МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра Прикладная математика и информатика
(наименование)

01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки / специальности)

Компьютерные технологии и математическое моделирование
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Программная реализация алгоритма линейной регрессии»

Обучающийся М.С. Горбачев _____
(Инициалы Фамилия) (личная подпись)

Руководитель канд. пед. наук, доцент, Т. А. Агошкова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант канд. пед. наук, доцент, С.А. Гудкова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Тема бакалаврской работы: Программная реализация алгоритма линейной регрессии.

Бакалаврская работа посвящена разработке программного обеспечения, позволяющее визуализировать статистику простоев оборудования и эффективности его обслуживания, а также проводить прогнозирование и анализ с использованием алгоритма линейной регрессии.

В ходе выполнения исследований по бакалаврской работе была сформулирована и поставлена задача, проведен сравнительный анализ и выбор подходящего языка программирования и интерфейса ПО, описан алгоритм линейной регрессии и построена его математическая модель, а также разработано и протестировано ПО для поставленной задачи.

Во введении описана актуальность темы, написаны цель и задачи.

В первом разделе произведен сравнительный анализ языков программирования и графических интерфейсов, а также описано практическое применение алгоритма.

Во втором разделе описана математическая модель алгоритма линейной регрессии и метода наименьших квадратов. Также определена и описана функциональность программы и процесс ее разработки.

В третьем разделе описана реализация ПО, произведено тестирование и анализ полученных результатов.

В заключении представлены результаты выполнения бакалаврской работы.

Бакалаврская работа состоит из введения, трёх разделов, заключения и списка использованной литературы.

Бакалаврская работа состоит из 48 страниц, 16 рисунков, 2 таблиц, 27 источников и 1 листинга.

Abstract

The topic of the bachelor's thesis is the software implementation of the linear regression algorithm.

The bachelor's thesis is devoted to the development of software that allows you to visualize the statistics of equipment downtime and maintenance efficiency, as well as to carry out forecasting and analysis using a linear regression algorithm.

During the research on the bachelor's thesis, a task was formulated and set, a comparative analysis and selection of a suitable programming language and software interface were carried out, a linear regression algorithm was described and its mathematical model was built, and software for the task was developed and tested.

The introduction describes the relevance of the topic, the purpose and objectives are written.

The first section provides a comparative analysis of programming languages and graphical interfaces, as well as describes the practical application of the algorithm.

The second section describes a mathematical model of the linear regression algorithm and the least squares method. The architecture of the system and the system for development are also designed.

The third section describes the software implementation, testing and analysis of the results obtained.

In conclusion, the results of the bachelor's work are presented.

The bachelor's thesis consists of an introduction, three sections, a conclusion and a list of references.

The bachelor's thesis consists of 48 pages, 16 figures, 2 tables, 27 sources and 1 listing.

Содержание

Введение

В современном мире, где данные становятся все более важным ресурсом, прогнозирование и анализ данных становятся ключевыми инструментами для принятия решений и оптимизации процессов. В этом контексте линейная регрессия является одним из самых распространенных и эффективных методов прогнозирования и анализа, который используется во многих отраслях, включая промышленность, финансы, здравоохранение и многие другие.

АвтоВАЗ, один из крупнейших автомобильных производителей в России, не является исключением. В процессе производства автомобилей, оборудование и машины являются основными активами предприятия, и их эффективное функционирование является ключевым фактором успеха. Однако, как и в любом другом производственном процессе, оборудование может выходить из строя, что приводит к потере времени и затраты на устранение неполадок. Поэтому, важно иметь возможность прогнозировать и анализировать время устранения неисправностей, чтобы повысить эффективность процессов и снизить затраты.

Объектом исследования бакалаврской работы является изучение и анализ основных методов реализации алгоритма линейной регрессии.

Предметом исследования бакалаврской работы является алгоритм построения модели линейной регрессии на основе файла с информацией о выявленных моментах неисправности оборудования на предприятии, программные решения и библиотеки выбранного языка программирования.

Цель работы - исследование и программная реализация алгоритма линейной регрессии с целью создания простого в использовании и эффективного инструмента для анализа данных, выявления зависимостей между переменными и прогнозирования значений целевой переменной на основе имеющихся данных.

Задачи выпускной квалификационной работы включают в себя:

– выбор наиболее подходящего языка программирования и интерфейса для разработки программного обеспечения;
– изучение основных принципов работы алгоритма линейной регрессии;
– анализ существующих методов реализации алгоритма линейной регрессии;
– разработка программного кода для реализации алгоритма линейной регрессии;
– проведение тестирования программной реализации на различных наборах данных;
– выводы и рекомендации по применению разработанного алгоритма.

Данная работа включает в себя введение, три главы, заключение и список литературы.

В первом разделе происходит постановка задачи и выбор языка программирования, на котором будет реализован программный код работы. Также производится сравнительный анализ графических интерфейсов и описывается практическое применения алгоритма линейной регрессии.

Во втором разделе описываются алгоритм линейной регрессии и происходит выбор метода линейной регрессии. Также описывается функциональность программы и процесс ее разработки.

В третьем разделе описана реализация ПО и его тестирование.

1 Постановка задачи, выбор языка программирования и программного интерфейса

1.1 Постановка задачи и выбор наиболее подходящего языка программирования

Задачей данной бакалаврской работы является применение алгоритма линейной регрессии для визуализации зависимости времени устранения неполадки оборудования от его коэффициента износа на предприятии АвтоВАЗ с течением времени.

Целью работы является создание программного обеспечения, дающее возможность проводить прогнозирование и анализ с использованием алгоритма линейной регрессии. Это необходимо для того, чтобы сотрудники АвтоВАЗа могли наглядно видеть, как изменилось время устранения неисправности оборудования в указанный в отчёте период, например с 2018 года по 2020 год. Программа будет в дальнейшем использоваться на предприятии для выявления эффективности исправления неполадок и выявления КПД сотрудников.

Линейная регрессия используется для прогнозирования значений, которые принимают непрерывные значения (то есть любые числа из определенного диапазона). Прогнозируемое значение называется целевой переменной (или зависимой), а числовые данные, используемые для прогнозирования целевой переменной, называются объясняющей переменной (или независимой).

Линейную регрессию можно реализовать на многих языках программирования, однако необходимо определить какой язык программирования будет наиболее подходящим для решения этой задачи. Это необходимо для того, чтобы узнать на каком языке программирования можно написать более корректный код, ввиду наличия встроенных функций и библиотек.

Для этого проведем сравнительный анализ языков программирования. Наиболее распространенными языками программирования для решения поставленной задачи являются Python и R, но также могут быть применимы языки, такие как Java, C++, и MATLAB.

Рассмотрим их сравнительный анализ (таблица 1).

Таблица 1– Сравнительный анализ языков программирования для решения задачи с помощью алгоритма линейной регрессии

Язык	Преимущества языков	Недостатки языков
Python	Широкое распространение и популярность в области машинного обучения и анализа данных, ввиду наличия множества различных библиотек для решения задач данной области; Простой синтаксис и удобство использования.	Иногда может быть медленнее по сравнению с компилируемыми языками, такими как C++; Не всегда подходит для высокопроизводительных вычислений из-за интерпретируемости.
R	Мощный инструмент для статистического анализа и визуализации данных. Богатый выбор пакетов и библиотек для анализа данных, включая пакеты для линейной регрессии, такие как stats и lmtest.	Менее удобен для обработки больших объемов данных по сравнению с Python. Имеет менее активное сообщество разработчиков и меньшее количество библиотек для машинного обучения.
Java	Высокая производительность и эффективность в обработке больших объемов данных. Подходит для разработки крупных и высоконагруженных приложений.	Больше сложностей в использовании и написании кода в сравнении с Python и R. Меньшее количество специализированных библиотек и инструментов для анализа данных и машинного обучения.
C++	Высокая производительность и эффективность в использовании памяти. Подходит для разработки быстрых и производительных вычислительных приложений.	Сложный синтаксис и более высокий порог вхождения для начинающих разработчиков. Отсутствие широкого выбора библиотек и инструментов для анализа данных по сравнению с Python и R.
MATLAB	Простота в использовании и понимании. Большое количество интегрированных функций для работы с матрицами и числовыми данными.	Платная лицензия, что делает его менее доступным для широкого круга пользователей. Меньшее количество возможностей для статистического анализа

На основе сравнительного анализа языков программирования можно сделать вывод, что наиболее подходящим можно считать Python. В нём существует множество встроенных функций и библиотек, которые работают с алгоритмом линейной регрессии, например numpy, pandas, scikit-learn и statsmodels [16][25]. Для написания и отладки кода мною будет использована платформа PyCharm, которая работает на языке программирования Python.

1.2 Выбор подходящего интерфейса ПО

Для программной реализации алгоритма линейной регрессии необходимо выбрать подходящий интерфейс, в котором будет осуществляться работа. Это необходимо в виду того, что в организации, в которой я прохожу преддипломную практику, в дальнейшем будут использовать разработанное мною программное обеспечение. В виду этого произведём сравнительный анализ графических интерфейсов, которые работают на языке программирования Python (таблица 2).

Таблица 2 – Сравнительный анализ графических интерфейсов на Python

Интерфейс	Преимущества интерфейса	Недостатки интерфейса
Tkinter	Встроенный в стандартную библиотеку Python; Простой в изучении и использовании; Поддерживает создание различных виджетов; Работает на различных платформах без необходимости установки дополнительных пакетов.	Ограниченный набор виджетов и возможностей стилизации по сравнению с некоторыми другими библиотеками; Требуется дополнительная работа для создания сложных и красочных интерфейсов.
PyQt PySide	Полнофункциональный набор инструментов для создания GUI; Поддерживает событийно-ориентированное программирование; Имеют активное сообщество пользователей;	Не является стандартной библиотекой Python и требует установки дополнительных пакетов; Может быть более сложным для начинающих пользователей из-за большого количества функций и опций.

