Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки (специальность) 09.03.03. Прикладная информатика
Кафедра Программной инженерии

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Применение методов машинного обучения для предсказания длительности цепочек вычислительных задач

УДК 004.85-049.8.004.832

Студенты

Группа	ФИО	Подпись	Дата
8К31	Дутов Иван Юрьевич		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ведущий программист	Губин М.Ю.			

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент кафедры МЕН	Тухватулина Л.Р.	к.ф.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент кафедры ЭБЖ	Пустовойтова М.И.	к.х.н		

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
ПИ	Иванов М.А.	к.т.н		

Томск – 2017г.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки (специальность) 09.03.03. Прикладная информатика
Кафедра Программной инженерии

УТВЕРЖДАЮ:
Зав. кафедрой
_____ _____ М.А.Иванов
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Бакалаврской работе

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
8К31	Дутов Иван Юрьевич

Тема работы:

Применение методов машинного обучения для предсказания длительности цепочек вычислительных задач	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	15 июня 2017 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	Работа направлена на создание инструментария, предназначенного для предсказания поведения вычислительной системы ATLAS Production System при расчёте цепочек вычислительных заданий. Реализация функционала предсказания длительности обработки цепочек вычислительных заданий является одним из компонентов системы мониторинга и обнаружения аномалий для ATLAS Production System. Исходными данными к работе являются набор данных, описывающий обработанные в системе задачи и техническое задание на разработки.

Перечень подлежащих исследованию, проектированию и разработке вопросов	1. Рассмотрение понятий и моделей машинного обучения. 2. Рассмотрение предметной области. 3. Анализ исходных данных. 4. Формирование набора данных, описывающего цепочки вычислительных задач. 5. Построение предсказательных моделей разных классов. 6. Сравнение результатов предсказаний моделей. 7. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение. 8. Социальная ответственность.
Перечень графического материала	Графики структуры цепочки вычислительных задач, точности предсказательных моделей. Диаграмма Ганта цепочки вычислительных задач. Гистограмма важности признаков наиболее точной модели.

Консультанты по разделам выпускной квалификационной работы

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность, ресурсосбережение.	Тухватулина Лилия Равильевна
Социальная ответственность	Пустовойтова Марина Игоревна

Названия разделов, которые должны быть написаны на русском и иностранном языках:

1. Теоритические основы.
2. Применение методов машинного обучения.
3. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение.
4. Социальная ответственность.

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ведущий программист	Губин М.Ю.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К31	Дутов Иван Юрьевич		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8К31	Дутов Иван Юрьевич

Институт	Кибернетики	**Кафедра**	Программной инженерии
Уровень образования	Бакалавриат	**Направление/специальность**	Прикладная информатика

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих* 2. *Нормы и нормативы расходования ресурсов* 3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	Применение методов машинного обучения для определения продолжительности обработки цепочек вычислительных задач для производственной системы компании «ATLAS»
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	Анализ перспективности проекта
2. *Планирование и формирование бюджета научных исследований*	Планирование этапов работ, определение трудоемкости и построение календарного графика, формирование бюджета.
3. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования*	Оценка сравнительной эффективности исследования
Перечень графического материала *(с точным указанием обязательных чертежей)***:**	
1. Матрица SWOT 2. График Ганта	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент кафедры МЕН	Тухватулина Л.Р.	к.ф.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К31	Дутов Иван Юрьевич		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8К31	Дутов Иван Юрьевич

Институт	Кибернетики	**Кафедра**	Программной инженерии
Уровень образования	Бакалавриат	**Направление/специальность**	Прикладная информатика

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования	Применение методов машинного обучения для определения продолжительности обработки цепочек вычислительных задач для производственной системы компании «ATLAS»
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – требования к микроклимату; – расчет уровням шума на рабочих местах, оборудованных ПЭВМ; – требования к освещенности; 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – требования к электробезопасности	**1. Производственная безопасность** 1.1. Анализ факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – показатели микроклимата; – уровни шума и вибрации; – уровни электромагнитных излучений; – освещенность рабочей зоны. 1.2. Анализ факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – электрический ток.
2. Экологическая безопасность: – воздействие на окружающую среду	**2. Экологическая безопасность:** Анализ негативного воздействия на окружающую природную среду: утилизация компьютеров и другой оргтехники. В том числе мусорные отходы(бумага)
3. Безопасность в чрезвычайных ситуациях:	**3. Безопасность в чрезвычайных ситуациях:** – перечень возможных ЧС при разработке и эксплуатации проектируемого решения; – выбор наиболее типичной ЧС;

	– разработка превентивных мер по предупреждению ЧС; – разработка действий в результате возникшей ЧС и мер по ликвидации её последствий.
4. Правовые и организационные вопросы обеспечения безопасности:	**4. Правовые и организационные вопросы обеспечения безопасности:** – специальные правовые нормы трудового законодательства при работе с компьютером и орг. техникой; – требования к организации рабочих мест пользователей.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцен кафедры ЭБЖ	Пустовойтова М.И.	к.х.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8К31	Дутов Иван Юрьевич		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики
Направление подготовки (специальность) Прикладная информатика
Уровень образования Бакалавр
Кафедра Программная инженерия
Период выполнения (осенний / весенний семестр 2016/2017 учебного года)

Форма представления работы:

бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
09.03.17	*Раздел 1. Теоритические Основы*	*20*
15.04.17	*Раздел 2. Применение методов машинного обучения*	*50*
25.05.17	*Раздел 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение*	*15*
27.05.17	*Раздел 5. Социальная ответственность*	*15*

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ведущий программист	Губин М.Ю.			

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
ПИ	Иванов М.А.	к.т.н.		

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ООП

Код результата	Результат обучения
Профессиональные компетенции	
Р1	Применять базовые и специальные естественно-научные и математические знания в области информатики, экономики, маркетинга и менеджмента, достаточные для комплексной инженерной деятельности.
Р2	Применять базовые и специальные знания в области современных информационных технологий для решения инженерных и экономических задач.
Р3	Ставить и решать задачи комплексного анализа, связанные с созданием новых информационных технологий и информационных систем в экономике, с использованием базовых и специальных знаний, современных аналитических методов и моделей.
Р4	Разрабатывать новые и модернизировать уже существующие информационные технологии и системы (в экономике) в соответствии с техническим заданием.
Р5	Проводить теоретические и экспериментальные исследования, включающие поиск и изучение необходимой научно-технической информации, математическое моделирование, проведение эксперимента, анализ и интерпретация полученных данных, в области прикладной информатики. Проводить исследования, связанные с оценкой информационной безопасности проектов.
Р6	Внедрять, эксплуатировать и обслуживать современные информационные технологии и системы, обеспечивать их

	высокую эффективность, соблюдать правила охраны здоровья, безопасность труда, выполнять требования по защите окружающей среды.
Универсальные компетенции	
Р7	Использовать базовые и специальные знания в области проектного менеджмента для ведения комплексной инженерной деятельности.
Р8	Владеть иностранным языком на уровне, позволяющем работать в иноязычной среде, разрабатывать документацию, презентовать и защищать результаты комплексной инженерной деятельности.
Р9	Эффективно работать индивидуально и в качестве члена группы, состоящей из специалистов различных направлений и квалификаций, демонстрировать ответственность за результаты работы и готовность следовать корпоративной культуре организации.
Р10	Демонстрировать знания правовых, социальных, экономических и культурных аспектов комплексной инженерной деятельности.
Р11	Демонстрировать способность к самостоятельной к самостоятельному обучению в течение всей жизни и непрерывному самосовершенствованию в инженерной профессии.

Реферат

Дипломная работа содержит: 95 страниц, 11 рисунков, 19 таблиц, 17 источников, 9 приложений.

Ключевые слова: машинное обучение, программирование, анализ данных, разработка, распределенные системы обработки данных.

Объектом исследования данной работы является закономерность времени обработки цепочек вычислительных задач, в зависимости от их характеристик. Предметом исследования является построение алгоритмов, способных решить поставленную задачу.

Целью данной работы является применение методов машинного обучения для построения модели, предсказывающей время обработки цепочек вычислительных задач в производственной системе производства и обработки данных эксперимента ATLAS.

В процессе исследования была исследована производственная система ProdSys2, а также данные, описывающие цепочки вычислительных задач.

В результате исследования была построена модель, предсказывающая время выполнения цепочек вычислительных задач. Данная модель, позволит организовать систему планирования загрузки цепочек в систему ProdSys2.

Степень внедрения: планируемое внедрение в течении следующего полугода.

Область применения: распределенные системы обработки данных, методы машинного обучения.

Содержание

Введение

В настоящее время, эксперименты по физике высоких энергий являются одним из наиболее актуальных видов физических исследований, вносящих неоспоримый вклад в фундаментальную науку. Как правило, результатом таких экспериментов является огромное количество данных, зарегистрированных детекторами ускорителя заряженных частиц. Обработка такого объема данных требует большие вычислительные мощности, в связи с чем создаются распределенные системы обработки данных, содержащие большое количество суперкомпьютеров. Однако использование такого оборудования обходится дорого, поэтому необходима организация рационального планирования обработки данных, во избежание простоя оборудования и неравномерного распределения задач по обработки, среди суперкомпьютеров системы обработки данных. Очевидно, что определение времени обработки данных являются ключевой задачей для организации системы планирования. Решением такой задачи может стать построение методами машинного обучения модели, способной предсказывать время обработки данных.

Целью данной работы является применение методов машинного обучения для построения модели, предсказывающей время обработки цепочек вычислительных задач в производственной системе производства и обработки данных эксперимента ATLAS. Для решения данной задачи используются реальные данные об обработки вычислительных задач за три месяца с начала 2017 года. Главной особенностью этого набора данных является тот факт, что он описывает вычислительные задачи а не цепочки, в связи с чем необходимо построить новый набор, характеризующий цепочки вычислительных задач.

Таким образом, для достижения поставленной цели необходимо выполнить ряд задач:

- рассмотрение понятий и методов машинного обучения;
- рассмотрение предметной области;
- анализ исходных данных;

- формирование набора данных, описывающего цепочки вычислительных задач;
- построение предсказательных моделей разных классов;
- сравнение результатов предсказаний моделей.

Объектом исследования данной работы является закономерность времени обработки цепочек вычислительных задач, в зависимости от их характеристик. Предметом исследования является построение моделей, способных решить поставленную задачу.

Сегодня, машинное обучение является активно развивающимся способом решения разного рода задач, находящим применение в различных отраслях. Подход применения методов машинного обучения для оценки длительности вычисления цепочек задач в среде, подобной системе эксперимента ATLAS является новым и не имеет аналогов в мире. Данные факты служат доказательством актуальности данной работы.

Глава 1. Теоретические основы

Целью данной главы является рассмотрение основных понятий машинного обучения, которые часто используются в рамках данной работы. Также, приводится формальное описание моделей машинного обучения, используемых для решения задачи предсказания длительности выполнения цепочек вычислительных задач, рассматриваются их особенности, преимущества и недостатки.

1.1. Основные понятия машинного обучения

Машинное обучение – это наука, изучающая способы извлечения закономерностей среди признаков реальных физических объектов или процессов из ограниченного количества примеров, для решения поставленных задач. Примерами такого рода задач являются: отнесение объекта или процесса к определенному классу, соответствующему его параметрам; построение прогноза поведения объекта или процесса, разделение набора объектов или процессов на группы, обладающие некоторыми свойствами.

Приведенные выше примеры соответствуют наиболее распространенным классам задач машинного обучения, называемым: задачи классификации, задачи анализа регрессии и задачи кластеризации. Так как в данной работе решалась задача регрессионного анализа, то подробное описание приведено только для этого класса задач. Однако, прежде чем перейти к непосредственному описанию задач регрессии, необходимо определить некоторые основные понятия машинного обучения [1].

Исходя из определения, приведенного выше, машинное обучение имеет дело с наборами примеров (данных) содержащими определенные признаки. Выявление закономерностей в наборе данных, содержащим метки объектов, т.е. ответы на интересующий нас вопрос для каждого объекта, называется обучением на размеченных данных. Обучающей выборкой называется набор данных вида:

$$X = \{(x_1, y_1), \dots, (x_l, y_l)\}, \quad (1)$$

где $x_1, \dots, x_l$ – обучающие объекты;

$y_1, \dots, y_l$ – метки объектов;

l – их количество.

Выявление закономерностей в обучающей выборке происходит с помощью определенного алгоритма (модели) из множества алгоритмов A, который представляет собой функцию перехода из пространства объектов в пространство ответов. Для оценки качества работы модели используется характеристика, называемая функционалом ошибки. $Q(a, X)$ – функционал алгоритма a на выборке X

Таким образом, постановку задачи регрессии можно определить следующим образом. Для обучения выборки X необходимо найти такой алгоритм $a \in A$ (множеству алгоритмов), на котором будет достигаться минимум функционала ошибки [2]:

$$Q(a, X) \to min_{a \in A}. \quad (2)$$

В результате построения предсказательной модели может возникнуть ситуация, при которой алгоритм показывает хороший результат на обучающей выборки, и плохой на реальных данных. Такая ситуация называется переобучением и возникает в случае, когда алгоритм хорошо подстраивается под объекты обучающей выборки, включая выбросы, не отражая при этом реальной зависимости. Ситуация, при которой алгоритм показывает плохой результат и на обучающей выборки, и на реальных данных называется недообучением. Для контроля качества моделей, используется отложенная выборка, т.е. выборка, объекты которой не участвуют в обучении [3].

Выше, было упомянуто, что существует некоторое множество моделей A. Далее рассматриваются некоторые модели этого множества, используемые в данной работе.

1.2. Константная модель

Под константной моделью понимается алгоритм, который возвращает одинаковое значение, полученное в результате обработки обучающей выборки, для любого входного объекта данной модели.

Такое значение может быть получено, путем применения некоторой функции к значениям признака или группы признаков обучающей выборки. Функцией могут служить: среднее значение, медиана, мода и др.

Данная модель очень проста и редко применяется на практике, однако в ряде случаев она может оказаться очень полезной.

1.3. Модель Лассо

Линейной моделью называется модель вида:

$$a(x) = w_0 + \sum_{j=1}^{d} w_j \cdot f_j(x_j) = \sum_{j=1}^{d+1} w_j \cdot f_j(x_j), \quad (3)$$

где w_0 – свободный коэффициент или сдвиг;

x_j – значения признаков;

f_j – некоторая функция преобразования значений признака x_j;

w_j – веса, d – количество признаков. Свободный коэффициент можно внести под знак суммы, если добавить $(d + 1)$-й признак, который на каждом объекте принимает значение 1. Исходя из определения скалярного произведения векторов приведенную выше формулу можно записать в виде:

$$a(x) = \langle w, f(x) \rangle, \quad (4)$$

где $\langle w, f(x) \rangle$ – скалярное произведение вектора функций значений признаков и вектора весов [4].

В качестве функционала ошибки в линейных моделях обычно используется метод подсчета среднеквадратичной ошибки алгоритма, который задается следующим образом:

$$Q(a, x) = \frac{1}{l} \sum_{i=1}^{l} (a(x_i) - y_i)^2, \quad (5)$$

где l – количество объектов в обучающей выборке.

Подсчет среднеквадратичной ошибки осуществляется путем подсчета среднего значений квадратов отклонений предсказаний модели по всем объектам обучающей выборки [4].

Таким образом, решение задач регрессии с помощью линейной модели можно определить как поиск такого вектора весов при значениях признаков, при которых будет достигаться минимум функционала ошибки:

$$Q(w,x) = \frac{1}{l}\sum_{i=1}^{l}(\langle w_i, f_i(x_i)\rangle - y_i)^2 \to min. \qquad (6)$$

Следует отметить, что линейные модели легко переобучаются. Для представления проблемы переобученности линейных моделей, далее приводится пример, где в качестве алгоритмов используются полиномы первой четвертой и пятнадцатой степени. На рисунке 1 представлены: синей линией графики полиномов, точками показаны положения объектов обучающей выборки, зеленой линией представлена истинная зависимость значений признаков объектов.

