УДК 004.[62+91]
DOI: 10.15827/0236-235X.030.3.439-446
Дата подачи статьи: 20.01.17
2017. Т. 30. № 3. С. 439–446

ФРЕЙМВОРК ДЛЯ АНАЛИЗА И ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ ПРИ РАЗРАБОТКЕ КОМПОНЕНТ ПРОАКТИВНЫХ СИСТЕМ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ

М.В. Щербаков, д.т.н., главный научный сотрудник, maxim.shcherbakov@vstu.ru;
К.С. Задиран, студент, konstantin.zadiran@gmail.com;
А.В. Голубев, аспирант, ax.golubev@gmail.com;
Аль-Гунаид Моххаммед Амин, преподаватель, mohammadalgunaid@gmail.com
(Волгоградский государственный технический университет,
просп. Ленина, 28, г. Волгоград, 400005, Россия)

В статье описан разработанный фреймворк для автоматизации исследований в области проектирования математического и ПО проактивных систем поддержки принятия решений. В частности, рассматривается проблема анализа и прогнозирования временных рядов для формирования компонент автоматизации прогнозирования различных процессов.

На практике для прототипирования компонент используются различные библиотеки анализа данных (на языках R, Python). Основная проблема – отсутствие или недостатки реализации методики анализа, то есть последовательности действий при анализе. Кроме этого, следует выделить ряд сдерживающих факторов, влияющих на эффективность проектирования компонент анализа и прогнозирования: рутинные операции при ручном анализе временных рядов, отнимающие много времени разработчика; недостаточная квалификация разработчика как сдерживающий фактор реализации компонент проактивных систем; зачастую необходимость анализа большого числа однотипных данных.

Предлагаемое программное решение, представляющее собой ПО на языке Python, позволяет автоматизировать процесс анализа временных рядов и формировать отчет в формате LaTeX. Формирование осуществляется в автоматическом режиме в соответствии с методологией CRISP-DM. Отчет содержит результат комплексного анализа временных рядов в соответствии с опубликованными и признанными методиками.

Все функции фреймворка можно отнести к одной из следующих групп: функции загрузки данных и формирование внутреннего фрейма данных; функции дескриптивного анализа временных рядов с визуализацией; функции прогнозирования и функции формирования отчета. Показан пример использования фреймворка для решения задачи поддержки принятия проактивных управленческих решений в системах энергетического менеджмента (прогнозирование потребления электроэнергии).

Ключевые слова: *проактивные системы, программный фреймворк, анализ временных рядов, прогнозирование временных рядов, визуализация данных.*

Низкая степень автоматизации бизнес-процессов все еще остается серьезной проблемой для производства и бизнеса, а следовательно, и для разработчиков автоматизированных систем и системных интеграторов [1]. Развитие современных подходов и методов анализа данных позволяет расширить область объектов автоматизации, автоматизировать операции, связанные с принятием решений (поддержка принятия решений), повышая уровень зрелости бизнес-процессов предприятий.

Одним из направлений автоматизации является совершенствование механизмов прогнозирования развития ситуации и оценка риска возникновения неблагоприятных исходов. В этом случае рассматривается новый класс систем поддержки принятия решений – проактивные системы принятия решений. Эти системы позволяют выявлять проактивные ситуации, требующие внимания или вмешательства заинтересованных лиц. В результате вырабатываются и принимаются превентивные меры для минимизации риска возникновения неблагоприятной ситуации [2].

В основе подобных систем лежат модели и методы прогнозирования временных рядов. Разработку ПО, выполняющего функции прогнозирования, целесообразно выносить в отдельные небольшие проекты, в которых осуществляется быстрое прототипирование решений с использованием современных средств и инструментов анализа данных. Несмотря на большое количество инструментов для проведения анализа временных рядов, например, язык программирования R или Python с его набором библиотек для работы с данными, остается проблема, связанная с реализацией методики построения модели прогнозирования. В зависимости от квалификации разработчика сроки реализации проекта могут существенно варьироваться. Выделим ряд факторов, имеющих большое влияние при решении задачи анализа и прогнозирования: ручной анализ временных рядов, отнимающий время разработчика; недостаточная квалификация разработчика как сдерживающий фактор реализации систем; зачастую необходимость анализа большого числа однотипных данных.

В качестве решения может быть предложен фреймворк (https://github.com/zadiran/Dataworks) для проведения исследований при анализе и прогнозировании временных рядов согласно методологии CRISP-DM [3]. В статье предлагается программное решение, позволяющее автоматизировать процессы анализа временных рядов и формирования отчетов в формате LaTeX.