Продолжение таблицы 2

Интерфейс	Преимущества интерфейса	Недостатки интерфейса
wxPython	Предоставляет GUI-инструменты, основанные на библиотеке wxWidgets; Хорошо поддерживается и имеет широкий набор виджетов; Поддерживает стандарты, такие как AUI и MDI;	Требует установки сторонней библиотеки и некоторых знаний C++ для полного использования всех возможностей.
Kivy	Поддерживает создание интерфейсов как для мобильных приложений, так и для ПК; Имеет свой DSL для описания пользовательского интерфейса; Позволяет создавать приложения с красивым и современным дизайном.	Может быть сложным для новичков из-за специфического синтаксиса и подхода к созданию интерфейса.

На основе сравнительного анализа графических интерфейсов, которые работают на Python, можно сделать вывод, что наиболее подходящим для визуализации алгоритма линейной регрессии будет встроенный в стандартную библиотеку Python графический интерфейс tkinter. Он является простым для понимания пользователя и разработчика. Также он не требует дополнительных знаний в front-end разработке и установки дополнительных пакетов расширений.

1.3 Практическое применение алгоритма линейной регрессии для решения поставленной задачи

Итоговым результатом моей работы станет программное обеспечение для сотрудников АвтоВАЗа. Я буду использовать Pycharm, который работает на Python. В графический интерфейс приложения, созданный на основе библиотеки tkinter, будет загружаться файл в формате csv. В файле будет храниться информация о времени устранения неисправности в минутах и коэффициент износа оборудования для каждой зафиксированной неисправности. На основе этого файла будет строиться график линейной регрессии, на котором будет наглядно видно, как изменялось время

устранения неполадок за выбранный период. Благодаря этому приложению сотрудники предприятия смогут своевременно реагировать на задержки в производстве, выявлять закономерности в работоспособности и эффективности сотрудников для выбранного временного ряда.

Для расчёта коэффициента фактического износа оборудования будет применяться формула 1.

$$К_ф = \frac{Т_ф}{Т_{ПИ}} * 100\%, \quad (1)$$

где

$Т_{ПИ}$ – срок полезного использования оборудования;

$Т_ф$ – фактический срок использования оборудования.

Фактический срок использования оборудования рассчитывается по формуле, предоставленной от предприятия (формула 2).

$$Т_ф = D_в - D_э, \quad (2)$$

где

$D_в$ – дата возникновения неисправности;

$D_э$ – дата ввода в эксплуатацию оборудования.

На основе этих формул рассчитывается зависимая переменная в csv-файле для визуализации графика алгоритма линейной регрессии.

Выводы по разделу 1:

Была поставлена задача, включающая в себя создание программного обеспечения, которое будет визуализировать время устранения неполадок на АвтоВАЗе за выбранный период с применением алгоритма линейной регрессии. Оно поможет сотрудникам своевременно оценивать тенденции и эффективность устранения неисправности в заданный период времени.

2 Описание алгоритма линейной регрессии и её методов

2.1 Описание математической модели алгоритма линейной регрессии

Линейная регрессия – это статистический метод анализа данных, который используется для изучения отношений между двумя переменными. Он предполагает, что существует линейная зависимость между независимой переменной (X) и зависимой переменной (Y) [19][8]. Цель линейной регрессии состоит в том, чтобы построить модель, которая описывает эту зависимость и позволяет делать прогнозы или оценивать значения зависимой переменной на основе значений независимой переменной [23][10].

В случае, когда имеется только одна независимая переменная, линейную регрессию можно проиллюстрировать графически: зависимая переменная перемещается по оси Y, а независимая – по оси X. На рисунке 1 изображен пример диаграммы рассеяния с десятью наблюдаемыми значениями. Линия регрессии представляет собой наилучшее возможное соответствие между зависимой и независимой переменными (рисунок 1).

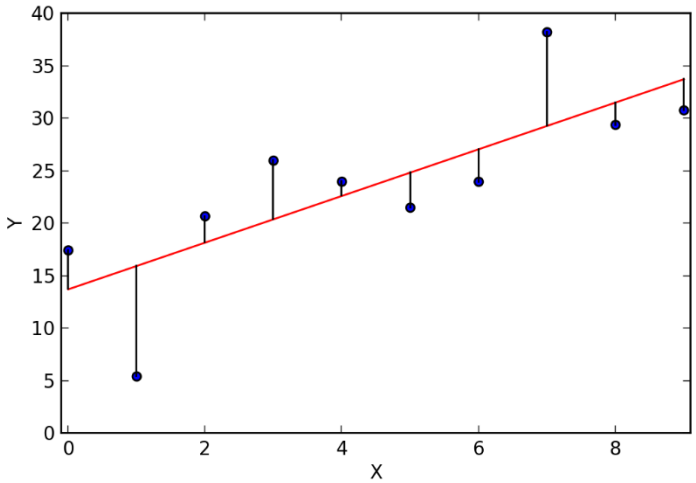

Рисунок 1 – Пример линейной регрессии

Модель линейной регрессии позволяет находить и оценивать связь между зависимой переменной и одной или несколькими независимыми переменными. Модель линейной регрессии стремится аппроксимировать зависимость между независимой переменной x и зависимой переменной y линейной функцией. Математически эту взаимосвязь можно выразить в виде уравнения (формула 3):

$$y_i = \beta_0 + \beta_1 * x_i + \varepsilon_i, \quad (3)$$

где

y_i – наблюдаемое значение зависимой переменной для i-го наблюдения;

x_i – значение независимой переменной для i-го наблюдения;

β_0 – коэффициент сдвига (интерсепт), который представляет собой значение зависимой переменной y при x=0;

β_1 – коэффициент наклона, который представляет собой изменение в y, вызванное изменением в x;

ε_i – случайная ошибка (остаток) для i-го наблюдения, которая представляет расхождение между наблюдаемым и предсказанным значением y.

Цель состоит в том, чтобы подобрать такие значения коэффициентов наклона и сдвига, которые минимизируют сумму квадратов остатков (расхождений между наблюдаемыми и предсказанными значениями).

Коэффициент β_0 называется свободным членом или константой регрессии. Он отображает точку пересечения с осью координат Y [22]. Это значение, которое принимает зависимая переменная, когда независимая равна нулю (формула 4):

$$y_i = \beta_0 + \beta_1 * 0 = \beta_0, \quad (4)$$

где

y_i – наблюдаемое значение зависимой переменной для i-го наблюдения;

β_0 – коэффициент сдвига (интерсепт), который представляет собой значение зависимой переменной y при x=0;

β_1 – коэффициент наклона, который представляет собой изменение в y, вызванное изменением в x.

Параметр β_1 является коэффициентом наклона и показывает во сколько раз значение будет увеличиваться по оси y при смещении на единицу по оси x. Если коэффициент положительный, то по мере увеличения независимой переменной будет увеличиваться также зависимая переменная y. В случае, если коэффициент отрицательный, это означает, что увеличение независимой переменной x приводит к уменьшению зависимой переменной y [9].

Для того, чтобы использовать алгоритм линейной регрессии для поставленной на предприятии задачи рассмотрим основные достоинства и недостатки этого алгоритма.

Основные достоинства алгоритма линейной регрессии.

Линейная регрессия является простым и легко интерпретируемым методом анализа данных. Ее интуитивно понятная природа позволяет исследователям и аналитикам быстро понять и объяснить результаты анализа. Коэффициенты регрессии, полученные в результате применения метода наименьших квадратов, могут быть непосредственно проанализированы для понимания вклада каждого признака в зависимую переменную. Это делает линейную регрессию чрезвычайно удобным инструментом для анализа данных и принятия решений на их основе.

Линейная регрессия идеально подходит для анализа и выявления тенденций во времени, что делает её особенно полезной в таких приложениях, как мониторинг изменений в процессе устранения неполадок с течением времени. Например, с помощью модели линейной регрессии можно определить, улучшается ли процесс устранения неисправностей на заводе по мере внедрения новых методов или технологий. Это может дать ценную

информацию о том, какие изменения были наиболее эффективными и в каких направлениях стоит двигаться дальше.

Одним из ключевых преимуществ линейной регрессии является её способность работать с большим количеством данных. В отличие от более сложных моделей, таких как нейронные сети, линейная регрессия имеет меньшую склонность к переобучению, особенно когда число наблюдений значительно превышает число признаков. Это делает её предпочтительным выбором для начального анализа данных и построения базовых моделей.

Коэффициенты линейной регрессии предоставляют четкую информацию о направлении и силе взаимосвязи между независимыми и зависимой переменными. Например, положительное значение коэффициента указывает на то, что с увеличением значения независимой переменной, значение зависимой переменной также увеличивается, и наоборот. Это позволяет исследователям не только прогнозировать будущие значения зависимой переменной, но и понимать природу и степень влияния различных факторов на интересующий их процесс.

Кроме того, линейная регрессия позволяет оценить влияние различных факторов на процесс устранения неисправностей оборудования. Путём анализа коэффициентов регрессии можно выявить ключевые аспекты, которые необходимо учитывать для повышения эффективности процесса. Например, если коэффициент, связанный с определённым признаком, является значительным и отрицательным, это может указывать на то, что данный фактор существенно замедляет процесс устранения неполадок, и его влияние следует минимизировать.

Линейная регрессия также обладает высокой вычислительной эффективностью. Процесс оценки модели с использованием метода наименьших квадратов может быть выполнен быстро даже для больших наборов данных. Это особенно важно в условиях реального времени или при необходимости регулярного обновления модели по мере поступления новых данных.

