МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Системы автоматического управления»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой
д.т.н., профессор
_____/ В.И. Ширяев
«____» _____ 2019 г.

Разработка программного обеспечения для анализа и прогнозирования временных рядов

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ – 09.03.01.2019.136.00 ПЗ ВКР

Руководитель работы
доцент каф. САУ, к.т.н.
_____/ В.О. Чернецкий
«____» _____ 2019 г.

Автор работы
студент группы КЭ-451
_____/ Д.А. Расаев
«____» _____ 2019 г.

Нормоконтролер
доцент каф. САУ, к.т.н.
_____/ В.О. Чернецкий
«____» _____ 2019 г.

Челябинск 2019

АННОТАЦИЯ

Расаев Д.А. Разработка программного обеспечения для анализа и прогнозирования временных рядов. – Челябинск: ЮУрГУ, ВШ ЭКН; 2019, 51 с., 13 ил., библиогр. список – 15 наим., 18 листов слайдов презентации ф.А4, 1 прил.

Работа состоит из введения, трех глав, заключения, списка использованной литературы и приложения.

Во введении раскрывается актуальность выбранной темы, описываются цели и задачи аттестационной работы.

В первой главе изложены обзор и анализ известных методов и решений, включающие в себя обзор существующих программных продуктов, обзор известных методов прогнозирования сравнительный анализ методов прогнозирования.

Во второй главе производится выбор необходимых и наиболее подходящих для разработки программного продукта инструментов среди имеющихся.

Третья глава посвящена непосредственно разработке программного обеспечения. Глава разбита на 4 части: разработка внутренней логики, разработка пользовательского интерфейса, интеграция инструмента визуализации данных и тестирование программного продукта на примере прогнозирования и анализа реальных данных.

В заключении обобщается проделанная в ходе дипломного проекта работа.

					09.03.01.2019.136.00 ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.		Расаев Д.А.			Разработка программного обеспечения для анализа и прогнозирования временных рядов	Лит.	Лист 4 / Листов 51
Провер.		Чернецкий В.О.					
Н. Контр.		Чернецкий В.О.				ЮУрГУ Кафедра САУ	
Утверд.		Ширяев В.И.					

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

«Прогнозирование – это вид познавательной деятельности человека, направленной на формирование прогнозов развития объектов, на основе анализа тенденций и закономерностей его развития»[1, с. 34].

Временным рядом называют некоторую последовательность наблюдений, упорядоченных по времени или другому признаку. Часто, наблюдения статистически зависимы. Во многих областях имеются данные, которые могут быть выражены временными рядами. Природа и структура процесса, которым формируются эти данные, могут влиять на вид ряда. Зная структуру и природу этого процесса, можно описать его математически, что позволит не только произвести детальный анализ ряда, но и получить предположение о его поведении в будущем, то есть получить прогноз временного ряда.

Процесс прогнозирования различных систем достаточно актуален в настоящее время. Сфера его применения очень широка. Прогнозирование широко используется в экономике, промышленности, военной отрасли и многих других отраслях. В менеджменте понятие «планирование» и «прогнозирование» тесно переплетены. Они не идентичны и не подменяют друг друга. Планы и прогнозы различаются между собой временными границами, степенью детализации содержащихся в них показателей, степенью точности и вероятности их достижения, адресностью и правовой основой. Прогнозы, как правило, носят индикативный характер, а планы обладают силой директивного характера.

В промышленности методы прогнозирования играют первостепенную роль. Используя эти методы, можно делать предварительные выводы относительно разных процессов, явлений, реакций, операций. Это позволяет определять основные направления ее развития с выделением основных аспектов деятельности: развитие материально-технической базы производства, организационно-технический уровень производства, потребность в продукции и степень ее удовлетворения, потребность в ресурсах, изменение структуры, темпов и объемов производства. Определённую нишу прогнозирование занимает и в военных дисциплинах. Используя методы прогнозирования, можно определить или оценить радиоактивную обстановку местности и т.п.

В настоящее время имеет место большое количество программных продуктов для прогнозирования с различным функционалом. В пункте 1 главы 1 будут рассмотрены и проанализированы наиболее популярные и востребованные и имеющихся на рынке программных продуктов. Однако в виду перегруженности функционала и наличия несвязанного, либо слабо связанного, с решением задачи прогнозирования инструментария, построение прогноза усложняется. Более того, ры-

нок информационных технологий испытывает дефицит высококвалифицированных специалистов, задачей которых является анализ и прогнозирование данных. Это увеличивает вероятность построения некачественного прогноза.

Таким образом, цель данной работы – разработка альтернативного программного продукта, предназначенного для анализа и прогнозирования временных рядов с максимальной автоматизацией данного процесса. Под программным продуктом понимается программа для персонального компьютера с графическим интерфейсом.

В ходе выполнения работы необходимо решить следующие задачи:

– проведение сравнительного анализа известных методов прогнозирования;

– выбор метода прогнозирования;

– выбор необходимых инструментов для разработки;

– разработка внутренней логики;

– разработка пользовательского интерфейса;

– визуализация данных.

1 ОБЗОР И АНАЛИЗ ИЗВЕСТНЫХ МЕТОДОВ И РЕШЕНИЙ

1.1 Обзор существующих программных продуктов

Из имеющихся на рынке программных продуктов выделим наиболее популярные. Одним из них является Microsoft Excel [2]. Microsoft Excel — программа для работы с электронными таблицами, созданная корпорацией Microsoft для Microsoft Windows, Windows NT и Mac OS, а также Android, iOS и Windows Phone. Она предоставляет возможности экономико-статистических расчетов, графические инструменты и, за исключением Excel 2008 под Mac OS X, язык макропрограммирования VBA (Visual Basic for Application). Microsoft Excel входит в состав Microsoft Office и на сегодняшний день Excel является одним из наиболее популярных приложений в мире. Этот продукт имеет большой функционал, однако, прогнозирование не является основной задачей данного программного обеспечения. Поэтому при решении задач прогнозирования в этой программе возможно возникновение ряда сложностей. Еще одним существенным недостатком данного продукта является его цена. Это может стать одной из основных причин отказа от данного продукта.

Программное обеспечение Novo Forecast [3] представляет собой решение для планирования продаж. Novo Forecast предлагает удобный способ сделать обоснованный план в MS Excel. Novo Forecast легко интегрировать в имеющиеся бизнес-процессы, так как прогноз можно рассчитать на основании данных OLAP-кубов, аналитических BI-систем, сводных таблиц и отчетов в Excel из учетных программ 1С, SAP и т. д. Недостаток этого продукта состоит в том, что он не является самостоятельным. Данный продукт лишь предоставляет дополнительные возможности к Microsoft Excel. Это значит, что все проблемы Excel присутствуют и в Novo Forecast.

Прикладной пакет PSPP [4] является альтернативой другому продукту, а именно SPSS [5]. PSPP – программа для статистического анализа выбранных данных. Она интерпретирует команды на языке SPSS и выводит таблицы в ASCII, HTML, или PostScript формате. Программа может служить свободной заменой проприетарной программы SPSS, во многом ей подобна за небольшими исключениями. Имеется графический интерфейс и можно пользоваться консольными командами. Синтаксис и файлы данных совместимы с SPSS. Обеспечено взаимодействие с Gnumeric, OpenOffice.Org и другим свободным программным обеспечением, импорт данных из электронных таблиц, текстовых файлов и баз данных. Лицензия: GNU GPL. Поддерживают следующие операционные системы: Linux, Mac OS X, Unix, Windows. Эти два продукта, как и Microsoft Excel имеют хороший функционал, однако задача прогнозирования лишь одна из задач, решаемых

в этом прикладном пакете, из-за чего возникают определенные сложности в решении задач прогнозирования. К примеру, большое число функций, не связанных с прогнозированием может усложнить поиск инструментов для прогнозирования и работу в целом.

Из вышесказанного можно сделать вывод о том, что эти программные решения не решают проблемы автоматизации процессов анализа и прогнозирования временных рядов. Значит, необходимо разработать собственное программное обеспечение, в котором эта проблема была бы решена.

1.2 Обзор известных методов прогнозирования

Важным понятием в прогнозировании является метод прогнозирования. Метод прогнозирования – это способ исследования объекта прогнозирования, направленный на разработку прогнозов. Представляет собой некоторую последовательность действий, которые необходимо выполнить для получения модели прогнозирования. В свою очередь, модель прогнозирования есть функциональное представление, адекватно описывающее исследуемый процесс и являющееся основой для получения его будущих значений. Совокупность метода и модели есть некоторый прогноз.

Все методы прогнозирования могут быть разделены на формальные и интуитивные. Интуитивные методы основаны на суждения и оценках экспертов. В основном используются в политике, экономике и т.п. Эти системы либо достаточно сложны для математического описания, либо слишком просты и не нуждаются в математическом обосновании. Формальные же методы, наоборот, используют математические модели для получения прогнозов, что удобно для реализации системы прогнозирования при помощи ЭВМ. В свою очередь, каждая из этих групп может быть разбита на другие подгруппы в зависимости от способа получения прогнозной информации. К интуитивным методам относятся следующие методы:

– метод исторических аналогий;

– метод прогнозирования по образцу;

– методы экспертных оценок.

К формальным методам относят:

– методы экстраполяции;

– системно-структурные методы;

– ассоциативные методы;

– методы опережающей информации.

Системно-структурные методы основаны на выявлении структурных взаимосвязей и их анализе. После чего делается предположение о возможном состоянии объекта в будущем. К системно-структурным методам можно отнести:

– морфологического анализа;

– сетевое моделирования.

Морфологический анализ — метод прогнозирования, в основу которого положено построение матрицы характеристик и их возможных значений. Далее на основе перебора характеристик и их значений получают различные варианты прогноза. Метод включает в себя этапы:

– формулировка решаемой задачи;

– выделение морфологических признаков системы $P_i(i = 0, \cdots, n)$, наиболее существенных для данной задачи, при этом каждый признак системы обладает некоторым количеством k_i свойств $[p_i^1, p_i^2, \cdots, p_i^{k_i}]$;

- раскрытие возможных свойств каждого признака и запись их в виде матриц-столбцов;
- рассмотрение возможных сочетаний свойств признаков объекта прогнозирования;
- разработка системы критериев для оценки полученных вариантов объекта прогнозирования;
- оценка и выбор наилучших вариантов объекта прогнозирования при помощи выбранных критериев.

Основная трудность при работе с данным методом – выбор параметров. Выбираемые параметры должны быть независимыми и наиболее полно описывать систему. Также существует проблема с быстрым возрастанием количества записей при увеличении количества параметров и решений. Поэтому, необходимо следить за тем, чтобы выбор параметров не приводил к слишком большому числу возможных вариантов. Морфологический анализ используют для решения неформализованных, когда нет математической модели, нет целевой функции в явном виде, но существует логическое описание функциональных зависимостей.

Сетевое моделирование основано на представлении структуры управляемого процесса в виде специального графа – сети. Сетью является ориентированный граф без контуров и кратных дуг. Элементы этого графа сопоставляются с числами. В зависимости от того, с чем сопоставлены числа, граф может быть взвешенным или отмеченным. Основными элементами сетевой модели являются операции и события. Операция – любое действие, целью которого является достижение некоторого результата. Событие – достижение какого-либо результата. Существует несколько видов операций. Действительной называется операция, для осуществления которой необходимо затратить некоторое количество времени. Фиктивная операция не требует затрат времени. Событие происходит мгновенно и не имеет времени выполнения. От цели моделирования системы разделяют на сети:

1. ориентированная на события;
2. ориентированная на операции;
3. ориентированная на операции и события.

В первом случае, вершинам сопоставлены события, а дугам связи между ними. Во втором, вершинам сопоставлены события, дугам – связи. В третьем варианте, вершинам сопоставлены события, а дугам – вершины. Также сети классифицируют по другим параметрам:

1. количество сетей – односетевые и многосетевые;
2. количество конечных целей – одноцелевые и многоцелевые;
3. количество исходных событий – с одним исходным событием и несколькими исходными событиями;
4. степень неопределенности – детерминированные и стохастические сетевые модели;
5. количество операций – большие, средние и малые.

Наличие фиктивных операций позволяет преобразовывать любую многосетевую многоцелевую модель с несколькими исходными событиями в односетевую одноцелевую модель с одним исходным событием. Требования к построению сетевой модели:

1. номер предшествующей вершины должен быть меньше номера текущей вершины;
2. в сети не должно быть вершин, не соответствующих исходным событиям и операциям, а также вершин, которые не соответствуют завершающим событиям и операциям;
3. сеть не должна иметь петель, контуров и кратных дуг;
4. сеть должна иметь одну исходную и одну завершающую вершины.

Сеть называют упорядоченной, если для нее справедливо правило 1. Для выполнения третьего и четвертого правил в сеть вводят фиктивные дуги и вершины. Анализ сетевой модели, представленной в графической или матричной форме, позволяет выявить взаимосвязи этапов реализации проекта и определить наиболее оптимальный порядок выполнения этих этапов работ.

Методы опережающей информации основаны на использовании свойства научно-технической информации опережать реализацию научно-технических достижений в общественном производстве. Если в статистических методах используется информация о ретроспективном периоде, то в опережающих методах — информация о непосредственном периоде упреждения. Эти методы применяются для прогнозирования НТП и относятся к такому его направлению, как инженерное прогнозирование.

Ассоциативные методы основаны на определении взаимосвязи переменных при помощи математического аппарата и последующем предсказании поведения системы. На практике, наиболее часто применяют регрессионный анализ.

Регрессионный анализ позволяет исследовать зависимость между результатом системы и некоторым количеством наиболее важных ее факторов. Этот способ хорошо зарекомендовал себя там, где исследуются взаимосвязи явлений. Примером может послужить изучение зависимости качества или количества труда в различных отраслях производства от различных внешних и внутренних факторов. Часто, при помощи регрессионного анализа оценивают взаимосвязь результата хозяйственной деятельности от различных факторов.

Данный метод предполагает решение двух задач:

1. выбор независимых переменных, которые наиболее существенно влияют на систему, и выбор функции регрессии;
2. расчет коэффициентов функции регрессии так, чтобы она была максимально приближена к исходным данным.

Прогнозные значения получаются путем подстановки новых значений факторов в уравнение регрессии. По количеству параметров, различают парную и множественную регрессию, а по виду функции регрессии на линейную и нелинейную. Простейшей регрессией является линейная однофакторная регрессия, которая имеет вид [6, с. 68]

$$Y^M(X) = b_0 + b_1 X, \tag{1.1}$$

где Y^M – прогнозные значения;

b_0 и b_1 – коэффициенты модели;

X – параметр модели.

В случае с линейной функцией регрессии от n переменных, модель описывается уравнением вида [6, с. 74]

$$Y^M(\vec{X}) = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n, \tag{1.2}$$

где Y^M – прогнозные значения;

$\vec{X}$ – вектор из n параметров модели;

X_i – i-ый параметр модели;

b_i – i-ый коэффициент модели.

Различие в этих функциях регрессии состоит лишь в количестве параметров. В нелинейных функциях регрессии для описания зависимости используются более сложные виды функций. Некоторые типы нелинейных функций регрессии:

1. $Y^M(X) = b_0 + b_1 X + b_2 X^2 + b_3 X^3 + \cdots + b_k X^k$ – полином степени k;

2. $Y^M(X) = b_0 + \frac{b_1}{X}$ – равносторонняя гипербола;
3. $Y^M(X) = b_0 b_1^X$ – показательная;
4. $Y^M(X) = b_0 e^{b_1 + b_2 X}$ – экспоненциальная.

Также в регрессионном анализе имеет место понятие авторегрессии. Суть ее заключается в линейной зависимости значений временного ряда от предыдущих значений этого же ряда. Или по-другому, параметрами модели авторегрессии являются значения ряда в прошлом. В общем виде математически описывается следующим образом [7, с. 192; 8, с.69]:

$$Y_t^M = \alpha_0 + \sum_{i=1}^{n} \alpha_i Y_{t-i} + \varepsilon_i, \quad (1.3)$$

где Y_t^M – прогнозные значения;

α_i – коэффициенты модели;

Y_i – фактические значения системы;

ε_i – случайная составляющая.

Простейшей моделью авторегрессии является уравнение первого порядка [7, с.192; 8, с.69]:

$$Y_t^M = \alpha_0 + \alpha_1 Y_{t-1} + \varepsilon_t, \quad (1.4)$$

где Y_t^M – прогнозные значения;

α_0 и α_1 – коэффициенты модели;

Y_{t-1} – фактические значения системы в момент времени t – 1;

ε_t – случайная составляющая.

Выше были приведены методы параметрического регрессионного анализа. Данный способ предполагает наличие некоторой кривой, имеющей функциональную форму. В этом случае предполагают, что функция описывается некоторым конечным количеством параметров.

В противовес этим методам имеется непараметрический регрессионный анализ, при котором функция регрессии, изначально, не имеет определенного вида. Данный метод еще называют сглаживанием. Основная идея сглаживания – процедура локального усреднения, которое заключается в получении среднего значения функции отклика в окрестностях некоторой точки. В данном методе, зависимость моделируется уравнением вида [9, с. 25]:

$$Y_t = m(X_t) + \varepsilon_t, \quad (1.5)$$

где Y_t – эмпирическое значение Y;

m – кривая среднего значения отклика;

X_t – эмпирическое значение X;

ε_t – случайная компонента.

В зависимости от используемого метода вводятся специфические весовые последовательности значений $W_{ni}(X)$. Наиболее распространенным методом непараметрической регрессии является метод ядерного сглаживания. Суть его заключается в представлении весовой функции $W_{ni}(X)$ при помощи функции ядра. Это вещественная симметричная функция, параметром которой является некоторое скалярное значение. Также эта функция должна обладать следующим свойством [9, с. 35]:

$$\int K(u)du = 1, \tag{1.6}$$

где K – ядро;

u – скалярный параметр ядра.