Авторы исследовали работы, в которых описаны методики для анализа и прогнозирования временных рядов [4–7]. Это позволило выделить две основные проблемы: исходный код для представленных методов или отсутствует, или не всегда понятен разработчику.

При работе с временными рядами для анализа и прогнозирования можно использовать разные пакеты, на приобретение которых требуются средства. Для работы с ними необходима соответствующая квалификация [8–16]. Немаловажным фактором использования ПО является возможность реализации своих методов. Интеграция существующего ПО в собственные решения ограничена лицензией.

Многие разработчики и аналитики обычно реализуют свои собственные наработки. На это затрачивается время, которое можно было бы использовать более эффективно.

В последнее время все большее распространение получает язык программирования R, который был создан для статистической обработки данных и работы с графикой. Для него имеется большое количество пакетов [17], расширяющих функциональные возможности по обработке и прогнозированию временных рядов [18, 19]. Одним из них является forecast [20], в котором реализованы основные модели, используемые для прогнозирования.

Помимо R, большой популярностью пользуется Python. Это высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода, а также имеющий минималистичный синтаксис. Репозиторий насчитывает порядка 90 000 пакетов [21], среди которых можно выделить такие, как pandas [22, 23], matplotlib [24] и numpy [25], являющиеся базовыми для любого исследователя.

Описание фреймворка

Основная цель фреймворка – уменьшение времени на первичный анализ временных рядов для дальнейшего построения модели прогнозирования. От пользователя (аналитика данных) требуется задать минимум значений параметров. Структуру фреймворка определяют следующие сущности.

- ***data_provider*** – абстрактный класс, предоставляющий доступ к источнику данных:

get_data(*string[] = None)* : *DataFrame* – метод, используемый для получения всего набора данных; в качестве параметра принимает список столбцов, для которых требуется получить данные (если параметр не указан, метод возвращает данные для всех столбцов);

get_training_set*(string[] = None)* : *DataFrame* – метод, используемый для получения тренировочного набора данных; в качестве параметра принимает список столбцов, для которых требуется получить данные (если параметр не указан, возвращает данные для всех столбцов);

get_testing_set*(string[] = None)* : *DataFrame* – метод, используемый для получения тестового набора данных; в качестве параметра принимает список столбцов, для которых требуется получить данные (если параметр не указан, возвращает данные для всех столбцов);

load_data() – метод, используемый для загрузки данных из источника в оперативную память (данные во фреймворке загружаются только по требованию, а не при создании экземпляра *data_provider*);

divide_data() – метод, используемый для разделения набора данных на тренировочный и тестовый наборы;

try_separate_timestamp() : *dict* – метод для автоматического выделения колонок с датами и формирования статистики по выделенным столбцам.

- ***csv_data_provider*** – реализация класса *data_provider* для CSV-файлов.
- ***visualizer*** – класс, предоставляющий API для построения графиков.
- ***forecast_handler*** – класс, предоставляющий API для прогнозирования:

get_benchmark_result() : *dict* – метод получения результатов прогнозирования для бенчмарк-модели;

get_error_measurements*(dict)* : *dict* – метод расчета значений показателей ошибок для всех столбцов в словаре, переданном в качестве параметра;

get_error_measurements_for*(dict, str)* : *dict* – метод расчета значения;

grid_search() : *dict* – метод, используемый для осуществления *grid search.*

- ***report_generator*** – абстрактный класс, который предоставляет API для генерации отчетов:

data_loading*(bool = None)* – метод, генерирующий отчет для этапа загрузки данных; параметр отвечает за добавление результата в файл с отчетом или его перезапись;

charts*(bool = None)* – метод, генерирующий графики по данным: временной ряд, гистограмма, агрегация, графики по времени дня; параметр отвечает за добавление результата в файл с отчетом или его перезапись;

grid_search*(bool = None)* – метод, генерирующий отчет по результату прогнозирования; параметр отвечает за добавление результата в файл с отчетом или его перезапись.

- ***latex_report_generator*** – реализация класса *report_generator* для генерации отчетов в формате LaTeX.

Данная структура графически представлена на рисунке 1.

Все функции фреймворка можно отнести к одной из следующих групп:

– загрузка данных и формирование внутреннего фрейма данных;

– дескриптивный анализ временных рядов с визуализацией;
– прогнозирование;
– формирование отчета.

