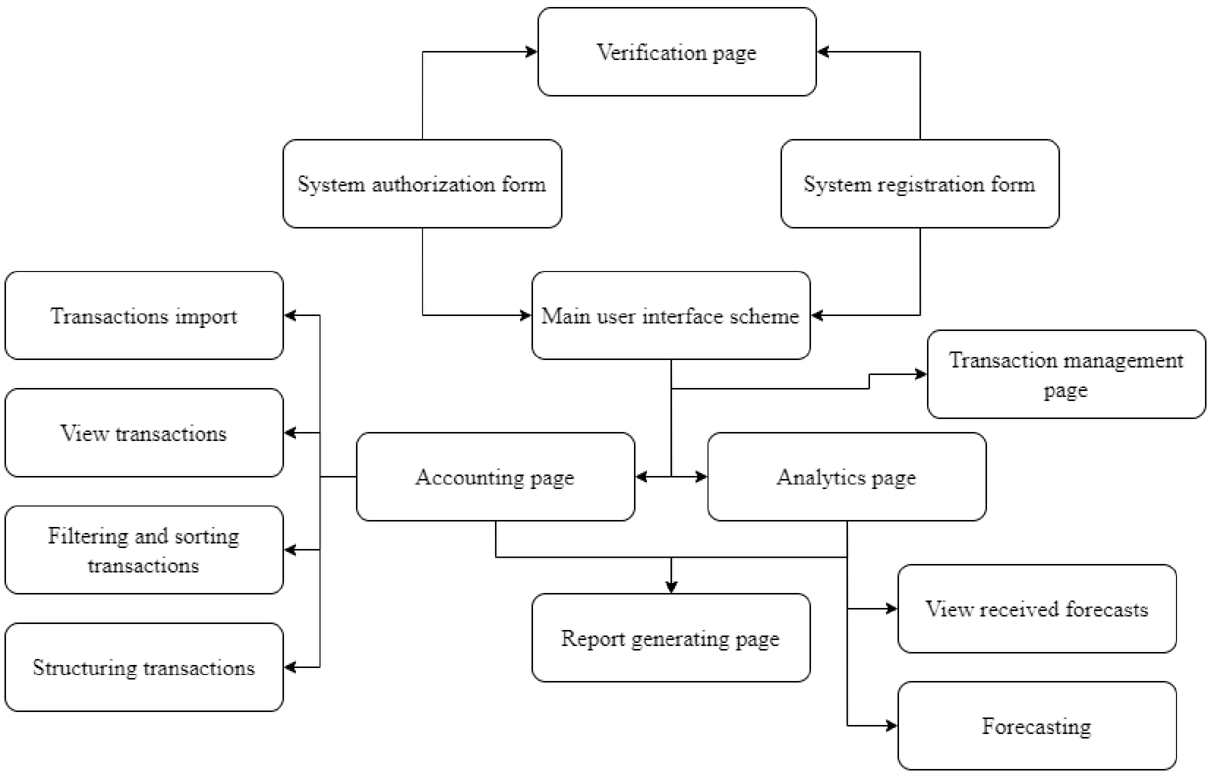


Figure 4: The pages of system’s interface

For interactions with the transaction import and forecasting modules, a custom interface has been developed. The web application interface consists of several endpoints:

- Authorization, a page that greets the user and has a form for creating an account or logging in with pre-existing data, namely email and password.
- Transactions, a page consisting of a table that displays transactions created by the user or imported through the import module for accounting. The table provides the ability to sort and filter data based on certain available parameters. The user menu includes a button that opens a form for creating a transaction. The Import button, located in the upper right corner of the page, opens a form where the user can select specific parameters for import and a CSV file with a bank statement.
- Analytics, the page contains a button in the upper right corner, Forecast, which opens a form with settings for generating a forecast. After generating the forecast, graphs appear on the page displaying the forecasted results in the form of a line chart, as well as transactions for this period for a more detailed analysis and correlation between real and forecasted data.