Ядерные веса рассчитываются из следующего выражения [9, с. 35]:

$$W_{n_i}(X) = \frac{K_{h_n}(X-X_i)}{\hat{f}_{h_n}(X)}, \tag{1.7}$$

где W_{n_i} – весовая функция;

n – размер выборки;

K_{h_n} – ядерная функция с шириной окна h_n, представленная в уравнении 1.8 [9, с. 35];

X_i – эмпирические значения X;

$\hat{f}_{h_n}$ – ядерная оценка плотности Розенблата – Парзена для ширины окна h_n, представленная в уравнении 1.9 [9, с. 35].

$$K_{h_n}(u) = \frac{1}{h_n} K\left(\frac{u}{h_n}\right), \tag{1.8}$$

$$\hat{f}_{h_n}(X) = \frac{1}{n}\sum_{i=1}^{n} K_{h_n}(X - X_i), \tag{1.9}$$

Функция m рассчитывается как [9, с. 35]:

$$\hat{m}_n(X) = \frac{\frac{1}{n}\sum_{i=1}^{n} K_h(X-X_i)Y_i}{\frac{1}{n}\sum_{i=1}^{n} K_h(X-X_i)}, \tag{1.10}$$

где $\hat{m}_n$ – оценка Надарая – Ватсона;

n – размер выборки;

X – параметр функции;

K_h – ядерная функция с шириной окна h;

X_i – эмпирические значения X;

Y_i – эмпирические значения Y.

Экстраполяционные методы представляют собой методы прогноза, основанные на математической экстраполяции. Эти методы предполагает наличие большого количества факторов, влияющих на систему, из которых не представляется

возможным выделить наиболее существенные. В них выбор аппроксимирующей функции подбирается на основе статистического наблюдения динамики развития системы или объекта в прошлом. Прогнозирование методами экстраполяции является одним из наиболее распространенных методов прогнозирования в силу простоты этих методов. К методам прогнозирования экстраполяцией можно отнести следующие методы:

- экстраполяцию тренда;
- прогнозирование на основе кривых роста.

Наиболее распространенным методом экстраполяции выступает экстраполяция тренда. Этот метод предполагает следующие допущения:

1. развитие системы может быть описано некоторой гладкой прямой, называемой трендом;
2. факторы, влиявшие на поведение системы в будущем, не претерпят изменений в будущем.

Основной задачей в экстраполяции является выбор вида математической модели. Однако, исходный ряд, часто, имеет выбросы различного характера. Поэтому, перед выбором математической модели для прогнозирования методами экстраполяции необходимо произвести предварительную обработку исходных данных. Она направлена на снижение влияния случайной составляющей в исходном числовом ряду, т. е. приблизить его к некоторой известной функции – тренду. Для этого используются различные методы сглаживания. Сглаживание используется для минимизации случайных отклонений значений ряда от предполагаемого тренда. Для этого могут быть использованы методы:

- скользящее среднее;
- простое экспоненциально взвешенное среднее;

Метод скользящего среднего основан на вычислении средних значений из n его последних значений для момента времени t. Математически он может быть представлен следующим образом [10, с. 19]:

$$m_t = \frac{1}{n}\sum_{i=t}^{t-n+1} d_i, \qquad (1.11)$$

где m_t – среднее значение в момент времени t;

d_i – фактическое значение;

n – число точек, входящих в среднее.

В данной формуле более свежим данным присваивается тот же вес, что и более старым. Т.к. в реальных объектах новые данные сильнее влияют на прогноз, то имеет смысл использовать разные веса для различных d_i. Но их сумма всегда

должна стремиться к единице. Данный метод очень прост в реализации, однако, имеет следующие недостатки:

- в расчете средних значений участвуют не все экспериментальные значения (это снижает качество прогноза);
- изменение чувствительности вызывает определенные трудности т.к. требуется изменить значение n, а значит, и снова произвести все расчеты (в случае, когда веса для разных d_i различны, требуется пересчет и самих весов).

Метод простого экспоненциально взвешенного среднего является частным случаем скользящего среднего. Математически он может быть представлен следующим образом [10, с. 21]:

$$u_t = \alpha d_t + (1 - \alpha)u_{t-1}, \quad (1.12)$$

где u_t – экспоненциально взвешенное среднее в момент времени t;

α – показатель экспоненциального сглаживания $\alpha \in (0;1)$.

Значение α представляет из себя вес t-го значения уровня временного ряда. Чем больше это значение, тем выше чувствительность среднего. Данный метод не имеет недостатков метода скользящего среднего. Коэффициент α будет подбираться путем перебора возможных значений до тех пор, пока среднеквадратичная ошибка не примет наименьшего значения. Принцип, при котором более новую информацию ассоциируют с более высоким значением степени информативности, называют принципом дисконтирования. В качестве начального значения u_0, обычно, принимают среднее значение среди всех экспериментальных.

Методы сглаживания также могут рассматриваться как методы прогнозирования стационарных рядов, т.е. рядов среднее которых, практически, неизменно.

Далее, необходимо определить тип тренда и подобрать его коэффициенты для более точного описания системы. Известные типы трендов [1, с. 46]:

1. линейный: $\hat{y}(t) = \alpha_0 + \alpha_1 t$;
2. параболический (разных порядков): $\hat{y}(t) = \alpha_0 + \alpha_1 t^2 + \alpha_2 t$, $\hat{y}(\tau) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3$ и т.д.;
3. показательный: $\hat{y}(t) = \alpha_0 \alpha_1^t$, $\hat{y} = \alpha_0 e^{\alpha_1 t}$;
4. логарифмический: $\hat{y}(t) = \alpha_0 + \alpha_1 lg(t)$;
5. гиперболический: $\hat{y}(t) = \alpha_0 + \frac{\alpha_1}{t}$;
6. Сезонный: $\hat{y}(t) = \alpha_0 + \sum_{k=0}^{m}\left(\alpha_k cos(kt) + b_k sin(kt)\right)$, $\alpha_0 = \frac{1}{n}\sum y$, $\alpha_k = \frac{1}{n}\sum y cos(kt)$, $b_k = \frac{1}{n}\sum y sin(kt)$.

Метод прогнозирования кривыми роста схож с экстраполяций тренда. В нем предполагается выбор математической функции и расчет ее параметров. Однако, кривые роста отражают кумулятивные возрастания до локального максимума.

Также в данном методе расчет коэффициентов модели производится другим способом, так как метод наименьших квадратов дает лишь приближенные значения. Еще одной особенностью является то, что по мере приближения к максимальному значению уменьшается значение приращения. Методы прогнозирования на основе кривых роста требуют выполнения условий [1, с. 55]:

- очень длинный временной ряд;
- отсутствие скачков в исходном временном ряду;
- точное определение предела насыщения.

Наиболее известные и распространенные кривые:

- Гомперца;
- Перля-Рида.

Обе кривые являются S-образными. Кривая Гомперца в математическом представлении имеет вид [1, с. 56]:

$$y = \alpha b^{cx}, \quad (1.13)$$

где α – максимальное значение;

b – расстояние, разделяющее значение уровня от максимального значения в момент времени t;

x – время первого значения уровня;

c – показатель темпа роста.

Алгоритм прогнозирования:

1. логарифмируют исходное уравнение: $\lg y = \lg\alpha + \lg b \cdot c^x$;
2. весь ряд разбивают на 3 части;
3. для каждой группы считают суммы: S_1, S_2, S_3;
4. рассчитывают первым разности по полученным суммам: $d_1 = S_2 - S_1$, $d_2 = S_3 - S_2$;
5. вычисляют коэффициенты: $c = \sqrt[n]{\frac{d_2}{d_1}}$, $\lg\alpha = \frac{1}{n}\left(S_1 - \frac{d_1}{c^n - 1}\right)$, $\lg b = \frac{d_1(c-1)}{(c^2-1)^1}$, где n – число уровней.

Кривая Перля-Рида имеет вид [1, с. 57]:

$$\frac{1}{y} = \alpha + bc^x, \quad (1.14)$$

где α – максимальное значение;

b – расстояние, разделяющее значение уровня от максимального значения в момент времени t;

x – время первого значения уровня;

c – показатель темпа роста.

Далее, производят те же действия, что и при расчетах параметров кривой Гомперца, но без логарифмирования. То есть, формулы для расчет коэффициентов по формулам выглядят следующим образом:

1. $c = \sqrt[n]{\frac{d_2}{d_1}}$;
2. $\alpha = \frac{1}{n}\left(S_1 - \frac{d_1}{c^n - 1}\right)$;
3. $b = \frac{d_1(c-1)}{(c^2-1)^1}$.

Экстраполяцию также рассматривают не как конечный способ прогнозирования, но и как некоторый промежуточный этап, на основе результатов которого разрабатываю прогноз.

Одним из основных моментов в прогнозировании методами экстраполяции и параметрического регрессионного анализа является подбор коэффициентов при параметрах модели α_i, i = 0..n, при которых отклонение значений функции от экспериментальных значений было бы минимальным. Наиболее известным методом расчета коэффициентов модели является метод наименьших квадратов. Математически метод наименьших квадратов может быть представлен выражением [6, с. 68]

$$S = \sum_{i=0}^{n} \beta_i (\widehat{y_i} - y_i)^2 \to \min, \tag{1.15}$$

где β_i – коэффициент, принимающий значения от 0 до 1;

$\widehat{y_i}$ – расчетные значения ряда;

y_i – фактические значения ряда.

Из необходимого условия минимума функции, в общем случае, выражение может быть преобразовано к системе вида [6, с. 68]

$$\begin{cases} \frac{ds}{d\alpha_0} = 0 \\ \frac{ds}{d\alpha_1} = 0 \\ \cdots \\ \frac{ds}{d\alpha_0} = 0 \end{cases}, \tag{1.16}$$

где α_i – коэффициент x_i-го параметра полученной модели.

1.3 Сравнительный анализ методов прогнозирования

Среди описанных выше методов прогнозирования для данной задачи наиболее подходящими являются ассоциативные методы и методы экстраполяции. Интуитивные методы не подходят для решения данной задачи, так как целью работы является разработка программного обеспечения, которое должно составлять прогноз с минимальным участием человека. Системно–структурные методы не под-

ходят по другой причине. В этих методах исследуются структурные зависимости. В данной же задаче нет известных структурных зависимостей, а лишь количественные данные. Это делает неподходящим системно–структурные методы для решения поставленной задачи.

Среди методов экстраполяции можно сразу исключить методы прогнозирования стационарных рядов, так как результаты такого прогнозирования будут неудовлетворительными в случае с нестационарными временными рядами, а информации о стационарности временного ряда на начальном этапе нет. Прогнозирование экстраполяцией тренда имеет один существенный недостаток, который может сильно сказаться на качестве прогноза в будущем. Не во всех случаях получается так, что факторы, влияющие на поведение системы, остаются неизменными с течением времени. При возникновении такой проблемы необходимо производить корректировку исходного ряда. Эта задача является нетривиальной и требует вмешательства прогнозиста. То же относится и к прогнозированию кривыми роста. Более того, наличие заранее выбранной математической модели ограничивает прогнозиста. В случае с кривыми роста, временной ряд должен быть очень длинным.

В случае с методами регрессионного анализа наиболее подходящим в рамках данной работы является непараметрическая регрессия. Этот подход является более гибким в сравнении с параметрической регрессией. Основной проблемой параметрического способа является то, что используемая математическая модель может иметь недостаточную либо слишком большую размерность, в то время как непараметрический метод предоставляет более гибкие способы получения зависимостей. Этот же недостаток имеют и методы экстраполяционного прогнозирования. Так же, непараметрические методы являются более чувствительными к малым скачкам в исследуемых данных. В качестве примера можно привести исследования кривых роста человека, представленные на рисунке 2.1 [9, с. 15]. На нем исходные данные представлены маленьким графиком. В верхней части, представлены графики, которые отображают зависимость скорости изменения роста девушек от их возраста, а в нижней части – ускорение. Штриховая линия отображает результат работы параметрического регрессионного анализа, а сплошная – непараметрического. На рисунке 1.1 видно, что непараметрические модели выявили скачки в данных, которые были сглажены параметрическими моделями. Выявление этих скачков параметрическими методами приводит к значительным трудностям.

Из описанного выше, можно сделать вывод о том, что для решения поставленной задачи наиболее подходящим методом прогнозирования являются методы непараметрической регрессии.

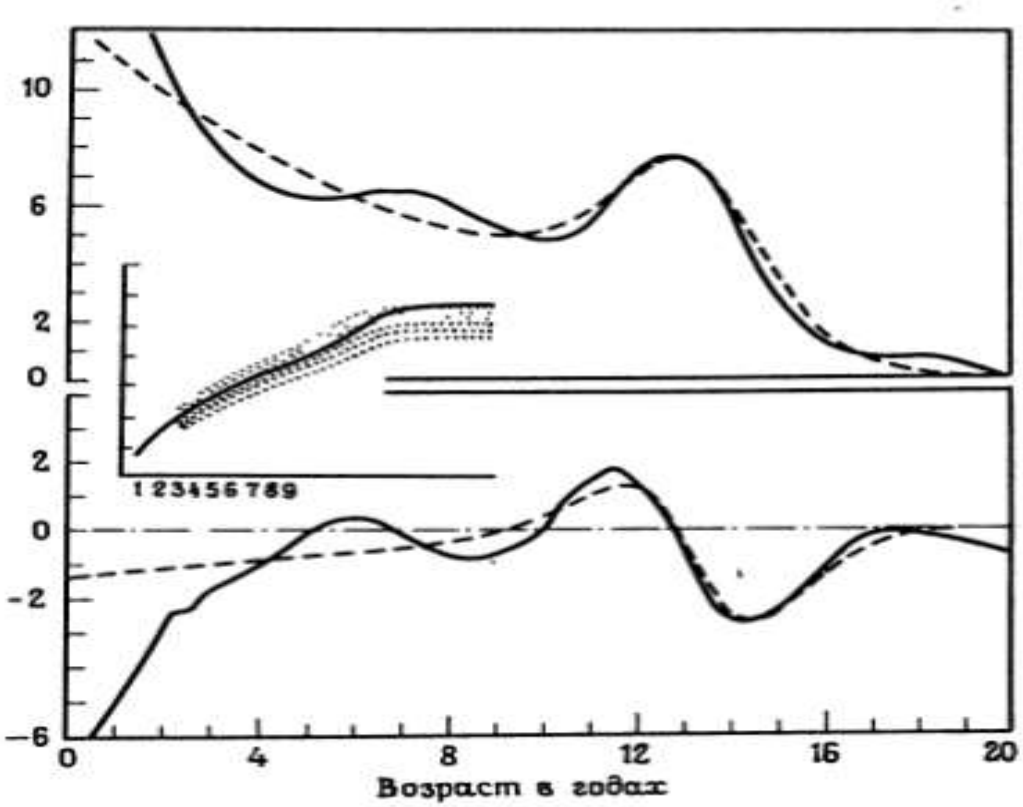

Рисунок 1.1 – Исследование кривых роста человека

Выводы по главе один

В данной главе рассмотрены основные методы прогнозирования и известные программные продукты. В виду своей сложности и неудобства в использовании, рассмотренные продукты требуют альтернативного решения. Из рассмотренных методов прогнозирования, как наиболее подходящие, выбраны методы непараметрической регрессии. Они являются более гибкими и чувствительными к изменениям в исходных данных. Далее, в работе будут использованы именно эти методы.

2 ВЫБОР ИНСТРУМЕНТОВ ДЛЯ РАЗРАБОТКИ

2.1 Задачи главы два

Перед началом разработки, необходимо определиться с наиболее подходящими инструментами для разработки программного обеспечения. Необходимо произвести выбор следующих инструментов:

- языка программирования;
- инструментов для анализа данных;
- инструментов для прогнозирования;
- инструментов визуализации данных;
- инструментов для реализации пользовательского интерфейса.

2.2 Выбор языка программирования

Перед началом разработки программного продукта необходимо определиться с используемыми инструментами. Выбор этих инструментов зависит от того, каким языком программирования будет пользоваться разработчик. В области науке о данных наиболее известными языками являются:

- MATLAB;
- R;
- Julia;
- Python.

MATLAB – это высокоуровневый интерпретируемый язык и интерактивная среда для программирования, численных расчетов и визуализации результатов. С помощью MATLAB можно анализировать данные, разрабатывать алгоритмы, создавать модели и приложения. Он является признанным языком для вычислений, используется в академических кругах и промышленности. Этот язык программирования обладает свойствами языков программирования общего назначения. Ярким примером может послужить объектно-ориентированное программирование(ООП), которое позволяет облегчить ряд задач в разработке программного обеспечения. Язык по–умолчанию, обладает большим функционалом для решения математических задач. Это дает ему преимущество над языками программирования общего назначения. Область его применения достаточно широка: IoT, финансы, медицина, космос, автоматика, робототехника, беспроводные системы и т.д. Разработан и лицензирован MathWorks, компанией, созданной в 1984 году, которая занимается разработкой программного обеспечения. К ключевым особенностям языка можно отнести:

- платформонезависимость;
- наличие средств интеграции с языками: C и C++;

– наличие встроенных средств разработки пользовательского интерфейса;
– большое количество средств визуализации данных;
– обширный функционал для анализа данных;
– наличие встроенной интерактивной среды для программирования и управления данными.

К недостаткам данного программного продукта можно отнести :

– высокая цена продукта;
– узконаправленнность;
– низкая скорость работы в сравнении с компилируемыми языками программирования.