Линейная регрессия предоставляет возможность легко интерпретировать результаты анализа и использовать их для практических рекомендаций. Например, на основе модели можно дать конкретные советы по оптимизации процесса устранения неисправностей, определив, какие факторы следует изменить или улучшить.

Основные недостатки алгоритма линейной регрессии.

Линейная регрессия предполагает линейную зависимость между независимыми и зависимой переменными, что может быть недостаточно для точного моделирования некоторых реальных процессов. В реальных приложениях часто встречаются нелинейные отношения, и линейная регрессия не способна их адекватно описать, что приводит к значительным ошибкам в прогнозах и интерпретации данных. Однако для многих процессов, где изменения происходят медленно и непрерывно, линейная регрессия все же может быть подходящим инструментом.

Линейная регрессия чувствительна к выбросам в данных. Выбросы могут искажать модель и приводить к неправильным выводам, так как метод наименьших квадратов минимизирует сумму квадратов ошибок, и большие ошибки, связанные с выбросами, оказывают непропорционально большое влияние на модель. В контексте поставленной задачи выбросы могут быть обнаружены и исключены как аномалии, что смягчает влияние этого недостатка. Тем не менее, в случае наличия множества выбросов, процесс их удаления может быть трудоемким и повлиять на достоверность модели.

Когда в модели присутствуют сильно коррелирующие между собой независимые переменные, линейная регрессия может давать нестабильные или недостоверные оценки коэффициентов. Мультиколлинеарность приводит к нестабильности оценок, делая их высокозависимыми от небольших изменений в данных, что затрудняет интерпретацию результатов. Влияние мультиколлинеарности может быть ограничено, если выбранные переменные хорошо обоснованы и проведен предварительный анализ корреляций. В

данном случае используется одна независимая переменная, что исключает возможную мультиколлинеарность.

Линейная регрессия предполагает, что ошибки модели (или остатки) являются нормально распределенными и имеют постоянную дисперсию (гомоскедастичность). На практике эти предположения часто нарушаются, что приводит к неэффективности оценок коэффициентов регрессии и занижению стандартных ошибок. Это делает проверку значимости коэффициентов ненадежной и может приводить к неверным выводам о значимости предикторов [15].

Линейная регрессия ограничена в способности работать с категориальными данными. Включение категориальных переменных требует использования метода дамми-переменных, что усложняет модель и может привести к проблемам интерпретации. Если категориальные переменные имеют много уровней, модель становится громоздкой и трудной для анализа [3]. В контексте задачи, связанной с техническим состоянием оборудования, это может ограничить применение линейной регрессии, если необходимо учитывать качественные характеристики оборудования.

Линейная регрессия не учитывает взаимодействия между независимыми переменными, если эти взаимодействия явно не включены в модель. В реальных приложениях взаимодействия могут играть важную роль в определении зависимой переменной, и их игнорирование может привести к потере важной информации и снижению точности модели. Кроме того, линейная регрессия не учитывает автокорреляцию в данных временных рядов, что может приводить к неверным оценкам и прогнозам. Для анализа временных рядов требуется использование специализированных методов, таких как авторегрессионные модели, которые могут учитывать временную зависимость данных.

Линейная регрессия не предоставляет средств для работы с пропущенными значениями в данных. Перед применением модели необходимо предварительно обработать пропущенные значения, что может

быть трудоемким и требовать дополнительных методологических решений, таких как использование методов импутации.

Линейная регрессия ограничена в способности справляться с большим количеством признаков, особенно если число признаков значительно превышает число наблюдений. В таких случаях модель становится переопределенной, что приводит к нестабильным оценкам и плохой обобщающей способности. Для решения этой проблемы могут потребоваться дополнительные методы, такие как регуляризация (например, Lasso или Ridge регрессия), которые не входят в базовую линейную регрессию. В данном случае используется только один признак, поэтому проблема переобучения или переопределенности модели не возникнет.

Учитывая всё вышесказанное недостатки линейной регрессии не являются значимыми для поставленной задачи. Несмотря на свои ограничения и недостатки, линейная регрессия эффективна в анализе данных и прогнозировании из-за своей простоты и интерпретируемости.

2.2 Выбор метода линейной регрессии и его математическая модель

На сегодняшний день существуют множество методов линейной регрессии. Важно выбрать метод, который наиболее подходит для задачи, которая была поставлена на предприятии для наиболее эффективной работы алгоритма:

- метод наименьших квадратов (МНК) прост в реализации, имеет аналитическое решение, позволяет получить оценки параметров модели с минимальной суммой квадратов остатков [17]. Однако он чувствителен к выбросам и наличию коррелированных признаков, требует выполнения предположения о нормальности остатков;
- градиентный спуск может быть эффективным для больших наборов данных, не требует аналитического решения. Несмотря на это он может застревать в локальных минимумах, требует тщательной

настройки параметров, таких как скорость обучения;

- регуляризация позволяет управлять переобучением путем добавления штрафа за сложность модели, может улучшить обобщающую способность модели. Тем не менее метод требует настройки гиперпараметров регуляризации, увеличивает сложность интерпретации модели;
- полиномиальная регрессия позволяет моделировать нелинейные отношения между переменными. Однако полиномиальная регрессия увеличивает сложность модели, что может привести к переобучению [24];
- взвешенная регрессия позволяет учитывать веса для различных наблюдений, что может быть полезно при наличии неоднородности, но требует корректной спецификации весов, которая может быть сложной.

На основе сравнительного анализа выше для нахождения оптимальных значений коэффициентов я буду использовать метод наименьших квадратов (МНК), который был предложен математиком Карлом Гауссом в начале 19 века. Метод заключается в том, чтобы сумма квадратов отклонений отдельных значений зависимой переменной от регрессионной линии была минимальной [20].

Разность между наблюдаемыми и предсказанными значениями может быть как положительной (если предсказанное значение меньше наблюдаемого), так и отрицательной (если предсказанное значение больше наблюдаемого). Однако, если сложить все разности, положительные и отрицательные значения взаимокомпенсируются. Чтобы этого избежать, разница возводится в квадрат. Цель этого метода состоит в том, чтобы минимизировать сумму квадратов остатков RSS [1][4]. Таким образом, сумма остаточных квадратов вычисляется по формуле (формула 5):

$$RSS = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 * x_i)^2, \quad (5)$$

где

y_i – наблюдаемое значение зависимой переменной для i-го наблюдения;

x_i – значение независимой переменной для i-го наблюдения;

β_0 – коэффициент сдвига (интерсепт), который представляет собой значение зависимой переменной y при x=0;

β_1 – коэффициент наклона, который представляет собой изменение в y, вызванное изменением в x;

ε_i – случайная ошибка (остаток) для i-го наблюдения, которая представляет расхождение между наблюдаемым и предсказанным значением y.

Чтобы найти оптимальные значения для β_0 и β_1, дифференцируются RSS по β_0 и β_1, и приравниваются производные к нулю. Решение этих уравнений дает оценки параметров β_0 и β_1, которые минимизируют сумму квадратов остатков [2][7].

Полученные формулы для оценок коэффициентов МНК для линейной регрессии выглядят следующим образом (формула 6 и формула 7):

$$\widehat{\beta_1} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})*(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}, \quad (6)$$

$$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1} * \bar{x}, \quad (7)$$

где:

$\bar{y}$ – среднее значение y;

$\bar{x}$ – среднее значение x;

$\widehat{\beta_1}$ – коэффициент сдвига МНК;

$\widehat{\beta_0}$ – коэффициент наклона МНК.

После оценки коэффициентов можно использовать данную модель для прогнозирования значений зависимой переменной y для новых значений

независимой переменной x.

2.3 Описание функциональности программы и процесса её разработки

Для того, чтобы создать программное обеспечение с пользовательским интерфейсом необходимо сначала определить его функциональность, которая состоит из нескольких основных компонентов: графического интерфейса пользователя (GUI), собственной реализации линейной регрессии, построения графиков и функций для обработки данных.

Программа будет иметь интерактивный графический интерфейс, предоставляющий пользователю удобный способ взаимодействия с приложением. Главное окно приложения будет иметь фиксированный размер и содержать различные элементы управления, такие как метки для отображения статуса, кнопки для запуска определенных функций и область для загрузки данных. Графический интерфейс будет поддерживать возможность перетаскивания файлов, что позволит пользователям загружать данные простым перетаскиванием их в окно приложения.

Для реализации линейной регрессии в программе будет создан специальный класс. Этот класс будет предоставлять методы для обучения модели на основе предоставленных данных, прогнозирования значений и оценки качества модели.

Программа будет включать в себя функционал для построения графиков, на которых будут отображаться исходные данные и линия линейной регрессии. Графики будут создаваться с использованием подходящей библиотеки или фреймворка для визуализации данных.

Также будет создан набор функций для обработки данных, которые будут вызываться в ответ на действия пользователя в графическом интерфейсе. К ним относятся функции для загрузки и обработки данных из файлов, построения графиков, очистки графического окна и т.д. Эти функции

будут координировать работу различных компонентов системы, обеспечивая последовательный и корректный процесс анализа данных.

Составим диаграмму вариантов использования программы, показанную на рисунке 2.