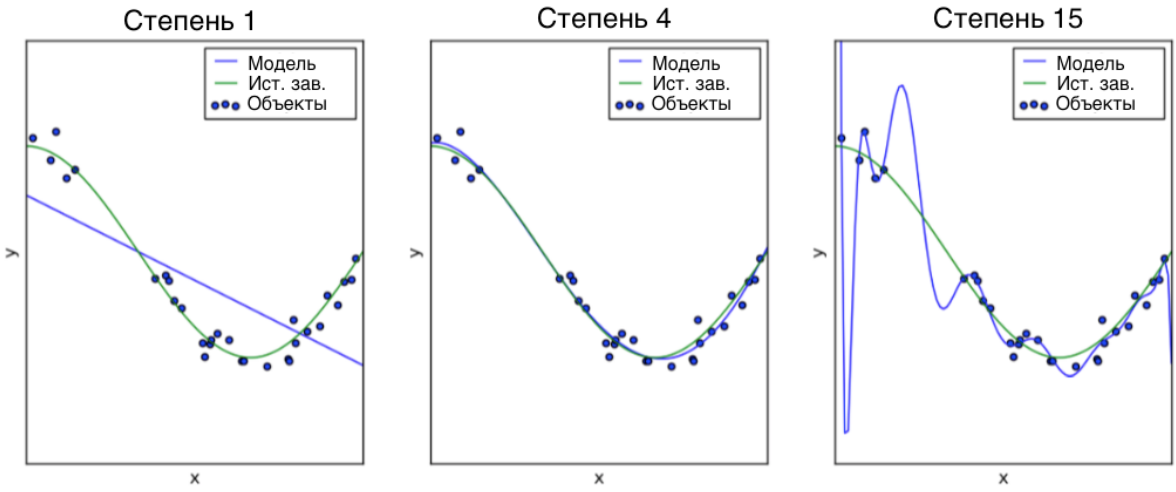

Рисунок 1 – Пример переобучения линейных моделей

Как показано на рисунке выше, полином пятнадцатой степени хорошо подстраивается под объекты обучающей выборки, но совсем не отражает истинной зависимости.

Одной из основных проблем переобученности линейных моделей являются большие веса при значениях признаков. Для борьбы с этим

используется видоизмененный функционал ошибки, получаемый прибавлением к нему регуляризатора, который штрафует модель за слишком большие веса.

Линейная модель называется моделью Лассо, если она реализует следующий функционал ошибки:

$$Q(w,x) = \frac{1}{l}\sum_{i=1}^{l}(\langle w_i, f_i(x_i)\rangle - y_i)^2 + \lambda \cdot \sum_{j=1}^{d}|w_j|, \qquad (7)$$

где сумма модулей весов при признаках – L_1 регуляризатор;

λ – коэффициентом регуляризации (чем он больше, тем ниже сложность модели).

Также, модель Лассо можно определить как минимизацию приведенного выше функционала ошибки:

$$Q(w,x) = \frac{1}{l}\sum_{i=1}^{l}(\langle w_i, f_i(x_i)\rangle - y_i)^2 + \lambda \cdot \sum_{j=1}^{d}|w_j| \to min. \qquad (8)$$

Особенностью регуляризатора L_1 является то, что при его применении, некоторые веса оказываются равными нулю. Другими словами, такой регуляризатор производит отбор признаков и позволяет использовать в модели не все признаки, а только самые важные[5].

Таким образом, все линейные модели обладают рядом важных преимуществ: они быстро обучаются, способны работать с большим количеством объектов и признаков, имеют небольшое количество параметров и легко регуляризуются. При этом у них есть один серьезный недостаток – они могут восстанавливать только линейные зависимости между целевой переменной и признаками.

1.4. Модель «Случайный лес»

Для перехода к рассмотрению модели «Случайный лес», необходимо рассмотреть такие понятия как модель решающих деревьев и композиции алгоритмов. Решающие деревья – семейство алгоритмов, позволяющее восстанавливать нелинейные зависимости произвольной сложности. Такие

деревья, как правило, являются бинарными, где в каждой внутренней вершине записано условие, а в каждом листе дерева прогноз. Условия во внутренних вершинах выбираются крайне простыми. Наиболее частый вариант – проверить, лежит ли значение некоторого признака x_j левее, заданного порога t. Прогноз в листе является вещественным числом, если решается задача регрессии.

Точность предсказания моделей обучающих деревьев во многом зависит от глубины дерева, т.е. максимального, по количеству вершин, пути от корня дерева до одного из его листьев. Далее рассматривается пример решения задачи регрессии с помощью решающих деревьев разной глубины. Пусть решается задача с одним признаком, по которому нужно восстановить значение целевой переменной. Не очень глубокое дерево восстанавливает зависимость примерно, как представлено на рисунке 2.

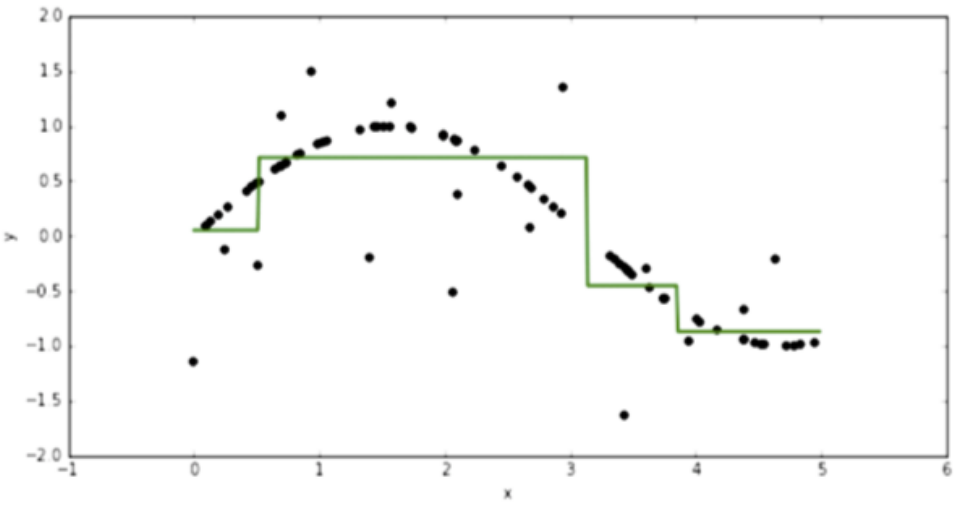

Рисунок 2 – Пример решения задачи регрессии неглубокого дерева

Восстановленная зависимость, показанная зеленным цветом, является кусочно-постоянной, но в целом имеет неплохое качество. При увеличении глубины дерева функция будет иметь следующий вид (рис. 3).

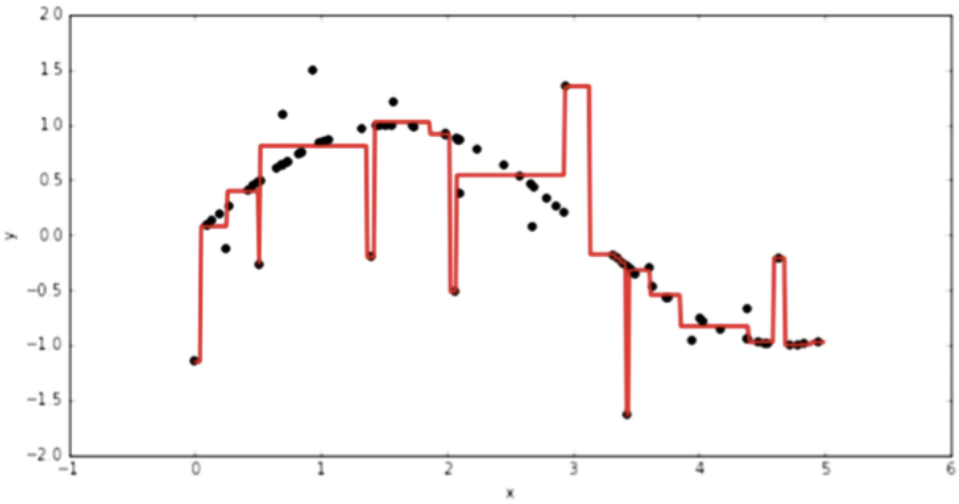

Рисунок 3 – Пример решения задачи регрессии глубокого дерева

На рисунке выше показано, что дерево (красная линия) подогналось под выбросы и его качество уже не будет таким хорошим. Дерево переобучилось из-за слишком большой глубины.

Рассмотрим алгоритм построения дерева. Сначала выбирается корень, который разбивает выборку на две, затем разбивается каждый из потомков этого корня до тех пор, пока этого не будет достаточно. Пусть в некоторую вершину m попало множество X_m объектов из обучающей выборки. Параметры в условии, значение признака x_j не превосходит порог t ($x_j \leq t$), выбираются так, чтобы минимизировать критерий ошибки $Q(X_m, j, t)$, зависящий от этих параметров:

$$Q(X_m, j, t) \rightarrow min. \quad (9)$$

После того, как параметры были выбраны, множество X_m объектов обучающей выборки разбивается на два множества:

$$X_l = \{x \in X_m | (x_j \leq t)\}, \quad (10)$$

$$X_r = \{x \in X_m | (x_j > t)\}, \quad (11)$$

где множество X_l – соответствует левой дочерней вершине;

множество X_r – правой.

Данная процедура продолжается для каждой из дочерних вершин, следовательно, дерево углубляется все больше. Однако, рано или поздно

процесс останавливается, и очередная вершина объявляется листом. Этот момент определяется критерием остановки, который может быть:

- попаданием в вершину только одного объекта обучающей выборки;
- достижением глубиной дерева определенного значения.

При объявлении вершины листом, определяется прогноз, который будет содержаться в данном листе. В задаче регрессии это может быть среднее значение, медиана или другая функция от целевых переменных в листе [6].

Стоит отметить, что в машинном обучении применяется жадный способ построения решающего дерева, т.е. способ при котором выбранный атрибут, дающий некоторое разбиение на подмножества не может быть заменен другим атрибутом, дающим лучшее разбиение.

Критерий ошибки решающего дерева $Q(X_m, j, t)$ записывается следующим образом:

$$Q(X_m, j, t) = \frac{|X_l|}{|X_m|} \cdot H(X_l) + \frac{|X_r|}{|X_m|} \cdot H(X_r), \qquad (12)$$

где $H(X)$ – критерий информативности, в случае регрессии выражающий разброс ответов в X.

Основными недостатками решающих деревьев является то, что они легко переобучаются и сильно изменяются при небольшом изменении обучающей выборки. Решающие деревья сами по себе редко используются на практике, однако популярны их композиции.

Композиция – объединение N алгоритмов $b_1(x), \dots, b_N(x)$ в один, с целью усреднения полученных от них ответов. Для задач регрессии композицией является [7]:

$$a(x) = \frac{1}{N} \cdot \sum_{n=1}^{N} b_n(x). \qquad (13)$$

Для построения композиции необходимо обучить N базовых алгоритмов, причем их обучение не должно проходить на всей обучающей выборке, т.к. в этом случае они все получаются одинаковыми, и их усреднение не имеет смысла. Поэтому обучение базовых алгоритмов происходит на

разных подвыборках обучающей выборки. Одним из популярных подходов построения обучающих повыборок является бутстрап. Он заключается в выборе из обучающей выборки длины l с возвращением l объектов. В результате, получается подвыборка длины l, в которой некоторые объекты повторяются, а часть объектов в нее не попадают.

Часто, в качестве базовых алгоритмов композиции, используются решающие деревья. Одним из способов построения такой композиции называется случайный лес. Пусть необходимо построить случайный лес из N решающих деревьев. На первом шаге, с помощью бутстрапа формируются N случайных подвыборок $\tilde{X}_n$, где $n = 1, \dots, N$. Каждая подвыборка $\tilde{X}_n$ используется как обучающая выборка для построения соответствующего решающего дерева $b_n(x)$. Дерево строится, пока в каждом листе не окажется не более n_{min} объектов. Очень часто деревья строятся до конца (n_{min} = 1), для получения сложных и переобученных решающих деревьев, так как их усреднение дает очень хороший результат. Процесс построения деревьев рандомизирован: на этапе выбора оптимального признака, по которому происходит разбиение, он ищется не среди всего множества признаков, а среди случайного их подмножества. Данное подмножество выбирается каждый раз, когда необходимо разбить очередную вершину. Далее построенные деревья объединяются в композицию [7].

Случайный лес – популярная модель в машинном обучении, которая показывает хороший результат, однако имеет ряд проблем. Первая проблема заключается в том, что обучение глубоких деревьев требует очень много вычислительных ресурсов, особенно в случае большой выборки или большого числа признаков. При ограничении глубины решающих деревьев, случайный лес не способен улавливать сложные закономерности, следовательно, их усреднение не покажет хороший результат. Вторая проблема заключается в том, что процесс построения деревьев является ненаправленным: каждое следующее дерево в композиции никак не зависит от предыдущих. Поэтому для решения сложных задач требуется огромное количество деревьев.

1.5. Модель «Градиентный бустинг решающих деревьев»

Бустинг – подход к построению композиций, в рамках которого базовые алгоритмы строятся последовательно, один за другим, причем, каждый следующий алгоритм строится таким образом, чтобы исправлять ошибки уже построенной композиции [8].

Рассмотрим пример построения композиции с помощью бустинга. Пусть дана задача регрессии, где в качестве ошибки используется среднеквадратичная ошибка:

$$Q(a,x) = \frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i)^2. \quad (14)$$

Для начала обучим первый простой алгоритм (например, неглубокое решающее дерево):

$$b_1(x) = argmin_b \frac{1}{l}\sum_{i=1}^{l}(b(x_i) - y_i)^2, \quad (15)$$

где $argmin_b$ – аргумент минимизации b, при котором достигается минимум функции.

Второй алгоритм должен быть обучен таким образом, чтобы композиция первого и второго алгоритмов $b_1(x_i) + b_2(x_i)$ имела наименьшую из возможных ошибку на обучающей выборке:

$$b_2(x) = argmin_b \frac{1}{l}\sum_{i=1}^{l} b\big((x_i) - (y_i - b_1(x_i))\big)^2. \quad (16)$$

Другими словами, алгоритм $b_2(x)$ должен улучшить качество работы алгоритма $b_1(x)$. Продолжая по аналогии на шаге N очередной алгоритм $b_N(x)$ будет определяться следующим образом:

$$b_N(x) = argmin_b \frac{1}{l}\sum_{i=1}^{l}\left(b(x_i) - (y_i - \sum_{n=1}^{N-1} b_n(x_i))\right)^2. \quad (17)$$

Градиентный бустинг является одним из лучших способов направленного построения композиции на сегодняшний день. В градиентном бустинге строящаяся композиция является суммой, а не усреднением базовых

алгоритмов $b_i(x)$. Это связано с тем, что алгоритмы обучаются последовательно и каждый следующий корректирует ошибки предыдущих.

Пусть задана функция потерь $L(y, z)$, где y – истинный ответ, z – прогноз алгоритма на некотором объекте. В задачах регрессии данная функция потерь имеет вид среднеквадратичной ошибки:

$$L(y, z) = (y - z)^2 \quad (18)$$

Рассмотрим алгоритм построения градиентного бустинга. В начале построения композиции необходимо построить первый базовый не сложный алгоритм. Например, можно использовать алгоритм $b_0(x) = 0$, который всегда возвращает ноль. Далее происходит последовательное обучение базовых алгоритмов. Допустим к некоторому моменту обучены $N - 1$ алгоритмов $b_1(x), \dots, b_{N-1}(x)$, то есть композиция имеет вид:

$$a_{N-1}(x) = \sum_{n=1}^{N-1} b_n(x). \quad (19)$$

Теперь к текущей композиции добавляется еще один алгоритм $b_N(x)$. Этот алгоритм обучается так, чтобы как можно сильнее уменьшить ошибку композиции на обучающей выборке:

$$\sum_{i=1}^{l} L\big(y_i, a_{N-1}(x_i) + b(x_i)\big) \to min. \quad (20)$$

Однако, сначала имеет смысл решить более простую задачу: определить, какие значения $s_1, \dots, s_l$ (расстояния от ответов алгоритма до истинных ответов) должен принимать алгоритм $b_N(x_i) = s_i$ на объектах обучающей выборки, чтобы ошибка на обучающей выборки была минимальной:

$$F(s) = \sum_{i=1}^{l} L\big(y_i, a_{N-1}(x_i) + b(x_i)\big) \to min, \quad (21)$$

где $s = (s_1, \dots, s_l)$ – вектор сдвигов.

Другими словами, необходимо найти такой вектор сдвигов s, который будет минимизировать функцию $F(s)$. Поскольку направление

наискорейшего убывания функции задается направлением антиградиента, его можно принять в качестве вектора s:

$$s = \begin{pmatrix} -L'_z\left(y_1, a_{N-1}(x_1)\right), \\ \cdots \\ -L'_z\left(y_l, a_{N-1}(x_l)\right) \end{pmatrix}. \quad (22)$$

Компоненты вектора сдвигов s, фактически, являются теми значениями, которые на объектах обучающей выборки должен принимать новый алгоритм $b_N(x)$, чтобы минимизировать ошибку строящейся композиции.

$$b_N(x) = argmin \frac{1}{l} \sum_{i=1}^{l} (b(x_i) - s_i)^2. \quad (23)$$

В конце, алгоритм добавляется в композицию:

$$a_N(x) = \sum_{m=1}^{N} b_m(x). \quad (24)$$

Градиентным бустингом решающих деревьев называется модель, где в качестве базовых алгоритмов используются решающие деревья. Как было упомянуто ранее, функция, которую восстанавливает решающее дерево – кусочно-постоянная. Таким образом, решающее дерево разбивает пространство объектов на J областей $R_1, R_2, \ldots, R_J$, в каждой из которых дерево возвращает постоянное предсказание. Пусть b_j – предсказание дерева в области R_j, тогда решающее дерево $b(x)$ можно записать в следующем виде:

$$b(x) = \sum_{j=1}^{J} \left[x \in R_J\right] b_j, \quad (25)$$

где $\left[x \in R_J\right]$ – индикатор того, что объект x попал в область R_J [8].