R — язык программирования для статистической обработки данных и работы с графикой, а также свободная программная среда вычислений с открытым исходным кодом в рамках проекта GNU. Этот продукт позволяет производить обработку данных, математическое моделирование и работу с графикой. Язык создавался как аналогичный языку S, разработанному в Bell labs, и является его альтернативной реализацией. R был разработан сотрудниками статистического факультета Оклендского университета Россом Айхэкой и Робертом Джентлменом. Среда разработки включает в себя большое количество пакетов для математических расчетов и может быть использована для решения задач различной сложности. На дынный момент R является одним из наиболее популярных статистических инструментов в мире. К достоинствам данного продукта можно отнести:

– бесплатность;
– кроссплатформенность;
– качественные средства визуализации;
– большое количество инструментов для работы с математическим аппаратом;
– наличие средств интеграции с языками: C, C++, java;
– возможность работы с форматами данных прикладных пакетов: SAS, SPSS, STATA.

К недостаткам можно отнести:

– относительно небольшой объем литературы;
– низкая производительность;
– узконаправленность;
– специфичность синтаксиса.

Julia — высокоуровневый высокопроизводительный свободный язык программирования с динамической типизацией, созданный для математических вычислений. Эффективен также и для написания программ общего назначения. В

стандартный комплект входит JIT-компилятор, благодаря чему, приложения, полностью написанные на этом языке, лишь немного уступают в производительности приложениям, написанным на статически компилируемых языках вроде Си или C++. Большая часть стандартной библиотеки языка написана на нём же. Этот язык программирования поддерживает следующие парадигмы программирования:

- объектно–ориентированное программирование;
- функциональное программирование.

Также язык имеет встроенную поддержку большого числа команд для распределенных вычислений. Достоинства языка :

- производительность;
- имеет большое количество библиотек для анализа данных;
- бесплатность;
- мультипарадигменность;
- возможность прямого вызова функций из библиотек языка программирования Си;
- масштабируемость.

Недостатки:

- малое количество собственных библиотек, что требует использование сторонних. Что сильно снижает производительность.

Python — «легкочитаемый» высокоуровневый интерпретируемый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Данный язык, ввиду его популярности, используется в самых разных задачах, в том числе в математических и статистических задачах. Для него имеется большое количество библиотек, что сильно расширяет его функционал. Он является мультипарадигменным. Поддерживаемые парадигмы программирования:

- объектно-ориентированное;
- функциональное;
- процедурное.

Также, python позволяет писать программный код в интерактивном режиме. Это значит, что есть возможность писать код в оболочке интерпретатора и вводить новые команды после выполнения предыдущих. К достоинствам данного языка можно отнести:

- бесплатность;
- кроссплатформенность;
- большое количество инструментов для анализа данных;
- возможность интеграции с языками: C и C++;

– большое количество литературы;
– мультипарадигменность.

К недостаткам можно отнести:

– низкая производительность относительно компилируемых языков программирования.

Также, для решения задач прогнозирования и анализа данных можно использовать и другие языки общего назначения: C++, Java, Ruby и т.п. Однако, основные задачи этих языков не состоят в анализе данных. Это значит, что инструментов для анализа данных в этих языках либо нет, либо их недостаточно. Однако, некоторые из этих языков удобно использовать в связке с предыдущими в некоторых задачах для повышения производительности.

Для данной задачи наиболее важными являются характеристики языка:

– производительность;
– наличие инструментов для прогнозирования временных рядов и анализа данных;
– наличие инструментов для визуализации данных;
– наличие инструментов для разработки пользовательского интерфейса;
– достаточное количество литературы;
– проработанность языка и удобство для разработчика;
– цена.

Для более удобного сравнительного анализа вынесем информацию о каждом языке программирования по этим критериям в таблицу 2.1.

Таблица 2.1 – Сравнительный анализ языков программирования

Характеристики	Julia	R	MATLAB	Python
Производительность	Наиболее производительный из перечисленных из-за наличия JIT компилятора	Относительно низкая производительность	Относительно низкая производительность	Относительно низкая производительность
Инструменты для прогнозирования и анализа данных	Имеются встроенные библиотеки	Большое число встроенных библиотек	Большое число встроенных библиотек	Большое число подключаемых библиотек

Продолжение таблицы

Характеристики	Julia	R	MATLAB	Python
Инструменты для разработки пользовательского интерфейса	Малое количество	Малое количество	Только встроенные инструменты	Большой выбор инструментов
Инструменты для визуализации данных	Большое количество библиотек	Только встроенные	Только встроенные	Большое количество библиотек
Цена	Бесплатный	Бесплатный	Платный	Бесплатный
Возможность использования языка для решения других задач	Может быть использован, но основной функционал рассчитан на анализ данных	Нет	Может быть использован, но основной функционал рассчитан на анализ данных	Большой спектр задач, для которых язык может быть использован

В данной работе необходимо разработать программное обеспечение для прогнозирования временных рядов с пользовательским интерфейсом. Это значит, что язык программирования должен иметь большое количество инструментов для прогнозирования, анализа данных, разработки графических интерфейсов и визуализации данных. Из таблицы 2.1 можно сделать вывод о том, что по выбранным критериям, для данной задачи наиболее подходящим языком программирования является python. Этот язык имеет две версии (2 и 3). Принципиальных различий для данной задачи в них нет. В работе был использован python версии 3.

2.3 Выбор инструментов для прогнозирования

Существуют различные библиотеки для решения задачи прогнозирования. Однако не все из них подходят для решения этой задачи. Необходимо выбрать существующую, либо разработать собственную. В прошлых пунктах были получены необходимые требования к библиотеке:

- прогнозирование методом непараметрической регрессии;
- язык программирования python.

Известные существующие решения на языке python:

- ARIMA;

– Prophet;
– Arch.

Среди этих решений ARIMA и Arch используют автогрегрессионный метод прогнозирования. Что не удовлетворяет выдвинутым ранее требованиям к инструментам прогнозирования. В отличии от них модель Prophet, реализованная в библиотеке fbprophet компанией Facebook, использует аддитивную непараметрическую регрессионную модель. Эта модель имеет вид [11]:

$$Y(t) = g(t) + s(t) + h(t) + \varepsilon_t, \quad (2.1)$$

где $g(t)$ – тренд;

$s(t)$ – сезонности;

$h(t)$ – аномальные дни;

ε_t – случайный компонент.

В пользу данной модели можно привести сравнение методов прогнозирования из [11]. Сравнение представлено на рисунке 2.2. На нем изображена зависимость средней абсолютной ошибки разных методов прогнозирования от количества дней в исходной выборке. На рисунке видно, что наименьшей средней абсолютной ошибкой обладает модель Prophet. Таким образом, для получения прогноза в данной работе будет использована именно эта модель.

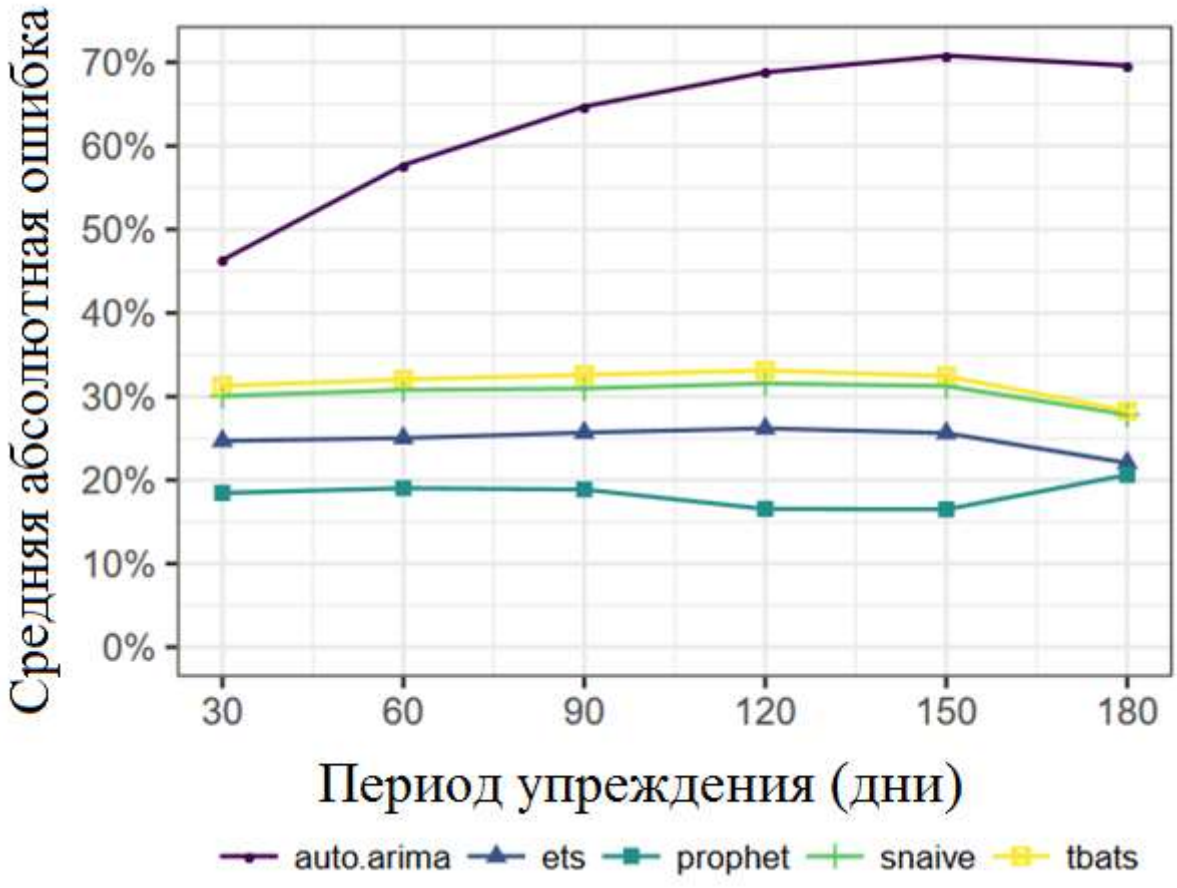

Рисунок 2.1 – Сравнение методов прогнозирования

2.4 Выбор инструментов для анализа данных

На данный момент существует большое количество инструментов для анализа данных для языка python. В контексте данной задачи необходимо определиться с инструментами для расчета статистических показателей и прогнозирования. Основными инструмента для расчета статистических показателей на языке python являются библиотеки:

- NumPy;
- Pandas;
- SciPy.

NumPy – свободная библиотека для языка python, написанная на языке Си. Предоставляет базовые методы и более удобные структуры для работы с данными. Основная особенностями этой библиотеки является поддержка одномерных и многомерных массивов, а также функционал для работы с ним.

Pandas – свободная библиотека языка python, для анализа и обработки данных написанная при помощи библиотеки NumPy. Предоставляет объекты: DataFrame и Series для более удобной работы с данными и богатый функционал для работы с ними. Эти типы данных являются ассоциативными массивами. Эта библиотека в большей степени предназначена для очистки и первичной обработки данных, чем для анализа.

SciPy – библиотека языка python, включающая в себя большой количество модулей для математических расчетов и анализа данных. Как и Pandas, основан на NumPy, но имеет набор функций для более сложных математических и статистических расчетов, таких как интегрирование, оптимизация и других.

Эти три инструмента, чаще всего, используют вместе для решения различных математических задач. Для решения поставленной задачи функционала библиотеки Pandas оказалось достаточно. Также, библиотека Fbprophet работает с типами данных, предоставляемыми Pandas. Поэтому в работе была использована только она.

2.5 Выбор инструментов для разработки пользовательского интерфейса

Следующим этапом является выбор средств разработки пользовательского интерфейса для языка python. Существует больше десятка библиотек для решения данной задачи. Из них, наиболее популярными являются библиотеки:

- Tkinter;
- PyGTK;
- PyQt.

Tkinter – кросс-платформенная свободная библиотека, предназначенная для разработки графических интерфейсов. Она входит в состав стандартной библиотеки python и является свободным программным обеспечением. Данный программный продукт подходит для разработки несложных программных продуктов с графическим интерфейсом. К преимуществам этого продукта можно отнести:

- простота в освоении;
- является свободным программным обеспечением;
- входит в стандартную библиотеку языка python;
- наличие достаточного количества литературы.

Недостатки:

- данный программный продукт является устаревшим;
- малое количество компонентов для разработки;
- отсутствие визуального редактора;
- нет возможности интегрировать инструменты для визуализации данных.

PyGTK – свободное кросс-платформенное программное обеспечение для разработки пользовательского интерфейса на языке python. Этот продукт является привязкой функционала библиотеки графического интерфейса GTK+, написанной на языке Си, к языку python. Преимущества этого языка:

- является свободным программным продуктом;
- большое количество компонентов пользовательского интерфейса;
- имеет достаточное количество литературы;
- наличие визуального редактора;
- есть возможность работы с инструментом matplotlib, который предназначен для визуализации данных.

PyQt – набор привязок фреймворка Qt для языка python. Qt является наиболее объемным инструментом графического программирования. К его достоинствам относятся:

- большее, относительно других инструментов, количество компонентов для разработки;
- большое количество литературы;
- подробная документация;
- наличие визуального редактора;
- есть возможность работы с инструментом matplotlib, который предназначен для визуализации данных.

Недостатки:

- сложен относительно других инструментов.

В контексте данной задачи, наиболее важными являются следующие критерии:

- количество компонентов;
- возможность интеграции инструментов для визуализации данных;
- количество документации.

Для более удобного сравнительного анализа этих библиотек, все критерии были вынесены в таблицу 2.2.

Из выше перечисленных инструментов наиболее подходящим является PyQt. Основное его достоинство перед остальными состоит в большем количестве компонентов доступных для разработки. Это предоставляет большое количество вариантов расширения программного продукта.

Qt framework имеет две версии (4 и 5), не совместимые друг с другом. Преимуществ у одной из них перед другой для данной задачи нет. Основное различие состоит в реорганизации модулей. В данной работе используется Qt версии 5.

Таблица 2.2. – Сравнительный анализ инструментов разработки пользовательского интерфейса

Критерии	TKinter	PYGTK	PyQt
Количество компонентов	Малое количество	Большое количество доступных компонентов	Имеет наибольшее количество из выбранных
Возможность интеграции инструментов для визуализации	Отсутствует	Имеется возможность интеграции с matplotlib	Имеется возможность интеграции с matplotlib
Количество документации	Большое количество информации	Большое количество информации	Большое количество информации и наиболее проработанная документация
Цена	Бесплатный	Бесплатный	Бесплатный

2.6 Выбор инструментов для визуализации данных

На предыдущем этапе был выбран PyQt 5, как инструмент для разработки пользовательского интерфейса разрабатываемого программного обеспечения. Это накладывает ограничение на инструменты визуализации. Этим ограничением является требование к возможности интеграции графиков в пользовательский ин-

терфейс. Наиболее известными программными продуктами для визуализации данных на языке python являются:

- plotly;
- matplotlib.

Plotly – библиотека с открытым исходным кодом, предназначенная для разработки интерактивных веб-приложений. Поддерживает языки:

- R;
- Python;
- MATLAB;
- Julia;
- Perl.

Предоставляет возможность отрисовки данных в режимах online и offline. К основным достоинствам данного языка можно отнести:

- простота;
- большой функционал;
- возможность построения интерактивных графиков.

К недостаткам:

- нет возможности интеграции графики в PyQt5;
- относительно, малое количество литературы.

Matplolib – бесплатная библиотека языка программирования python, предназначенная для визуализации двухмерных данных в задачах анализа данных. Является наиболее популярным инструментом визуализации данных в python. К основным достоинствам можно отнести:

- большой функционал;
- Большое количество информации и проработанная документация;
- возможность интеграции графики в PyQt5.

Недостатки:

- более сложный инструмент в сравнении с plotly.

Для данной задачи наиболее значимыми являются следующие критерии:

- проработанность документации;
- возможность интеграции в интерфейс, разработанный при помощи PyQt5;
- цена.

Сравнительный анализ описанных инструментов по этим критериям представлена в таблице 2.3.

Таблица 2.3 – Сравнительный анализ инструментов визуализации данных

Критерии	Plotly	Matplotlib
Проработанность документации	Малый объем информации	Достаточно проработанная документация
Возможность интеграции в интерфейс PyQt5	Отсутствует	Имеется
Цена	Бесплатен	Бесплатен

Из таблицы 2.3 видно, что matplotlib является наиболее подходящей для решения этой задачи.

Выводы по главе два

В данной главе рассмотрены основные инструменты, применяемые для решения подобных и смежных задач. В качестве языка программирования выбран python. Стек используемых технологий следующий:

- Prophet;
- Pandas;
- PyQt5;
- Matplotlib.

На основе этих инструментов необходимо разработать программный продукт для анализа и прогнозирования временных рядов.

3 РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

3.1 Разработка внутренней логики

При создании любого программного продукта всегда встает вопрос о разработке его внутренней логики. Кроме требований, указанных в техническом задании необходимыми требованиями к внутренней структуре являются:

- читабельность;
- расширяемость.

Читабельность подразумевает организацию структуры программы таким образом, чтобы разработчик тратил как можно меньшее время на понимание того, как работает программа, и что является результатом ее работы в будущем. Расширяемость предполагает минимальную связность программного кода, то есть организацию структуры программы таким образом, чтобы изменение части программы не требовало изменения другой ее части, либо требовало несущественных изменений.

Эти требования могут выполняться различным образом в зависимости от используемой парадигмы программирования. На данный момент известными и наиболее подходящими для данной задачи являются следующие парадигмы:

- процедурное;
- функциональное;
- объектно-ориентированное.

Процедурное программирование – парадигма программирования, которая основана на использовании процедур (подпрограмм) при написании программного кода. Процедура может быть вызвана сколько угодно раз в любой части программы, в том числе и внутри себя. Данный метод написания программы позволяет

- упростить написание кода;
- уменьшить размер программы за счет повторного использования подпрограмм.

Главным недостатком такого подхода к программированию является наличие большого числа глобальных переменных. Это затрудняет поиск ошибок, а значит, увеличивает время разработки и расширения программного продукта.