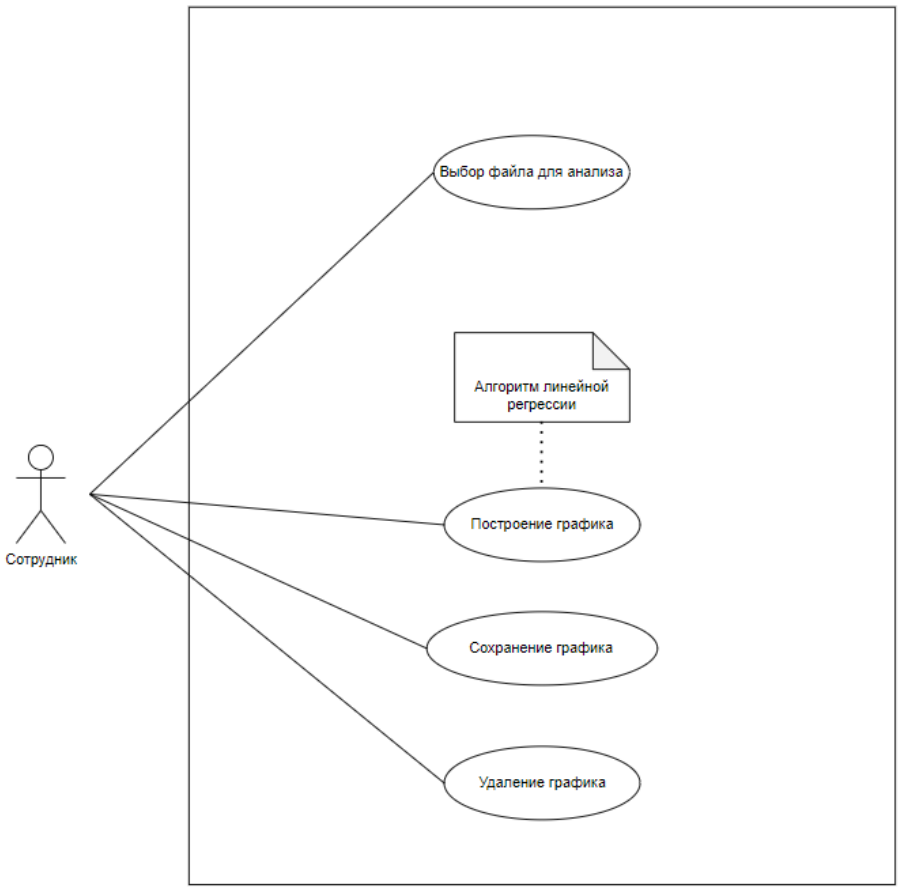

Рисунок 2 – Диаграмма вариантов использования

На основе диаграммы вариантов использования мы можем описать систему для дальнейшей разработки. Программный код будет строить график линейной регрессии для визуализации данных об устранении неполадок на предприятии с учетом коэффициента износа оборудования в течение выбранного периода времени. Система для дальнейшей разработки программного кода представлена ниже (рисунок 3).

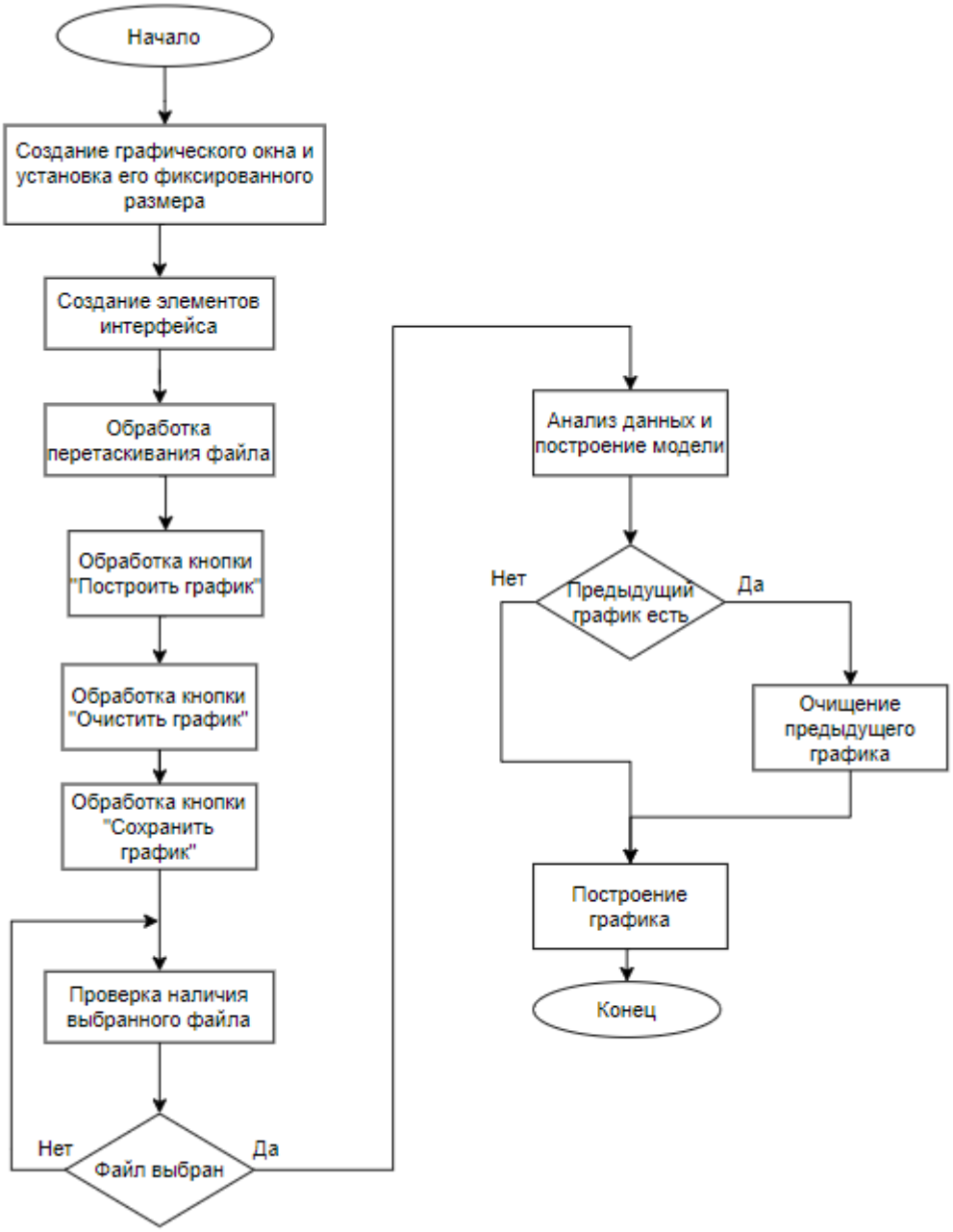

Рисунок 3 – Система для разработки программного кода

Из диаграммы, которая представлена на рисунке 3, мы можем сделать вывод, что в начале работы программы происходит создание окна графического интерфейса и установка его фиксированного размера для взаимодействия с пользователем программы. После этого создаются элементы интерфейса и происходит обработка поля для перетаскивания файла из проводника.

После этих операций производится обработка кнопок построения графика, его очистки и сохранения для того, чтобы с ними можно было взаимодействовать. Затем цикл проверяет наличие файла и в случае успешной загрузки отчёта выводится сообщение «Файл выбран».

Следующим этапом станет анализ данных и построение модели линейной регрессии, а также построение графика этой модели. После завершения анализа полученных результатов пользователь может нажать на кнопку «Очистить график» и ПО проверит есть ли график в данный момент и в случае, если он есть, очистит экран для того, чтобы можно было вывести следующий график.

Выводы по разделу 2:

Во втором разделе была описана математическая модель линейной регрессии, а также был произведен сравнительный анализ и выбран метод линейной регрессии. Был выбран метод наименьших квадратов, а также описана математическая модель данного метода.

Также была описана функциональность программы и система для дальнейшей разработки программного кода.

3 Реализация и тестирование программного обеспечения

3.1 Реализация программного обеспечения

Для разработки и тестирования кода был выбран PyCharm, который является одной из самых используемых сред разработки.

Реализация кода начинается с импорта подходящих библиотек. Сначала импортируются библиотеки, которые необходимы для создания графического дизайна. Для этого импортируется модуль tkinter для создания графического пользовательского интерфейса, а также модули filedialog и ttk из tkinter для работы с файловыми диалогами и элементами интерфейса соответственно, ещё для работы необходимы модули Tk и DND_FILES из tkinterdnd2 для поддержки перетаскивания файлов.

После импорта библиотек для создания графического интерфейса понадобятся технические библиотеки – numpy для работы с массивами и матрицами и pandas для работы с данными. Для построения графиков импортируется модуль pyplot из библиотеки matplotlib и FigureCanvasTkAgg из matplotlib.backends.backend_tkagg для вставки графиков в интерфейс Tkinter.

В начале работы над проектом было необходимо скачать и установить некоторые из упомянутых библиотек. Это можно сделать в командной строке с помощью менеджера пакетов pip, который обычно поставляется вместе с установкой Python.

Приступаем к реализации программного кода. Класс MyLinearRegression создает собственную реализацию линейной регрессии. В этот класс входят три метода и конструктор класса (рисунок 4).

```python
# Класс для собственной реализации линейной регрессии
class MyLinearRegression:
    def __init__(self):
        self.coef_ = None
        self.intercept_ = None

    def fit(self, X, y):
        X = np.array(X)
        y = np.array(y)

        X = np.hstack((np.ones((X.shape[0], 1)), X))

        k = np.linalg.inv(X.T @ X) @ X.T @ y

        self.coef_ = k[1:]
        self.intercept_ = k[0]

    def predict(self, X):
        y_pred = np.dot(X, self.coef_) + self.intercept_
        return y_pred

    def score(self, X, y):
        y_pred = self.predict(X)
        y_mean = np.mean(y)
        ss_res = np.sum((y - y_pred) ** 2)
        ss_tot = np.sum((y - y_mean) ** 2)
        r2 = 1 - (ss_res / ss_tot)
        return r2
```

Рисунок 4 – Класс реализации линейной регрессии

Конструктор класса инициализирует атрибуты coef_ (коэффициенты наклона) и intercept_ (пересечение) значением None при создании объекта класса.