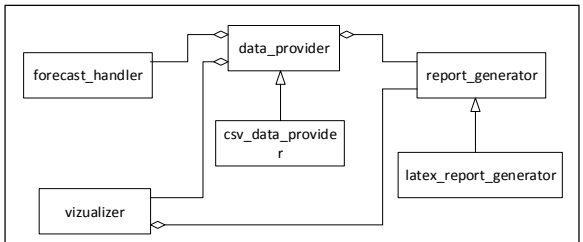


Рис. 1. Графическое представление структуры

Fig. 1. A graphical representation for the structure

Загрузка данных и формирование внутреннего фрейма данных. Для загрузки данных пользователь задает значения параметров *<data_provider_parameters, forecast_handler_params, vis_params>*.

Параметр *data_provider_parameters* включает:

'filepath' – имя файла с путем;

'testing_set_size': – размер выборки для тестирования.

Параметр *forecast_handler_params* включает:

forecasting_horizon – опциональный параметр, горизонт прогнозирования (задание параметров сезонности (частоты));

forecasting_columns – список столбцов, для которых выполняется прогнозирование.

Параметр *vis_params* включает:

'output_directory' – путь к директории с отчетом и картинками;

'timestamp_column' – имя столбца, содержащего отметки дата-время;

'timestamp_parameters' – параметры для обработки колонок, при автоматическом определении колонок с отметками даты-времени; включает логические параметры *'dayfirst'* и *'yearfirst'*, указывающие положение дня и года в строке с датой (в случае, если этот параметр не указан, значения параметров подбираются исходя из оптимального варианта – наименьшего количества различных интервалов между соседними отметками даты-времени функции прогнозирования).

Пример кода по заданию параметров прогнозирования:

```
from framework.data import csv_data_provider as dp
from framework.forecasting import forecast_handler

# create data provider
data_provider_parameters = {
    'filepath': 'data/demo1.csv',
    'testing_set_size': 96
}
data_provider = dp(data_provider_parameters)

# create forecast handler
    forecast_handler_params = {
    'forecasting_columns' :
['Open','High','Low','Close','Volume'],
    'forecast_horizon' : 96
}
fh = forecast_handler(data_provider, forecast_han-
dler_params, benchmark_model, models, error_measure-
ments)
```

Далее пользователь задает имя временной колонки, по которой фреймворк может произвести загрузку в *dataframe*, выделить столбцы для прогнозирования, выявить пропуски, произвести индексацию по временным меткам.

Дескриптивный анализ временных рядов с визуализацией. Фреймворк загружает данные и формирует отчет о загрузке, который включает следующую информацию:

– ошибки загрузки (если имеются);
– список колонок в файле с загрузкой;
– число записей;
– детальная информация по каждой колонке, имеющей значения, представленные в числовом виде: минимальное значение, максимальное значение, среднее, медиана;
– информация о столбце с временной меткой (если можно выделить автоматически): дата начала наблюдений, дата окончания наблюдений, интервалы наблюдений.

За визуализацию данных во фреймворке отвечает модуль *visualizer.py*.

Пример кода для визуализации данных:

```
from framework.data import csv_data_provider as dp
from framework.visualization import visualizer as vs

    # create data provider
data_provider_parameters = {
    'filepath': 'data/demo1.csv', # path to data
    'testing_set_size': 96,         # size of testing set
    'missing_values_processing_policy' : None # missing
values processing method
}
data_provider = dp(data_provider_parameters)

# plot charts
vis_params = {
    'output_directory' : 'report/images/', # directory, where
charts will be placed
    'timestamp_column' : 'DateTime', # timestamp column
    'timestamp_parameters' : {'dayfirst' : True} # parameters
for parsing timestamp
}
v = vs(vis_params, data_provider)
v.plot_everything()
```

Модуль поддерживает построение временных рядов, построение гистограмм, построение агрегированных данных по дням, неделям, месяцам (группировка данных по часам).

Прогнозирование. Задание списка моделей кандидатов с параметрами осуществляется в списке *models* словарями с параметрами. Представим задание областей изменения параметров моделей кандидатов:

```
    models = [
```

```
    { 'model' : naive_model() },
    { 'model' : random_model() },
    { 'model' : moving_average_model(),
        'parameters' : [
            model_param('lookback', 7, (4, 10), 1),
            model_param('lag', 30, None, None)
        ]
    },
    { 'model' : simple_average_model(),
        'parameters' : [
            model_param('frequency', 365, None, None),
            model_param('lag', 2, None, None)
        ]
    },
    { 'model' : arma_model(),
        'parameters' : [
            model_param('p', 2, None, None),
            model_param('q', 0, None, None)
        ]
    }
]
```

Меры точности задаются списком функций, который передается в *forecast_handler* вместе с моделями и данными. На данный момент реализованы следующие меры точности: MAE, RMSE, MSE, MdAE, MAPE, MdAPE, RMSPE, RMdSPE, MASE, MdASE, RMSSE, SMAPE, SMdAPE [26].