Таким образом композиция a_N градиентного бустинга решающих деревьев будет выглядеть следующим образом:

$$a_N(x) = a_{N-1}(x) + \sum_{j=1}^{J} \left[x \in R_{Nj}\right] b_{Nj}. \quad (26)$$

Данное выражение можно проинтерпретировать не только как прибавление одного решающего дерева, но и как прибавление J очень простых алгоритмов, каждый из которых возвращает постоянное значение в некоторой области и ноль во всем остальном пространстве. Можно подобрать каждый прогноз b_{Nj}, где N – номер дерева, j – номер листа в этом дереве, таким образом, чтобы он был оптимальным с точки зрения исходной функции потерь:

$$\sum_{i=1}^{l} L\left(y_i, a_{N-1}(x) + \sum_{j=1}^{J}\left[x \in R_{Nj}\right] b_{Nj}\right) \to min. \qquad (27)$$

Итак, структура базового решающего дерева (структура областей R_j) в градиентном бустинге настраивается минимизацией среднеквадратичной ошибки. Потом можно переподобрать ответы в листьях, то есть перенастроить их, так, чтобы они были оптимальны не с точки зрения среднеквадратичной ошибки (с помощью которой строилось дерево), а с точки зрения исходной функции потерь L. Это позволяет существенно увеличить скорость сходимости градиентного бустинга.

Таким образом, градиентный бустинг решающих деревьев устраняет главный недостаток модели случайного леса, а именно делает алгоритм построения деревьев направленным, т.е. при котором каждый последующий базовый алгоритм композиции исправляет ошибки предыдущих. Следовательно для реализации данного алгоритма достаточно использовать неглубокие решающие деревья в качестве базовых алгоритмов, и в гораздо меньшем количестве, чем в случайном лесе, что существенно снижает затраты вычислительных ресурсов.

1.6. Оценка качества построения моделей

После обучения моделей, необходимо проверить точность их предсказаний. Для оценки регрессионных моделей обычно используют метрику, называемую коэффициентом детерминации R^2:

$$R^2 = \left(1 - \sum_{i=1}^{n} \frac{(y_{true} - y_{pred})^2}{\left(y_{true} - mean(y_{true})\right)^2}\right) \cdot 100\%, \quad (28)$$

где y_{true}– правильный ответ;

y_{pred} – ответ предсказанный моделью;

$mean(y_{true})$ – среднее значение среди всех ответов;

n – количество объектов.

Таким образом, коэффициент детерминации – это единица за вычетом дисперсии случайной ошибки модели. Данная метрика принимает значения от 0 до 1, однако может принимать и отрицательные значения. Модель, с отрицательным коэффициентом детерминации абсолютно не восстанавливает искомую зависимость. Такая метрика показывает насколько хорошо модель понимает данные[9].

Для оценки ошибки ответов можно использовать абсолютную среднюю ошибку (MAE), которая находится по формуле:

$$MAE = \frac{1}{n}\left(\sum_{i=1}^{n} |y_{true} - y_{pred}|\right), \quad (29)$$

где y_{true}– правильный ответ;

y_{pred} – ответ предсказанный моделью;

n – количество объектов.

Такая метрика показывает на сколько ошибочны предсказания модели, чем MAE меньше, тем точнее модель[10].

Ранее, было упомянуто, что для оценки качества моделей используется отложенная выборка, которая обычно составляет 30-40% от исходного набора данных. Однако, при разбиении может возникнуть ситуация, при которой объекты разных типов оказались в разных выборках. В таком случае, модель обучится на данных определенного типа, и при тестировании на отложенной выборке не будет отражать реальной точности модели. Во избежание подобной ситуации, следует использовать метод кросс-валидации. Данный метод заключается в случайном разбиении исходной выборки определенного

количества раз, в указанном соотношении, и усреднении полученных результатов на данных выборках.

Общий вывод по разделу

В данном разделе были определены основные понятия и методы машинного обучения. Кроме того, были рассмотрены некоторые разнородные модели машинного обучения, алгоритмы их построения, некоторое математическое обоснование целесообразности использования этих моделей и определены их достоинства и недостатки. В следующей главе рассматривается применение этих моделей на практике, для решения поставленной ранее задачи.

Глава 2. Применение методов машинного обучения

Данный раздел выпускной квалификационной работы призван дать описание методов и подходов, которые были использованы для построения предсказательной модели времени обработки цепочек вычислительных задач. Также, приводится описание предметной области, в рамках которой выполнялось задание.

2.1. Обзор инструментальных средств

Производственная система для которой была выполнена данная работа, реализована с помощью определенной инструментальная базы, поэтому во избежание сложностей при внедрении модели, было решено использовать программные средства, используемые для построения данной системы. В связи с этим, обоснования выбора инструментальных средств не приводится.

Для анализа данных и построения моделей машинного обучения использовался язык программирования Python 3.6. В качестве среды разработки использовалась интерактивная веб оболочка для языка Python – Jupyter Notebook. Данная среда позволяет объединять код, текст и диаграммы в одном файле, и распространять их для других пользователей, что делает ее незаменимым инструментом для выполнения работ, требующих постоянного обсуждения результатов участниками проекта.

Помимо стандартных библиотек языка Python, были использованы такие библиотеки как: Pandas, NumPy, MatPlotLib, Sklearn, FastParquet, SciPy и DateTime. Библиотека NumPy предназначена для работы с большими многомерными массивами данных, а также содержит большое количество математических функция для операций над ними. Pandas предоставляет специальные структуры данных, такие как DataFrame и Series, также операции для манипулирования числовыми таблицами и временными рядами. Библиотека SciPy содержит математические функции для обработки данных, не реализованные в NumPy. Библиотека Sklearn содержит большое количество инструментов машинного обучения, такие как предсказательные модели, метрики оценки качества моделей, и инструменты разбиения данных на

обучающие и тестовые подвыборки. Также были использованы библиотека DateTime, которая предназначена для работы со временным типом данных и библиотека FastParquet, которая импортирует файлы формата parquet и загружает их в DataFrame структуру.

2.2. Описание предметной области

Целью данной ВКР является построение модели, способной предсказывать длительности обработки цепочек вычислительных задач для планирования их загрузки в производственной системе ProdSys2 в рамках эксперимента «ATLAS». Данная система отвечает за обработку данных для групп ученых-физиков, а также за предварительную обработку и анализ данных, используя суперкомпьютеры, облачные и GRID-технологии (распределенные вычисления). GRID-система Prodsys2 содержит более 170 центров обработки данных по всему миру.

Работа с системой происходит следующим образом: пользователи формируют запрос системе, в котором указываются входные данные; преобразования, которые необходимо сделать с данными и форма результата. На основании запроса создаются вычислительные задачи. Такого рода задачи представляют собой набор работ, каждая из которых обрабатывает определенные части данных, называемые событиями. Часто, запросы формируемые пользователями, требуют обработки данных в несколько этапов, в таких случаях создаются цепочки задач. В производственной системе ProdSys2, цепочка - структура связанных задач, где для обработки определенных из них, необходим результат обработки их предшественника. Причем не всегда необходима полная обработка задачи для перехода к следующей, порой достаточно обработать лишь некоторые ее работы. Также, задачи в цепочке могут обрабатываться одновременно, чаще всего это задачи имеющие одного предшественника. Ниже представлен пример структуры цепочки вычислительных задач (рис. 4).

Рисунок 4 – Пример структуры цепочки вычислительных задач

На данном рисунке представлена достаточно сложная цепочка с несколькими задачами, зависящими от результата обработки первой. Однако, не все цепочки имеют такую структуру. Порой встречаются цепочки двух последовательно соединенных задач.

После создания, задача или цепочка задач встает в очередь на обработку с помощью одного из средств системы ProdSys2. Таким образом, не известно время получения результата обработки задач и цепочек задач, поэтому невозможна рациональная организация планирования загрузки ProdSys2 цепочками задач.

2.3. Обработка входных данных

Данные, полученные для проведения данного исследования представляют собой набор признаков, описывающий вычислительные задачи, обработанные в производственной системе ProdSys2 за три месяца с начала 2017 года.

Данный набор содержит уникальный идентификатор для каждой задачи и идентификатор задачи предшественника, для формирования структуры цепочки задач. Задача, не имеющая предшественников, т.е. первая задача цепочки или единственная, имеет в качестве значения атрибута предшественника свой идентификатор. Кроме того данные содержат сведения о времени начала и окончания обработки задачи; день недели в который началась обработка цепочки; продолжительность обработки задач, полученную с помощью предсказательной модели, использующейся в системе ProdSys2. Также данные описывают тип задачи, пользователей и группу пользователей которым принадлежит данная задача, количество работ, вид

обработки, приоритет, статус, место обработки задачи, и технические сведения о средствах, с помощью которых происходила обработка. Описание данных приведено в приложении А.

Таким образом, исходный набор данных дает описание задачам, а не цепочкам задач, поэтому необходима структура, новый набор данных, который будет описывать цепочки вычислительных задач. Для этого был создан метод, способный извлечь все цепочки, содержащие минимально две задачи, из текущего набора данных. Алгоритм метода следующий:

1. Определяются первые задачи всех цепочек, т.е. те задачи, у которых значение идентификатора совпадает со значением идентификатора предшественника.
2. В цикле, для каждой первой задачи, определяются задачи, значения идентификаторов предшественника которых совпадают с идентификатором первой задачи. Найденные задачи сохраняются в DataFrame структуру.
3. Во вложенном цикле, выбираются задачи, значения идентификаторов предшественника которых совпадают с идентификаторами, найденных на предыдущем шаге задач. Текущие задачи, сохраняются в уже существующей таблице с найденными задачами на предыдущих шагах.
4. Процесс продолжается до тех пор пока у набора текущих задач не окажется потомков. На выходе, получается массив DataFrame структур, в каждой из которых содержится одна цепочка. Реализация данного алгоритма представлена в приложении Б.

Для построения нового набора данных, необходимо определить состав признаков. Выбор атрибутов планировалось проводить на двух этапах: до построения моделей, на основе экспертного мнения и после построения моделей, используя результаты о весах признаков, при построении наиболее точной модели. На первом этапе, были выбраны только те признаки, значения которых известны до обработки задач. После было определено, существует

ли потенциальная зависимость между значениями выбранных признаков и временем обработки цепочек задач. Для этого использовались рекомендации эксперта, занимающегося проектированием системы ProdSys2. Ни один из выбранных атрибутов не содержит пустых значений. Набор атрибутов для нового набора данных представлен в приложении В.

Однако, для построения нового набора данных этого не достаточно. Дело в том, что значения одних признаков задач в цепочке могут различаться, поэтому необходимо усреднить данные значения с минимальной потерей информации. Для усреднения значений использовалось: сумма, значения первой задачи цепочки, наиболее часто встречающееся значение и объединение всех значений в одно. Данные методы выбирались в зависимости от характеристики, которую описывает атрибут.

В качестве идентификатора цепочки использовалось значение идентификатора первой задачи. Фактическая продолжительность обработки цепочки($FACT\_DURATION$) находилось путем вычитания из максимального времени окончания обработки задачи в цепочки $\max(END\_TIME)$, минимального времени начала обработки задачи $\min(START\_TIME)$:

$$FACT\_DURATION = \max(END\_TIME) - \min(START\_TIME).$$

Значения предсказанной продолжительности задач суммировались, так как, несмотря на то, что перед началом обработки, известна структура цепочки, остается неизвестным каким образом она будет обрабатываться, т.е. на каком этапе обработки задачи предшественника начнется обработка задачи потомка. Также суммировались: количество файлов, содержащих данные; количество работ и событий; количество задач цепочки и количество ядер обрабатывающих цепочку, т.к. логически именно сумма данных атрибутов характеризует цепочку задач.

Для атрибутов, описывающих пользователя, сформировавшего запрос, было использовано значение для первой задачи в цепочке. К ним относятся: имя пользователя; название его рабочей группы; компания, которой пользователь принадлежит. Данный метод усреднения также использовался

для признаков: идентификатор цепочки; названия проекта, в рамках которого происходит обработка данных; название центра, где назначена обработка задачи; номер задачи в очереди на обработку; день и номер недели, в который начинается обработка задачи. Значения атрибутов, приведенных выше, усреднялись именно таким способом, потому что значения данных признаков либо совпадают, либо логически являются значениями первой задачи для цепочки.

Объединение всех значений атрибутов задач в цепочке проводилось для категориальных признаков: тип задачи, вид обработки, вид архитектуры обрабатывающей машины, источник задачи, доля ресурсов на которую может претендовать пользователь, формирующий запрос. Такое усреднение характерно для категориальных признаков, т.к. именно объединение значений признаков задач является новой категорией, характеризующей цепочку. Для определения приоритета цепочки использовалось наиболее часто встречающееся значение приоритетов среди задач одной цепочки, т.к. значения приоритета могут принимать только определенные числовые значения, поэтому нельзя применять сумму и объединение значений.

С помощью нового набора данных были выявлены: максимальная, минимальная и средняя продолжительности цепочек, которые составили 70, 0,18 и 9 суток соответственно.

Получившийся набор данных содержит несколько категориальных признаков, поэтому такой набор непригоден для обучения моделей, т.к. модели могут работать только с числовыми признаками. В связи с этим категориальные признаки были закодированы в числовые с помощью класса LabelEncoder из библиотеки sklearn. Данный класс принимает все значения категориального признаков и трансформирует каждое из них в число, характеризующее определенное значение.

2.4. Построение предсказательных моделей

После формирования нового набора данных, описывающего цепочки вычислительных задач, можно перейти к построению предсказательных

моделей. В данном разделе приводится описание построения константной модели, Lasso, Случайный лес и Градиентный бустинг решающих деревьев. Также представлена оценка качества предсказаний данных моделей с помощью метрик: коэффициент детерминации R^2, средняя абсолютная ошибка (MAE). Доля тестовых данных от исходного набора составляет 40%. Данные разбивались случайным образом в заданном соотношении 30 раз, на каждом разбиении модель обучалась и определялась ее точность. Полученные результаты усреднялись. Оптимальное количество разбиений определялось эмпирическим путем, минимизируя разброс усредненных ответов. Построение нового набора данных приведено в приложении Г.

2.4.1. Константная модель

Новый набор данных, описывающий цепочки вычислительных задач было решено проверить на наличие постоянной зависимости реального времени обработки цепочек от суммы продолжительностей всех задач в цепочке, предсказанных моделью, использующейся в системе ProdSys2. Такое решение было принято на основе анализа диаграмм Ганта нескольких цепочек. Реализация функции построения диаграмм Ганта приведена в приложении Д. Пример такой диаграммы представлен на рисунке 5.

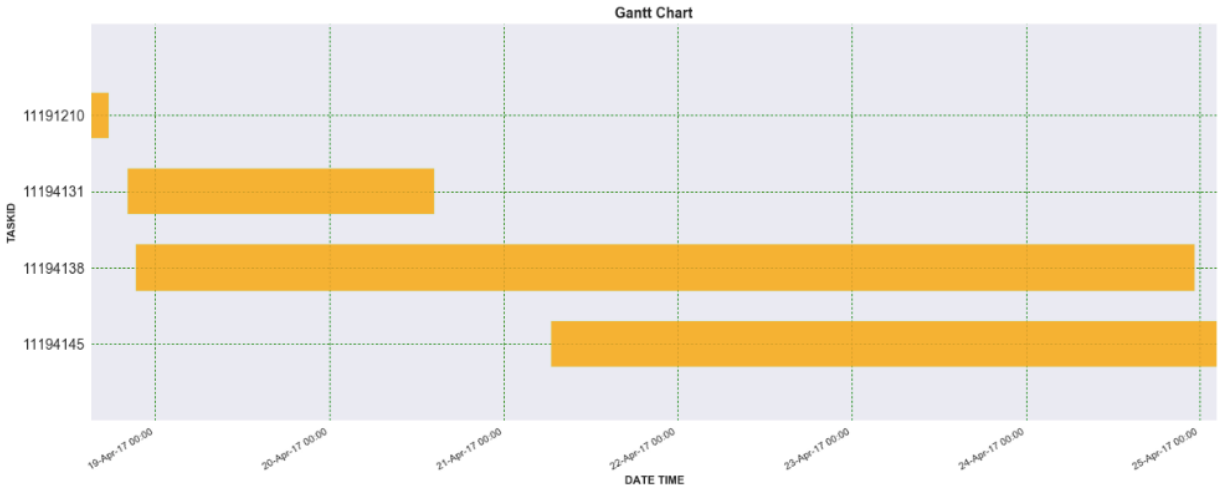

Рисунок 5 – Пример диаграммы Ганта цепочки вычислительных задач

В результате анализа диаграмм Ганта цепочек вычислительных задач, было выявлено, что продолжительности обработки похожих по структуре диаграмм цепочек примерно равны. В связи с этим была построена

константная модель, которая возвращает среднее значение сумм продолжительностей обработки задач. Таким образом, обучающая выборка состоит всего из одного атрибута – суммы продолжительности задач цепочки, в качестве меток используется фактическое время обработки цепочек. Для построения такой модели использовался класс DummyRegressor из библиотеки sklearn. Реализация построения данной модели приведена в приложении Е. На рисунке ниже изображено графическое представление точности предсказаний данной модели (рис. 6).