Метод fit подгоняет модель к данным. Он принимает матрицу признаков X и вектор целевых значений y. Сначала он преобразует X и y в массивы NumPy. Затем он добавляет столбец единиц к X, чтобы учесть свободный член

в уравнении линейной регрессии. Затем метод решает уравнение нормальных уравнений для получения коэффициентов наклона и пересечения с помощью метода наименьших квадратов. Значения коэффициентов сохраняются в атрибутах coef_ и intercept_.

Метод predict делает предсказания на основе входных данных X. Он вычисляет предсказанные значения y путем умножения X на коэффициенты наклона и добавления пересечения.

Метод score вычисляет коэффициент детерминации R^2 для модели. Он принимает входные данные X и целевые значения y. Сначала он делает предсказания на основе входных данных с помощью метода predict. Затем он вычисляет сумму квадратов остатков и сумму квадратов отклонений от среднего. Наконец, он вычисляет коэффициент детерминации и выводит на консоль.

Далее объявляем функцию plot_graph, которая строит график для данных и предсказаний модели линейной регрессии. Ей на вход приходят следующие параметры:

- X: данные независимой переменной;
- y: данные зависимой переменной;
- my_lr: модель собственной линейной регрессии.

График содержит рассеяние данных, а также прямую линейной регрессии.

Следующая функция drop_file обрабатывает событие перетаскивания файла и сохраняет путь к файлу в переменной file_path.

Две функции build_graph и clear_graph необходимы для построения графика и очищения его после реализации программы. Функция build_graph вызывает функцию analyze_data, если файл был выбран. Данные функции изображены на рисунке 5.

```python
# Функция для построения графика
def plot_graph(X, y, my_lr):
    fig, ax = plt.subplots()
    ax.scatter(X, y, color='blue', s=10)
    x = np.array([np.min(X), np.max(X)])
    y_pred = my_lr.predict(x.reshape(-1, 1))
    ax.plot(x, y_pred, color='green', label='Линейная регрессия')

    ax.set_xlabel('Коэффициент износа')
    ax.set_ylabel('Время устранения')
    ax.legend()

    return fig

# Функция для обработки перетаскивания файла
def drop_file(event):
    global file_path
    file_path = event.data
    status_label.config(text="Файл выбран")

# Функция для обработки кнопки "Построить график"
def build_graph():
    global file_path
    try:
        if file_path:
            analyze_data(file_path)
    except Exception:
        messagebox.showerror( title: "Ошибка",  message: "Выберите файл для анализа")

# Функция для очистки графика
def clear_graph():
    global fig
    for widget in plot_frame.winfo_children():
        widget.destroy()
    fig = None
    status_label.config(text="График очищен")
    mean_time_label.config(text="Среднее время устранения: N/A")
```

Рисунок 5 – Функции для построения графика

Функция analyze_data в свою очередь анализирует данные из CSV-файла, заменяет запятые на точки для корректной работы, создает модель линейной регрессии, а также строит график.

Функция draw_figure предназначена для отображения графика в

интерфейсе Tkinter. В первую очередь функция проходит по всем виджетам (графическим элементам), которые находятся внутри фрейма plot_frame (это область, где будет отображаться график). Для каждого виджета вызывается метод destroy(), который удаляет виджет из фрейма. Это нужно для того, чтобы очистить область отображения графика перед добавлением нового графика [11]. Далее создается объект canvas класса FigureCanvasTkAgg, который является интерфейсом между объектом Figure из Matplotlib и графическим интерфейсом Tkinter. В качестве аргументов передаются график, который нужно отобразить. Вызывается метод draw() для объекта canvas, который отрисовывает график на холсте. Отображение холста с графиком на графическом интерфейсе Tkinter. Добавляем виджет холста в фрейм plot_frame с помощью метода pack(). Указываем, что холст должен занимать всю доступную верхнюю часть фрейма (side=tk.TOP), заполнять всю доступную область по горизонтали и вертикали (fill=tk.BOTH) и растягиваться при изменении размера фрейма (expand=1).

Для возможности сохранения графика создаем функцию save_graph(). Функция запрашивает у пользователя путь для сохранения файла, предлагая сохранить график в формате PNG или JPEG. Если пользователь выбрал путь для сохранения, то график сохраняется с помощью метода savefig() объекта fig из библиотеки Matplotlib. Данные функции изображены на рисунке 6.

В рассматриваемых функциях выше используется переменная fig, которая хранит объект типа Figure из библиотеки Matplotlib. Этот объект представляет собой контейнер для графических элементов, таких как оси, линии, текстовые метки и т.д [6]. В данном случае fig используется для хранения графического контекста, в котором строится и отображается график линейной регрессии. В функциях перед именем этой переменной используется ключевое слово global. Это связано с тем, что fig является глобальной переменной, а не локальной[12]. Глобальные переменные доступны из любой части программы, в то время как локальные переменные доступны только внутри той функции, в которой они объявлены [21][26]. В данном случае fig

используется в нескольких функциях (например, в plot_graph(), analyze_data(), draw_figure()), и его значение может изменяться в процессе выполнения программы.

```python
# Функция для анализа данных и построения моделей
def analyze_data(file_path):
    global X, y, my_lr, fig
    try:
        # Загрузка данных из CSV-файла
        data = pd.read_csv(file_path, encoding='cp1251', sep=';')
        data[['Коэффициент']] = data[['Коэффициент']].apply(lambda x: x.str.replace(',', '.'))
        X = data[['Коэффициент']].astype(float)
        data['Время'] = data['Время'].str.replace(',', '.')
        y = data['Время'].astype(float)

        my_lr = MyLinearRegression()
        my_lr.fit(X, y)

        my_lr.score(X, y)

        # Расчет среднего времени устранения
        mean_time = y.mean()
        mean_time_label.config(text=f"Среднее время устранения: {mean_time:.2f}")

        # Построение графика
        fig = plot_graph(X, y, my_lr)
        draw_figure(fig)
    except Exception as e:
        status_label.config(text=f"Ошибка: {str(e)}")

# Функция для отрисовки графика в Tkinter
def draw_figure(fig):
    for widget in plot_frame.winfo_children():
        widget.destroy()
    canvas = FigureCanvasTkAgg(fig, master=plot_frame)
    canvas.draw()
    canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

# Функция для сохранения графика
def save_graph():
    global fig
    if fig is not None:
        file_path = tk.filedialog.asksaveasfilename(defaultextension=".png",
                                                    filetypes=[("PNG", "*.png"), ("JPEG", "*.jpg")])
        if file_path:
            fig.savefig(file_path)
            status_label.config(text="График сохранен")
    else:
        messagebox.showerror( title: "Ошибка",  message: "Нет графика для сохранения")
```

Рисунок 6 – Функции для работы с графиком

Когда функции save_graph() нужно изменить или использовать значение глобальной переменной fig, она должна указать это явно с помощью ключевого слова global [5][13][18]. В противном случае Python будет считать, что fig является локальной переменной функции save_graph(), и попытается создать новую локальную переменную с таким именем, что приведет к ошибке, так как значение глобальной переменной не будет использовано.

После объявления создаем графический интерфейс. Для этого создаем окно Tkinter с заголовком "Анализ данных линейной регрессии" и фиксированным размером 800x600, а также включаем поддержку перетаскивания файлов.

Создаем виджеты и элементы управления: кнопки, метки, области перетаскивания.

После чего привязываем обработчик события перетаскивания файла к соответствующему виджету.

В итоге, запускаем главный цикл графического интерфейса.

Фрагменты программного кода представлены в Приложении А.

3.2 Тестирование программного обеспечения и анализ результатов

Произведём тестирование программы. Конечным результатом работы программного кода должна стать визуализация метода алгоритма линейной регрессии для выявления изменения времени устроения неполадок в оборудовании на предприятии в течение выбранного срока.

При запуске программы перед пользователем выводится интерфейс, в котором он должен перетащить файл отчёта в формате csv в указанное поле для того, чтобы на основе полученных данных построить график линейной регрессии (рисунок 7).