Использование grid search для выбранных моделей с мерами точности:

```
from framework.data import csv_data_provider as dp
from framework.visualization import visualizer as vs

from framework.forecasting import naive_model, random_model

from framework.measurement.absolute import mean_absolute_error as mae
from framework.measurement.absolute import root_mean_square_error as rmse
from framework.measurement.absolute import mean_square_error as mse

from framework.measurement.absolute import median_absolute_error as mdae

from framework.forecasting import forecast_handler

# create data provider
data_provider_parameters = { … }
data_provider = dp(data_provider_parameters)

# create charts builder
vis_params = { … }
v = vs(vis_params, data_provider)

# estimate models
naive = {'model': naive_model(), 'parameters': None}
forecast_handler_params = { … }

# error measurements
err_m = [mae(), rmse(), mse(), mdae()]

models = [
    {'model': naive_model()},
    {'model': random_model()}
]

# create forecast handler
fh = forecast_handler(data_provider, forecast_handler_params, naive, models, err_m)

# get a benchmark result
bench_result = fh.get_benchmark_result()

# get error measurements for benchmark model
bench_measurements = fh.get_error_measurements(bench_result)

# perform grid search
grid_search_result = fh.grid_search()
```

Далее для получения прогноза по выбранным моделям используется метод *grid search*, в котором реализован цикл по моделям и параметрам. *Grid search* реализован следующим образом: на верхнем уровне идет цикл по колонкам данных, уровнем ниже – цикл по используемым моделям для прогнозирования и на самом нижнем уровне – цикл по параметрам. Расчет оценки точности производится методом *get_error_measurements*.

Формирование отчета. На данный момент формирование результирующего отчета реализовано с использованием формата LaTeX. Представим код, отвечающий за данный процесс формирования отчета:

```
from framework.data import csv_data_provider as dp
from framework.reporting import latex_report_generator
from framework.visualization import visualizer as vs

# create data provider
data_provider_parameters = {
    'filepath': 'data/demo1.csv',
    'testing_set_size': 96
}
data_provider = dp(data_provider_parameters)

# create charts builder
vis_params = {
    'output_directory' : 'report/images/', # directory, where charts will be placed
    'timestamp_column' : 'DateTime', # timestamp column
    'timestamp_parameters' : {'dayfirst' : True} # parameters for parsing timestamp column
}
v = vs(vis_params, data_provider)

generator_parameters = {
    'path_to_report' : 'report/body.tex', # file, where to write report
    'include_timestamp_info' : True # if true, try to find timestamp columns automatically
}

# create latex report generator
rg = latex_report_generator(generator_parameters, data_provider, v)

# generate report
rg.data_loading()
rg.charts(True)
```

В корне репозитория с кодом доступны файлы с примерами использования данного фреймворка. Тестовые примеры описывают основной функционал фреймворка и разделены на несколько файлов:

– *example_load_data.py* представляет информацию по загрузке данных из демонстрационного примера;

– *example_grid_search.py* выполняет работу по формированию прогноза и оценке точности используемой модели;

– *example_create_charts.py* представляет функционал по визуализации данных;

– *example_generate_report.py* описывает работу по генерации отчета в формате LaTeX, включая все предыдущие пункты.

Пример использования

Настоящий фреймворк использовался для анализа временных рядов о потреблении электроэнергии в офисных зданиях [27]. Анализ проводился с целью повышения эффективности энергетического менеджмента за счет внедрения принципов проактивного управления [28]. В частности, требуется разработать модели прогнозирования для планирования бюджета на потребление энергии и выявления неэффективных режимов работы. Несмотря на существование алгоритмов автоматического прогнозирования временных рядов [19, 29], процедура совершенствования моделей прогнозирования осуществляется аналитиком данных.

Данные содержатся в CSV-файле, который представляет собой результат выгрузки из SCADA системы EcoSCADA [27]. По умолчанию в выгрузке содержатся данные по всем измерениям для здания.

На рисунке 2 представлены графики, полученные после визуализации данных.

Для генерации отчета пользователь задает параметры провайдера данных, параметры *forecast_handler*, параметры визуализации данных, используемые модели и меры точности, параметры генератора отчетов.

Выполнив скрипт *example_generate_report.py*, получим отчет в формате LaTeX, который необходимо собрать вручную, используя соответствующую среду, например TexLive. В результате получаем документ report.pdf.