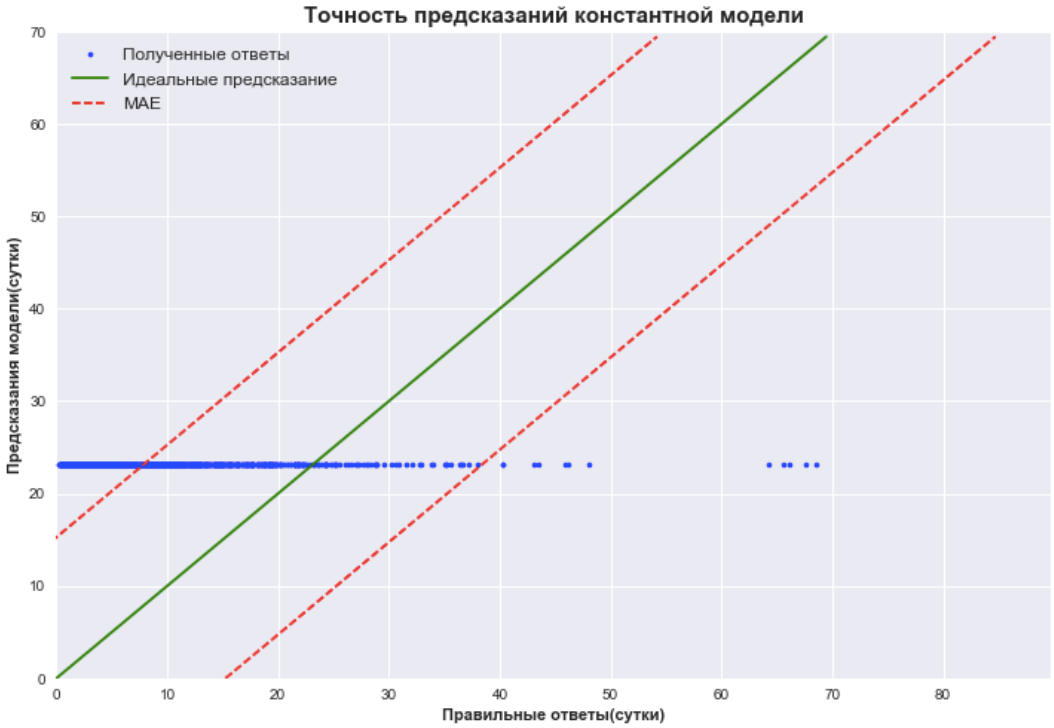

Рисунок 6 – Точность предсказаний константной модели

На данном рисунке, синими точками, представлена зависимость предсказаний модели от истинных значений. В связи с тем, что предсказания модели для любого объекта одинаковы, данная зависимость приняла форму прямой. Зеленой линией показана идеальная зависимость предсказаний от ответов (предсказания верные с точностью 100%), таким образом чем больше точек располагаются вдоль линии и чем они ближе к ней, тем точнее предсказания модели. Красной пунктирной линией показана средняя абсолютная ошибка предсказаний, чем меньше область ограниченная

пунктирными линиями, тем меньше ошибка предсказаний. Данная область построена с помощью метрики средней абсолютной ошибки(MAE).

В результате построения модели получили: коэффициент детерминации равный -0,026% и среднюю абсолютную ошибку в размере 14,5 суток. Отрицательный коэффициент детерминации говорит о практической непригодности такой модели, также как и слишком большое значение MA-метрики.

2.4.2. Модель Лассо

В связи с тем, что предыдущая модель оказалась непригодной для практического применения, было решено построить линейную регуляризуемую модель Лассо. Для построения данной модели использовались все признаки набора данных, кроме идентификатора цепочки. Построение осуществлялось с помощью класса Lasso из библиотека sklearn. Реализация построения модели Лассо представлена в приложении Ж. На рисунке 7 представлен результат построения данной модели.

Рисунок 7 – Точность предсказаний модели Лассо

На рисунке показано, что гораздо больше точек разместились вдоль зеленой линии. Также уменьшилась область ошибки предсказаний, что

говорит об уменьшении их ошибки. Коэффициент детерминации данной модели составил 47%, средняя абсолютная ошибка 4 дня. Также, на рисунке показано, что некоторые предсказания получились отрицательными. Такая ситуация невозможна, так как предсказывается время обработки цепочек вычислительных задач, которое не может принимать отрицательные значения. Из этого следует непригодность такой модели. Во избежание отрицательных ответов, необходимо установить ограничение на отрицательные значения коэффициентов при значениях признаков. Точность модели Лассо с ограничениями на коэффициенты представлена на рисунке 8.

Рисунок 8 – Точность предсказаний модели Лассо с ограничениями

После ограничения коэффициентов, качество модели значительно ухудшилось. На рисунке видно, что теперь значительно меньше точек располагается вдоль прямой, характеризующей идеальные предсказания. Облако точек стремится к прямой, не совпадающей с зеленой линией. Также заметно увеличилась ошибка предсказаний. Коэффициент детерминации такой модели составил 14,37%, среднее абсолютное отклонение 5,5 суток.

Модель Лассо показала лучший результат, чем константная, однако ее точность слишком низка для внедрения в производственную систему ProdSys2.

2.4.3. Модель Случайный лес

После построения модели Лассо, стало ясно, что линейные модели не способны восстанавливать искомую закономерность, поэтому было решено построить модель, способную восстанавливать сложные нелинейные зависимости. Как было упомянуто ранее, одной из таких моделей является Случайный лес. Для построения этой модели использовался построенный набор данных, со всеми признаками, кроме идентификатора цепочки. Для построения модели применялся класс RandomForestRegressor из библиотеки sklearn. Построение модели Случайный лес представлено в приложении И. На рисунке ниже показан результат построения данной модели (рис. 9).

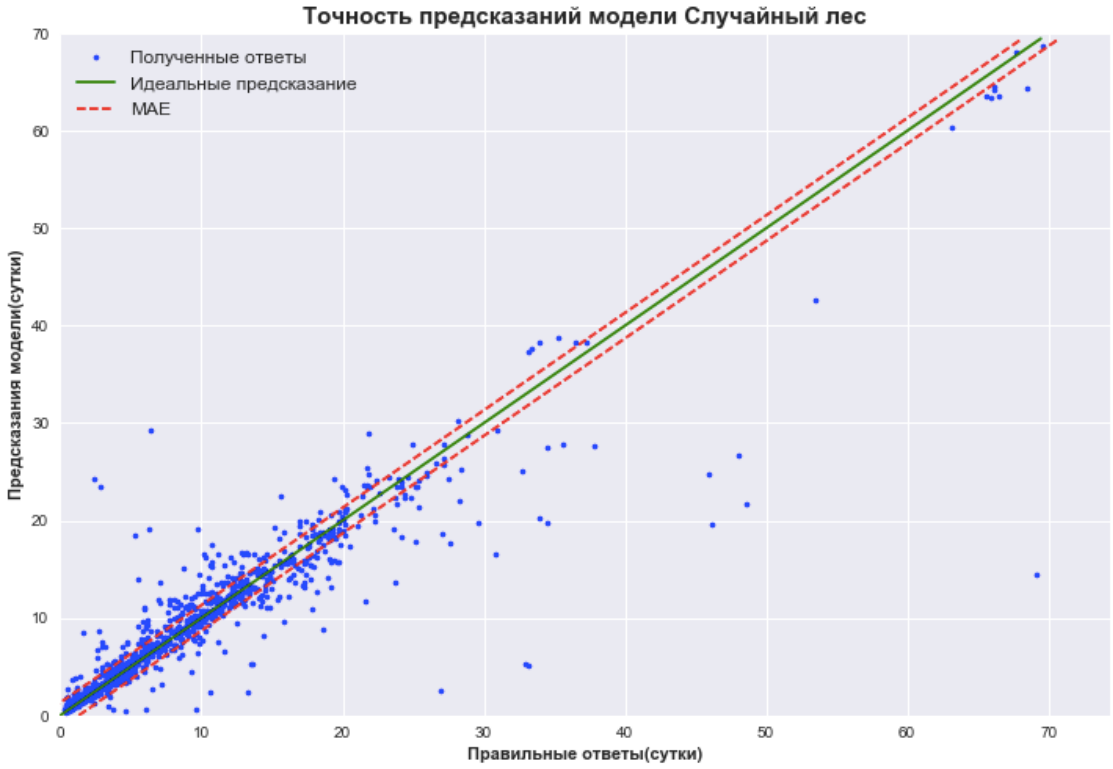

Рисунок 9 – Точность предсказаний модели Случайный лес

На рисунке выше показано, что Случайный лес дает более точные предсказания, чем предыдущие две модели. Облако точек вытянулось вдоль зеленой прямой, а область ошибки значительно уменьшилась. Метрика R^2

данной модели оказалась равной 86,9%, средняя абсолютная ошибка составила 1,3 суток, что при средней продолжительности цепочек в 9 суток является хорошим результатом.

Данная модель показала хороший результат, и, согласно мнениям экспертов системы ProdSys2, является приемлемой для внедрения в производственную систему ProdSys2.

2.4.4. Модель Градиентный бустинг решающих деревьев

В теоритической части данной работы говорится о том, что решающие деревья в модели Случайный лес строятся независимо друг от друга. Модель Градиентный бустинг решающих деревьев делает процесс построения деревьев направленным, каждое последующее дерево строится таким образом, чтобы исправлять ошибки уже построенных. В связи с тем, что случайный лес показал хорошую точность предсказаний, было решено проверить, каким будет точность Градиентного бустинга решающих деревьев. Для построения модели применялся класс GradientBoostingRegressor из библиотеки sklearn Реализация построения модели Градиентного бустинга решающих деревьев представлена в приложении К. На рисунке ниже представлены результаты построения данной модели (рис. 10).

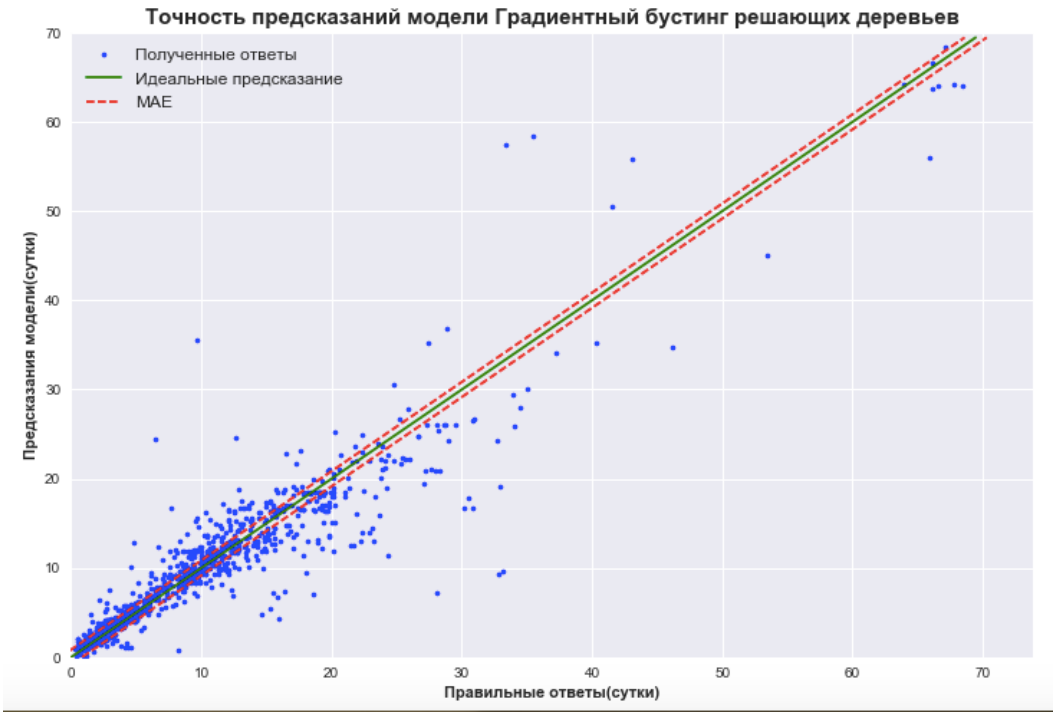

Рисунок 10 – Точность предсказаний модели Градиентного бустинга решающих деревьев

Приведенный график, на рисунке выше, напоминает график точности случайного леса (рисунок 2.6), но на данном рисунке большее количество точек располагается у зеленой прямой. Коэффициент детерминации данной модели составил 87,8%, средняя абсолютная ошибка оказалась равной 0,84 суток.

Таким образом, модель Градиентного бустинга решающих деревьев показала лучший результат из всех рассмотренных моделей. Определим веса признаков данной модели с помощью функции feature_importances_. Результаты представлены на рисунке 11.

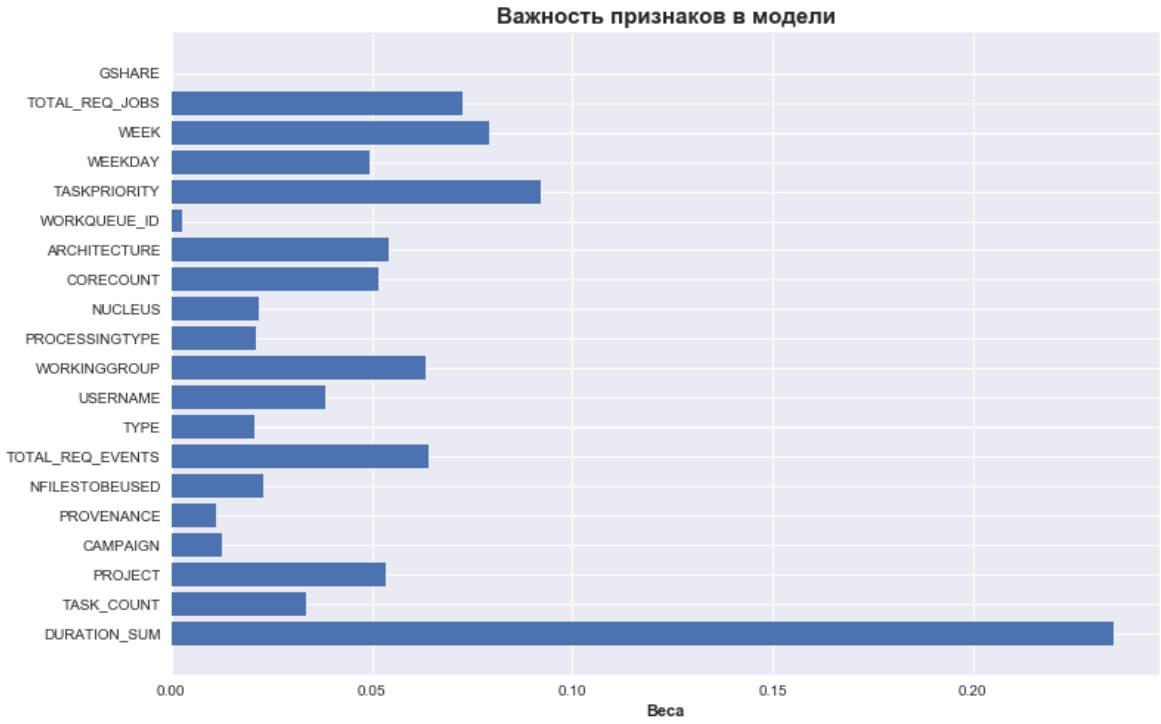

Рисунок 11 – Веса признаков модели Градиентного бустинга решающих деревьев.

На данном рисунке показано, что атрибут GSHARE не влияет на точность модели, поэтому данный признак можно исключить из текущего набора данных.

Общий вывод по разделу

В данном разделе были рассмотрены некоторые особенности предметной области, в рамках которой выполнялась данная работа. Также был изучен исходный набор данных, на основе которого строились предсказательные модели. Лучшей моделью оказался Градиентный бустинг решающих деревьев, имеющий точность предсказаний 87,8%, и среднюю абсолютная ошибка в размере 0,84 суток. Кроме того был выявлен оптимальный набор признаков для использования такой данной. Согласно мнениям экспертов системы ProdSys2, такой результат является хорошим для поставленной задачи, и модель пригодна для внедрения в производственную систему ProdSys2. На данный момент, проводятся подготовительные работы по внедрению полученной предсказательной модели. Данная модель будет внедрена в систему ProdSys2 в течении следующего полугода.