Функциональное программирование – парадигма программирования, в основу которой положено понимание функций в математическом представлении, а не как подпрограмм или процедур в привычном понимании. В других парадигмах функция рассматривается как некоторая подпрограмма, которая может менять состояние других частей программы, внешних по отношению к ней. В основе этой парадигмы лежат чистые функции. Чистыми называют функции, которые не производят изменения внешних, по отношению к ней, частей программы. Это решает

проблемы процедурного программирования, и, как следствие, оптимизирует процесс написания программного продукта.

Объектно-ориентированное программирование – парадигма программирования, в которой основной функционал представлен в виде объектов и их взаимодействия друг с другом. Объектом, в данном случае, является некоторая сущность, представленная программно. Эта сущность имеет свое внутренне состояние и поведение. Объекты являются экземплярами некоторого класса, который и описывает их внутреннюю структуру. Объектно-ориентированный подход основан на следующих принципах:

1. абстракция – выделение наиболее значимых свойств объекта в контексте данной задачи;
2. инкапсуляция – механизм объединения и сокрытия данных, с целью не допустить изменения внутреннего состояния объекта из внешней части программы;
3. полиморфизм – способ взаимодействия с объектами разных классов, как с объектами одного класса в пределах одной иерархии;
4. наследование – способ расширения и переопределения функционала базового класса классом наследником.

В языке программирования python возможно разрабатывать программный продукт с использованием любой из трех парадигм. Объектно-ориентированное и функциональное программирование не имеет недостатков присущих процедурному стилю. Объектно-ориентированный и функциональный способы не имеют принципиальных преимуществ для решения данной задачи. Поэтому, возможно использование любого из них. Для разработки продукта был выбран объектно-ориентированный подход.

Для выполнения требований читабельности и расширяемости недостаточно просто выбрать некоторую парадигму программирования. Необходимо также реализовать такую структуру, в пределах выбранной парадигмы, при помощи которой выполнялись бы требования, описанные выше. В объектно-ориентированном подходе для формирования этой структуры используются методы, называемые паттернами проектирования. Наиболее подходящим для выполнения требований данной задачи, является паттерн «Компонент» [12]. Этот паттерн используется в ситуации, когда некоторый класс должен охватывать большое количество задач, не связанных или слабо связанных друг с другом. В этом паттерне происходит делегирование отдельных задач класса экземплярам других классов. Для графического представления отношений между классами в объектно-ориентированном программировании используют UML-диаграммы. UML-диаграмма данного паттерна изображена на рисунке 3.1. Учитывая особенности языка программирова-

ния python, можно исключить интерфейсы из этой диаграммы. UML диаграмма программы изображена на рисунке 3.2.

В данной работе реализован класс Analyzer, который предназначен не только для прогнозирования, но и для анализа данных. Расчет статистических показателей и прогнозирование – это разные и независимые задачи в контексте данной работы. Экземпляры класса Analyzer делегируют эти задачи экземплярам следующих классов:

- Statistics – расчет статистических показателей;
- Forecast – прогнозирование.

Класс Statistics содержит методы:

- analysis – производит расчет статических показателей (кроме корреляции и ковариации);
- pearsCorr – расчет корреляции Пирсона между рядами;
- covr – расчет ковариации между двумя рядами.

В качестве единственного параметра, analysis принимает временной ряд, характеристики которого необходимо рассчитать. Возвращаемым значением является значение типа bool. Значение True говорит о том, что метод выполнился без ошибок, иначе возвращается False. pearsCorr и covr в качестве параметров принимают два ряда, между которыми необходимо рассчитать корреляцию Пирсона и ковариацию соответственно. Стоит отметить, что экземпляры этого класса не сохраняют результаты работы методов pearsCorr и covr в качестве своих полей, а лишь возвращают их после выполнения. Такая реализация обусловлена тем, что другие характеристики ряда зависят лишь от исходных значений ряда. В отличии от них, значения корреляции и ковариации зависят не только от исходного, но и от ряда, относительного которого необходимо рассчитать эти характеристики. Это означает, что эти показатели характеризуют не ряд, а отношения двух рядов. При отсутствии параметров у этих методов, им передается значение None.

Класс содержит следующие поля:

- min – минимальное значение ряда;
- max – максимальное значение ряда;
- mean – среднее значение ряда;
- median – медиана ряда;
- var – дисперсия ряда;
- std – среднеквадратичное отклонение ряда;
- mad – медиана абсолютных отклонений ряда.

По умолчанию, все значения содержат None. Для каждого из полей настроено свойство-геттер.

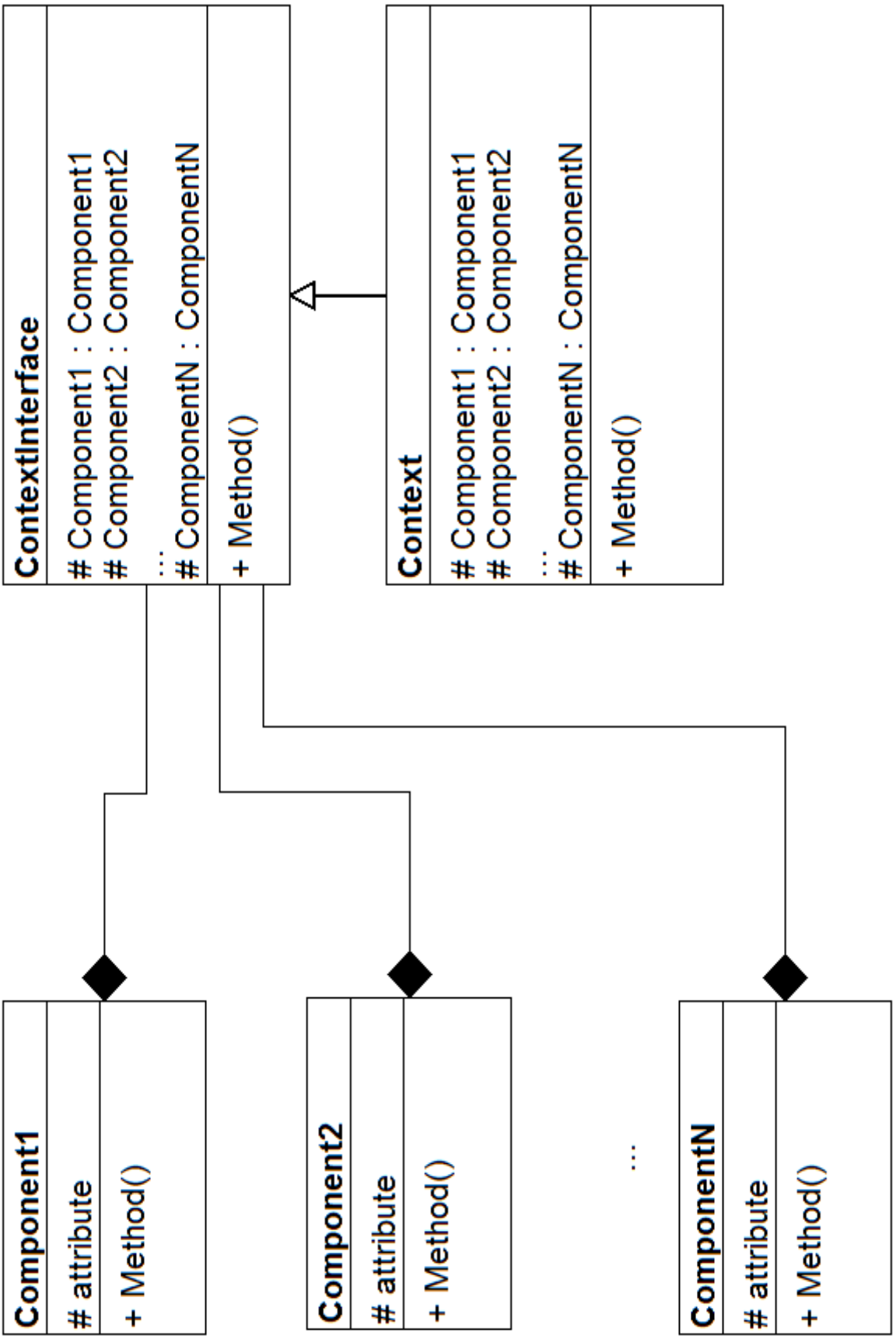

Рисунок 3.1 – UML диаграмма паттерна «Компонент»

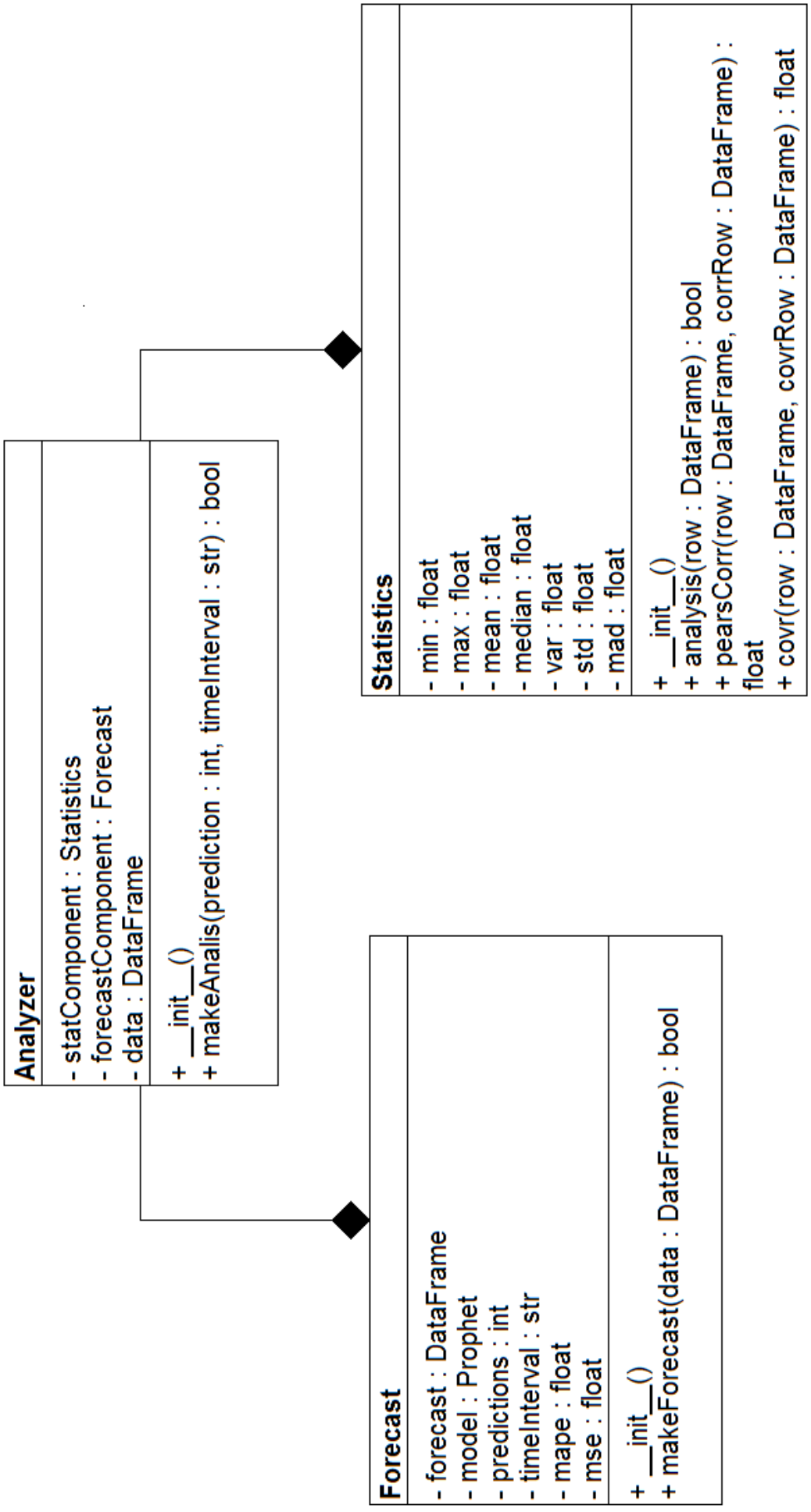

Рисунок 3.2 – UML диаграмма программы

Класс Forecast инкапсулирует следующие поля:

- predictions – период упреждения прогноза(по умолчанию равен 3);
- forecast – прогноз(по умолчанию равен None);
- timeIntervals – символ, который показывает, по каким промежуткам времени производится прогноз(по умолчанию равен ‘D’);
- model – модель Prophet(по умолчанию равен None);
- mse – среднеквадратичная ошибка(по умолчанию равен None);
- mape – средняя абсолютная ошибка в процентах(по умолчанию равен None).

Возможные значения для timeIntervals:

- ‘D’ – прогнозирование по дням;
- ‘W’ – прогнозирование по неделям;
- ‘Y’ – прогнозирование по годам.

Для каждого поля настроено свойство-геттер, а для predictions и timeIntervals еще и свойство-сеттер.

Forecast содержит метод makeForecast, которое в качестве параметра принимает временной ряд, прогнозирование которого необходимо произвести. После расчета прогноза, метод считает среднеквадратичную и среднюю абсолютную ошибки. Возвращаемое значение – значение типа bool. Значение True говорит об отсутствии ошибок во время выполнения метода, а False – об их наличии.

Экземпляры класса Analyzer содержат поля:

- statComponent – экземпляр класса Statistics;
- forecastComponent – экземпляр класса Forecast;
- data – временной ряд.

По умолчанию, поле data равен None. Поэтому перед анализом и прогнозирования следует инициализировать это поле. Класс содержит метод makeAnalisis. Этот метод производит анализ и прогнозирования временного ряда, при помощи вызова методов соответствующих компонентов. В качестве параметров принимает:

- prediction – символ, который показывает, по каким промежуткам времени производится прогноз(по умолчанию равен ‘D’);
- timeInterval период упреждения прогноза(по умолчанию равен 3).

Эти параметры передаются необходимым методам его компонентов. Возвращаемое значение – значение типа bool. Значение True говорит об отсутствии ошибок во время выполнения метода, а False – об их наличии.

Стоит отметить, что в качестве временных рядов используется тип DataFrame, определенный в библиотеке Pandas [13]. Он представляет из себя таблицу значений. Эта таблица должна иметь два столбца:

– ds – значения типа DateTime;
– y – значения ряда.

Так же экземпляры класса содержат свойство-геттер data. Методы __init__ всех классов, вызываемый при создании нового экземпляра, производит инициализацию полей объекта.

Программный код внутренней логики программы содержится в файлах:

– statistics.py – определение класса Statistics;
– forecast.py – определение класса Forecast;
– TRAnalyzer.py – определение класса Analyzer.

3.2 Разработка пользовательского интерфейса

В состав Qt framework входит свободная кроссплатформенная среда разработки графических интерфейсов Qt Designer. Эта среда предназначена для облегчения и ускорения разработки графического пользовательского интерфейса. Изменение вида пользовательского интерфейса происходит при помощи перетаскивания соответствующего компонента с панели компонентов на экран приложения. Данные о расположении компонентов на экране сохраняются в текстовый файл с расширением ui. Этот файл имеет xml формат. В данной работе Qt Designer был использован для начального проектирования интерфейса.

Далее, по данным из этого файла необходимо производить отображение компонентов пользовательского интерфейса. Для этого существует два способа:

– статический;
– динамический.

Статический предполагает получение модуля с кодом python, в котором производится начальная настройка компонентов. При динамическом способе производится вызов функции loadUi модуля uic прямо во время выполнения кода. Параметрами этой функции являются путь к файлу и ссылка на объект, в котором она вызывается. Принципиальной разницы между этими способами нет. В работе используется статический метод.

Пользовательский интерфейс представляет собой единственное окно с набором компонентов. Поэтому, результатом работы утилиты uic является файл uiDesign.py, в котором находится определение единственного класса Ui_MainWindow. При использовании статического метода было решено произвести наследование функционала класса Ui_MainWindow классом AppClass. В нем производится определение всех методов программы пользовательского интерфейса. Это необходимо, так как в ходе работы пользовательский интерфейс может быть изменен, а значит, данный файл будет перезаписан. Это возможно и при доработке этого проекта в будущем. Определение класса AppClass находится в файле appClass.py.

Для запуска пользовательского интерфейса необходимо произвести первоначальные настройки для корректной работы приложения при помощи Qtframework и создать экземпляр класса appClass. Все это производится в файле main.py.

Все используемые компоненты записаны в таблице 3.1. Кроме компонентов, описанных в таблице 3.1, в программе используется большое количество компонентов класса QLabel. Они используются как подписи к другим компонентам и рассмотрения не требуют. Поэтому, они не были вынесены в таблицу 3.1. Внешний вид программы с подписями представлен на рисунке 3.4.

Для описания поведения программы при наступлении некоторых событий в Qt framework используется механизм сигналов и слотов [14]. При наступлении определенного события происходит генерация соответствующего сигнала. Сигнал вызывает метод-слот. Слотом является метод класса, присоединенный к сигналу. Связь слотов и сигналов представлена на рисунке 3.3. Сигналы и слоты, используемые в этой работе, представлены в таблицах 3.2 и 3.3 соответственно.