After considering the main aspects of the system implementation, it is necessary to explore the possibilities of its application for financial data forecasting tasks.

4. ML models implementation and research

To conduct research using the created ML models, it is necessary to form training and testing datasets, which are formed based on a sample of 50,000 client transactions of Monobank over a 3-year period. The scikit-learn library’s train_test_split method was used to create the datasets, allowing the dataset to be split using the test_size parameter.

The training dataset is provided to the hybrid model during its training along with corresponding evaluation results, enabling the model to establish a connection between input characteristics and accurate responses.

The testing dataset is used to assess the model’s effectiveness. In this case, the model is not given the target characteristic; instead, it must predict it based on other characteristics. According to the obtained results, predictive values are compared with actual results.

The ACF and PACF plots based on (1) and (2) are shown in Fig. 5 depict the three parameters of the created hybrid ARIMA model: $p = 1$, $d = 1$, $q = 0$.

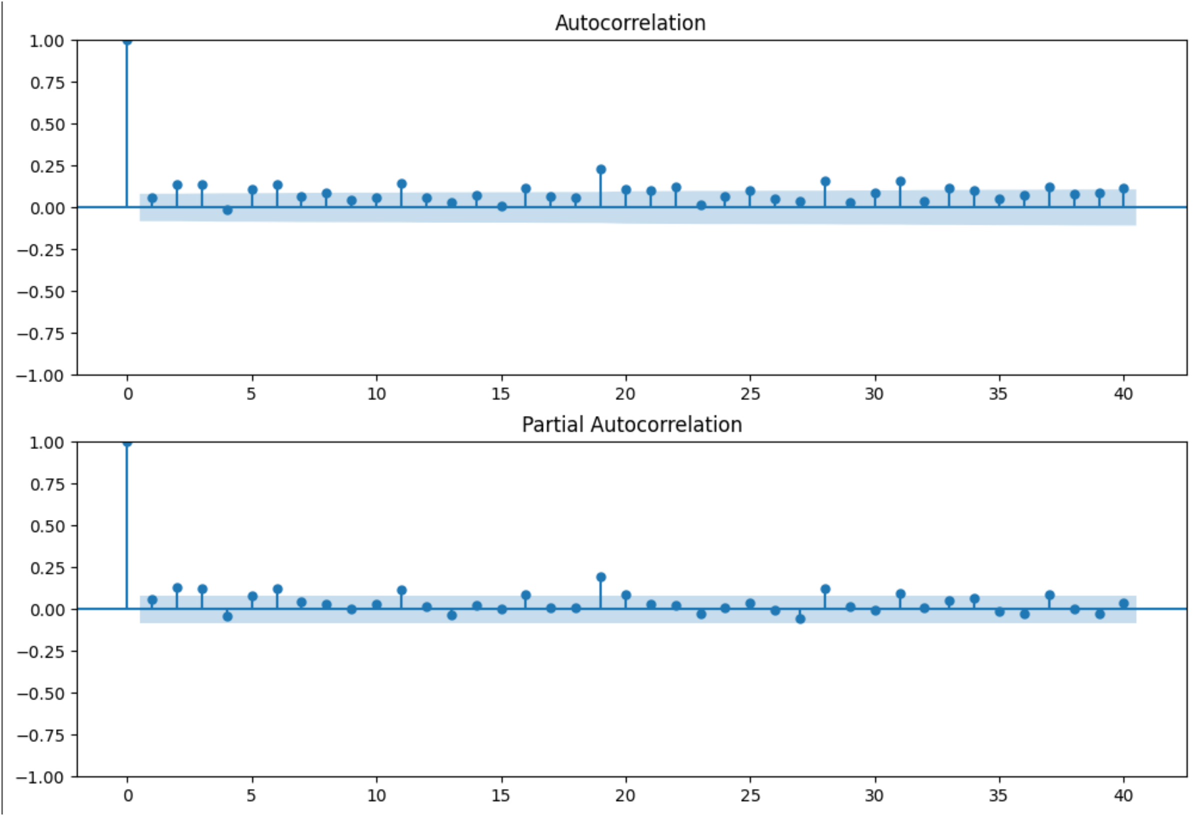


Figure 5: The ACF and PACF plots

Before starting the model training, it is important to establish criteria for evaluating its effectiveness to ensure the chosen algorithm's feasibility. One way to evaluate could be comparing the model's performance with results obtained through random generation of the target characteristic, i.e., a baseline model.