Формируется отчет из 103 страниц, включающий секции Data Loading, Charts и Forecast result.

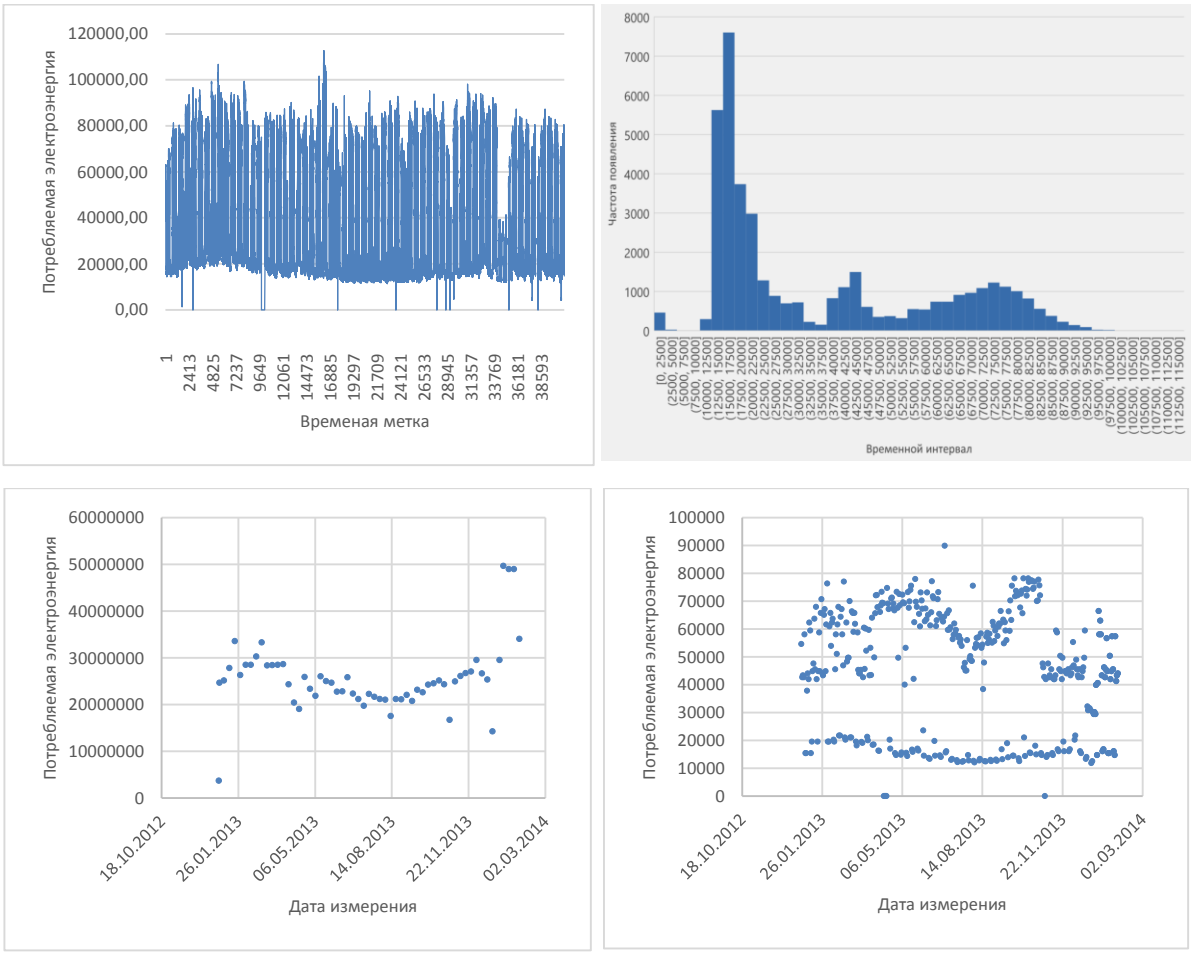


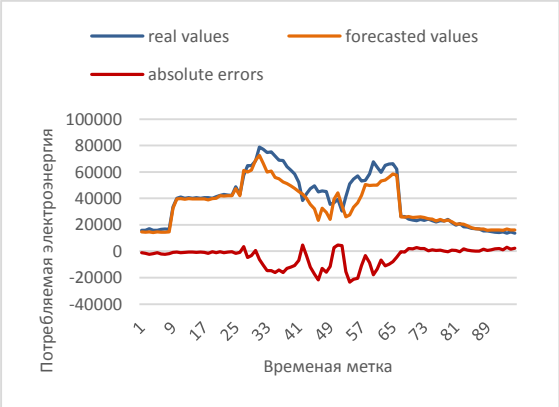
Рис. 2. Примеры графиков, полученных при визуализации