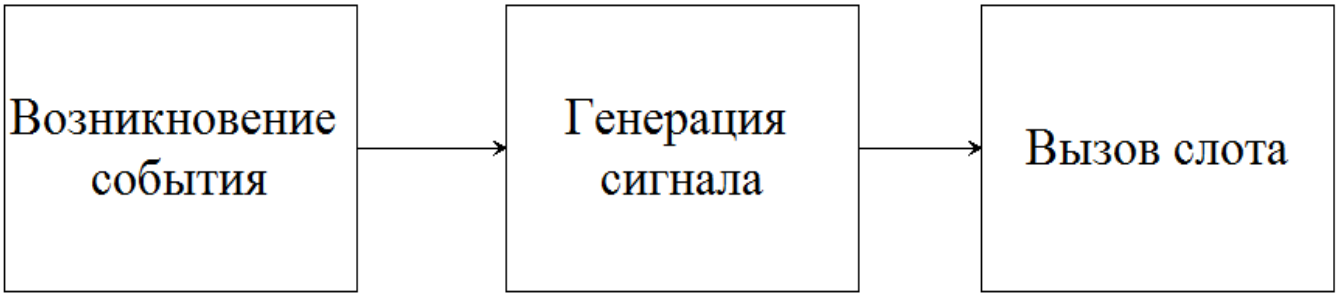

Рисунок 3.3 – Связь слотов и сигналов в Qt framework

Таблица 3.1 – Список используемых компонентов

Имя	Класс	Описание
centralWidget	QMainWindow	Окно программы
graphBox	QGroupBox	Используется как компонент, на котором будет производится отрисовки графиков
statBox	QGroupBox	Используется для группировки компонентов, предназначенных для вывода статистических показателей
setBox	QGroupBox	Используется для группировки компонентов, используемых для предварительной настройки

Продолжение таблицы

Имя	Класс	Описание
nameEdit	QLineEdit	Используется для вывода имени файла с данными
performButton	QPushButton	При нажатии производится анализ и прогнозирование
toolButton	QTollButton	Используется для открытия диалогового окна и получения имени, выбранного файла
rowList	QComboBox	Позволяет выбрать временной ряд из файла, для которого необходимо произвести анализ и прогнозирование
rowList_2	QComboBox	Позволяет выбрать временной ряд из файла, для которого необходимо рассчитать корреляцию и ковариацию по отношению к ряду из rowList
checkCorr	QCheckBox	Если отмечен, становится доступным расчет корреляции
checkCovr	QCheckBox	Если отмечен, становится доступным расчет ковариации
timeIntervals	QComboBox	Позволяет выбрать временные промежутки, по которым необходимо производить прогноз (дни, недели, месяцы, года)
intervalSpin	QSpinBox	Позволяет выбрать период упреждения
menuBar	QMenuBar	Представляет собой строку меню, которая содержит набор выпадающих списков меню
menu	QMenu	Представляет выпадающий список меню
actionexport_to_csv	QAction	Элемент списка меню, предназначенный для импорта расчетов в файл с расширением csv
actionexport_to_png	QAction	Элемент списка меню, предназначенный для импорта графиков в файлы с расширением png
actionabout	QAction	Элемент списка меню, предназначенный для получения краткой справки о программе
actionexit	QAction	Элемент списка меню, предназначенный для выхода из программы

Таблица 3.2 – Сигналы, используемые в данной работе

Название	Генерирующие компоненты	Описание
Clicked	performButton, toolButton	Генерируется при нажатии на экземпляр QButton и QToolButton
stateChanged	checkCorr, checkCovr	Генерируется при изменении состояния экземпляров QCheckBox
triggered	actionimport_to_csv, actionimport_to_png, actionabout, actionexit	Генерируется при выборе опции экземпляра QMenu
resizeEvent	centralWidget	Генерируется при изменении размеров окна приложения

Таблица 3.3 – Слоты, реализованные в работе

Название	Компонент	Сигнал
actionExit_triggered	actionexit	triggered
actionAbout_triggered	actionabout	triggered
actionExportToPng_triggered	actionexport_to_png	triggered
actionExportToCsv_triggered	actionexport_to_csv	triggered
checkCorr_StateChanged	checkCorr	stateChanged
toolButton_onClick	toolButton	clicked
resizeEvent	centralWidget	resize
checkCovr_StateChanged	checkCovr	stateChanged
performButton_onClick	performButton	clicked

Стоит отметить, что компонент actionexport_to_png производит экспорт данных в 3 файла:

- defaultRow.png – исходный ряд;
- trend.png – тренд;
- forecast.png – прогноз.

actionexport_to_csv производит экспорт данных в файл csv. Имя файла указывается пользователем. Файл csv представляет собой несколько столбцов с наборами значений. В данной работе, программа экспортирует данные в файл со столбцами:

- ds – даты;
- yhat – прогноз;
- trend – тренд.

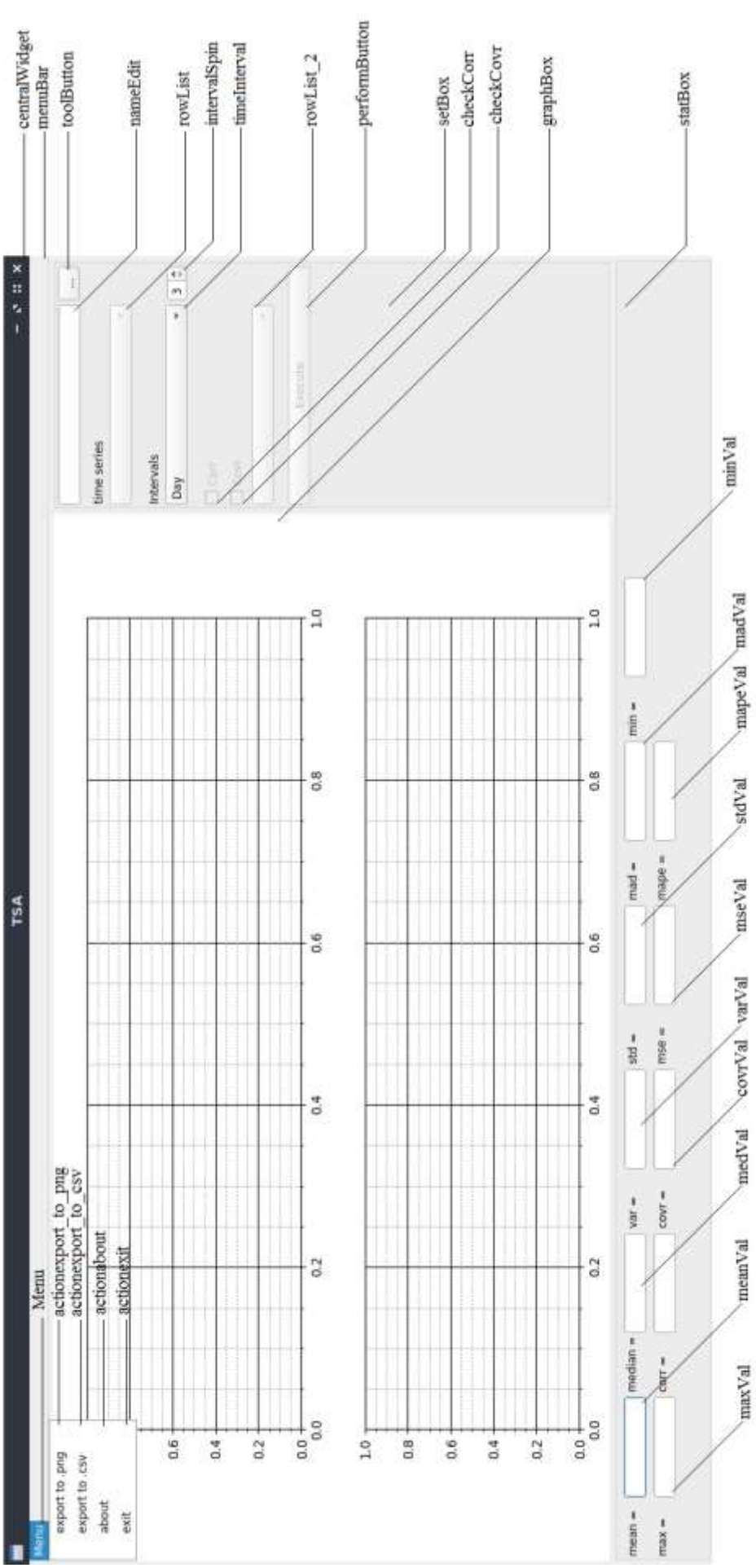

Рисунок 3.4 – Внешний вид пользовательского интерфейса программы с подписями компонентов

3.3 Визуализация данных

Для визуализации данных, реализованной методами библиотеки matplotlib, был написан класс PlotCanvas. Этот класс наследует функционал класса FigureCanvasQTAgg из matplotlib.backends.backend_qt5agg[15]. FigureCanvasQTAgg является классом-контейнером верхнего уровня для работы с графикой в Qt5. Особенность этого класса FigureCanvasQTAgg является метод setParent. Он позволяет выбрать компонент Qt5, в контексте которого необходимо производить отрисовку графической информации. В качестве параметра принимает ссылку на этот компонент. Класс инкапсулирует объекты:

- figure – экземпляр класса Figure библиотеки matplotlib;
- graph – экземпляр класса Axes библиотеки matplotlib.

Объект figure является контейнером для graph. А на объекте graph производится отрисовка данных. Далее, figure передается как параметр конструктору класса FigureCanvasQTAgg, и, при помощи метода draw этого же класса, проиводится отрисовка данных на соответствующем компоненте.

Метод __init__ класса PlotCanvas производит начальную отрисовку и некоторые настройки. Параметры метода:

- parent – объект, на котором необходимо произвести отрисовку графиков;
- width – ширина полотна;
- height – высота полотна;
- dpi – количество точек на дюйм.

Отрисовка данных производится методом plot. Параметры метода:

- data – данные для графиков;
- title – подпись к графику;
- color – цвет линии;
- linewidth – толщина линии;
- label – подпись линии;
- legend – легенда.

Также класс предусматривает очистку содержимого объекта. Для этого используется метод clear, который не принимает параметров. Этот метод является оберткой для метода clear объекта figure. Вывод графиков производится в компонент graphBox класса QGroupBox. Описание этого компонента представлено в таблице 3.1. При этом для графиков созданы два экземпляра класса PlotCanvas:

- forecastPlot – содержит графики исходных данных, прогноза и доверительного интервала;
- trendPlot – содержит график тренда.

Эти объекты представлены на рисунке 3.5.

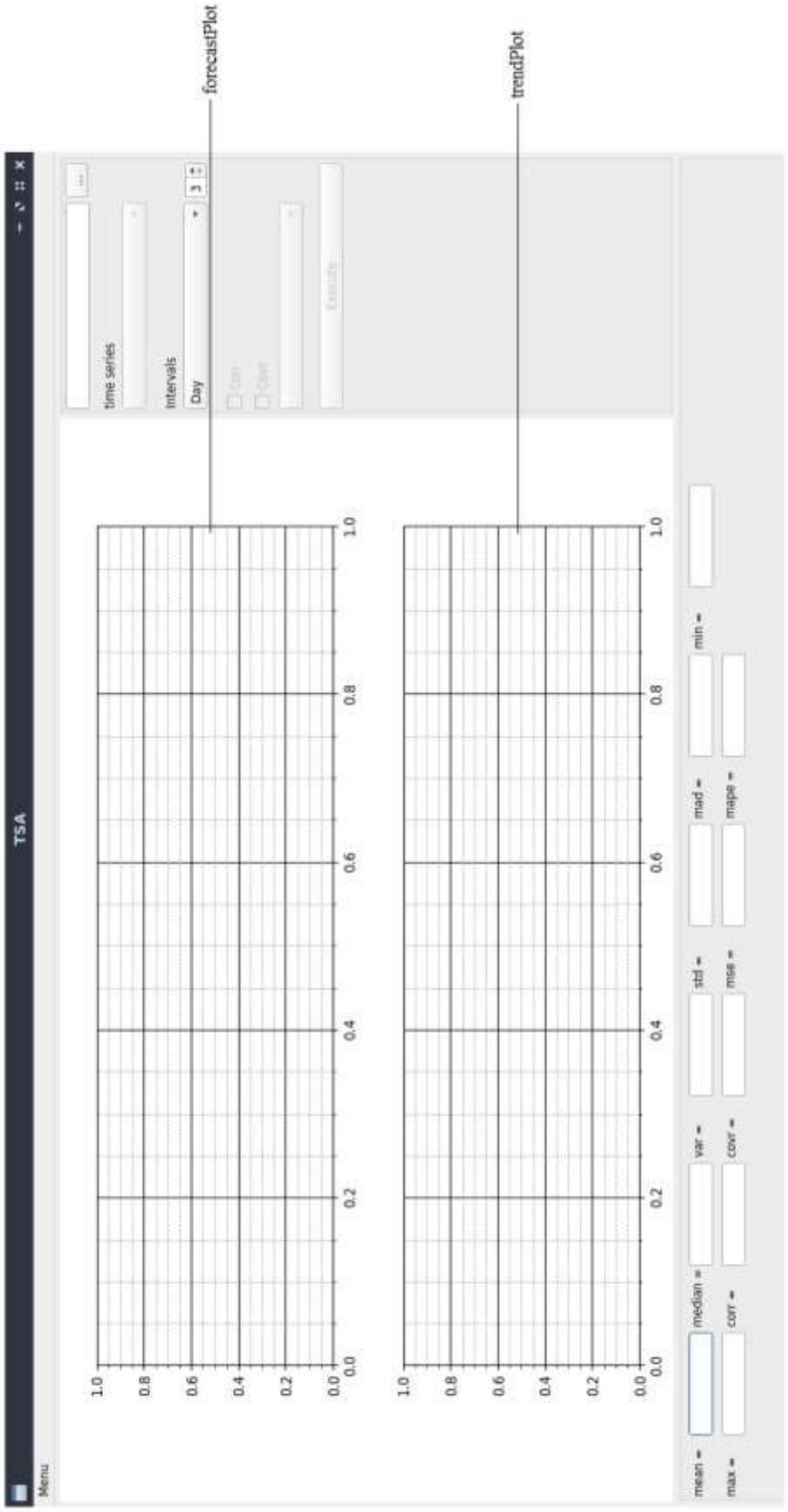

Рисунок 3.5 – Внешний вид пользовательского интерфейса программы с подписями объектов типа PlotCanvas

3.4 Тестирование программного продукта

Под тестированием программного продукта будем понимать работу программы в процессе прогнозирования и анализа реальных данных. В качестве исходных данных используется информация об отказе устройств предприятия АО «Златмаш». Далее, для проверки качества работы программы необходимо произвести удаление части данных из исходной выборки и сравнение полученных результатов с реальными данными. В ходе проверки была удалена информация о последних 40 днях. Результат работы программы представлен на рисунке 3.9. Сравнение исходных данных с прогнозом представлено на рисунке 3.10. Из рисунка 3.9 видно, что средняя абсолютная ошибка в процентах составляет 18,7309%, а из рисунка 3.10, что полученный прогноз достаточно точно предсказывает изменение данных на последующие 40 дней в сравнении с исходными, что говорит о хорошем качестве программного продукта. На рисунках 3.6 – 3.8 представлены результаты экспорта данных в файлы png. Формат экспорта данных в файл csv представлен на рисунке 3.11.