If the chosen algorithm shows worse results than simple random guessing, it may signal the need to reconsider the approach, possibly using alternative methods. In our case, for solving the financial risk regression task, it is appropriate to compare with the mean value of the training dataset. To compare the performance of the created hybrid model based on ARIMA, models such as Facebook Prophet and XGBoost were also chosen.

The results of measuring MAE for the constructed models are presented in Fig.6.

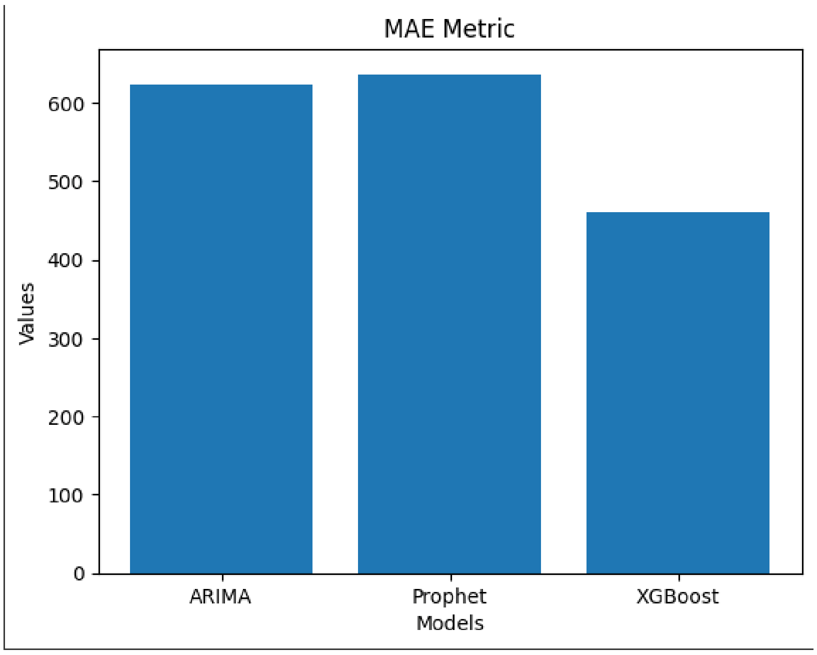


Figure 6: The results of measuring MAE for the constructed models

It should be noted that the overall nature of the distribution of the obtained evaluation results for the ACF and PACF models is quite balanced, with a spread not exceeding values of 0.25 on the positive side and 0.075 on the negative side.

The data sample is uniform, resulting in financial risk values ranging from 0.05 to 0.2. This indicates a fairly high accuracy in forecasting financial asset values and a moderate growth in calculation error.

As can be seen, the lowest MAE value, around 445, is achieved by XGBoost. However, the computational time of this model is nearly three times greater than the hybrid ARIMA-based model, which has an MAE of over 610. The Prophet model is the least accurate and fastest.

5. Conclusions

As a result of this study, software capable of automated processing, tracking, and forecasting financial assets and risks was developed, utilizing hybrid ML models with ARIMA.

The research involved exploring models for time series analysis, specifically the proposed hybrid ARIMA, Prophet, and XGBoost regression models within the context of predicting expenses and profits based on a prepared dataset of user transactions over a 3-year period. MAE metrics for each model are presented to aid in selecting the best-performing model for further performance enhancement.

Considering the chosen algorithm, it is advisable to further calculate the weight values of factors and dynamically adjust the hyperparameters of the created hybrid ARIMA model to increase its productivity and accuracy. An optimized model can be used for more precise forecasting of user spending and profit risks.