Глава 3. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Целью данного раздела является экономическое обоснование проекта, предлагаемого в выпускной квалификационной работе.

Задачи раздела:

- проведение SWOT-анализа для определения стратегий реализации и дальнейшего развития проекта;
- построения линейного графика планирования НТИ для согласования и организации занятости участников проекта;
- расчет себестоимости разработки для определения экономической эффективности проекта.

3.1 SWOT-анализ

Для исследования внешней и внутренней среды проекта был проведен SWOT-анализ, показывающий сильные и слабые стороны разрабатываемого проекта.

Таблица 1 – SWOT-анализ

	Сильные стороны научно-исследовательского проекта: С1. Простота эксплуатации. С2. Повышение производительности труда. С3. Гарантия получения результата обработки данных. С4. Техническая поддержка после внедрения проекта.	**Слабые стороны научно-исследовательского проекта:** Сл1. Узкая направленность. Сл2. Потребность больших вычислительных мощностей. Сл3. Существует вероятность получения неверных предсказаний
Возможности: В1. Спрос на улучшение разработки В2. Устойчивый спрос инициаторов проекта на разработку	Техническая поддержка проекта после внедрения и дальнейшее развитие разработки.	Работа по увеличению точности предсказаний работ цепочек вычислительных задач.

Угрозы: У1. Нехватка вычислительных ресурсов. У2. Отсутствие дополнительного спроса У3. Отсутствие возможности расширения функционала. У4. Невозможность выхода на новых потребителей	Ориентация на потребности инициаторов проекта.	Рассмотрение менее ресурсоемких моделей.

Таблица 2 – Интерактивная матрица проекта

		Сильные стороны проекта				**Слабые стороны проекта**		
		С1	С2	С3	С4	Сл1	Сл2	Сл3
Возможности проекта	В1	+	+	0	+	-	0	+
	В2	+	+	+	+	-	0	+
Угрозы проекта	У1	-	0	0	0	-	+	0
	У2	-	-	-	-	+	0	0
	У3	-	0	-	-	+	-	-
	У4	-	-	-	-	+	0	-

3.2 Планирование научно-исследовательских работ

3.2.1. Организация и планирование работ

Для реализации конкретного проекта прежде всего необходимо определить и согласовать способ организации занятости каждого из его участников, что можно достичь с помощью построения линейного графика работ. Для этого нужно выявить полный перечень проводимых работ, их продолжительность и исполнителей. Полученные данные приведены в таблице 3.

Таблица 3 – Перечень работ и продолжительность их выполнения

Этапы работы	**Исполнители**	**Загрузка исполнителя НР, %**	**Загрузка исполнителя И, %**
Постановка целей и задач, получение исходных данных	НР, Б	90	10
Составление и утверждение ТЗ	НР, Б	100	10
Подбор и изучение материалов по тематике	НР, Б	40	100
Разработка календарного плана	НР, Б	100	10

Выбор обучающих моделей	НР, Б	80	50
Предварительная обработка данных, обучение моделей и тестирование	НР, Б	20	100
Анализ результатов	НР, Б	20	100
Оформление расчетно-пояснительной записки	Б		100
Подведение итогов	НР, Б	45	100

В таблице представлена загруженность исполнителей проекта на каждом этапе выполнения проекта.

3.2.2 Продолжительность этапов работ

Расчет продолжительности этапов работ был определен опытно-статистическим экспертным методом.

На первом шаге, было найдено значение ожидаемой трудоемкости, по формуле:

$$t_{\text{ож}} = \frac{3 \cdot t_{min} + 2 \cdot t_{max}}{5}, \qquad (30)$$

где t_{min} – минимально возможная трудоемкость выполнения, чел.-дн.;

t_{max}– максимально возможная трудоемкость выполнения, чел.-дн.

По примеру, приведенному ниже, рассчитываются значения ожидаемой трудоемкости для каждого этапа.

$$t_{\text{ож1}} = \frac{3 \cdot 5 + 2 \cdot 8}{5} = 6{,}2 \text{ дн.}$$

После, исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях по формуле ниже.

$$T_{\text{РД}} = \frac{t_{\text{ож}}}{K_{\text{ВН}}} \cdot K_{\text{Д}}, \qquad (31)$$

где $t_{\text{ож}}$ – ожидаемая трудоемкость выполнения работы, чел.-дн.

$K_{\text{ВН}}$– коэффициент выполнения работ, учитывающий влияние внешних факторов на соблюдение предварительно определенных длительностей, в частности, возможно $K_{\text{ВН}} = 1$;

$K_Д$ – коэффициент, учитывающий дополнительное время на компенсацию непредвиденных задержек и согласование работ ($K_Д$ = 1–1,2). Коэффициент дополнительного времени был выбран равным 1,1.

Ниже показан расчет продолжительности первой работы для научного руководителя (НР). Аналогичным образом рассчитываются остальные значения продолжительности работ.

$$T_{РД1} = \frac{6,2}{1} \cdot 1,1 = 6,19 \text{ чел.} - \text{дн.}$$

Продолжительность этапа в календарных днях вычисляется по следующей формуле:

$$T_{КД} = T_{РД} \cdot T_К, \quad (32)$$

где $T_{КД}$ – продолжительность выполнения этапа в календарных днях;

$T_К$– коэффициент календарности, с помощью которого можно перейти от длительности работ в рабочих днях к соответствующим длительностям в календарных днях. Коэффициент календарности рассчитывается по формуле:

$$T_К = \frac{T_{КАЛ}}{T_{КАЛ} - T_{ВД} - T_{ПД}} = \frac{365}{365 - 52 - 10} = 1,205,$$

где $T_{КАЛ}$– календарные дни ($T_{КАЛ}$= 365);

$T_{ВД}$– выходные дни ($T_{ВД}$= 52);

$T_{ПД}$– праздничные дни ($T_{ПД}$= 10).

Пример расчета $T_{КД}$ для научного руководителя на первом этапе работ:

$$T_{КД1} = 6,19 \cdot 1,205 = 7,4 \text{ чел.} - \text{дн.}$$

В таблице 4 представлены продолжительности этапов работ и их трудоемкости по исполнителю, на каждом этапе.

Таблица 4 – Трудозатраты на выполнение проекта

№ этапа	Вид этапа	Исполнители	Продолжительность работ, дни			Трудоемкость работ по исполнителям чел.- дн.			
						$T_{РД}$		$T_{КД}$	
			t_{min}	t_{max}	$t_{ож}$	НР	И	НР	И
1	Постановка целей и задач, получение исходных данных	НР, Б	5	8	6,2	6,19	0,68	7,4	0,82
2	Составление и утверждение ТЗ	НР, Б	4	5	4,4	4,84	0,48	5,83	0,58
3	Подбор и изучение материалов по тематике	НР, Б	18	30	22,8	10,03	25,08	12,09	30,22
4	Разработка календарного плана	НР, Б	1	2	1,4	1,54	0,15	1,86	0,19
5	Выбор обучающих моделей	НР, Б	2	4	2,8	2,46	1,54	2,97	1,86
6	Предварительная обработка данных, обучение моделей и тестирование	НР, Б	10	15	12	6,6	13,2	7,95	15,91
7	Анализ результатов	НР, Б	4	8	5,6	1,23	6,16	1,48	7,42
8	Оформление расчетно-пояснительной записки	Б	15	17	15,8	0	17,38	0	20,94
9	Подведение итогов	НР, Б	3	6	4,2	2,08	4,62	2,51	5,57
	Итого:				**75,2**	**34,93**	**69,3**	**42,09**	**83,51**

С помощью округленных до большего значений трудоемкости этапов по исполнителям $T_{КД}$ был построен линейный график осуществления проекта (таблица 5).

Таблица 5 – Трудозатраты на выполнение проекта

№ работ	Вид работ	Исполнители	$T_{кд}$	Продолжительность выполнения работ								
				февраль			март			апрель		
				10	20	30	10	20	30	10	20	30
1	Постановка целей и задач, получение исходных данных	НР, Б	8; 1									
2	Составление и утверждение ТЗ	НР, Б	6; 1									
3	Подбор и изучение материалов по тематике	НР, Б	13; 31									
4	Разработка календарного плана	НР, Б	2; 1									
5	Выбор обучающих моделей	НР, Б	3; 2									
6	Предварительная обработка данных, обучение моделей и тестирование	НР, Б	8; 16									
7	Анализ результатов	НР, Б	2; 8									
8	Оформление расчетно-пояснительной записки	Б	21									
9	Подведение итогов	НР, Б	3; 6									

- НР - Б

3.3. Бюджет научно-технического исследования

3.3.1. Расчет материальных затрат

При написание ВКР не использовалось определенное материальное оборудование. Так как оборудование и лицензии на платное ПО предоставлялись университетом, и использовалось свободно распространяемо ПО, то для данной статьи расходов расчет не проводился.

3.3.2. Расчет амортизационных расходов

Расчета амортизации оборудования проводился по следующей формуле:

$$С_{АМ} = \frac{Н_А \cdot Ц_{об} \cdot t_{рф} \cdot n}{F_д}, \quad (33)$$

где $Н_А$ – годовая норма амортизации единицы оборудования;

$Ц_{об}$ – балансовая стоимость единицы оборудования с учетом ТЗР;

$F_д$ – действительный годовой фонд времени работы соответствующего оборудования;

$t_{рф}$ – фактическое время работы оборудования в ходе выполнения проекта, учитывается исполнителем проекта;

n – число задействованных однотипных единиц оборудования.

Расчет амортизации для персонального компьютера:

$$С_{АМк} = \frac{0{,}33 \cdot 67000 \cdot 722{,}7}{2384} = 6702{,}56 \text{ руб.}$$

Таблица 6 – Расчет амортизационных расходов

Наименование оборудования	$F_д$	$t_{рф}$	$Н_А$	$Ц_{об}$	$С_{АМ}$
Персональный компьютер	2384	722,7	0,33	67000	6702,56
Итого:					6702,56

3.3.3. Расчет затрат на электроэнергию

Данная статья расходов является затратами на электроэнергию, потраченную при эксплуатации оборудования, в ходе выполнения проекта. Для расчета затрат была использована формула:

$$С_{эл.об.} = P_{об} \cdot t_{об} \cdot Ц_э, \quad (34)$$

где $P_{об}$ – мощность, потребляемая оборудованием, кВт. $P_{об}$ для данной модели персонального компьютера равна 0,045 кВт.;

$Ц_э$ – тариф на 1 кВт·час; для Томска $Ц_э$ = 4,21 руб./квт·час (с НДС).

$t_{об}$ – время работы оборудования, час., учитывается исполнителем проекта.

Расчет затрат на электроэнергию для персонального компьютера:

$$С_{эл.об.} = 0{,}045 \cdot 722{,}7 \cdot 4{,}21 = 136{,}92 \text{ руб.}$$

Расчета затрат на электроэнергию для технологических целей приведен в таблице 7.

Таблица 7 – Расчет затрат на электроэнергию

Наименование оборудования	**Потребляемая мощность $P_{об}$, кВт**	**Время работы оборудования $t_{об}$, час**	**Затраты $С_{эл.об.}$, руб.**
Персональный компьютер	0,045	722,7	136,92
Итого:			**136,92**

3.3.4 Основная заработная плата исполнителей темы

В данном разделе была рассчитана основная заработная плата работников, непосредственно занятых выполнением НТИ, (включая премии, доплаты).

На первом этапе рассчитывается месячный должностной оклад работника:

$$З_м = З_{тс} \cdot k_р \quad (35)$$

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_р$ – районный коэффициент, равный 1,3 (для Томска).

В качестве примера, ниже представлен расчет $З_м$ для научного руководителя. $З_{тс}$ для младшего научного сотрудника равен 14874,45.

$$З_{м1} = 14874{,}45 \cdot 1{,}3 = 19336{,}79 \text{ руб.}$$

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot М}{F_д} \quad (36)$$

где $З_м$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года:

при отпуске в 24 раб. дня. М =11,2 месяца, 6-дневная неделя;

$F_д$ – действительный годовой фонд рабочего времени научно-технического персонала, раб. дн.

Таблица 8 – Баланс рабочего времени

Показатели рабочего времени	**Руководитель**	**Студент**
Календарное число дней	365	365
Количество нерабочих дней - выходные дни - праздничные дни	66	66
Потери рабочего времени - отпуск - невыходы по болезни	24	24
Действительный годовой фонд рабочего времени	275	275

Расчет среднедневной заработной платы для научного руководителя:

$$З_{дн1} = \frac{19336{,}79 \cdot 11{,}2}{275} = 787{,}53 \text{ руб.}$$

Основная заработная плата ($З_{осн}$) рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot T_p, \quad (37)$$

где T_p – продолжительность работ, выполняемых научно-техническим работником, раб. дн.

Расчет основной заработной платы для научного руководителя:

$$З_{осн} = 787{,}53 \cdot 24{,}9 = 19609{,}49 \text{ руб.}$$

Так как бакалавр, являясь студентом ТПУ, получает фиксированные стипендиальные выплаты, то расчет основной заработной платы проводится только для научного руководителя. Расчёт основной заработной платы приведён в таблице 9.

Таблица 9 – Расчет основной заработной платы

Участники	Разряд	$З_{тс}$, руб.	$k_р$	$З_м$, руб	$З_{дн}$, руб.	$Т_р$, раб. дн.	$З_{осн}$, руб.
Научный руководитель	Мл. науч. сотр.	14874,45	1,3	19336,79	787,53	20,8	19609,49
Бакалавр							2275
Итого:							**21884,49**

3.3.5 Дополнительная заработная плата исполнителей темы

Расчет дополнительной заработной платы ведется по следующей формуле:

$$З_{доп} = k_{доп} \cdot З_{осн}, \quad (38)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы (на стадии проектирования принимается равным 0,12 – 0,15). Примем коэффициент равный 0, 12.

Так как бакалавр не получал дополнительной заработной платы, расчет проводился только для научного руководителя.

$$З_{доп1} = 0{,}12 \cdot 19609{,}49 = 2353{,}14 \text{ руб.}$$

Таблица 10 – Расчет основной заработной платы

Участники	Основная заработная плата, руб.	$k_{доп}$	Дополнительная заработная плата, руб.
НР	19609,49	0,12	2353,14
Итого:			**2353,14**

3.3.6 Отчисления во внебюджетные фонды (страховые отчисления)

Затраты во внебюджетные фонды включают в себя отчисления в пенсионный фонд, социальное и медицинское страхование. Доля отчислений для ТПУ составляет 27,1% от полной заработной платы по проекту.

$$З_{внеб.} = 0{,}271 \cdot (З_{осн.} + З_{доп.}) \quad (39)$$

Расчет $З_{внеб.}$ для научного руководителя представлен ниже. Для бакалавра расчет не проводился, так как стипендия относитится к выплатам, не облагающимся взносам во внебюджетные фонды.

$$З_{внеб1} = 0{,}271 \cdot (19609{,}49 + 2353{,}14) = 5951{,}87 \text{ руб.}$$

Таблица 11 – Расчет отчислений во внебюджетные фонды

Исполнитель	**$З_{внеб.}$**
НР	5951,87
Итого:	**5951,87**

3.3.7 Расчет общей себестоимости разработки

В данном разделе был произведен расчет общей себестоимости разработки, путем суммирования результатов расчета затрат, приведенных выше.

Таблица 12 – Расчет общей себестоимости разработки

Статья затрат	**Условное обозначение**	**Сумма, руб**
Амортизационные отчисления	$С_{АМ}$	6702,56
Расходы на электроэнергию	$С_{эл.об.}$	136,92
Основная заработная плата	$З_{осн.}$	21884,49
Дополнительная заработная плата	$З_{доп.}$	2353,14
Отчисления в социальные фонды	$З_{внеб.}$	5951,87
Итого:		**45690,85**

В соответствии с таблицей 12 основную долю себестоимости разработки занимают расходы на основную заработную плату исполнителей. Данный результат характерен для научно-исследовательских работ, без применения дополнительных материалов.

Общий вывод по разделу

В ходе выполнения данного раздела ВКР были выявлены сильные и слабые стороны разработки, также проанализированы факторы внешней среды. Это позволило сформировать стратегии дальнейшей работы с проектом, которые показывают заинтересованность инициаторов проекта в его реализации и дальнейшем развитии. Кроме того, была подсчитана общая стоимость реализации программного модуля, которая составила 45690,85 руб. Разработка решает проблему рационального планирования загрузки вычислительных задач, из чего можно сделать вывод об необходимости

реализации проекта, вне зависимости от его себестоимости, так как инициаторами проекта не было установлено ограничений на сумму его реализации.

Глава 4. Социальная ответственность

Выпускная квалификационная работа по применению методов машинного обучения для предсказания длительности выполнения цепочек вычислительных задач выполнялась в одной из лабораторий Кибернетического центра ТПУ. Лаборатория представляет собой помещение, рассчитанное на десять рабочих мест, оборудованное персональными компьютерами и офисной мебелью. В лаборатории находятся два окна, имеется кондиционер, используется искусственное и естественное освещение.