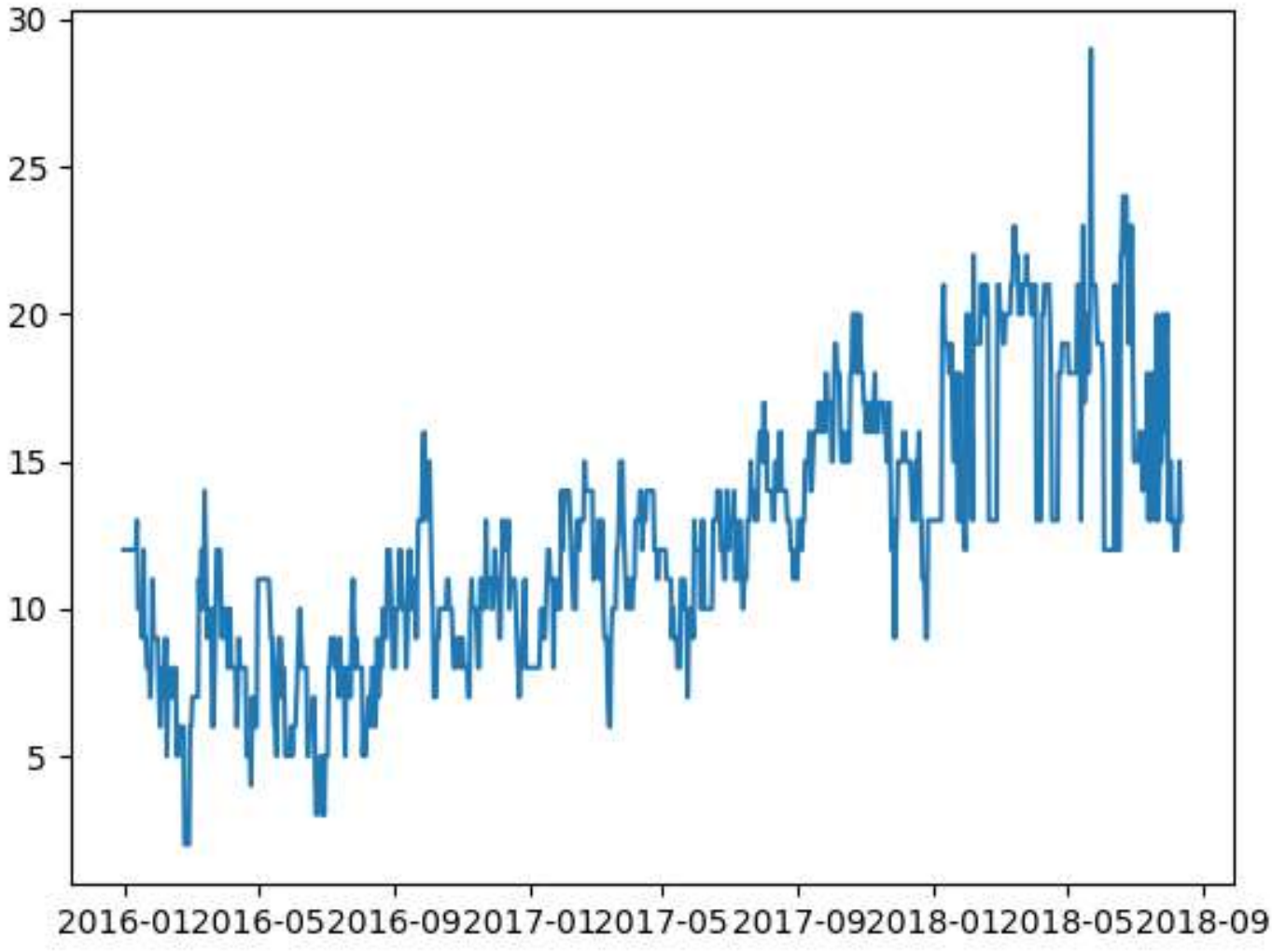

Рисунок 3.6 - Результат экспорта исходных данных в файл defaultRow.png

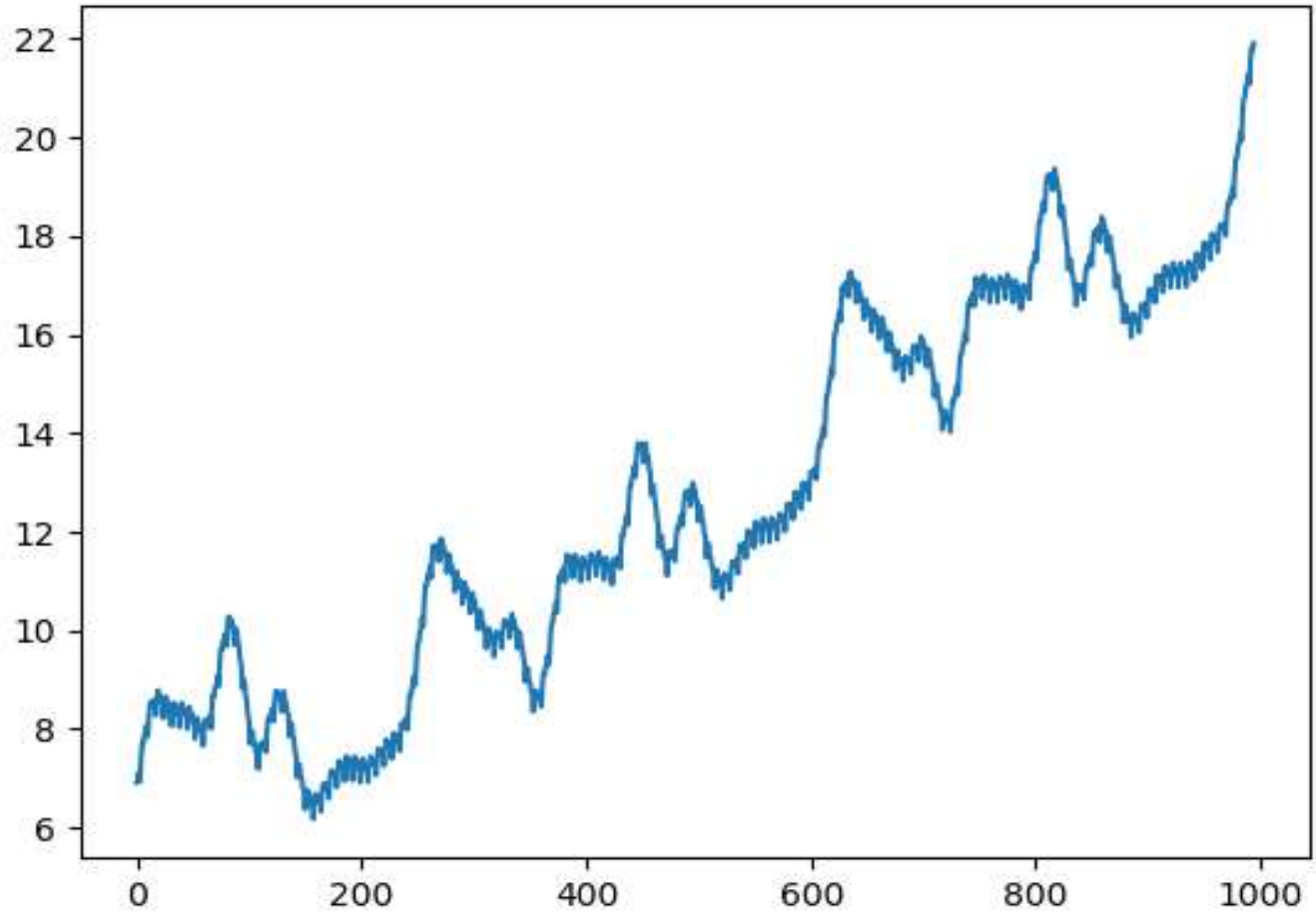

Рисунок 3.7 - Результат экспорта исходных данных в файл forecast.png

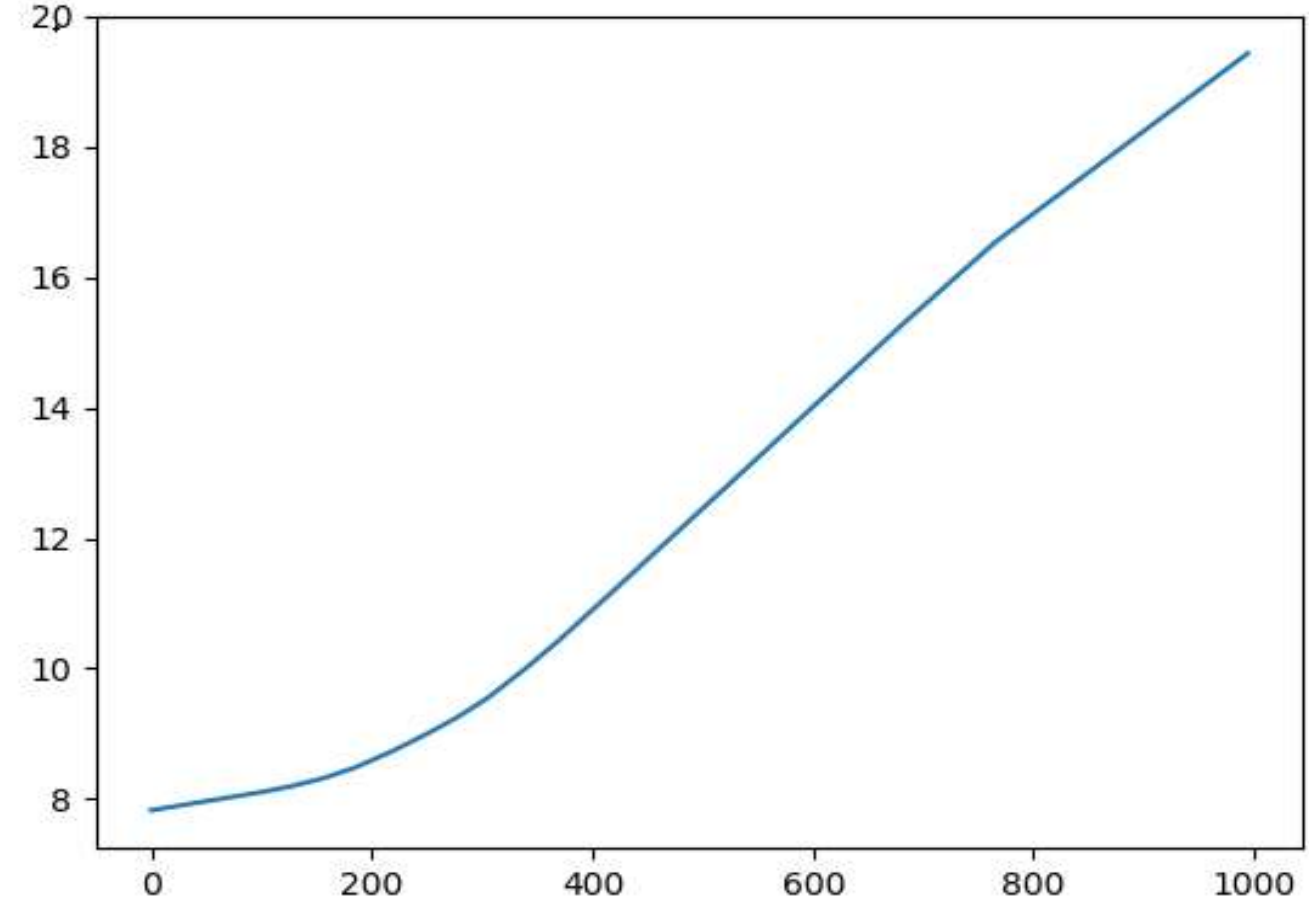

Рисунок 3.8 – Результат экспорта исходных данных в файл trend.png

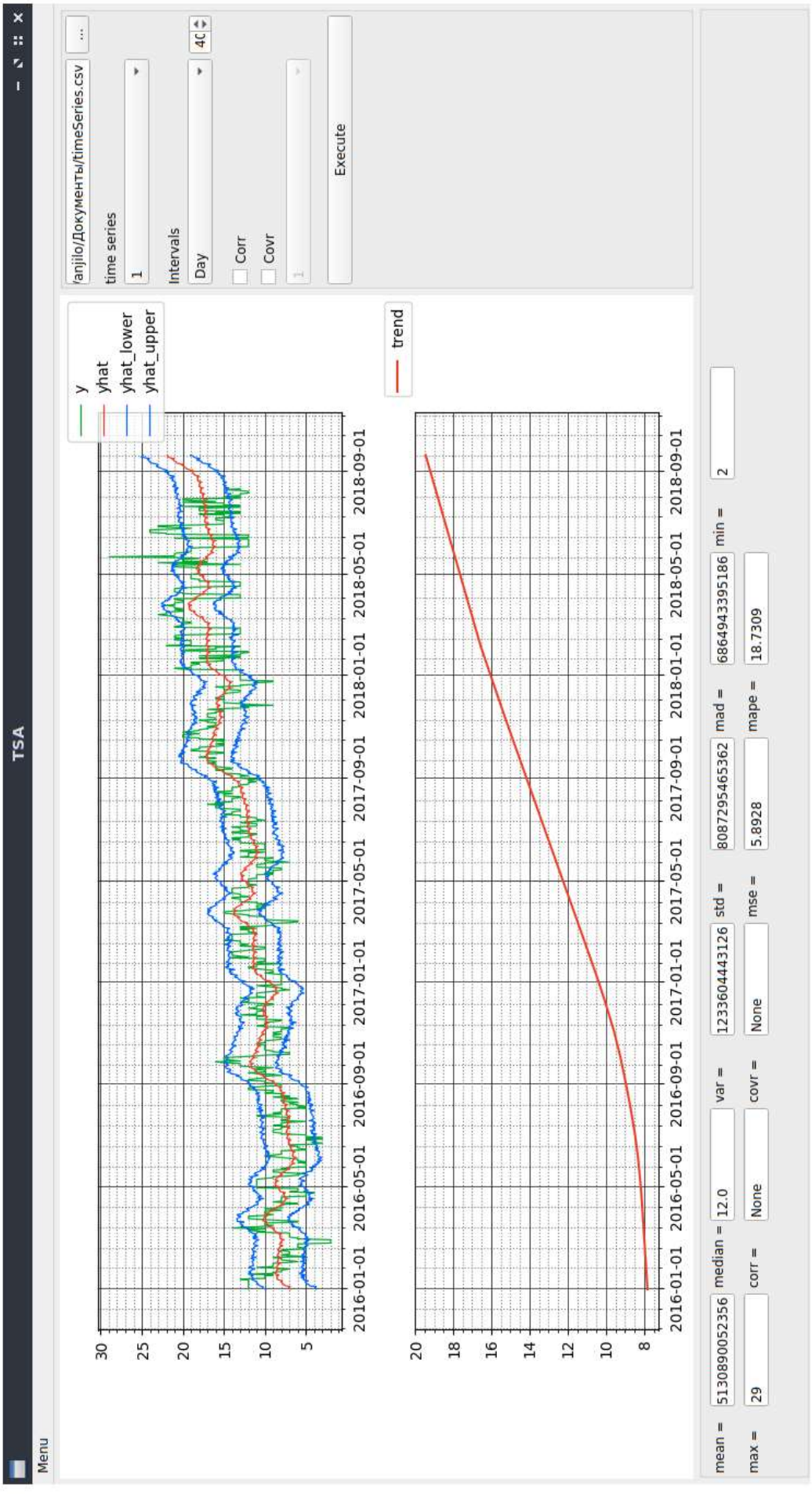

Рисунок 3.9 - Результат прогнозирования ряда с периодом упреждения 40 дней

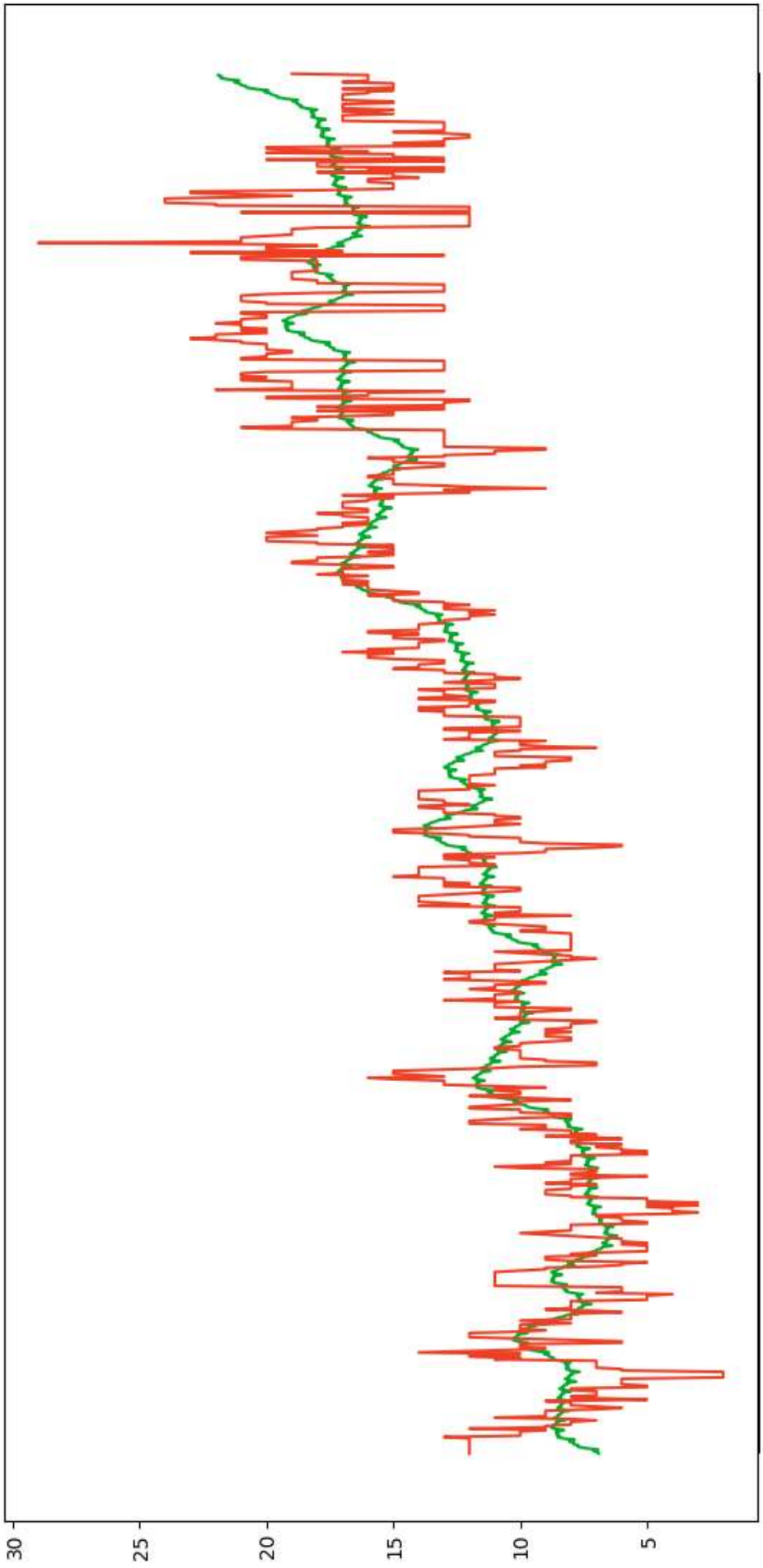

Рисунок 3.10 – Сравнение исходных данных с прогнозными

```
ds,yhat,trend
2016-01-01,6.910277,7.823334
2016-01-02,6.924559,7.826079
2016-01-03,7.119648,7.828824
2016-01-04,6.934966,7.831569
2016-01-05,7.42913,7.834314
2016-01-06,7.716203,7.837059
2016-01-07,7.713185,7.839804
2016-01-08,7.896677,7.842549
2016-01-09,7.903146,7.845293
2016-01-10,8.076007,7.848038
2016-01-11,7.855645,7.850783
2016-01-12,8.301917,7.853528
2016-01-13,8.530373,7.856273
2016-01-14,8.459698,7.859018
```

Рисунок 3.11 - Формат экспорта данных в файл csv

Выводы по главе три

Выполнена разработка программного продукта на основе инструментов, выбранных во второй главе. Программа включается в себя файлы:

- statistics.py;
- forecast.py;
- TRAnalyzer;
- uiDesign.py
- appClass.py;
- plot.py;
- main.py.

Листинг программы представлен в приложении А.

Качество работы программы оценено при помощи прогнозирования и анализа реальных данных для компании АО «Златмаш». Результаты тестирования представлены в пункте 3.4.

ЗАКЛЮЧЕНИЕ

Разработано программное обеспечение для прогнозирования и анализа временных рядов. Разработка является более удобной и простой альтернативой имеющимся на рынке решениям. Пользователю предоставляется возможности:

– представление прогноза в графическом виде в окне программы;

– расчет статистических показателей;

– экспорт данных в png и csv форматы.

Исходные данные представляются в виде csv файла, в котором обязательно должен присутствовать столбец ds. Этот столбец содержит даты в формате yy-mm-dd. Остальные столбцы – данные для прогнозирования и анализа.

Программный продукт показал себя состоятельным при прогнозировании и анализе реальных данных для предприятия АО «Златмаш». Средняя абсолютная ошибка в процентах составляет 18,7309%, а полученный прогноз достаточно точно предсказывает изменение данных на последующие 40 дней в сравнении с исходными.