Fig. 2. Examples of charts obtained in visualization

Naive model

Forecast horizon: 96
Model doesn't has any parameters

Forecast



Testing set forecast

Testing set forecast absolute error distribution

	RMSE	MSE	MdAE	MAE
No parameters	8019.03984277	64305000.0	2100.0	5181.25

Рис. 3. Страница из отчета по результату прогнозирования с использованием наивной модели

Fig. 3. A page from the report on a naive model forecasting result

На рисунке 3 представлена страница из отчета прогнозирования временного ряда с использованием наивной модели.

Заключение

Для решения задачи автоматического анализа временных рядов, данные которых представлены в формате таблиц (в частности, в текстовом формате с разделителями запятых), предложено программное решение в виде фреймворка на языке Python с использованием модульной архитектуры. Основное назначение – формирование отчета о результатах анализа временных рядов в формате LaTeX.

Все функции фреймворка разделены на определенные группы. В результате исследователь получает возможность вести журнал исследований (экспериментов), получать отчет в автоматическом режиме, адаптировать отчет под его нужды, а также формировать пояснительную записку к техническому заданию в автоматическом режиме. Фреймворк также предусматривает возможность расширения функциональной составляющей, то есть исследователь может работать и с другими форматами, а также может дополнить его необходимыми методами для анализа и прогнозирования временных рядов.

Работа частично поддержана РФФИ (проект № 16-37-60066_мол_дк) и грантом Президента РФ (проект № МД-6964.2016.9).

Литература

1. Muller P. The CIO of 2020. The future of the chief information officer. URL: http://www.slideshare.net/HPESoftwareSolutions/the-cio-of-2020 (дата обращения: 18.01.2017).
2. Engel Y., Etzion O. and Feldman Z. 2012. A basic model for proactive event-driven computing. Proc. 6th ACM Intern. Conf. on Distributed Event-Based Systems (DEBS '12). ACM, NY, USA, 2012, pp. 107–118. DOI: 10.1145/2335484.2335496.
3. IBM SPSS Modeler CRISP-DM Guide. URL: ftp://public.dhe.ibm.com/software/analytics/spss/documentation/modeler/14.2/en/CRISP_DM.pdf (дата обращения: 18.01.2017).
4. Hyndman R.J., Athanasopoulos G. Forecasting: principles and practice. Paperback. October 17, 2013. URL: https://www.otexts.org/fpp (дата обращения: 18.01.2017).
5. Yarushev S.A., Averkin A.N. Review of studies on time se-

ries forecasting based on hybrid methods, neural networks and multiple regression // Программные продукты и системы. 2016. № 1. С. 75–82 (англ.).

6. Лесик И.А. Решение задачи прогнозирования с использованием нейронных сетей прямого распространения на примере построения прогноза роста курса акций // Программные продукты и системы. 2015. № 2. С. 70–74.

7. Kovalev S.M., Sukhanov A.V. Anomaly detection based on Markov chain model with production rules // Программные продукты и системы. 2014. №. 3. С. 40–43 (англ.).

8. STATISTICA. URL: http://statsoft.ru/ (дата обращения: 18.01.2017).

9. EViews. URL: http://www.eviews.com/home.html (дата обращения: 18.01.2017).

10. SPSS Statistics. URL: https://www-03.ibm.com/software/products/ru/spss-statistics (дата обращения: 18.01.2017).

11. SAS. URL: http://www.sas.com (дата обращения: 18.01.2017).

12. STATA. URL: https://www.stata.com/ (дата обращения: 18.01.2017).

13. MathWorks Matlab. URL: https://www.mathworks.com/ (дата обращения: 18.01.2017).

14. Wolfram Mathematica. URL: https://www.wolfram.com/mathematica/ (дата обращения: 18.01.2017).

15. Scilab. URL: http://www.scilab.org/ (дата обращения: 18.01.2017).

16. MAXIMA. URL: http://maxima.sourceforge.net/ru/ (дата обращения: 18.01.2017).

17. The Comprehensive R Archive Network. URL: https://cran.r-project.org/ (дата обращения: 18.01.2017).

18. Tavish Srivastava. A complete tutorial on time series modeling in R. URL: https://www.analyticsvidhya.com/blog/2015/12/complete-tutorial-time-series-modeling/ (дата обращения: 18.01.2017).