Целью данного раздела является выявление и анализ вредных и опасных факторов, которые могут возникнуть в рассматриваемом помещении и оказать негативное воздействие на работника, а также рассмотрение мер защиты от них. Также будут рассмотрены вопросы техники безопасности, охраны окружающей среды и пожарной профилактики.

Разработка программного модуля, сама по себе, никоим образом не оказывает негативного воздействия на общество и окружающую среду, однако в ходе его реализации могут образоваться твердые отходы. В связи с этим будет рассмотрен вопрос утилизации отходов.

4.1 Производственная безопасность.

Все опасные и вредные производственные факторы, оказывающие какое либо негативное влияние на организм человека подразделяются на следующие группы воздействия:

- физического;
- химического;
- биологического;
- психофизиологического.

В связи с тем, что биологические и химические факторы не оказывает существенного влияния на организм исполнителя данной работы, далее подробно будут рассмотрены только физические факторы.

Для наглядного представления всех вредных и опасных факторов, которые могут быть выявлены в рассматриваемой лаборатории, ниже

приведена их классификация с соответствующими нормативными документами (таблица 13).

Таблица 13 – Классификация вредных и опасных факторов

Источник фактора, наименование видов работ	Факторы (ГОСТ 12.0.003-2015 ССБТ)		Нормативные документы
	Вредные	Опасные	
1. Работа с компьютером и орг. техникой.	1. Отклонение показателей микроклимата; 2. Превышение уровней шума и вибрации; 3. Превышение уровня электромагнитных излучений; 4. Отклонение показателей освещенности рабочего места.	1. Электрический ток. 2. Взрывоопасные и пожароопасные материалы	1. ГОСТ 12.0.003-2015 2. СанПиН 2.2.4.548-96 3. ГОСТ 12.1.006–84 4. СанПиН 2.2.1/2.1.1.1278-03 5. СанПиН 2.2.2/2.4.2732-10 6. СНиП 2.04.05-91 7. ГОСТ 12.1.003-2014 8. СанПиН 2.2.4.1191-03 9. СанПиН 2.2.4.1340-03 10. СН 2.2.4/2.1.8.562-96 11. НПБ 105-95

4.1.1 Микроклимат рабочего помещения

Для анализа микроклимата на рабочем месте необходимо оценить следующие параметры: температуру, скорость движения и влажность воздуха. Эти факторы определяют микроклимат рабочих помещений и оказывают влияние на организм человека, поэтому необходимо соблюдение соответствия данных показателей санитарным нормам.

Оптимальные микроклиматические условия - установленные по критериям оптимального теплового и функционального состояния человека характеристики микроклимата. Они обеспечивают общее и локальное ощущение теплового комфорта в течение 8-часовой рабочей смены при минимальном напряжении механизмов терморегуляции, не вызывают отклонений в состоянии здоровья, создают предпосылки для высокого уровня работоспособности и являются предпочтительными на рабочих местах.

Так как работа исполнителя проводилась сидя за компьютером и сопровождалась незначительными физическими нагрузками, то в соответствии с СанПиН 2.2.4.548-96 [13], данный вид работ относится к

легкой физической работе (Iа). Ниже приведены оптимальные значения характеристик микроклимата (таблица 14).

Таблица 14 – Оптимальные значения характеристик микроклимата

Период года	Категория работ по уровню энергозатрат, Вт	Темпе-ратура воздуха, °С	Темпе-ратура поверх-ностей, °С	Относитель-ная влаж-ность воз-духа, %	Скорость движения воздуха, м/с
Холодный	Iа (до 139)	22-24	21-25	60-40	0,1
Теплый	Iа (до 139)	23-25	22-26	60-40	0,1

Допустимые микроклиматические условия - установленные по критериям допустимого теплового и функционального состояния человека на период 8-часовой рабочей смены характеристики микроклимата. Они не вызывают повреждений или нарушений состояния здоровья, но могут приводить к возникновению общих и локальных ощущений теплового дискомфорта, напряжению механизмов терморегуляции, ухудшению самочувствия и понижению работоспособности. Допустимые величины показателей микроклимата устанавливаются в случаях, когда по технологическим требованиям, техническим и экономически обоснованным причинам не могут быть обеспечены оптимальные величины. В таблице 15 представлены допустимые значения характеристик микроклимата.

Таблица 15 – Допустимые значения характеристик микроклимата

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С		Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с	
		диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более**
Холодный	Iа (до 139)	20,0-21,9	24,1-25,0	19,0-26,0	15-75	0,1	0,1
Теплый	Iа (до 139)	21,0-22,9	25,1-28,0	20,0-29,0	15-75	0,1	0,2

Параметры микроклимата данного рабочего помещения, регулируются системой центрального отопления, а также приточно-вытяжной вентиляцией через окно. В лаборатории отсутствуют приборы для измерения вышеприведенных параметров микроклимата.

4.1.2 Производственные шумы

Шум - звуковые колебания в диапазоне слышимых частот, способные оказать вредное воздействие на безопасность и здоровье работника.

Шум на рабочем месте оказывает раздражающее влияние на работника, повышает его утомляемость, а при выполнении задач, требующих внимания и сосредоточенности, способен привести к росту ошибок и увеличению продолжительности выполнения задания. Длительное воздействие шума влечет тугоухость работника вплоть до его полной глухоты.

Внезапные шумы высокой интенсивности, даже кратковременные (взрывы, удары и т.п.), могут вызвать как острые нейросенсорные эффекты (головокружение, звон в ушах, снижение слуха), так и физические повреждения (разрыв барабанной перепонки с кровотечением, поражения среднего уха и улитки).

В рассматриваемом рабочем помещении основными источниками шума являются персональные компьютеры. Согласно требованиям СН 2.2.4/2.1.8.562-96 [14], в рабочих помещениях при выполнении легкой физической работы, уровни шума на рабочих местах не должны превышать предельно допустимых значений, т.е. 80 дБА.

4.1.3 Электромагнитное излучение

Во время работы с персональным компьютером организм человека подвергается воздействию электромагнитного излучения, которое может оказать негативное влияние.

Электромагнитное поле (ЭМП), создаваемое ПК представляет собой совокупность электрического и магнитного полей, которые при определенных условиях могут порождать друг друга. Воздействие такого поля зависит от

величин напряженности электрического поля (E), магнитного поля (H), размера облучаемого тела, частоты колебаний и потока энергии.

Согласно СанПиН 2.2.4.1191-03 [15] время пребывания человека в зоне воздействия электромагнитного излучения зависит от значений напряженности электрического поля. Количество часов допустимого пребывания в рабочей зоне определяется по следующей формуле:

$$T = \frac{50}{E} - 2 \qquad (40)$$

К примеру, работа в условиях облучения электрическим полем с напряженностью 20–25 кВ/м не может продолжается более 10 минут. При напряженности до 5 кВ/м разрешается присутствие людей в течение 8 часов.

В таблице 16 представлено допустимое время пребывания человека в зоне воздействия электромагнитного излучения в зависимости от напряженности поля и магнитной индукции.

Таблица 16 – Допустимое время пребывания в зоне воздействия ЭМП

Время воздействия за рабочий день, минуты	Условия воздействия			
	Общее		Локальное	
	ПДУ напряженности, кА/м	ПДУ магнитной индукции, мТл	ПДУ напряженности, кА/м	ПДУ магнитной индукции, мТл
0-10	24	30	40	50
11-60	16	20	24	30
61-480	8	10	12	15

Также временные допустимые уровни (ВДУ) электромагнитных полей регламентируются СанПиН 2.2.2/2.4.2732-10 [16]. В таблице ниже показаны временные допустимые уровни ЭМП, создаваемые ПЭВМ.

Таблица 17 – Временные допустимые уровни ЭМП

Наименование параметров		ВДУ ЭМП
Напряженность электрического поля	в диапазоне частот 5 Гц - 2 кГц	25 В/м
	в диапазоне частот 2 кГц - 400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот 5 Гц - 2 кГц	250 нТл
	в диапазоне частот 2 кГц - 400 кГц	25 нТл
Электростатический потенциал экрана видеомонитора		500 В

Способы защиты от ЭМП:

- использование защитных экранов и фильтров для мониторов;
- увеличение расстояния от источника излучения до облучаемого тела (расстояние должно быть не меньше 50см);
- использование профилактических напитков;
- применение средств индивидуальной защиты путем экранирования пользователя ПЭВМ целиком или отдельных зон его тела;
- использование сертифицированных ПЭВМ;

4.1.4 Производственное освещение

Освещение - использование световой энергии солнца и искусственных источников света для обеспечения зрительной работоспособности в производственных помещениях. Правильная организация освещения в помещении очень важна, так как любое его отклонение от нормы может оказать негативное влияние на состояние человека. Так, к примеру, недостаток освещения вызывает усталость центральной нервной системы, возникающей в результате прилагаемых усилий для опознания четких или сомнительных сигналов. Также, в результате нерационально организованной системы освещения рабочего помещения могут возникнуть: слепые зоны, резкие тени и слепящие источники света, что может послужить причиной травматизма работника.

Согласно гигиеническим требованиям к персональным электронно-вычислительным машинам [16], освещение помещения, предназначенного для работы с компьютером должно быть оснащено как искусственным, так и естественным источниками освещения, причем рабочее место должно располагаться боковой стороной к световым проемам, а естественный свет должен падать преимущественно слева. Уровень искусственного освещения должен быть не менее 300 лк. Коэффициент пульсации освещенности не должен быть выше 5%. Яркость светящихся поверхностей, находящихся в поле зрения, и светильников общего освещения в зоне углов излучения от 50 до 90° с вертикалью в продольной и поперечной плоскостях не должны

превышать 200 кд/м^2. Освещенность поверхности рабочего стола должна варьироваться в пределах 300 - 500лк.

В помещении, где была выполнена ВКР организовано сочетание естественного и искусственного освещений. Для искусственного освещения используются 6 светильников ЛПО36, расположенных равномерно в два ряда, по всей поверхности потолка. Каждый светильник содержит по 4 люминесцентных лампы ЛБ-40. В помещение находится два оконных проема, и рассматриваемое рабочее место располагается слева от него.

Таблица 18 – Параметры систем естественного и искусственного освещения

Наименование рабочего места	Тип светильника и источника света	Коэффициент естественной освещенности, КЕО, %		Освещенность при совмещенной системе, лк	
		Фактически	Норм. значение	Фактически	Норм. значение
Помещение для работы с ПЭВМ	ОДР ЛБ-40	---	0,7	1021 лк	300÷500 лк

Для данного типа помещений коэффициент естественного освещения (КЕО) должен равняется 0,7 при совмещенном и боковом естественном освещении.

4.1.5 Электробезопасность

Электрическая безопасность (ЭБ) – комплекс организационных мер в совокупности с техническими средствами, призванных предотвращать вредное и опасное воздействие на работника электрического тока, электрической дуги и статического электричества.

Для помещений, оборудованных ПЭВМ, электричество представляет особую опасность, так как большое количество элементов находится под напряжением. Риск электропоражения возрастает при превышении относительной влажности в помещении (больше 75%), при наличии токопроводящих полов и токопроводящей пыли, при высокой температуры (больше 35°С) и при возможности одновременного соприкосновения с металлическим элементом, соединенным с землей, и металлическим корпусом

электрооборудования. Поэтому, опасность поражения человека электрическим током напрямую зависит от правильного размещения оборудования и соблюдения правил электробезопасности.

Опасность поражения током наступает при:

- прикосновении к токоведущим частям;
- прикосновении нетоковедущим частям и поверхностям, под напряжением;
- коротком замыкании в высоковольтных блоках.

В помещении, где проводились работы, используются ПЭВМ и другие устройства, использующие электрический ток, поэтому необходимо использовать следующие меры предосторожности:

1. перед использованием убедиться в исправности оборудования;
2. при обнаружении неисправностей, не предпринимая никаких самостоятельных действий по исправлению, сообщить ответственному за оборудование;
3. содержать рабочее место свободным от лишних предметов.

В случае же возникновения несчастного случая следует немедленно освободить пострадавшего от действия электрического тока и, вызвав врача, оказать ему необходимую помощь.

В помещениях, оборудованных электрическими устройствами, часто возникает статическое электричество в результате сопрокосновения человека с элементами электрических устройств. Однако, его разряды не опасны для людей, но могут могут привести к выходу устройств из строя. Чтобы снизить величины токов статического электричества, необходимо использование специальных половых покрытий с антистатической пропиткой и регулярное увлажнение воздуха.

4.2 Экологическая безопасность

Экологическая безопасность - это комплекс организационно-технических мер, направленных на обеспечение соответствия природоохранной деятельности предприятия нормативным требованиям.

Для типа органиций, в которой выполнялся описываемый проект, меропроятия по экологической безопасности сводятся к утилизации отходов жизнедеятельности человека и бытового мусора. В офисных помещениях основными отходами являются бумага, канцелярские принадлежности, батарейки, отходы от продуктов питания и личной гигиены, использованные картриджи, вышедшие из строя электронные устройства, люминесцентные лампы и др.

В связи с таким разнообразием отходов необходимы мероприятия по сортированию и сбору мусора в зависимости от его происхождения перед устранением. Это позволяет намного упростить процесс переработки отходов для вторичного использования и избежать его гниения и горения в окружающей среде. Отходы имеющие статус опасный для окружающей среды, устанавливающийся в Федеральном Классификационном Каталоге Отходов (ФККО) должны устранятся специальными утилизирующими компаниями.

4.3 Безопасность в чрезвычайных ситуациях.

Для рассматриваемого типа помещения наиболее распространенным видом чрезвычайных ситуаций является пожар, поэтому в данной части работы будут рассмотрены вопросы пожарной безопасности.

Пожарная безопасность - это такое состояние производственного помещения, при котором с установленной вероятностью исключается возможность возникновения и развития пожара и воздействия на людей его опасных факторов, а также созданы условия для защиты материальных ценностей.

4.3.1 Оценка пожарной безопасности помещения

Согласно противопожарным нормам нормам НПБ 105-95 все виды помещения и сооружений классифицируются по пожарной и взрывной опасности. Рассматриваемая лаборатория оборудована электронными устройствами напряжением 220В и деревянной мебелью, что соответствует категории В.

Причиной возникновения пожара в лаборатории может послужить:

1) короткое замыкание;

2) работа с неисправными или дефектными электрическими устройствами;

3) неисправности в проводке, розетках и выключателях;

4) несоблюдение правил пожарной безопасности;

5) работа с открытой электроаппаратурой.

В связи с этим необходимо соблюдать правила пожарной безопасности.

4.3.2 Основные правила пожарной безопасности помещения

Для исследуемого помещения характерны правила пожарной безопасности приведенные ниже.

1) Содержание служебных и вспомогательных помещений в чистоте и систематическая очистка их от документов с истекшим сроком хранения, мусор в зданиях и помещениях должен удаляться ежедневно.

2) Эксплуатация осветительного электрооборудования, электропроводки и других потребители электроэнергии должна соответствовать Правилам устройства электроустановок, Правилам технической эксплуатации электроустановок потребителей и Правилам техники безопасности при эксплуатации электроустановок потребителей.

3) Обеспечение свободного доступа к электрощитам и электрооборудованию.

4) Подключение новых потребителей электрической энергии должно проводиться после согласования с рабочим по обслуживанию здания и ответственным за пожарную безопасность.

5) Вывод из эксплуатации неисправных электроприборов, которые могут послужить причиной возникновения пожара, для дальнейшей замены либо ремонта.

6) Подключение к электросети настольных ламп, вентиляторов, ПЭВМ и других электроприборов должно проводиться только через исправные штепсельные розетки и электрошнуры. Запрещается эксплуатация

временных электросетей. Замеры сопротивления изоляции в силовых и осветительных сетях необходимо проводить не реже одного раза в год.

7) Работники, в конце рабочего дня, должны обеспечить порядок на рабочем месте, закрыть окна, а также обесточить все электронные устройства.

8) Допускается применение приборов местного отопления только заводского изготовления, имеющих автоматическое отключение при нештатных ситуациях (короткое замыкание, повышение температуры теплоносителя выше допустимой, утечка теплоносителя и т.п.).

9) Уборка помещений и чистка электротехнического оборудования должна проводиться после отключения электроприборов из сети.

10) Использование только исправного оборудования, имеющего технический паспорт.

4.3.3 Мероприятия по устранению и предупреждению пожаров

Для предупреждения возникновения пожара помещение должно быть оборудовано средствами тушения пожара (огнетушителями, ящиком с песком, стендом с противопожарным инвентарем); средствами связи; должна быть исправна электрическая проводка осветительных приборов и электрооборудования. У каждый сотрудника должна иметься информация о месте нахождения средств пожаротушения и средств связи; должны быть номера телефонов для сообщения о пожаре; сотрудники должны уметь пользоваться средствами пожаротушения. Для этого необходимо регулярное проведение инструктажа работников о пожаробезопасности.