В работе использованы следующие инструменты:

– Prophet;

– Pandas;

– PyQt5;

– Matplotlib.

Все требования технического задания успешно выполнены.

В дальнейшем планируется расширение функционала. Расширение предполагает предоставление возможности удаления данных из исходной выборки, в случае, если они не удовлетворяют пользователя.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Садовникова, Н.А. Анализ временных рядов и прогнозирование. Учебное пособие / Н.А. Садовникова, Р.А. Шмойлова. – М.: Изд-во МГУЭСИ, 2001. – 67 с.

2 Microsoft Excel [Электронный ресурс]. Режим доступа – https://products.office.com/ru-RU/Excel – Заглавие с экрана.

3 Novo Forecast [Электронный ресурс]. Режим доступа – https://novoforecast.com – Заглавие с экрана.

4 GNU PSPP [Электронный ресурс]. Режим доступа – http://www.gnu.org/software/pspp/ – Заглавие с экрана.

5 SPSS [Электронный ресурс]. Режим доступа – https://ru.wikipedia.org/wiki/SPSS – Заглавие с экрана.

6 Четыркин, Е.М. Статистические методы прогнозирования / Е.М. Четыркин. – 2-е изд., перераб. и доп. – М.: Изд-во «Статистика», 2007. – 199 с.

7 Anderson, T.W. The statisctical analysis of time series. / T.W. Anderson; пер. с англ. И.Г. Журбенко, В.П. Носко. – М.: Изд-во «Мир», 1976. – 756 с.

8 Бокс, Дж. Анализ временных рядов. Прогноз и управление: в 3 т. / Дж. Бокс, Г. Дженкинс. – М.: Изд-во «Мир», 1974. – Т. 1. – 406 с.

9 Hardle, W. Applied nonparametric regression. / W. Hardle; пер. с англ. А.В. Назин; под ред. М.Б. Малютова. – М.: Изд-во «Мир», 1993. – 352 с.

10 Lewis, C.D. Industrial and business forecasting methods / C.D. Lewis; пер. с англ. Е.З. Демиденко. – М.: Изд-во «Финансы и статистика», 1986. – 135 с.

11 Taylor, S.J. Forecasting at scale / S.J. Taylor, B. Letham // The American Statistician. – 2018. – 72(1). – P. 37-45.

12 Game programming patterns [Электронный ресурс]. Режим доступа – https://www.gameprogrammingpatterns.com/ – Заглавие с экрана.

13 Pandas 0.24.2 documentation [Электронный ресурс]. Режим доступа – http://pandas.pydata.org/pandas-docs/stable/reference/ – Заглавие с экрана.

14 Qt documentation [Электронный ресурс]. Режим доступа – https://doc.qt.io/ – Заглавие с экрана.

15 Matplotlib [Электронный ресурс]. Режим доступа – https://matplotlib.org/ – Заглавие с экрана.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А. Листинг программы

statistics.py:

```
# -*- coding: utf8 -*-
import pandas as pd
import numpy as nm

class Statistics(object):
    def __init__(self):
        self.__min = None
        self.__max = None
        self.__mean = None
        self.__median = None
        self.__var = None
        self.__std = None
        self.__mad = None

    def analysis(self,row = None):
        try:
            self.__min = row['y'].min()
            self.__max = row['y'].max()
            self.__mean = row['y'].mean()
            self.__median = row['y'].median()
            self.__var = row['y'].var()
            self.__std = row['y'].std()
            self.__mad = row['y'].mad()
        except Exception as ex:
            print(ex)
            return False
        return True

    def pearsCorr(self,row = None,corrRow = None):
        try:
            data = {'col1' : row['y'], 'col2' : corrRow['y']}
        except Exception as ex:
            print(ex)
            return None
        newDataFrame = pd.DataFrame(data)
        corr = newDataFrame.corr(method = 'pearson')
        return corr['col1'][1]

    def covr(self,row = None,covrRow = None):
        try:
            data = {'col1' : row['y'], 'col2' : covrRow['y']}
        except Exception as ex:
            print(ex)
            return None
        newDataFrame = pd.DataFrame(data)
        covr = newDataFrame.cov(min_periods = 1)
```

```
        return covr['col1'][1]

    @property
    def min(self):
        return self.__min
    @property
    def max(self):
        return self.__max
    @property
    def mean(self):
        return self.__mean
    @property
    def median(self):
        return self.__median
    @property
    def var(self):
        return self.__var
    @property
    def std(self):
        return self.__std
    @property
    def mad(self):
        return self.__mad
```

forecast.py:

```
# -*- coding: utf8 -*-
import pandas as pd
from fbprophet import Prophet
import numpy as np
import matplotlib.pyplot as plt
import datetime as dt

class Forecast(object):
    def __init__(self):
        self.__forecast = None
        self.__model = None
        self.__predictions = 3
        self.__timeInterval = 'D'
        self.__mape = None
        self.__mse = None

    def makeForecast(self,data = None):
        self.__model = Prophet()
        try:
            data = data.groupby('ds')[['y']].sum()
            data = data.reset_index()[['ds','y']]
```

```
            self.__model.fit(data)
            future = self.__model.make_future_dataframe(periods =
self.__predictions,freq = self.__timeInterval,
include_history = True)
        except Exception as ex:
            print(ex)
            return False

        self.__forecast = self.__model.predict(future)
        self.__forecast = round(self.__forecast,6)
        er = data['y'] - self.__forecast['yhat']
        er_m = er/data['y']*100

        if self.__predictions != 0:
            self.__mape = round(np.mean(np.abs(er_m[:
               -self.__predictions])),4)
            self.__mse = round(np.mean(np.square(er[:
               -self.__predictions])),4)
        else:
            self.__mape = round(np.mean(np.abs(er_m)),4)
            self.__mse = round(np.mean(np.square(er)),4)
        return True

    @property
    def predictions(self):
        return self.__predictions
    @property
    def forecast(self):
        return self.__forecast
    @property
    def mape(self):
        return self.__mape
    @property
    def mse(self):
        return self.__mse
    @property
    def model(self):
        return self.__model
    @property
    def timeInterval(self):
        return self.__timeInterval

    @predictions.setter
```

```
    def predictions(self,predict):
        self.__predictions = predict

    @timeInterval.setter
    def timeInterval(self,val):
        self.__timeInterval = val
```

TRAnalyzer.py:

```
# -*- coding: utf8 -*-
import statistics as st
import forecast as fr
import pandas as pd

class Analyzer(object):
    def __init__(self):
        self.__statComponent = st.Statistics()
        self.__forecastComponent = fr.Forecast()
             self.__data = None
    def makeAnalisis(self,prediction = 3,timeInterval = 'D'):
        try:
            self.__statComponent.analysis(self.__data)
            self.__forecastComponent.predictions = prediction
            self.__forecastComponent.timeInterval = timeInterval
            self.__forecastComponent.makeForecast(self.__data)
        except Exception as ex:
            print(ex)
            return False

        return True

    @property
    def statComponent(self):
        return self.__statComponent
    @property
    def forecastComponent(self):
        return self.__forecastComponent
    @property
    def data(self):
        return self.__data

    @data.setter
    def data(self,newData):
        self.__data = newData
```

```
        self.__data = self.__data.groupby('ds')[['y']].sum()
```

uiDesign.py:

```
# -*- coding: utf-8 -*-
from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(823, 501)
        MainWindow.setWindowTitle("TSA")
        self.centralWidget = QtWidgets.QWidget(MainWindow)
        self.centralWidget.setObjectName("centralWidget")
        self.graphBox = QtWidgets.QGroupBox(self.centralWidget)
        self.graphBox.setGeometry(QtCore.QRect(10, -10, 631, 371))
        self.graphBox.setTitle("")
        self.graphBox.setObjectName("graphBox")
        self.statBox = QtWidgets.QGroupBox(self.centralWidget)
        self.statBox.setGeometry(QtCore.QRect(10, 350, 801, 101))
        self.statBox.setTitle("")
        self.statBox.setObjectName("statBox")
        self.label = QtWidgets.QLabel(self.statBox)
        self.label.setGeometry(QtCore.QRect(10, 30, 57, 16))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(self.statBox)
        self.label_2.setGeometry(QtCore.QRect(130, 30, 57, 15))
        self.label_2.setObjectName("label_2")
        self.label_5 = QtWidgets.QLabel(self.statBox)
        self.label_5.setGeometry(QtCore.QRect(270, 30, 57, 16))
        self.label_5.setObjectName("label_5")
        self.label_7 = QtWidgets.QLabel(self.statBox)
        self.label_7.setGeometry(QtCore.QRect(380, 30, 57, 16))
        self.label_7.setObjectName("label_7")
        self.label_9 = QtWidgets.QLabel(self.statBox)
        self.label_9.setGeometry(QtCore.QRect(490, 30, 57, 16))
        self.label_9.setObjectName("label_9")
        self.label_3 = QtWidgets.QLabel(self.statBox)
        self.label_3.setGeometry(QtCore.QRect(610, 30, 57, 16))
        self.label_3.setObjectName("label_3")
        self.label_4 = QtWidgets.QLabel(self.statBox)
        self.label_4.setGeometry(QtCore.QRect(10, 60, 57, 16))
        self.label_4.setObjectName("label_4")
        self.label_6 = QtWidgets.QLabel(self.statBox)
```

```
self.label_6.setGeometry(QtCore.QRect(130, 60, 57, 16))
self.label_6.setObjectName("label_6")
self.label_8 = QtWidgets.QLabel(self.statBox)
self.label_8.setGeometry(QtCore.QRect(250, 60, 57, 16))
self.label_8.setObjectName("label_8")
self.label_12 = QtWidgets.QLabel(self.statBox)
self.label_12.setGeometry(QtCore.QRect(370, 60, 57, 15))
self.label_12.setObjectName("label_12")
self.label_13 = QtWidgets.QLabel(self.statBox)
self.label_13.setGeometry(QtCore.QRect(500, 60, 57, 15))
self.label_13.setObjectName("label_13")
self.meanVal = QtWidgets.QLineEdit(self.statBox)
self.meanVal.setEnabled(True)
self.meanVal.setGeometry(QtCore.QRect(60, 30, 51, 23))
self.meanVal.setReadOnly(True)
self.meanVal.setObjectName("meanVal")
self.medVal = QtWidgets.QLineEdit(self.statBox)
self.medVal.setEnabled(True)
self.medVal.setGeometry(QtCore.QRect(200, 30, 61, 23))
self.medVal.setReadOnly(True)
self.medVal.setObjectName("medVal")
self.varVal = QtWidgets.QLineEdit(self.statBox)
self.varVal.setEnabled(True)
self.varVal.setGeometry(QtCore.QRect(310, 30, 61, 23))
self.varVal.setReadOnly(True)
self.varVal.setObjectName("varVal")
self.stdVal = QtWidgets.QLineEdit(self.statBox)
self.stdVal.setEnabled(True)
self.stdVal.setGeometry(QtCore.QRect(420, 30, 61, 23))
self.stdVal.setReadOnly(True)
self.stdVal.setObjectName("stdVal")
self.madVal = QtWidgets.QLineEdit(self.statBox)
self.madVal.setEnabled(True)
self.madVal.setGeometry(QtCore.QRect(530, 30, 71, 23))
self.madVal.setReadOnly(True)
self.madVal.setObjectName("madVal")
self.minVal = QtWidgets.QLineEdit(self.statBox)
self.minVal.setEnabled(True)
self.minVal.setGeometry(QtCore.QRect(650, 30, 81, 23))
self.minVal.setReadOnly(True)
self.minVal.setObjectName("minVal")
self.maxVal = QtWidgets.QLineEdit(self.statBox)
self.maxVal.setEnabled(True)
```

```
self.maxVal.setGeometry(QtCore.QRect(60, 60, 61, 23))
self.maxVal.setReadOnly(True)
self.maxVal.setObjectName("maxVal")
self.corrVal = QtWidgets.QLineEdit(self.statBox)
self.corrVal.setEnabled(True)
self.corrVal.setGeometry(QtCore.QRect(170, 60, 71, 23))
self.corrVal.setReadOnly(True)
self.corrVal.setObjectName("corrVal")
self.covrVal = QtWidgets.QLineEdit(self.statBox)
self.covrVal.setEnabled(True)
self.covrVal.setGeometry(QtCore.QRect(300, 60, 61, 23))
self.covrVal.setReadOnly(True)
self.covrVal.setObjectName("covrVal")
self.mseVal = QtWidgets.QLineEdit(self.statBox)
self.mseVal.setEnabled(True)
self.mseVal.setGeometry(QtCore.QRect(410, 50, 81, 23))
self.mseVal.setReadOnly(True)
self.mseVal.setObjectName("mseVal")
self.mapeVal = QtWidgets.QLineEdit(self.statBox)
self.mapeVal.setEnabled(True)
self.mapeVal.setGeometry(QtCore.QRect(560, 60, 71, 23))
self.mapeVal.setReadOnly(True)
self.mapeVal.setObjectName("mapeVal")
self.setBox = QtWidgets.QGroupBox(self.centralWidget)
self.setBox.setGeometry(QtCore.QRect(650, -10, 161, 341))
self.setBox.setTitle("")
self.setBox.setObjectName("setBox")
self.nameEdit = QtWidgets.QLineEdit(self.setBox)
self.nameEdit.setEnabled(True)
self.nameEdit.setGeometry(QtCore.QRect(10, 50, 113, 23))
self.nameEdit.setObjectName("nameEdit")
self.toolButton = QtWidgets.QToolButton(self.setBox)
self.toolButton.setGeometry(QtCore.QRect(130, 50, 35, 22))
self.toolButton.setObjectName("toolButton")
self.checkCorr = QtWidgets.QCheckBox(self.setBox)
self.checkCorr.setEnabled(False)
self.checkCorr.setGeometry(QtCore.QRect(10, 220, 111, 21))
self.checkCorr.setObjectName("checkCorr")
self.checkCovr = QtWidgets.QCheckBox(self.setBox)
self.checkCovr.setEnabled(False)
self.checkCovr.setGeometry(QtCore.QRect(10, 250, 111, 21))
self.checkCovr.setObjectName("checkCovr")
self.label_10 = QtWidgets.QLabel(self.setBox)
```

```
        self.label_10.setGeometry(QtCore.QRect(10, 80, 111, 16))
        self.label_10.setObjectName("label_10")
        self.rowList = QtWidgets.QComboBox(self.setBox)
        self.rowList.setEnabled(False)
        self.rowList.setGeometry(QtCore.QRect(10, 110, 79, 23))
        self.rowList.setObjectName("rowList")
        self.rowList_2 = QtWidgets.QComboBox(self.setBox)
        self.rowList_2.setEnabled(False)
        self.rowList_2.setGeometry(QtCore.QRect(10, 270, 79, 23))
        self.rowList_2.setObjectName("rowList_2")
        self.performButton = QtWidgets.QPushButton(self.setBox)
        self.performButton.setEnabled(False)
        self.performButton.setGeometry(QtCore.QRect(10, 300, 80,23))
        self.performButton.setObjectName("performButton")
        self.timeIntervals = QtWidgets.QComboBox(self.setBox)
        self.timeIntervals.setGeometry(QtCore.QRect(10, 190, 79,23))
        self.timeIntervals.setObjectName("timeIntervals")
        self.intervalSpin = QtWidgets.QSpinBox(self.setBox)
        self.intervalSpin.setGeometry(QtCore.QRect(100, 190, 47,24))
        self.intervalSpin.setMinimum(0)
        self.intervalSpin.setMaximum(999)
        self.intervalSpin.setObjectName("intervalSpin")
        self.label_11 = QtWidgets.QLabel(self.setBox)
        self.label_11.setGeometry(QtCore.QRect(10, 160, 59, 15))
        self.label_11.setObjectName("label_11")
        MainWindow.setCentralWidget(self.centralWidget)
        self.menuBar = QtWidgets.QMenuBar(MainWindow)
        self.menuBar.setGeometry(QtCore.QRect(0, 0, 823, 20))
        self.menuBar.setObjectName("menuBar")
        self.menu = QtWidgets.QMenu(self.menuBar)
        self.menu.setObjectName("menu")
        MainWindow.setMenuBar(self.menuBar)
        self.actionexport_to_png = QtWidgets.QAction(MainWindow)
        self.actionexport_to_png.setObjectName("actionimport_to_png")
        self.actionexport_to_csv = QtWidgets.QAction(MainWindow)
        self.actionexport_to_csv.setObjectName(
"actionimport_to_csv")
        self.actionexit = QtWidgets.QAction(MainWindow)
        self.actionexit.setObjectName("actionexit")
        self.actionabout = QtWidgets.QAction(MainWindow)
        self.actionabout.setObjectName("actionabout")
        self.menu.addAction(self.actionexport_to_png)
        self.menu.addAction(self.actionexport_to_csv)
```

```
        self.menu.addAction(self.actionabout)
        self.menu.addAction(self.actionexit)
        self.menuBar.addAction(self.menu.menuAction())
        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        self.label.setText(_translate("MainWindow", "mean ="))
        self.label_2.setText(_translate("MainWindow", "median ="))
        self.label_5.setText(_translate("MainWindow", "var ="))
        self.label_7.setText(_translate("MainWindow", "std ="))
        self.label_9.setText(_translate("MainWindow", "mad ="))
        self.label_3.setText(_translate("MainWindow", "min ="))
        self.label_4.setText(_translate("MainWindow", "max ="))
        self.label_6.setText(_translate("MainWindow", "corr ="))
        self.label_8.setText(_translate("MainWindow", "covr ="))
        self.label_12.setText(_translate("MainWindow", "mse ="))
        self.label_13.setText(_translate("MainWindow", "mape ="))
        self.toolButton.setText(_translate("MainWindow", "..."))
        self.checkCorr.setText(_translate("MainWindow", "Corr"))
        self.checkCovr.setText(_translate("MainWindow", "Covr"))
        self.label_10.setText(_translate("MainWindow", "time
series"))
        self.performButton.setText(_translate("MainWindow",
"Execute"))
        self.label_11.setText(_translate("MainWindow", "Intervals"))
        self.menu.setTitle(_translate("MainWindow", "Menu"))
        self.actionexport_to_png.setText(_translate("MainWindow",
"export to .png"))
        self.actionexport_to_csv.setText(_translate("MainWindow",
"export to .csv"))
        self.actionexit.setText(_translate("MainWindow", "exit"))
        self.actionabout.setText(_translate("MainWindow", "about"))
```