It’s worth noting that obtaining predictive values of financial risks, taking into account user behavioral factors within their investment strategy, enables a clearer and more predictable assessment of profits. Predicting expenses for stable profits is less relevant.

A subsequent step in algorithm research involves testing the model on different datasets, such as data from different users. Adding new features to the dataset, such as holidays or weather data, may improve the model’s effectiveness and prediction results, enhancing its generalization capability.

References

[1] C. Peng, Digital Inclusive Finance Data Mining and Model-Driven Analysis of the Impact of Urban-Rural Income Gap, Wireless Communications and Mobile Computing 7 (2022) 1-8. DOI:10.1155/2022/5820145.

[2] N. Rudnichenko, V. Vychuzhanin, I. Petrov, D. Shibaev, Decision Support System for the Machine Learning Methods Selection in Big Data Mining, in: Proceedings of The Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020), CEUR-WS, 2608, 2020, pp. 872-885.

[3] V. Vychuzhanin, N. Rudnichenko, Z. Sagova, M. Smieszek, V. Cherniavskyi, A. Golovan, Analysis and structuring diagnostic large volume data of technical condition of complex equipment in transport, in: 24th Slovak-Polish International Scientific Conference on Machine Modelling and Simulations - MMS 2019, Liptovský Ján, Slovakia, 2019. DOI:10.1088/1757-899X/776/1/012049

[4] J. Bertomeu, Machine learning improves accounting: discussion, implementation and research opportunities. Review of Accounting Studies 25 3 (2020) 1135-1155. DOI:10.1007/s11142-020-09554-9

[5] Y. Baştanlar, M. Ozuysal, Introduction to machine learning. Methods in Molecular Biology (Clifton, N.J.) 1107 (2014) 105-128. DOI: 10.1007/978-1-62703-748-8_7

[6] J. Bertomeu, E. Cheynel, E. Floyd, W. Pan, Using machine learning to detect misstatements. Review of Accounting Studies 26 2 (2019) 468-519. DOI: 10.1007/s11142-020-09563-8

[7] M. Alloghani, D. Al-Jumeily, J. Mustafina, A. Hussain, A.J. Aljaaf, A systematic review on supervised and unsupervised machine learning algorithms for data. Supervised and Unsupervised Learning for Data Science, Unsupervised and Semi-supervised Learning, Springer International Publishing, Cham (2014) 3-21. DOI: 0.1007/978-3-030-22475-2_1

[8] D. Alex, C. Timothy, A regression based approach to short term load forecasting. IEEE transaction on power systems 5 4 (1990) 1535-1550. DOI: 10.1109/59.99410

[9] I. Moghram, S. Rahman, Analysis evaluation of five short term load forecasting techniques. IEEE transaction on power systems 4 4 (1989) 1484-1491. DOI: 10.1109/59.41700

[10] X.L. Dong, T. Rekatsinas, Data integration and machine learning: a natural synergy, in ACM SIGMOD International Conference on Management of Data, Houston, 2018, pp. 1645-1650. DOI: 10.1145/3183713.3197387

[11] I. Ali, N. Bilal, F. Marwa, Forecasting Stock Prices with an Integrated Approach Combining ARIMA and Machine Learning Techniques ARIMAML. Journal of Computer and Communications. 11 (2023) 58-70. DOI:10.4236/jcc.2023.118005.

[12] M. Alhomsi, H. Ahmed, Forecasting of ExchangeRate: Autoregressive modelsvs. XGBoost : thesis, 2020.

[13] V. Arumugam, V. Natarajan, Time Series Modeling and Forecasting Using Autoregressive Integrated Moving Average and Seasonal Autoregressive Integrated Moving Average Models. Instrumentation Mesure Metrologie 4 (2023) 161-168.

[14] W. Charles, Jr. Chase, ARIMA Models. Demand-Driven Forecasting 13 (2013) 203-237.