Для целей пожаротушения в помещении применяются углекислотный огнетушитель типа ОУ–2 - для тушения первичных загораний, а также электроустановок, находящихся под напряжением. Также в лаборатории находится электрический щит, с помощью которого обесточиваются все электроприборы в случае опасности.

4.3.4 Действия работников в случае пожара

При обнаружении пожара работник должен немедленно сообщить об этом по телефону 01 и спокойно доложить:

- что горит, чему угрожает;
- адрес, где располагается лаборатория;
- есть ли опасность для людей;
- назвать свою фамилию.

Далее, сообщение следует продублировать руководителю или дежурному смены и приступить к тушению пожара огнетушителями и подручными средствами, протянуть рукавную линию от внутреннего пожарного крана. Огнетушитель необходимо поднести к очагу пожара, удалить на огнетушителе чеку, направить раструб в сторону очага пожара и нажать на рычаг. Работник должен слушать распоряжения дежурного по смене, организованно покинуть здание. При невозможности покинуть здание (задымление, высокая температура) следует плотно закрыть дверь помещения, уплотнить тканью щели, вентиляционные отверстия, открыть окно и ждать пожарных. Следует запомнить, что при задымлении над полом воздух более чистый, что может пригодиться при эвакуации и ожидании помощи.

4.4 Особенности законодательного регулирования проектных решений.

Основная работа, выполняемая исполнителем данной работы, проводится сидя за компьютером. В связи с этим, необходимо соблюдение режима работы и отдыха. В соответствии с [16], данный вид работы, выполняемый с использованием ПЭВМ, относится к группе В, так как заключается в творческой работе в режиме диалога с ПК. В таблице 5.6 представлено время регламентированных перерывов в зависимости от продолжительности работы.

Таблица 19 – Время регламентированных перерывов при работе с ПЭВМ

Категория работы с ПЭВМ	Уровень нагрузки за рабочую смену при видах работ с ПЭВМ			Суммарное время регламентированных перерывов, мин	
	группа А, количество знаков	группа Б, количество знаков	группа В, ч	при 8-часовой смене	при 12-часовой смене
I	до 20000	до 15000	до 2	50	80
II	до 40000	до 30000	до 4	70	110
III	до 60000	до 40000	до 6	90	140

Рекомендуется, во избежание преждевременной утомляемости пользователей ПЭВМ, организовывать рабочую смену чередуя работы с использованием ПЭВМ и без него. В случае, когда несмотря на соблюдение всех санитарно-гигиенических и эргономических требований, у работника с ПЭВМ возникает зрительный дискомфорт и другие неблагоприятные субъективные ощущения, рекомендуется применять индивидуальный подход с ограничением времени работы с ПЭВМ.

Для работ, требующих постоянного взаимодействия с ВДТ (набор текстов или ввод данных и т.п.) с напряжением внимания и сосредоточенности, при невозможности периодического переключения на другие виды трудовой деятельности, не связанные с ПЭВМ, необходима организация перерывов на 10 - 15 мин каждые 45 - 60 мин работы.

Кроме того, для снижения нервно-эмоционального напряжения, утомления зрительного анализатора, предотвращения развития утомления, во время регламентированных перерывов рекомендуется выполнять комплексы специальных упражнений.

Заключение

В результате выполнения выпускной квалификационной работы, была построена модель методами машинного обучения, способная предсказывать время обработки цепочек вычислительных задач с точностью 87,9%. Внедрение данной модели в производственную систему ProdSys2 эксперимента ATLAS позволит обеспечить рациональную организацию планирования загрузки цепочек в систему, во избежание простоя оборудования и их неравномерного распределения среди центров обработки данных системы. Данная работа является доказательством эффективности применения методов машинного обучения для анализа задач систем масштаба ProdSys2.

Список использованных источников

1. Е. А. Соколов. Курс лекций ВШЭ, Лекция 1. Введение в машинное обучение [Электронный ресурс]. – 2016.- Режим доступа: https://github.com/esokolov/ml-course-hse/blob/master/2016-fall/lecture-notes/lecture01-intro.pdf
2. Общая постановка задачи обучения по прецедентам [Электронный ресурс]. – 2015, Режим доступа: http://www.machinelearning.ru/
3. Л. П. Коэльо, В. Ричард Построение систем машинного обучения на языке Python. - М.: ДМК Пресс, 2016. - 302 с.
4. Е. А. Соколов. Курс лекций ВШЭ, Лекция 2. Линейная регрессия [Электронный ресурс]. – 2016.- Режим доступа: https://github.com/esokolov/ml-course-hse/blob/master/2016-fall/lecture-notes/lecture02-linregr.pdf
5. Е. А. Соколов. Курс лекций ВШЭ, Лекция 3. Линейная регрессия [Электронный ресурс]. – 2016.- Режим доступа: https://github.com/esokolov/ml-course-hse/blob/master/2016-fall/lecture-notes/lecture03-linregr.pdf
6. Электронный курс «Обучение на размеченных данных», Решающие деревья [Электронный ресурс]. – 2016.- Режим доступа: https://www.coursera.org/learn/supervised-learning/supplement/flyUL/konspiekt
7. Электронный курс «Обучение на размеченных данных», Случайный лес [Электронный ресурс]. – 2016.- Режим доступа: https://www.coursera.org/learn/supervised-learning/supplement/25tKa/konspiekt
8. Электронный курс «Обучение на размеченных данных», Градиентный бустинг решающих деревьев [Электронный ресурс]. – 2016.- Режим доступа: https://www.coursera.org/learn/supervised-learning/supplement/DIly0/konspiekt
9. Критерии оценки качества регрессионной модели, или какая модель хорошая, а какая лучше [Электронный ресурс]. – 2016.- Режим доступа: http://www.prognoz.ru/blog/platform/kriterii-otsenki-modeli/
10. Mean absolute error [Электронный ресурс]. – 2015.- Режим доступа: https://en.wikipedia.org/wiki/Mean_absolute_error

11. Финансовый менеджмент, ресурсоэффективность и ресурсо-сбережение: учебно-методическое пособие / И.Г. Видяев, Г.Н. Се- рикова, Н.А. Гаврикова, Н.В. Шаповалова, Л.Р. Тухватулина З.В. Криницына; Томский политехнический университет. − Томск: Изд-во Томского политехнического университета, 2014. – 36 с.

12. Безопасность жизнедеятельности: Справочное пособие по дипломному проектированию / Под редакцией Иванова Н.И. и Фадина И.М. – СПб.: БГТУ, 1995.

13. СанПиН 2.2.4.548-96. Гигиенические требования к микроклимату производственных помещений.

14. СН 2.2.4/2.1.8.562-96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки.

15. СанПиН 2.2.4.1191-03. Электромагнитные поля в производственных условиях.

16. СанПиН 2.2.2/2.4.2732-10. Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.

17. СанПиН 2.2.1/2.1.1.1278-03. Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий.

Приложение А. Описание исходных данных.

Название атрибута	Описание
“TASKID”	идентификатор вычислительной задачи
“DURATION”	продолжительность вычислительной задачи, получаемая с помощью предсказательной модели в производственной системе ProdSys2
“STARTTIME”	время начала обработки задачи в системе ProdSys2
“ENDTIME”	время окончания обработки задачи в системе ProdSys2
“WEEK”	номер недели, с начала года, в которую началась обработка
“WEEKDAY”	день недели, в который началась обработка
“TOTAL_DONE_JOBS”	количество обработанных работ, в результате выполнения задачи
“PROVENANCE”	источник формирования задачи
“PROJECT”	название проекта, в рамках которого происходит обработка данных
“TYPE”	тип обрабатываемой задачи
“TOTAL_EVENTS”	общее количество событий
“TOTAL_REQ_JOBS”	общее количество работ в задаче
“TOTAL_REQ_EVENTS”	количество событий, которое необходимо обработать
“NFILESTOBEUSED”	количество файлов с данными, которые необходимо обработать
“CURRENT_PRIORITY”	текущий приоритет задачи
“STATUS”	статус обработки
“USERNAME”	имя пользователя, сформировавшего запрос на обработку данных

“REQID”	идентификатор запроса, на обработку которого формируется задача
“CLOUD”	название группы центров обработки данных, где назначена обработка цепочки
“SITE”	название центра обработки данных, где назначена обработка цепочки
“PRODSOURCELABEL”	метка источника
“WORKINGGROUP”	группа пользователей, которой принадлежит задача
“CORECOUNT”	количество процессоров на обработку одной работы задачи
“PROCESSINGTYPE”	вид обработки задачи
“TASKPRIORITY”	приоритет задачи
“ARCHITECTURE”	архитектура системы, на которой происходит обработка задачи
“SPLITRULE”	правила разбиение задачи на работы. Например, ограничение количества файлов на одну работу
“WORKQUEUE_ID”	идентификатор очереди в которой находится задача
“ERRORDIALOG”	сообщения об ошибки во время обработки задачи
“PARENT_TID”	идентификатор задачи предшественника. Если предшественника нет, то данное поле принимает значение идентификатора данной задачи
“CAMPAIGN”	способ обработки задачи
“GOAL”	цель выполнения задачи
“NUCLEUS”	содержит сведения “CLOUD” и “SITE”
“RAMCOUNT”	среднее потребляемое количество оперативной памяти задачей

“RAMUNIT”	единица измерения “RAMCOUNT”
“WALLTIME”	среднее время обработки задачи
“WALLTIMEUNIT”	единица измерения “WALLTIME”
“GSHARE”	доля глобальных ресурсов на которую может претендовать пользователь, который ставит задачу

Приложение Б. Алгоритм извлечения цепочек.

```
#Импорт parquet файла с данными в DataFtame струтктуру
parquet_file = ParquetFile('newdata.parquet')
dataFrame = parquet_file.to_pandas()

#Приведение значений числовых признаков к числовому типу данных
#Перевод значений, характеризующих время из миллисекунд в секунды
dataFrame['DURATION'] = dataFrame['DURATION'].apply(lambda x: x / 1000)
dataFrame['STARTTIME'] = dataFrame['STARTTIME'].apply(lambda x: x /
1000)
dataFrame['ENDTIME'] = dataFrame['ENDTIME'].apply(lambda x: x / 1000)
dataFrame['CORECOUNT'] = dataFrame['CORECOUNT'].apply(lambda x: int(x))
dataFrame['TOTAL_REQ_JOBS'] =
dataFrame['TOTAL_REQ_JOBS'].apply(lambda x: int(x))
dataFrame['NFILESTOBEUSED'] =
dataFrame['NFILESTOBEUSED'].apply(lambda x: int(x))
dataFrame['TOTAL_REQ_EVENTS'] =
dataFrame['TOTAL_REQ_EVENTS'].apply(lambda x: int(x))
dataFrame['TOTAL_EVENTS'] = dataFrame['TOTAL_EVENTS'].apply(lambda
x: int(x))
dataFrame['PARENT_TID'] = dataFrame['PARENT_TID'].apply(lambda x: int(x))

#Массив, в который сохраняются DataFtame цепочки
chains = []

#Нахождение всех первых вершин цепочек или одиночных задач.
roots = dataFrame['PARENT_TID'][dataFrame['TASKID'] ==
dataFrame['PARENT_TID'].values]

#Нахождение потомков первых задач
```

```
for root in roots:
    chain = dataFrame[dataFrame.PARENT_TID == root]
    tid = chain
    #Нахождение всех задач цепочки
    while tid.shape[0] != 0:
        tids = tid[tid.TASKID != tid.PARENT_TID]['TASKID']
        tid = dataFrame[dataFrame.PARENT_TID.isin(tids)]
        chain = chain.append(tid)

    #Сортировка задач цепочки по идентификаторам задач
    chain = chain.sort_values('TASKID')

    #Создание новых индексов задач цепчки
    chain.index = range(0,len(chain))

    #Добавление найденой цепочки в массив, при условии, что она содержит минимум 2 задачи
    if chain.shape[0] >= 2:
        chains.append(chain)
```

Приложение В. Описание нового набора данных.

Название атрибута	Описание
“TASKID”	идентификатор вычислительной задачи
“DURATION”	продолжительность вычислительной задачи, получаемая с помощью предсказательной модели в производственной системе ProdSys2
“STARTTIME”	время начала обработки задачи в системе ProdSys2
“ENDTIME”	время окончания обработки задачи в системе ProdSys2
“WEEK”	номер недели, с начала года, в которую началась обработка
“WEEKDAY”	день недели, в который началась обработка
“PROVENANCE”	источник формирования задачи
“PROJECT”	название проекта, в рамках которого происходит обработка данных
“TYPE”	тип обрабатываемой задачи
“TOTAL_REQ_JOBS”	общее количество работ в задаче
“TOTAL_REQ_EVENTS”	количество событий, которое необходимо обработать
“NFILESTOBEUSED”	количество файлов с данными, которые необходимо обработать
“USERNAME”	имя пользователя, сформировавшего запрос на обработку данных
“PRODSOURCELABEL”	метка источника
“WORKINGGROUP”	группа пользователей, которой принадлежит задача
“CORECOUNT”	количество процессоров на обработку одной работы задачи

“PROCESSINGTYPE”	вид обработки задачи
“TASKPRIORITY”	приоритет задачи
“ARCHITECTURE”	архитектура системы, на которой происходит обработка задачи
“WORKQUEUE_ID”	идентификатор очереди, в которой находится задача
“PARENT_TID”	идентификатор задачи предшественника. Если предшественника нет, то данное поле принимает значение идентификатора данной задачи
“CAMPAIGN”	способ обработки задачи
“NUCLEUS”	содержит сведения “CLOUD” и “SITE”
“GSHARE”	доля глобальных ресурсов на которую может претендовать пользователь, который ставит задачу

Приложение Г. Построение набора данных для обучения моделей.

```
#Названия признаков, которые учавтвуеют в обучении модели
columns = ['FIRST_TASKID', 'FACT_DURATION',
'DURATION_SUM','TASK_COUNT', 'PROJECT', 'CAMPAIGN',
'PROVENANCE',
        'NFILESTOBEUSED', 'TOTAL_REQ_EVENTS', 'TYPE', 'USERNAME',
      'WORKINGGROUP', 'PROCESSINGTYPE', 'NUCLEUS', 'CORECOUNT',
'ARCHITECTURE', 'WORKQUEUE_ID',
      'TASKPRIORITY', 'WEEKDAY', 'WEEK', 'TOTAL_REQ_JOBS',
'GSHARE']

#Инициализация набора данных
dataset = pd.DataFrame(columns = columns)

#Добавление цепочек в набор данных, с усредненными значениями
for i, chain in enumerate(chains):

   #Добавление идентификатора первой цепочки, реальной
продолжительности обработки цепочки и суммы продолжительностей задач
   row = [str(chain['TASKID'].ix[0]),(chain['ENDTIME'].max() -
chain['STARTTIME'].min()), chain['DURATION'].sum()]

   #Добавление количества задач в цепочке
   row.append(chain.shape[0])

   #Добавление значения Проекта первой задачи в цепочке
   row.append(chain['PROJECT'][0])

   #Добавление значения Проекта первой задачи в цепочке
   row.append(chain['CAMPAIGN'][0])
```

```
#Добавление всех значений Источника данных в цепочке
prov = ''
for p in range(len(chain['PROVENANCE'].unique())):
    prov += str(chain['PROVENANCE'].unique()[p]) + ', '
row.append(prov)

#Добавление суммы всех файлов в цепочке для обработки
row.append(chain['NFILESTOBEUSED'].sum())

#Добавление суммы всех файлов в цепочке для обработки
row.append(chain['TOTAL_EVENTS'].sum())

#Добавление всех типов задач в цепочке
type = ''
for k in range(len(chain['TYPE'].unique())):
    type += str(chain['TYPE'].unique()[k]) + ', '
row.append(type)

#Добавление имени пользователя первой задачи
row.append(chain['USERNAME'][0])

#Добавление рабочей группы пользователя первой задачи
row.append(chain['WORKINGGROUP'][0])

#Добавление всех видов обработки задач в цепочке
ptype = ''
for pt in range(len(chain['PROCESSINGTYPE'].unique())):
    ptype += str(chain['PROCESSINGTYPE'].unique()[pt]) + ', '
row.append(ptype)
```

```
    #Добавление сайта облака первой задачи
    row.append(chain['NUCLEUS'][0])

    #Добавление суммы ядер процесссоров для обработки цепочки
    row.append(chain['CORECOUNT'].sum())

    #Добавление всех видов архитектур аппаратных средств, на которых
обрабатывается цепочка
    arc = ''
    for ar in range(len(chain['ARCHITECTURE'].unique())):
        arc += str(chain['ARCHITECTURE'].unique()[ar]) + ', '
    row.append(arc)

    #Добавление идентификатора первой задачи в очереди на обработку
    row.append(chain['WORKQUEUE_ID'][0])

    #Добавление наиболее часто встречащегося значения приоритета задач
    row.append(mode(chain['TASKPRIORITY'])[0][0])

    #Добавление дня недели в который начинается обраотка
    row.append(chain['WEEKDAY'][0])

    #Добавление номера недели в который начинается обраотка
    row.append(chain['WEEK'][0])

    #Добавление суммы работ задач в цепочке
    row.append(chain['TOTAL_REQ_JOBS'].sum())

    #Добавление всех значений доли ресурсов задач для пользователя
```

```
    gs = ''
    for g in range(len(chain['GSHARE'].unique())):
        prov += str(chain['GSHARE'].unique()[g]) + ', '
    row.append(gs)

    #Запись цепочки в набор данных
    dataset.loc[i] = row

#Привеение значений к целочисленному типу
dataset['WEEKDAY'] = dataset['WEEKDAY'].apply(lambda x: int(x))
dataset['FACT_DURATION'] = dataset['FACT_DURATION'].apply(lambda x:
int(x))
dataset['DURATION_SUM'] = dataset['DURATION_SUM'].apply(lambda x:
int(x))
dataset['WEEK'] = dataset['WEEK'].apply(lambda x: int(x))

#Создание объекта кодировщика категориальных признаков
le = LabelEncoder()

#Кодирование категориальных признаков
le.fit(dataset['PROJECT'])
pr = le.transform(dataset['PROJECT'])
dataset['PROJECT'] = pr
le.fit(dataset['TYPE'])
type = le.transform(dataset['TYPE'])
dataset['TYPE'] = type
le.fit(dataset['PROVENANCE'])
prov = le.transform(dataset['PROVENANCE'])
dataset['PROVENANCE'] = prov
le.fit(dataset['GSHARE'])
```

```
gs = le.transform(dataset['GSHARE'])
dataset['GSHARE'] = gs
le.fit(dataset['USERNAME'])
un = le.transform(dataset['USERNAME'])
dataset['USERNAME'] = un
le.fit(dataset['WORKINGGROUP'])
wgroup = le.transform(dataset['WORKINGGROUP'])
dataset['WORKINGGROUP'] = wgroup
le.fit(dataset['PROCESSINGTYPE'])
prtype = le.transform(dataset['PROCESSINGTYPE'])
dataset['PROCESSINGTYPE'] = prtype
le.fit(dataset['NUCLEUS'])
nucl = le.transform(dataset['NUCLEUS'])
dataset['NUCLEUS'] = nucl
le.fit(dataset['CAMPAIGN'])
cam = le.transform(dataset['CAMPAIGN'])
dataset['CAMPAIGN'] = cam
le.fit(dataset['ARCHITECTURE'])
ar = le.transform(dataset['ARCHITECTURE'])
dataset['ARCHITECTURE'] = ar
le.fit(dataset['WORKQUEUE_ID'])
wqid = le.transform(dataset['WORKQUEUE_ID'])
dataset['WORKQUEUE_ID'] = wqid
le.fit(dataset['TASKPRIORITY'])
tpr = le.transform(dataset['TASKPRIORITY'])
dataset['TASKPRIORITY'] = tpr
le.fit(dataset['WEEKDAY'])
wd = le.transform(dataset['WEEKDAY'])
dataset['WEEKDAY'] = wd
le.fit(dataset['WEEK'])
```

```
w = le.transform(dataset['WEEK'])
dataset['WEEK'] = w
```