19. Rob J. Hyndman, Yeasmin Khandakar. Automatic time series forecasting: the forecast package for R, Jour. of Statistical Software, 2008, vol. 27, no. 1, pp. 1–22.

20. Rob J. Hyndman. R package 'forecast'. URL: https://cran.r-project.org/web/packages/forecast/forecast.pdf (дата обращения: 18.01.2017).

21. The Python Package Index. URL: https://pypi.python.org/pypi (дата обращения: 18.01.2017).

22. Aarshay Jain. A comprehensive beginner's guide to create a Time Series Forecast. URL: https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python (дата обращения: 18.01.2017).

23. Python Data Analysis Library. URL: http://pandas.pydata.org/ (дата обращения: 18.01.2017).

24. Matplotlib. URL: http://matplotlib.org/ (дата обращения: 18.01.2017).

25. NumPy. URL: http://www.numpy.org/ (дата обращения: 18.01.2017).

26. Shcherbakov M.V., Brebels A., Shcherbakova N.L., Tyukov A.P., Janovsky T.A., and Kamaev V.A. A survey of forecast error measures. World Applied Sciences Jour., 2013, vol. 24, iss. 24, pp. 171–176.

27. Kamaev V.A., Scherbarov M.V., Panchenko D.P., Scherbakova N.L., Brebels A. Using connectionist systems for electric energy consumption forecasting in shopping centers. Automation and Remote Control, 2012, vol. 73, no. 6, pp. 1075–1084.

28. Tennenhouse D.L. Proactive computing. Communications of the ACM, 2000, vol. 43, pp. 43–50.

29. Golubev A., Shcherbakov M., Shcherbakova N., Kamaev V. Automatic multi-steps forecasting method for multi seasonal time series based on symbolic aggregate approximation and grid search approaches. Jour. of Fundamental and Applied Sciences, 2016, vol. 8, no. 3S, pp. 2429–2441.

Software & Systems
DOI: 10.15827/0236-235X.030.3.439-446

Received 20.01.17
2017, vol. 30, no. 3, pp. 439–446

A FRAMEWORK FOR ANALYSIS AND FORECASTING OF TIME SERIES IN THE DEVELOPMENT OF PROACTIVE DECISION SUPPORT SYSTEM COMPONENTS

M.V. Shcherbakov [1], *Dr.Sc. (Engineering), Chief Researcher, maxim.shcherbakov@vstu.ru*
K.S. Zadiran [1], *Student, konstantin.zadiran@gmail.com*
A.V. Golubev [1], *Postgraduate Student, ax.golubev@gmail.com*
Al-Gunaid Mohammed Amin [1], *Lecturer, mohammadalgunaid@gmail.com*

[1] *Volgograd State Technical University, Lenin Ave. 28, Volgograd, 400005, Russian Federation*

Abstract. The article describes the developed framework for automation research on software design for decision support proactive systems. In particular, it considers the problem of analysis and forecasting time series to form automation components in predicting various processes.

In practice, there are various data analysis libraries (R, Python) used for prototyping the component uses. The main problem is a lack (or insufficiency) of implementation of analysis methods, i.e. the sequence of actions during analysis. In addition, there is a number of limiting factors that affect the efficiency of analysis and forecasting components design: (i) time-consuming routine operations during time series manual analysis; (ii) developer's insufficient qualifications as a deterrent in implementing proactive system components; (iii) a frequent need in the analysis of a large number of similar data.

The proposed software solution (a software in Python) automates the process of time series analysis and generate a LaTeX report. Formation is automatic in accordance with CRISP-DM methodology. The report provides a comprehensive analysis of time series in accordance with published and accepted methods.

All functions of the framework can be attributed to one of the following groups: data loading and the internal data frame; descriptive time series analysis with visualization; prediction and report making. The paper shows an example of using a

framework to solve the problem of proactive management decisions making support in energy management systems (forecasting of electricity consumption).

Keywords: proactive systems, software framework, time series analysis, time series forecasting, data visualization.

Acknowledgements. *The work has ben partially supported by RFBR (project no. 16-37-60066_мол_дк) and a grant of the President of the Russian Federation (project no. МД-6964.2016.9).*

References

1. Muller P. *The CIO of 2020: The future of the Chief Information Officer*. Available at: http://www.slideshare.net/HPE-SoftwareSolutions/the-cio-of-2020 (accessed January 18, 2017).

2. Engel Y., Etzion O., Feldman Z. A basic model for proactive event-driven computing. *Proc. 6th ACM Int. Conf. on Distributed Event-Based Systems (DEBS '12)*. ACM, NY, USA, 2012, pp. 107–118.