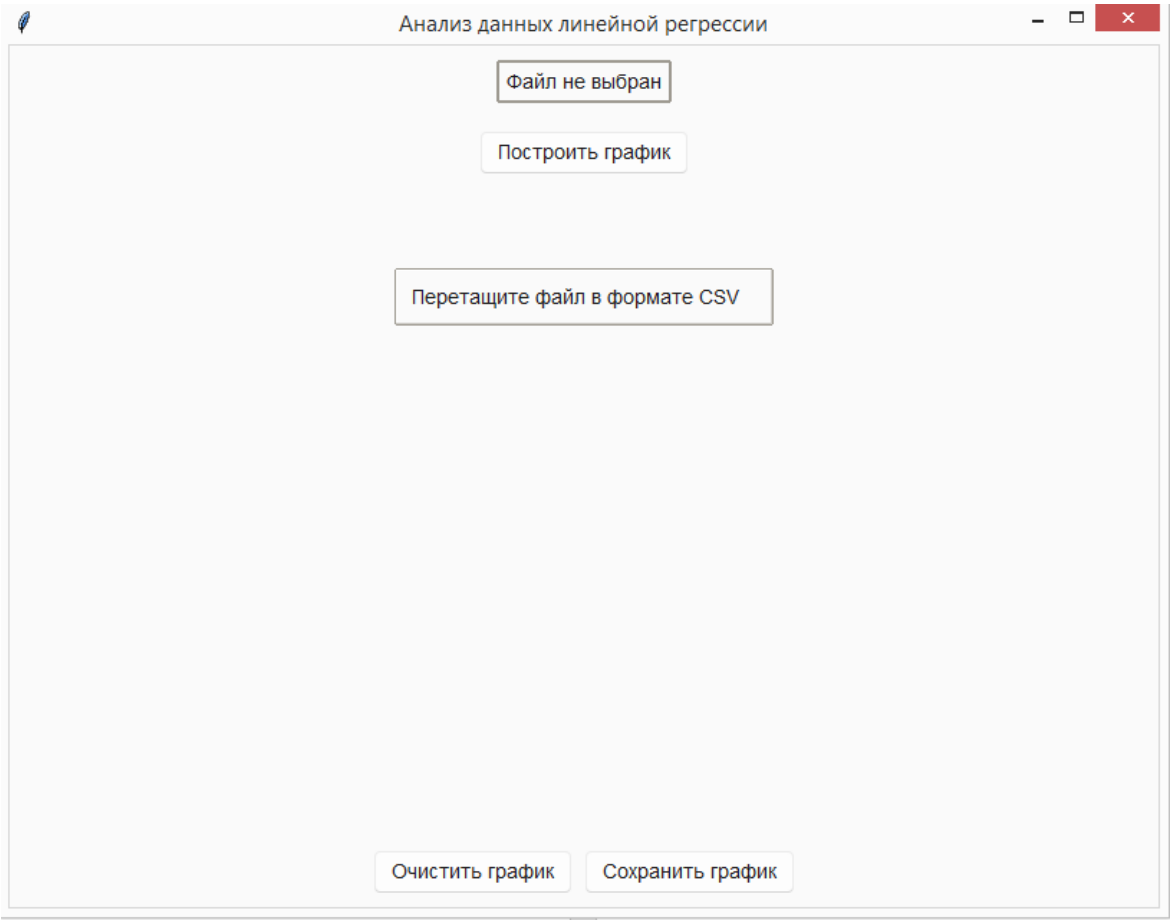

Рисунок 7 – Вывод интерфейса

Если не загрузить файл, то при нажатии на кнопку «Построить график» выведется оконное сообщение об ошибке (рисунок 8).

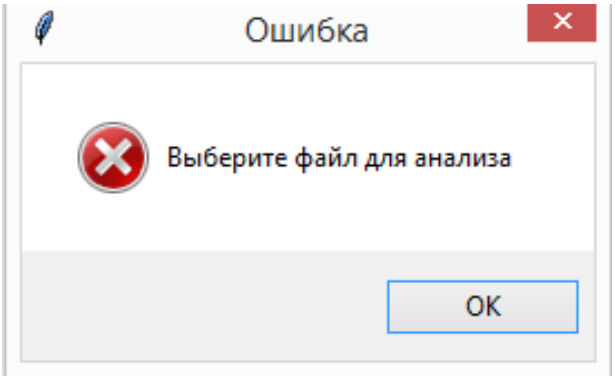

Рисунок 8 – Сообщение об ошибке при построении графика

Аналогично выведется ошибка при нажатии на кнопку сохранения графика, если он отсутствует (рисунок 9).

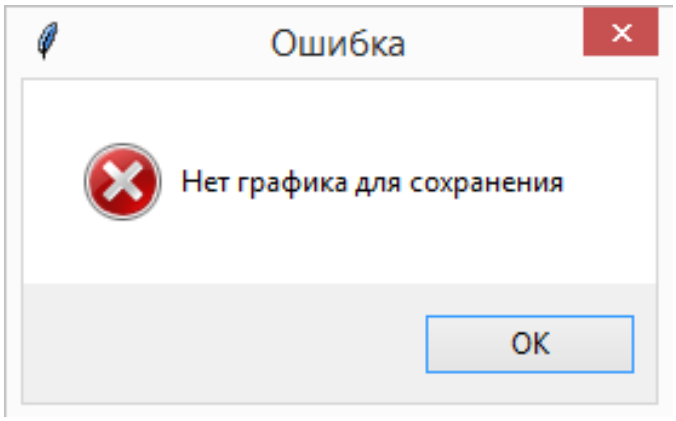

Рисунок 9 – Сообщение об ошибке при сохранении графика

После загрузки файла в интерфейсе появится сообщение, что файл успешно выбран (рисунок 10).

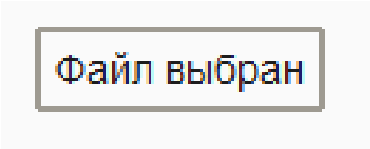

Рисунок 10 – Сообщение об успешной загрузки файла

Отчёт в формате CSV-файла, который мы загружаем, представляет собой набор больших данных, в котором столбец «Время» является временем устранения неполадки в минутах, а столбец «Коэффициент» представляет собой коэффициент износа оборудования. Формат CSV был выбран из-за легкости обработки и анализа данных независимо от используемого инструмента, так как эти файлы могут быть свободно открыты и редактированы как в мощных средах вроде Excel (рисунок 11), так и в обычных текстовых редакторах, что делает их универсальным и гибким средством для работы с информацией. CSV-файлы значительно легче и

быстрее открываются и обрабатываются, поскольку не содержат сложных структур и форматов, присущих файлам XLSX.

	A	B
1	Время	Коэффициент
2	12,5	220,1166528
3	10,93	220,2261636
4	32,4	220,2535412
5	0,92	220,2535412
6	0,27	220,2535412
7	0,83	220,2535412

Рисунок 11 – Отчёт в формате csv

После того, как пользователь выбрал файл он нажимает на кнопку «Построить график» и ПО визуализирует график линейной регрессии. На этом графике каждая точка это устранение неисправности. По вертикальной оси указано время устранения неисправности в минутах, а на горизонтальной оси представлен коэффициент износа оборудования. Вместе с графиком в интерфейсе отображается информация о среднем времени устранения за выбранный период.

Анализ проводился по двум месяцам: по данным сентября 2023 года и января 2023 года, а также по всему 2023 году. Сначала я проверил значения за 2023 год (рисунок 12).

Прямая линейной регрессии отклонена не сильно от горизонтальной оси. Это объясняется тем, что годовой масштаб включает в себя множество различных факторов, которые могут компенсировать друг друга или усредняться. Влияние отдельных месяцев сглаживается на общей динамике.

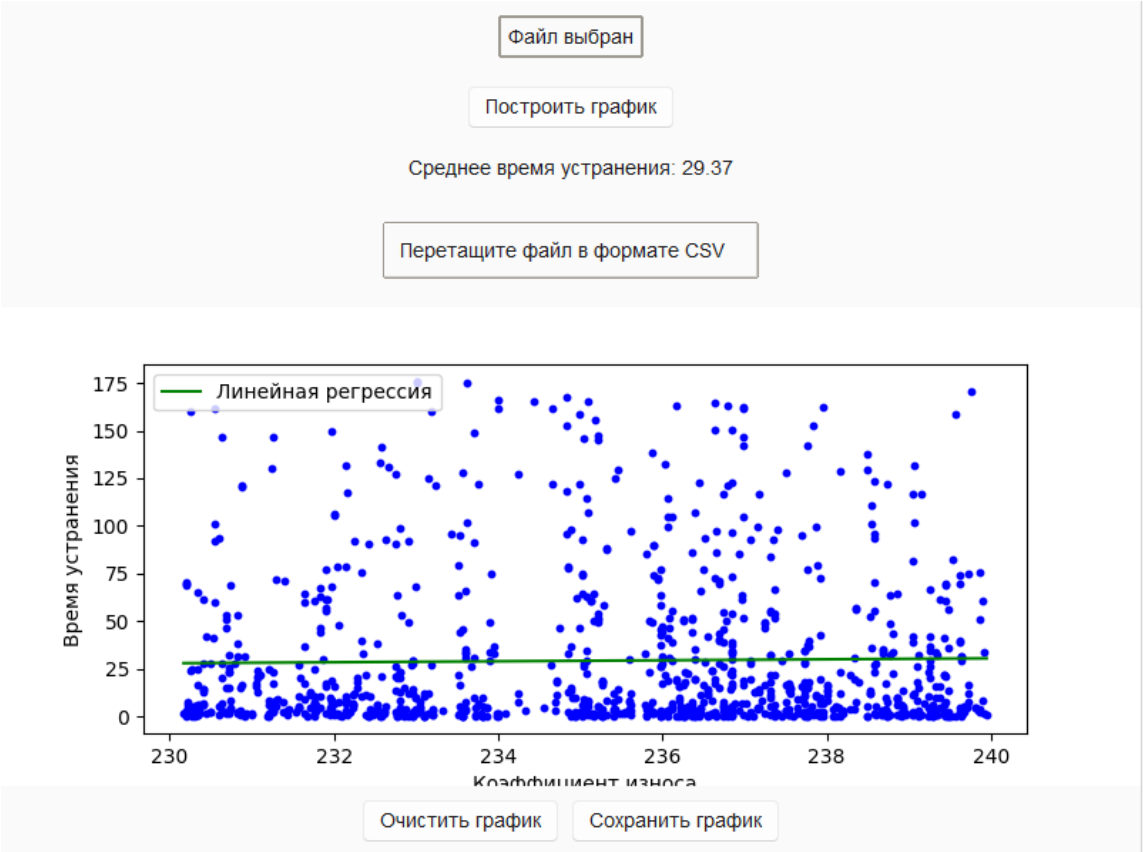

Рисунок 12 – Результат за 2023 год

Рассмотрим также графики линейной регрессии за январь 2023 года и сентябрь 2023 года. На них мы должны будем увидеть более показательную прямую, так как выборка в разы меньше. Результат представлен ниже (рисунок 13 и 14).

Из графиков, представленных на рисунках 13 и 14, можно сделать вывод, что предположение оказалось правдивым и в действительности прямая линейной регрессии имеет более крутой наклон. Это происходит потому, что месячный масштаб позволяет более точно уловить краткосрочные тенденции и изменения, которые могут быть незаметны на годовом уровне. В свою очередь, это означает, что проводить анализ в данной программе эффективнее, если брать более короткие промежутки времени для анализа.