appClass.py:

```
# -*- coding: utf8 -*-
import sys
from PyQt5 import QtWidgets, QtCore, QtGui
from PyQt5.QtWidgets import QMessageBox
import uiDesign as uids
import pandas as pd
import TRAnalyzer as tra
```

```
import plot
import matplotlib.pyplot as plt
from matplotlib.figure import Figure

#Пользовательский интерфейс
class AppClass(QtWidgets.QMainWindow, uids.Ui_MainWindow):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.__blocksOffset = 5
        self.__fileName = None
        self.__Analyzer = tra.Analyzer()
        self.__defaultData = None
        self.__forecastPlot = plot.PlotCanvas(parent = self.graphBox)
        self.__trendPlot = plot.PlotCanvas(parent = self.graphBox)

        self.toolButton.clicked.connect(self.toolButton_OnClick)
        self.performButton.clicked.connect(
self.performButton_OnClick)
        self.checkCorr.stateChanged.connect(
self.checkCorr_StateChanged)
        self.checkCovr.stateChanged.connect(
self.checkCovr_StateChanged)
        self.actionexport_to_csv.triggered.connect(
self.actionExportToCsv_triggered)
        self.actionexport_to_png.triggered.connect(
self.actionExportToPng_triggered)
        self.actionabout.triggered.connect(
self.actionAbout_triggered)
        self.actionexit.triggered.connect(
self.actionExit_triggered)

        self.setBox.setStyleSheet("QGroupBox{padding-top:18px; mar-
gin-top:-18px}")
        self.graphBox.setStyleSheet("QGroupBox{padding-top:18px; mar-
gin-top:-18px}")
        self.statBox.setStyleSheet("QGroupBox{padding-top:18px; mar-
gin-top:-18px}")

self.centralWidget.setMinimumSize(720,540)
self.timeIntervals.addItems(['Day','Week','Month','Year'])

        def actionExit_triggered(self):
```

```
            QtCore.QCoreApplication.quit()

        def actionAbout_triggered(self):
            aboutBox = QMessageBox.about(self,'About',
'Time series analysis and forecasting\nAutor: Rasaev Dmitriy')

        def actionExportToPng_triggered(self):
            fileName =
QtWidgets.QFileDialog.getExistingDirectory(self,"Import to
png",QtCore.QDir.homePath())#получаем директорию

            if fileName != '':
                try:
                    data = {'yhat' :
self.__Analyzer.forecastComponent.forecast['yhat'],
'trend' : self.__Analyzer.forecastComponent.forecast['trend']}
                    row = pd.DataFrame(data)
                    plt.plot(data['yhat'])
                    plt.savefig(''.join([fileName,'/forecast.png']),
dpi = 100)
                    plt.clf()

                    plt.plot(data['trend'])
                    plt.savefig(''.join([fileName,'/trend.png']),
dpi = 100)
                    plt.clf()

                    plt.plot(self.__Analyzer.data['y'])
plt.savefig(''.join([fileName, '/defaultRow.png']),dpi = 100)
                    plt.clf()
                except Exception as ex:
                    print(ex)
                    errorBox = QMessageBox.critical(self,'Error',
'Bad parametres')

                    return

        def actionExportToCsv_triggered(self):
            fileName =
QtWidgets.QFileDialog.getSaveFileName(self,"Import to
csv",QtCore.QDir.homePath(),'Text files (*.csv)')[0]

            if fileName != '':
```

```
                try:
                    data = {'ds' :
self.__Analyzer.forecastComponent.forecast['ds'],
'yhat' : self.__Analyzer.forecastComponent.forecast['yhat'],
'trend' : self.__Analyzer.forecastComponent.forecast['trend']}
                except Exception as ex:
                    errorBox = QMessageBox.critical(self,'Error','Bad
parametres')
                    return

                if fileName.find('.csv') == -1:
                    fileName = ''.join([fileName,'.csv'])
                saveDataFrame = pd.DataFrame(data)
                saveDataFrame.to_csv(fileName,index = False)

        def checkCorr_StateChanged(self):
            self.rowList_2.setEnabled(True)

        def checkCovr_StateChanged(self):
            self.rowList_2.setEnabled(True)
        def performButton_OnClick(self):
            try:
                data = {'ds' : self.__defaultData['ds'],
'y' : self.__defaultData[self.rowList.currentText()]}
                data['ds'] = pd.to_datetime(data['ds'])
            except Exception as ex:
                errorBox = QMessageBox.critical(self,
'Error',' '.join(['File has not series',str(ex)]))
                return

            newRow = pd.DataFrame(data)
            self.__Analyzer.data = newRow
            self.__Analyzer.makeAnalisis(prediction =
self.intervalSpin.value(),timeInterval =
self.timeIntervals.currentText()[0])
            self.meanVal.setText(str(self.__Analyzer.
statComponent.mean))
            self.medVal.setText(str(self.__Analyzer.
statComponent.median))
            self.varVal.setText(str(self.__Analyzer.
statComponent.var))
            self.stdVal.setText(str(self.__Analyzer.
statComponent.std))
```

```
            self.madVal.setText(str(self.__Analyzer.
statComponent.mad))
            self.minVal.setText(str(self.__Analyzer.
statComponent.min))
            self.maxVal.setText(str(self.__Analyzer.
statComponent.max))
            if self.checkCorr.isChecked() == True:
                try:
                    data = {'ds' : self.__defaultData['ds'],
'y' : self.__defaultData[self.rowList_2.currentText()]}
                    data['ds'] = pd.to_datetime(data['ds'])
                except Exception as ex:
                    errorBox = QMessageBox.critical(self,
'Error',str(ex))
                    return
                lastRow = pd.DataFrame(data)
                self.corrVal.setText(str(self.__Analyzer.
statComponent.pearsCorr(self.__Analyzer.data, lastRow)))
            else:
                self.corrVal.setText(str(None))

            if self.checkCovr.isChecked() == True:
                try:
                    data = {'ds' : self.__defaultData['ds'],
'y' : self.__defaultData[self.rowList_2.currentText()]}
                    data['ds'] = pd.to_datetime(data['ds'])
                except Exception as ex:
                    errorBox = QMessageBox.critical(self,
'Error',str(ex))
                    return

                lastRow = pd.DataFrame(data)
                self.covrVal.setText(str(self.
__Analyzer.statComponent. pearsCorr(self.__Analyzer.data, lastRow)))
            else:
                self.covrVal.setText(str(None))
            self.__forecastPlot.clear()
            self.__forecastPlot.plot(self.__Analyzer.data[['y']],
linewidth = 0.9,label = 'y',color = 'xkcd:kelly green')
            self.__forecastPlot.plot(self.__Analyzer. 0.9,label =
forecastComponent.forecast[['ds','yhat']].set_index('ds'),
linewidth = 'yhat', color = 'xkcd:tomato')
            self.__forecastPlot.plot(self.__Analyzer.
```

```
forecastComponent.forecast[['ds','yhat_lower']].set_index('ds'),
color = 'xkcd:bright blue',linewidth = 0.9,label = 'yhat_lower')
            self.__forecastPlot.plot(self.__Analyzer.
forecastComponent.forecast[['ds','yhat_upper']].set_index('ds'),
color = 'xkcd:bright blue',linewidth = 0.9,
label = 'yhat_upper',legend = True)

            self.__trendPlot.clear()
            self.__trendPlot.plot(self.__Analyzer.
forecastComponent.forecast[['ds','trend']].set_index('ds'),linewidth
= 1.5,label = 'trend',legend = True,
color = 'xkcd:tomato')
            self.mseVal.setText(str(self.
          __Analyzer.forecastComponent.
mse))
            self.mapeVal.setText(str(self.__Analyzer.
forecastComponent.mape))
        def toolButton_OnClick(self):
            file =
QtWidgets.QFileDialog.getOpenFileName(self,'Open',QtCore.QDir.homePa
th(),'Text files (*.csv)')[0]
            if file != '':
                self.nameEdit.setText(file)
                self.__fileName = file
                try:
                    self.__defaultData = pd.read_csv(self.__fileName)
                except Exception as ex:
                    errorBox = QMessageBox.critical(self,
'Error',str(ex))
                   return

                self.rowList.setEnabled(True)
                self.performButton.setEnabled(True)
                self.checkCorr.setEnabled(True)
                self.checkCovr.setEnabled(True)
                self.performButton.setEnabled(True)

                self.rowList.clear()
                self.rowList_2.clear()

                for i in (self.__defaultData.columns):
                    if i != 'ds':
                        self.rowList.addItem(i)
```

```
                        self.rowList_2.addItem(i)

                print(file)

        def resizeEvent(self, event):
            self.graphBox.setGeometry(self.__blocksOffset,
self.__blocksOffset,self.centralWidget.width()/5*4,
self.centralWidget.height()/6*5)
            self.__forecastPlot.setGeometry(0,0,
self.graphBox.width(),self.graphBox.height()/2)
            self.__trendPlot.setGeometry(0,
self.graphBox.height()/2,self.graphBox.width(),
self.graphBox.height()/2)
            self.setBox.setGeometry(self.graphBox.x() +
self.graphBox.width() + self.__blocksOffset,
self.__blocksOffset,self.centralWidget.width()/5 -
3*self.__blocksOffset,self.graphBox.height())
            self.statBox.setGeometry(self.graphBox.x(),
self.graphBox.y() + self.graphBox.height() + self.__blocksOffset,
self.centralWidget.width() - 2*self.__blocksOffset,
self.centralWidget.height() - self.graphBox.height() -
3*self.__blocksOffset)
            self.nameEdit.setGeometry(self.__blocksOffset,
self.__blocksOffset,self.setBox.width() - 3*self.__blocksOffset -
self.toolButton.width(),self.nameEdit.height())
            self.toolButton.setGeometry(self.nameEdit.width() +
2*self.__blocksOffset,self.nameEdit.y(),self.toolButton.width(),
self.toolButton.height())
            self.label_10.setGeometry(self.__blocksOffset,
self.nameEdit.y() + 2*self.__blocksOffset + self.nameEdit.height(),
self.label_10.width(),self.label_10.height())
            self.rowList.setGeometry(self.__blocksOffset,
self.label_10.y() + self.__blocksOffset + self.label_10.height(),
self.nameEdit.width(),self.rowList.height())
            self.label_11.setGeometry(self.__blocksOffset,
self.rowList.y() + 3*self.__blocksOffset + self.rowList.height(),
self.label_11.width(),self.label_11.height())
            self.timeIntervals.setGeometry(self.__blocksOffset,
self.label_11.y() + self.__blocksOffset + self.label_11.height(),
self.nameEdit.width(),self.timeIntervals.height())
            self.intervalSpin.setGeometry(self.toolButton.x(),
self.timeIntervals.y(),self.toolButton.width(),
self.timeIntervals.height())
```

```
            self.checkCorr.setGeometry(self.__blocksOffset,
self.timeIntervals.y() + 3*self.__blocksOffset +
self.timeIntervals.height(),
self.checkCorr.width(),self.checkCorr.height())
            self.checkCovr.setGeometry(self.__blocksOffset,
self.checkCorr.y() + self.__blocksOffset + self.checkCorr.height(),
self.checkCovr.width(),self.checkCovr.height())
self.rowList_2.setGeometry(self.__blocksOffset,
self.checkCovr.y() + self.__blocksOffset + self.checkCovr.height(),
self.rowList.width(),self.rowList_2.height())
            self.performButton.setGeometry(self.__blocksOffset,
self.rowList_2.height() + self.rowList_2.y() +
3*self.__blocksOffset,self.setBox.width() -
2*self.__blocksOffset,self.performButton.height())
            self.label.setGeometry(self.__blocksOffset,
self.__blocksOffset*2,self.label.width(),self.label.height())
self.meanVal.setGeometry(self.label.x() + self.label.width() +
self.__blocksOffset,self.label.y(),
self.statBox.width()/13,self.meanVal.height())
            self.label_2.setGeometry(self.meanVal.x() +
self.meanVal.width() + self.__blocksOffset,self.meanVal.y(),
self.label_2.width(),self.label_2.height())
            self.medVal.setGeometry(self.label_2.x() +
self.label_2.width() + self.__blocksOffset,self.label_2.y(),
self.statBox.width()/13,self.medVal.height())
            self.label_5.setGeometry(self.medVal.x() +
self.medVal.width() + self.__blocksOffset,self.medVal.y(),
self.label_5.width(),self.label_5.height())
            self.varVal.setGeometry(self.label_5.x() +
self.label_5.width() + self.__blocksOffset,self.label_5.y(),
self.statBox.width()/13,self.varVal.height())
            self.label_7.setGeometry(self.varVal.x() +
self.varVal.width() + self.__blocksOffset,self.varVal.y(),
self.label_7.width(),self.label_7.height())
            self.stdVal.setGeometry(self.label_7.x() +
self.label_7.width() + self.__blocksOffset,self.label_7.y(),
self.statBox.width()/13,self.stdVal.height())
            self.label_9.setGeometry(self.stdVal.x() +
self.stdVal.width() + self.__blocksOffset,self.stdVal.y(),
self.label_9.width(),self.label_9.height())
            self.madVal.setGeometry(self.label_9.x() +
self.label_9.width() + self.__blocksOffset,self.label_9.y(),
self.statBox.width()/13,self.madVal.height())
```

```
            self.label_3.setGeometry(self.madVal.x() +
self.madVal.width() + self.__blocksOffset,self.madVal.y(),
self.label_3.width(),self.label_3.height())
            self.minVal.setGeometry(self.label_3.x() +
self.label_3.width() + self.__blocksOffset,self.label_3.y(),
self.statBox.width()/13,self.minVal.height())
            self.label_4.setGeometry(self.__blocksOffset,
self.label.y() + self.label.height() + self.__blocksOffset*3,
self.label_4.width(),self.label_4.height())
            self.maxVal.setGeometry(self.label_4.x() +
self.label_4.width() + self.__blocksOffset,self.label_4.y(),
self.statBox.width()/13,self.mapeVal.height())
            self.label_6.setGeometry(self.maxVal.x() +
self.maxVal.width() + self.__blocksOffset,self.maxVal.y(),
self.label_6.width(),self.label_6.height())
            self.corrVal.setGeometry(self.label_6.x() +
self.label_6.width() + self.__blocksOffset,self.label_6.y(),
self.statBox.width()/13,self.corrVal.height())
            self.label_8.setGeometry(self.corrVal.x() +
self.corrVal.width() + self.__blocksOffset,self.corrVal.y(),
self.label_8.width(),self.label_8.height())
            self.covrVal.setGeometry(self.label_8.x() +
self.label_8.width() + self.__blocksOffset,self.label_8.y(),
self.statBox.width()/13,self.corrVal.height())
            self.label_12.setGeometry(self.covrVal.x() +
self.covrVal.width() + self.__blocksOffset,self.covrVal.y(),
self.label_12.width(),self.label_12.height())
            self.mseVal.setGeometry(self.label_12.x() +
self.label_12.width() + self.__blocksOffset,self.label_12.y(),
self.statBox.width()/13,self.mseVal.height())
            self.label_13.setGeometry(self.mseVal.x() +
self.mseVal.width() + self.__blocksOffset,self.mseVal.y(),
self.label_13.width(),self.label_13.height())
            self.mapeVal.setGeometry(self.label_13.x() +
self.label_13.width() + self.__blocksOffset,self.label_13.y(),
self.statBox.width()/13,self.mapeVal.height())
```

plot.py:

```
# -*- coding: utf8 -*-
import sys
import matplotlib
from PyQt5 import QtCore
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as
FigureCanvas
```

```
from matplotlib.figure import Figure
import datetime as dt
from matplotlib import dates
import pandas as pd

class PlotCanvas(FigureCanvas):
    def __init__(self,parent = None,width = 4,height = 4,dpi = 100):
        self.__figure = Figure(figsize = (width, height),dpi = dpi)
        self.__graph = self.__figure.add_subplot(111)
        self.__graph.grid(which='major',color = 'k')
        self.__graph.grid(which='minor',color = 'gray',
linestyle = ':')
        self.__graph.minorticks_on()
        self.__graph.set_title('')

        FigureCanvas.__init__(self,self.__figure)

        self.setParent(parent)
        self.move(0,0)

    def plot(self,data = None,title = '',color = 'r',
linewidth = 1,label = '',legend = False):
        self.__graph = self.__figure.add_subplot(111)

        try:
            self.__graph.plot(data, color = color,
linewidth = linewidth,label = label)
        except Exception as ex:
            print(ex)
            return

        self.__graph.set_title(title)
        self.__graph.get_xaxis().set_major_formatter(
dates.DateFormatter('%Y-%m-%d'))

        self.__graph.grid(which='major',color = 'k',linewidth = 0.5)
        self.__graph.grid(which='minor',color = 'gray',
linestyle = ':')
        self.__graph.minorticks_on()

        if legend == True:
            self.__figure.legend()

        self.draw()

    def clear(self):
        self.__figure.clear()
```

main.py:

```
# -*- coding: utf8 -*-
import sys
import os
from PyQt5 import QtWidgets
import appClass

def main():
    app = QtWidgets.QApplication(sys.argv)
    window = appClass.AppClass()
    window.show()
    app.exec_()
if __name__ == '__main__':
    main()
```