3. *IBM SPSS Modeler CRISP-DM Guide*. Available at: ftp://public.dhe.ibm.com/software/analytics/spss/documentation/modeler/14.2/en/CRISP_DM.pdf (accessed January 18, 2017).

4. Hyndman R.J., Athanasopoulos G. *Forecasting: Principles and Practice*. 2013. Available at: https://www.otexts.org/fpp (accessed January 18, 2017).

5. Yarushev S.A., Averkin A.N. Review of studies on time series forecasting based on hybrid methods, neural networks and multiple regression. *Programmnye produkty i sistemy* [Software & Systems]. 2016, no. 1 (113), pp. 75–82.

6. Lesik I. A. Meeting the challenge of forecasting using neural networks of direct distribution by the example of the construction of the forecast growth rate of shares. *Programmnye produkty i sistemy* [Software & Systems]. 2015, no. 2 (110), pp. 70–74 (in Russ.).

7. Kovalev S.M., Sukhanov A.V. Anomaly detection based on Markov chain model with production rules. *Programmnye produkty i sistemy* [Software & Systems]. 2014, no. 3 (107), pp. 40–43.

8. *STATISTICA*. Available at: http://statsoft.ru/ (accessed January 18, 2017).

9. *EViews*. Available at: http://www.eviews.com/home.html (accessed January 18, 2017).

10. *SPSS Statistics*. Available at: https://www-03.ibm.com/software/products/ru/spss-statistics (accessed January 18, 2017).

11. *SAS*. Available at: http://www.sas.com (accessed January 18, 2017).

12. *STATA*. Available at: https://www.stata.com/ (accessed January 18, 2017).

13. *MathWorks Matlab*. Available at: https://www.mathworks.com/ (accessed January 18, 2017).

14. *Wolfram Mathematica*. Available at: https://www.wolfram.com/mathematica/ (accessed January 18, 2017).

15. *Scilab*. Available at: http://www.scilab.org/ (accessed January 18, 2017).

16. *MAXIMA*. Available at: http://maxima.sourceforge.net/ru/ (accessed January 18, 2017).

17. *The Comprehensive R Archive Network*. Available at: https://cran.r-project.org/ (accessed January 18, 2017).

18. Tavish Srivastava. *A Complete Tutorial on Time Series Modeling in R*. Available at: https://www.analyticsvidhya.com/blog/2015/12/complete-tutorial-time-series-modeling/ (accessed January 18, 2017).

19. Hyndman R.J., Khandakar Y. Automatic Time Series Forecasting: The forecast Package for R. *Jour. of Statistical Software*. 2008, vol. 27, no. 1, pp. 1–22.

20. Hyndman R.J. *R Package 'forecast'*. Available at: https://cran.r-project.org/web/packages/forecast/forecast.pdf (accessed January 18, 2017).

21. *The Python Package Index*. Available at: https://pypi.python.org/pypi (accessed January 18, 2017).

22. Aarshay Jain. *A comprehensive beginner's guide to create a Time Series Forecast*. Available at: https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python (accessed January 18, 2017).

23. *Python Data Analysis Library*. Available at: http://pandas.pydata.org/ (accessed January 18, 2017).

24. *Matplotlib*. Available at: http://matplotlib.org/ (accessed January 18, 2017).

25. *NumPy*. Available at: http://www.numpy.org/ (accessed January 18, 2017).

26. Shcherbakov M.V., Brebels A., Shcherbakova N.L., Tyukov A.P., Yanovsky T.A., Kamaev V.A. A Survey of Forecast Error Measures. *World Applied Sciences Jour. (WASJ)*. 2013, vol. 24, spec. iss. 24: Information Technologies in Modern Industry, Education & Society, pp. 171–176.

27. Kamaev V.A., Shcherbakov M.V., Panchenko D.P., Shcherbakova N.L., Brebels A. Using connectionist systems for electric energy consumption forecasting in shopping centers. *Automation and Remote Control*. 2012, vol. 73, no. 6, pp. 1075–1084.

28. Tennenhouse D. Proactive computing. *Communications of the ACM*. 2000, vol. 43, pp. 43–50.

29. Golubev A.V., Shcherbakov M.V., Shcherbakova N.L., Kamaev V.A. Automatic multi-steps forecasting method for multi seasonal time series based on symbolic aggregate approximation and grid search approaches. *Jour. of Fundamental and Applied Sciences*. 2016, vol. 8, no. 3, pp. 2429–2441.