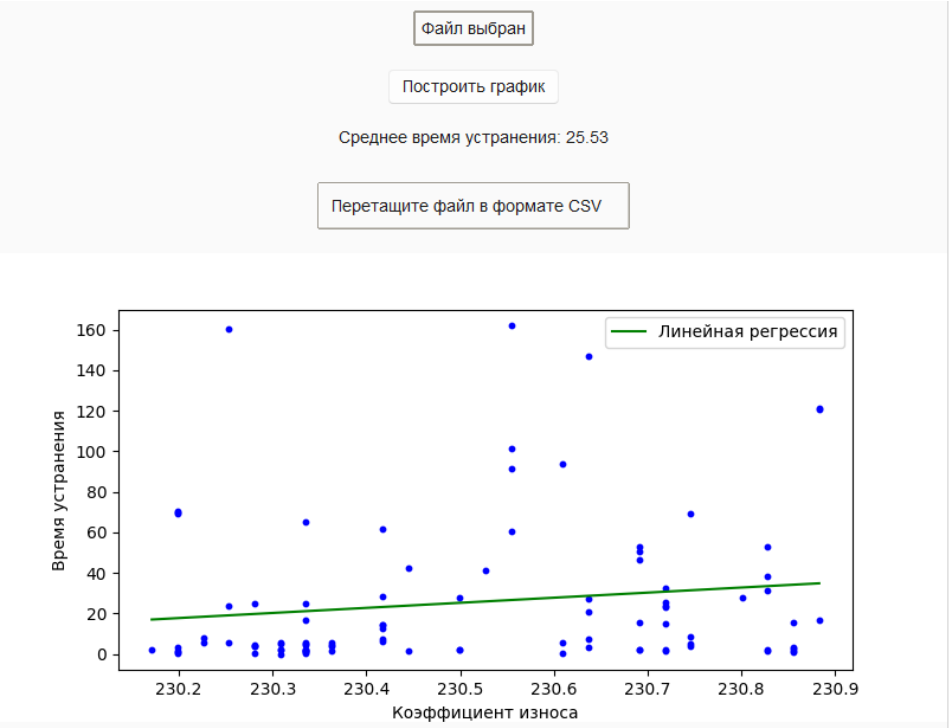

Рисунок 13 – Результат за январь 2023 года

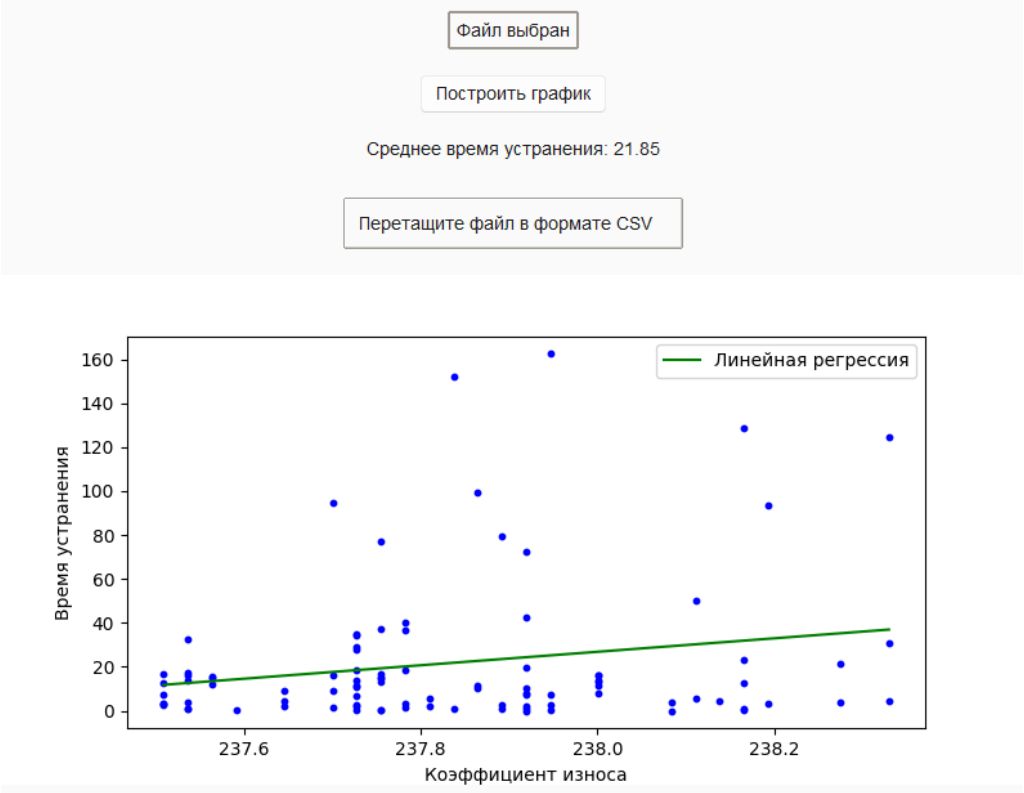

Рисунок 14 – Результат за сентябрь 2023 года

В данном случае мы видим, что на предприятии в сентябре к концу месяца среднее время устранения неполадок увеличилось. Можно сделать предположение, что это могло произойти из-за того, что во второй половине сентября была выше производственная мощность предприятия и оборудование работало дольше и интенсивнее.

Для оценки корректности и точности вычислений, было проведено сравнение коэффициента детерминации, полученного с использованием разработанного программного кода, и аналогичного показателя, рассчитанного в программе Microsoft Excel. Коэффициент детерминации является статистическим показателем, характеризующим качество подгонки регрессионной модели к данным [14]. Он показывает, какая доля вариации зависимой переменной объясняется регрессионной моделью [27].

Для сравнения использовался отчет за январь 2023 года. В Microsoft Excel была использована встроенная функция RSQ, которая вычисляет квадрат коэффициента корреляции Пирсона, что эквивалентно коэффициенту детерминации для линейной регрессии. После проведения расчетов в обеих средах было установлено, что значения коэффициента детерминации, полученные с помощью программного кода и в Excel, совпали (рисунки 15 и 16). Это подтверждает корректность работы алгоритма линейной регрессии в программе и соответствие стандартным статистическим методам расчета.

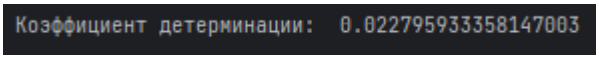

Рисунок 15 – Коэффициент детерминации, полученный с использованием программы

Для оценки точности вычислений, в ходе проведенного исследования был проведен сравнительный анализ показателя коэффициента детерминации, который рассчитывался как с помощью разработанного программного кода, так и в программе Microsoft Excel. Коэффициент детерминации является

важным статистическим показателем, характеризующим качество подгонки регрессионной модели к имеющимся данным, отражая долю вариации зависимой переменной, объясняемую регрессионной моделью.

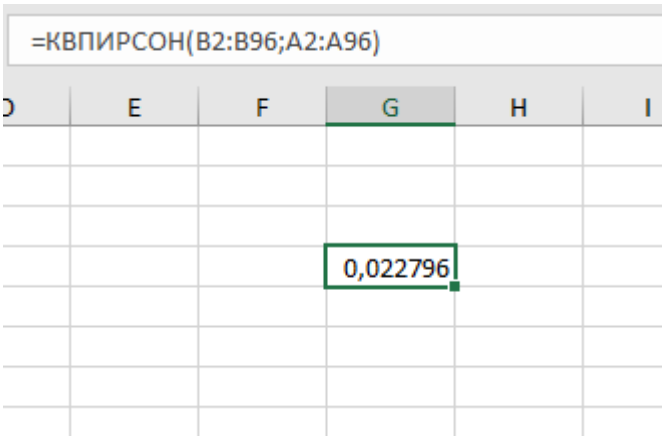

Рисунок 16 – Коэффициент детерминации, рассчитанный в Excel

В ходе сравнения, проведенного на основе данных отчета за январь 2023 года, было установлено, что значения коэффициента детерминации, полученные с использованием программного кода и с помощью встроенной функции RSQ в Excel (вычисляющей квадрат коэффициента корреляции Пирсона, эквивалентный коэффициенту детерминации для линейной регрессии), полностью совпали. Этот факт подтверждает корректность работы алгоритма линейной регрессии, реализованного в программном обеспечении, и соответствие его стандартным статистическим методам расчета данного показателя.

Выводы по разделу 3:

В третьем разделе был подробно описан процесс написания программного кода и его реализация. Также было произведено тестирование и проанализированы результаты работы программы на различных наборах данных.

Заключение

Бакалаврская работа посвящена программной реализации алгоритма линейной регрессии.

В ходе выполнения выпускной квалификационной работы были поставлены задачи на исследования.

Сначала был произведен сравнительный анализ языков программирования и графических интерфейсов для решения поставленной задачи. Выбор был в пользу Python и графического интерфейса tkinter. Также было описано практическое применение метода линейной регрессии в рамках поставленной на предприятии задачи.

Далее было описание математической модели линейной регрессии и выбор метода линейной регрессии. Выбор был в пользу метода наименьших квадратов. Также была описана математическая модель метода наименьших квадратов.

Затем была описана функциональность программы с использованием диаграммы вариантов использования и спроектирована система для дальнейшего написания программного кода.

В заключительном разделе ВКР была описана реализация программного обеспечения и произведено его тестирование, а также был произведён анализ полученных результатов. Анализ результатов был произведен на основе данных за январь 2023 года, сентябрь 2023 года, а также в целом за весь 2023 год.

Цель работы была выполнена: разработано программное обеспечение для визуализации результатов времени устранения неполадок на предприятии в течение выбранного срока.