Приложение Д. Функция построения диаграмм Ганта для цепочек задач.

```
#В качестве аргумента, функция принимает DataFrame цепочку
def makeGanttChart(chain):
    #Массивы меток диаграммы (идентификаторов задач) и дат начала и
окончания обработки
    ylabels = []
    customDates = []

    #Заполнение массива меток диаграммы
    for task in chain.iterrows():
        ylabels.append(task[1]["TASKID"])
        customDates.append(

[matplotlib.dates.datestr2num(dt.datetime.fromtimestamp(task[1]["STARTTIME"]
).strftime('%Y-%m-%d %H:%M:%S')),

matplotlib.dates.datestr2num(dt.datetime.fromtimestamp(task[1]["ENDTIME"]).st
rftime('%Y-%m-%d %H:%M:%S'))])

    #Генерация положений полосок диаграммы
    ilen = len(ylabels)
    pos = np.arange(0.5, 0.5+0.5*ilen, 0.5)

    #Создание словаря с датами
    task_dates = {}
    for i,task in enumerate(ylabels):
        task_dates[task] = customDates[i]

    #Создание фигуры, на которой располагается диаграмма
    fig = plt.figure(figsize=(20,8))
```

```
    ax = fig.add_subplot(111)

    #Размещение полос продолжительностей задач
    for i in range(len(ylabels)):
        start_date, end_date = task_dates[ylabels[i]]
        ax.barh((i*0.5)+0.5, end_date - start_date, left=start_date, height=0.3,
align='center',
                edgecolor='lightgreen', color='orange', alpha = 0.8)
    locsy, labelsy = plt.yticks(pos, ylabels)
    plt.setp(labelsy, fontsize = 14)

    #Установка параметров отображения диаграммы и ее показ
    ax.set_ylim(ymin = -0.1, ymax = ilen*0.5+0.5)
    ax.grid(color = 'g', linestyle = ':')
    ax.xaxis_date()
    formatter = DateFormatter("%d-%b-%y %H:%M")
    ax.xaxis.set_major_locator(AutoDateLocator())
    ax.xaxis.set_major_formatter(formatter)
    labelsx = ax.get_xticklabels()
    plt.setp(labelsx, rotation=30, fontsize=10)
    ax.set_ylabel("TASKID", fontweight='bold')
    ax.set_xlabel("DATE TIME", fontweight='bold')
    ax.set_title('Gantt Chart',fontsize=14, fontweight='bold')
    font = font_manager.FontProperties(size='small')
    ax.legend(loc=1,prop=font)
    ax.invert_yaxis()
    fig.autofmt_xdate()
    plt.show()
```

Приложение Е. Построение константной модели.

```
#Выбор признаков для обучения константной модели
const_dataset = dataset[['FIRST_TASKID', 'DURATION_SUM',
'FACT_DURATION']]

#Разделение набора данных на данные и ответы
data = np.array(const_dataset['DURATION_SUM'])
target = np.array(const_dataset['FACT_DURATION'])

#Создание объекта константной модели, возвращающей среднее значение
dummy = DummyRegressor(strategy='mean')

#Количество разбиений и переменные, хранящие результаты предсказаний
модели
n_iter = 30
dm_r2 = 0
dm_mae = 0

for train_indices, test_indices in cross_validation.ShuffleSplit(data.shape[0], n_iter
= n_iter, test_size = 0.4):
    #Разбиение данных и отвветов на обучающие и тестовые подвыборки
    train_data = data[train_indices]
    test_data = data[test_indices]
    train_target = target[train_indices]
    test_target = target[test_indices]

    #Обучение модели и сложение точностей модели на разнах разбиениях
    dm = dummy.fit(train_target, train_data)
    predicts = [dm.predict(test_data)[0]] * len(test_target)
    dm_r2 += r2_score(test_target, predicts)
```

```
    dm_mae += mean_absolute_error(test_target, predicts)

#Вывод средних оценок метрик точности модели
print('R^2 score : ', (dm_r2) / n_iter)
print('MAE score : ', (dm_mae) / n_iter)

#Создание фигуры, на которой располагается график
plt.figure(figsize=(12,8))
plt.cla()
plt.clf()

#Нанесение на график точек, характеризующих ответы модели
scatter = plt.plot(list(map(lambda x: x/86400 ,test_target)),
                   list(map(lambda x: x/86400,[dm.predict(test_data)[0]] *
len(test_target)))
                   ,"b.", label ='Полученные ответы')

#Нанесение на график прямой, характеризующей идеальные предсказания
модели
line1 = plt.plot(np.linspace(0,6000000/86400),np.linspace(0,6000000/86400),"g-",
label ='Идеальные предсказание')

#Нанесение на график области средней ошибки предсказаний модели
line2 = plt.plot(np.linspace((dm_mae / n_iter)/86400,(6000000+dm_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--",label ='MAE')
line3 = plt.plot(np.linspace(-(dm_mae / n_iter)/86400,(6000000-dm_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--")

#Установка параметров отображения графика
xlim(0)
```

```
ylim(0,70)
xlabel('Правильные ответы(сутки)', fontweight='bold')
ylabel('Предсказания модели(сутки)', fontweight='bold')
plt.title("Точность предсказаний константной модели", fontsize=15,
fontweight='bold')
l = legend(bbox_to_anchor=(0, 0, 1, 1), fontsize = 12)
plt.show()
```

Приложение Ж. Построение модели Лассо.

```
#Исключение идентификатора цепочки и разделение набора данных на данные и ответы
dataset = dataset.drop(['FIRST_TASKID'], 1)
target = np.array(dataset['FACT_DURATION']).astype(int)
ndata = dataset.drop(['FACT_DURATION'], 1)
data = np.array(ndata)

#Создание объекта модели Лассо
lasso = Lasso()

#Количество разбиений и переменные, хранящие результаты предсказаний модели
n_iter = 30
ls_r2 = 0
ls_mae = 0

for train_indices, test_indices in cross_validation.ShuffleSplit(data.shape[0], n_iter = n_iter, test_size = 0.4):
    #Разбиение данных и отвветов на обучающие и тестовые подвыборки
    train_data = data[train_indices]
    test_data = data[test_indices]
    train_target = target[train_indices]
    test_target = target[test_indices]

    #Обучение модели и сложение точностей модели на разнах разбиениях
    ls = lasso.fit(train_data, train_target)
    ls_r2 += r2_score(test_target, ls.predict(test_data))
    ls_mae += mean_absolute_error(test_target, ls.predict(test_data))
```

```
#Вывод средних оценок метрик точности модели
print('R^2 score : ', (ls_r2) / n_iter)
print('MAE score : ', (ls_mae) / n_iter)

#Создание фигуры, на которой располагается график
plt.figure(figsize=(12,8))
plt.cla()
plt.clf()

#Нанесение на график точек, характеризующих ответы модели
scatter = plt.plot(list(map(lambda x: x/86400 ,test_target)),
                   list(map(lambda x: x/86400,ls.predict(test_data)))
                   ,"b.", label ='Полученные ответы')

#Нанесение на график прямой, характеризующей идеальные предсказания
модели
line1 = plt.plot(np.linspace(0,6000000/86400),np.linspace(0,6000000/86400),"g-",
label ='Идеальные предсказание')

#Нанесение на график области средней ошибки предсказаний модели
line2 = plt.plot(np.linspace((ls_mae / n_iter)/86400,(6000000+ls_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--",label ='MAE')
line3 = plt.plot(np.linspace(-(ls_mae / n_iter)/86400,(6000000-ls_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--")

#Установка параметров отображения графика
xlim(0)
ylim(-20,70)
xlabel('Правильные ответы(сутки)', fontweight='bold')
ylabel('Предсказания модели(сутки)', fontweight='bold')
```

```
plt.title("Точность предсказаний модели Лассо", fontsize=15, fontweight='bold')
l = legend(bbox_to_anchor=(0, 0, 1, 1), fontsize = 12)
plt.show()
```

Приложение И. Построение модели Случайный лес.

```
#Исключение идентификатора цепочки и разделение набора данных на данные и ответы
dataset = dataset.drop(['FIRST_TASKID'], 1)
target = np.array(dataset['FACT_DURATION']).astype(int)
ndata = dataset.drop(['FACT_DURATION'], 1)
data = np.array(ndata)

#Создание объекта модели Случайный лес
random_forest = RandomForestRegressor()

#Количество разбиений и переменные, хранящие результаты предсказаний модели
n_iter = 30
rf_r2 = 0
rf_mae = 0

for train_indices, test_indices in cross_validation.ShuffleSplit(data.shape[0], n_iter = n_iter, test_size = 0.4):
    #Разбиение данных и отвветов на обучающие и тестовые подвыборки
    train_data = data[train_indices]
    test_data = data[test_indices]
    train_target = target[train_indices]
    test_target = target[test_indices]

    #Обучение модели и сложение точностей модели на разнах разбиениях
    rf = random_forest.fit(train_data, train_target)
    rf_r2 += rf.score(test_data, test_target)
    rf_mae += mean_absolute_error(test_target, rf.predict(test_data))
```

```
#Вывод средних оценок метрик точности модели
print('RandomForest R^2 score : ', (rf_r2) / n_iter)
print('RandomForest MAE : ', (rf_mae) / n_iter)

#Создание фигуры, на которой располагается график
plt.figure(figsize=(12,8))
plt.cla()
plt.clf()

#Нанесение на график точек, характеризующих ответы модели
scatter = plt.plot(list(map(lambda x: x/86400 ,test_target)),
                   list(map(lambda x: x/86400,rf.predict(test_data)))
                   ,"b.", label ='Полученные ответы')

#Нанесение на график прямой, характеризующей идеальные предсказания
модели
line1 = plt.plot(np.linspace(0,6000000/86400),np.linspace(0,6000000/86400),"g-",
label ='Идеальные предсказание')

#Нанесение на график области средней ошибки предсказаний модели
line2 = plt.plot(np.linspace((rf_mae / n_iter)/86400,(6000000+rf_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--",label ='MAE')
line3 = plt.plot(np.linspace(-(rf_mae / n_iter)/86400,(6000000-rf_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--")

#Установка параметров отображения графика
xlim(0)
ylim(0,70)
xlabel('Правильные ответы(сутки)', fontweight='bold')
ylabel('Предсказания модели(сутки)', fontweight='bold')
```

```
plt.title("Точность предсказаний модели Случайный лес", fontsize=15,
fontweight='bold')
l = legend(bbox_to_anchor=(0, 0, 1, 1), fontsize = 12)
plt.show()
```

Приложение К. Построение модели Градиентный бустинг решающих деревьев.

```
#Исключение идентификатора цепочки и разделение набора данных на данные и ответы
dataset = dataset.drop(['FIRST_TASKID'], 1)
target = np.array(dataset['FACT_DURATION']).astype(int)
ndata = dataset.drop(['FACT_DURATION'], 1)
data = np.array(ndata)

#Создание объекта модели Градиентный бустинг решающих деревьев
grad_boost = GradientBoostingRegressor()

#Количество разбиений и переменные, хранящие результаты предсказаний модели
n_iter = 30
gb_r2 = 0
gb_mae = 0

for train_indices, test_indices in cross_validation.ShuffleSplit(data.shape[0], n_iter = n_iter, test_size = 0.4):
    #Разбиение данных и отвветов на обучающие и тестовые подвыборки
    train_data = data[train_indices]
    test_data = data[test_indices]
    train_target = target[train_indices]
    test_target = target[test_indices]

    #Обучение модели и сложение точностей модели на разнах разбиениях
    gb = grad_boost.fit(train_data, train_target)
    gb_r2 += gb.score(test_data, test_target)
    gb_mae += mean_absolute_error(test_target, rf.predict(test_data))
```

```
#Вывод средних оценок метрик точности модели
print('RandomForest R^2 score : ', (gb_r2) / n_iter)
print('RandomForest MAE : ', (gb_mae) / n_iter)

#Создание фигуры, на которой располагается график
plt.figure(figsize=(12,8))
plt.cla()
plt.clf()

#Нанесение на график точек, характеризующих ответы модели
scatter = plt.plot(list(map(lambda x: x/86400, test_target)),
                   list(map(lambda x: x/86400, gb.predict(test_data)))
                   ,"b.", label ='Полученные ответы')

#Нанесение на график прямой, характеризующей идеальные предсказания
модели
line1 = plt.plot(np.linspace(0,6000000/86400),np.linspace(0,6000000/86400),"g-",
label ='Идеальные предсказание')

#Нанесение на график области средней ошибки предсказаний модели
line2 = plt.plot(np.linspace((gb_mae / n_iter)/86400,(6000000+gb_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--",label ='MAE')
line3 = plt.plot(np.linspace(-(gb_mae / n_iter)/86400,(6000000-gb_mae /
n_iter)/86400),np.linspace(0,6000000/86400),"r--")

#Установка параметров отображения графика
xlim(0)
ylim(0,70)
xlabel('Правильные ответы(сутки)', fontweight='bold')
```

```
ylabel('Предсказания модели(сутки)', fontweight='bold')
plt.title("Точность предсказаний модели Градиентный бустинг решающих деревьев", fontsize=15, fontweight='bold')
l = legend(bbox_to_anchor=(0, 0, 1, 1), fontsize = 12)
plt.show()

#Формирование таблицы с весами признаков
scores = pd.DataFrame(columns=['Признак', 'Вес'])
n = 0
for i, c in sorted(zip(gb.feature_importances_, ndata.columns)):
    scores.loc[n] = ([i, c])
    n+=1

#Построение диаграммы
val = gb.feature_importances_
pos = np.arange(len(ndata.columns))+.5
plt.figure(figsize=(12,8))
plt.barh(pos,val, align='center')
yticks(pos, ndata.columns)
xlabel('Веса', fontweight='bold')
title('Важность признаков в модели', fontsize=15, fontweight='bold')
grid(True)
```