Задачи, определённые для достижения цели работы, были выполнены в полном объёме.

Список используемой литературы

1. Андреев Н.А. Программирование на Python для анализа данных. М.: Издательский дом "Лори", 2019. 384 с.
2. Беляев В.И., Громакова Л.Г. Python и интерфейс tkinter: создание графических пользовательских приложений. СПб.: БХВ-Петербург, 2017. 352 с.
3. Васильев Е.С., Ларин В.И. Линейная регрессия: методы и программная реализация. М.: Книга по Требованию, 2020. 168 с.
4. Горбунов А.В., Козлова И.В. Программирование на Python и анализ данных: примеры и задачи. М.: Издательство "Наука", 2021. 240 с.
5. Жуков Н.С., Иванов Д.А. Анализ и обработка данных на Python: практикум. М.: ДМК Пресс, 2019. 336 с.
6. Зайцев В.К., Коршунов А.В. Программирование на Python для анализа и обработки данных. М.: Издательский дом "Лори", 2020. 224 с.
7. Колесов Д.П., Макарова А.В. Введение в машинное обучение и анализ данных на Python. М.: Питер, 2021. 416 с.
8. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
9. Лебедев Г.В., Михайлов А.А. Методы анализа данных и машинного обучения на Python. СПб.: БХВ-Петербург, 2018. 352 с.
10. Мартынов А.Г., Наумова Е.С. Программирование на Python для анализа данных и построения моделей. М.: Издательский дом "Лори", 2021. 288 с.
11. Петров Н.М. Введение в анализ данных и машинное обучение на Python. М.: Издательский дом "Лори", 2020. 272 с.
12. Прохоренок Н.А. Python 3 и PyQt. Разработка приложений. – СПб.: БХВ-Петербург, 2012. – 704 с.
13. Россум, Г. Язык программирования Python / Г. Россум, Ф.Л.Дж. Дрейк, Д.С. Откидач и др.. - М.: [не указано], 2019. - 263 с.

14. Слуцкин, Л. Н. Анализ стабильности модели линейной регрессии во времени / Л.Н. Слуцкин. - М.: Синергия, 2007. - 955 с.

15. Смирнов Е.И., Тарасов А.В. Линейная регрессия: теория и практика. СПб.: Питер, 2018. 240 с.

16. Тихонов А.С., Устинова М.В. Программирование на Python для анализа данных и визуализации результатов. М.: ДМК Пресс, 2021. 320 с.

17. Устинов Д.А., Федотов П.Н. Программирование на Python для анализа и обработки данных. М.: Проспект, 2019. 384 с.

18. Чернов А.П., Карпов Д.С. Программирование на Python и анализ данных. М.: Издательство "Наука", 2019. 304 с.

19. Шаров С.П., Смирнов Д.В. Python и анализ данных: методы и приложения. М.: Проспект, 2020. 432 с.

20. Шестаков А.Г., Леонтьева Л.М. Программирование на Python для анализа и обработки данных: практикум. М.: Издательство "Наука", 2018. 224 с.

21. Щербаков П.В., Родионова И.А. Введение в программирование на Python и анализ данных. М.: Питер, 2021. 368 с.

22. Aggarwal C.C. Data Mining: The Textbook. Springer, 2015. 734 p.

23. Bishop C.M. Pattern Recognition and Machine Learning. Springer, 2006. 738 c.

24. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, 2009. 745 c.

25. Kim H., Lee I., Lee K. Introduction to Data Science. CRC Press, 2017. 394 c.

26. Raschka S., Mirjalili V. Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow. Packt Publishing, 2019. 770 c.

27. Wu X., Kumar V., Quinlan J.R. et al. Top 10 Algorithms in Data Mining. CRC Press, 2009. 362 c.

Приложение А

Фрагменты программного кода

Листинг:

```
import tkinter as tk
import sv_ttk
from tkinter import ttk, messagebox
from tkinterdnd2 import Tk, DND_FILES
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

# Класс для собственной реализации линейной регрессии
class MyLinearRegression:
    def __init__(self):
        self.coef_ = None
        self.intercept_ = None

    def fit(self, X, y):
        X = np.array(X)
        y = np.array(y)

        # Добавляем столбец единиц для расчета свободного члена (интерсепта)
        X = np.hstack((np.ones((X.shape[0], 1)), X))

        # Метод наименьших квадратов для нахождения коэффициентов
        k = np.linalg.inv(X.T @ X) @ X.T @ y

        self.coef_ = k[1:]
```

```
        self.intercept_ = k[0]

    def predict(self, X):
        X = np.array(X)
        y_pred = np.dot(X, self.coef_) + self.intercept_
        return y_pred

    def score(self, X, y):
        y_pred = self.predict(X)
        y_mean = np.mean(y)
        ss_res = np.sum((y - y_pred) ** 2)
        ss_tot = np.sum((y - y_mean) ** 2)
        r2 = 1 - (ss_res / ss_tot)
        print("Коэффициент детерминации: ", r2)

# Функция для построения графика
def plot_graph(X, y, my_lr):
    fig, ax = plt.subplots()
    ax.scatter(X, y, color='blue', s=10)
    x = np.array([np.min(X), np.max(X)])
    y_pred = my_lr.predict(x.reshape(-1, 1))
    ax.plot(x, y_pred, color='green', label='Линейная регрессия')

    ax.set_xlabel('Коэффициент износа')
    ax.set_ylabel('Время устранения')
    ax.legend()

    return fig
```

```
# Функция для обработки перетаскивания файла
def drop_file(event):
    global file_path
    file_path = event.data
    status_label.config(text="Файл выбран")

# Функция для обработки кнопки "Построить график"
def build_graph():
    global file_path
    try:
        if file_path:
            analyze_data(file_path)
    except Exception:
        messagebox.showerror("Ошибка", "Выберите файл для анализа")

# Функция для очистки графика
def clear_graph():
    global fig
    for widget in plot_frame.winfo_children():
        widget.destroy()
    fig = None
    status_label.config(text="График очищен")
    mean_time_label.config(text="")

# Функция для анализа данных и построения моделей
def analyze_data(file_path):
    global X, y, my_lr, fig
    try:
```

```
        # Загрузка данных из CSV-файла
        data = pd.read_csv(file_path, encoding='cp1251', sep=';')
        data[['Коэффициент']] = data[['Коэффициент']].apply(lambda x: x.str.replace(',', '.'))
        X = data[['Коэффициент']].astype(float)
        data['Время'] = data['Время'].str.replace(',', '.')
        y = data['Время'].astype(float)

        my_lr = MyLinearRegression()
        my_lr.fit(X, y)

        # Расчет среднего времени устранения
        mean_time = y.mean()
        mean_time_label.config(text=f"Среднее время устранения: {mean_time:.2f}")

        # Построение графика
        fig = plot_graph(X, y, my_lr)
        draw_figure(fig)
    except Exception as e:
        status_label.config(text=f"Ошибка: {str(e)}")

# Функция для отрисовки графика в Tkinter
def draw_figure(fig):
    for widget in plot_frame.winfo_children():
        widget.destroy()
    canvas = FigureCanvasTkAgg(fig, master=plot_frame)
    canvas.draw()
```

```
    canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

# Функция для сохранения графика
def save_graph():
    global fig
    if fig is not None:
        file_path = tk.filedialog.asksaveasfilename(defaultextension=".png",
                                                    filetypes=[("PNG", "*.png"), ("JPEG", "*.jpg")])
        if file_path:
            fig.savefig(file_path)
            status_label.config(text="График сохранен")
    else:
        messagebox.showerror("Ошибка", "Нет графика для сохранения")

# Создание графического интерфейса
root = Tk()
root.title("Анализ данных линейной регрессии")

# Устанавливаем фиксированный размер окна
root.geometry("800x600")

# Устанавливаем начальную тему
sv_ttk.set_theme("light")

# Разрешаем перетаскивание файлов
root.drop_target_register(DND_FILES)

# Фрейм для размещения кнопок и метки
```

```
top_frame = ttk.Frame(root)
top_frame.pack(side=tk.TOP, pady=10)

# Метка для отображения статуса выбора файла
status_label = ttk.Label(top_frame, text="Файл не выбран", relief="solid", borderwidth=1, padding=5)
status_label.pack(side=tk.LEFT, padx=10)

# Кнопка для построения графика
plot_button = ttk.Button(root, text="Построить график", command=build_graph)
plot_button.pack(side=tk.TOP, pady=10)

# Метка для отображения среднего времени устранения
mean_time_label = ttk.Label(root)
mean_time_label.pack(side=tk.TOP, pady=10)

# Фрейм для размещения кнопок "Очистить график" и "Сохранить график"
button_frame = ttk.Frame(root)
button_frame.pack(side=tk.BOTTOM, pady=10)

# Кнопка для очистки графика
clear_button = ttk.Button(button_frame, text="Очистить график", command=clear_graph)
clear_button.pack(side=tk.LEFT, padx=5)

# Кнопка для сохранения графика
save_button = ttk.Button(button_frame, text="Сохранить график", command=save_graph)
```

```
save_button.pack(side=tk.RIGHT, padx=5)

# Информативная область для перетаскивания файлов
drop_area = ttk.Label(root, text="Перетащите файл в формате CSV", relief="groove", borderwidth=2, width=30, padding=10)
drop_area.pack(side=tk.TOP, pady=20)

# Привязываем обработчик события <<Drop>> к drop_area
drop_area.drop_target_register(DND_FILES)
drop_area.dnd_bind('<<Drop>>', drop_file)

# Фрейм для отображения графиков
plot_frame = ttk.Frame(root)
plot_frame.pack(side=tk.TOP, fill=tk.BOTH, expand=True)

# Запуск главного цикла графического интерфейса
root.mainloop()
```