МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«Новосибирский национальный исследовательский государственный университет»

(Новосибирский государственный университет)

Структурное подразделение Новосибирского государственного университета –

Высший колледж информатики НГУ

КАФЕДРА ЕСТЕСТВЕННОНАУЧНЫХ ДИСЦИПЛИН

**СОЗДАНИЕ БИБЛИОТЕКИ АЛГОРИТМОВ ДЛЯ СТАТИСТИЧЕСКОГО АНАЛИЗА ДАННЫХ КЛИНИЧЕСКИХ ИССЛЕДОВАНИЙ В ПАРАЛЛЕЛЬНЫХ ГРУППАХ**

Дипломный проект

на квалификацию техник

Научный руководитель

к.ф.-м.н. ИВМиМГ СО РАН

Лукинов В.Л.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Студент IV курса,

гр. 14214

Кошкарева С. В.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Новосибирск

2017

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ ТЕРМИНОВ И СОКРАЩЕНИЙ 3](#_Toc493464896)

[ВВЕДЕНИЕ 4](#_Toc493464897)

[1. ПОСТАНОВКА ЗАДАЧИ 7](#_Toc493464898)

[1.1. Описание предметной области 7](#_Toc493464899)

[1.2. Формулировка задачи 7](#_Toc493464900)

[1.3. Функциональные требования 8](#_Toc493464901)

[1.4. Нефункциональные требования 9](#_Toc493464902)

[1.5. Характеристики выбранных технических средств 9](#_Toc493464903)

[2. РЕАЛИЗАЦИЯ 10](#_Toc493464904)

[2.1. ООП модель 10](#_Toc493464905)

[2.1.1. Описание алгоритма поиска опечаток 13](#_Toc493464906)

[2.1.1.1. Реализация метода FindMisprints() для дискретных и категориальных значений 13](#_Toc493464907)

[2.1.1.2. Реализация метода FindMisprints() для непрерывных значений 14](#_Toc493464908)

[2.1.1.3. Реализация метода FindMisprints() для значений, содержащих даты 15](#_Toc493464909)

[2.1.2. Описание алгоритма поиска выбросов 15](#_Toc493464910)

[2.1.3. Описание алгоритма поиска непоследовательных дат 18](#_Toc493464911)

[3. РАЗРАБОТКА БИБЛИОТЕКИ 20](#_Toc493464912)

[4. ОТЛАДКА И ТЕСТИРОВАНИЕ 21](#_Toc493464913)

[ЗАКЛЮЧЕНИЕ 22](#_Toc493464914)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 23](#_Toc493464915)

[ПРИЛОЖЕНИЯ 24](#_Toc493464916)

ПЕРЕЧЕНЬ ТЕРМИНОВ И СОКРАЩЕНИЙ

ВВЕДЕНИЕ

Целью данной работы является разработка программного комплекса алгоритмов, предназначенного для проведения статистического анализа данных, полученных в результате клинических исследований в параллельных группах.

На протяжении своего развития медицина всегда старалась найти более эффективные способы лечения и диагностики. Пройдя путь от интуитивных обобщений, метода проб и ошибок, объединяя разрозненный эмпирический опыт, она перешла к доказательности. Доказательная медицина подразумевает такой подход к медицинской практике, при котором каждое решение, относящееся к выбору метода лечения, должно иметь научное обоснование, которое основывается на данных, полученных в ходе четко спланированного исследования, использующего адекватные методы статистического анализа.

Статистический анализ является неотъемлемой частью практически любого исследования, и только с его помощью можно объективно судить о результатах исследования пополнить доказательную базу. Расчет статистических показателей, которые позволяют оценить достоверность различия, корреляцию и взаимное влияние анализируемых факторов происходит по определенной технологии с использованием математических функций и создания моделей.

В большинстве случаев клинические исследования проводятся в параллельных группах. При проведении исследований в параллельных группах испытуемые двух или более групп получают различные курсы лечения или различные дозы ЛС. Для достижения статистической достоверности испытуемые распределяются по группам методом случайной выборки. Модели исследований в параллельных группах считаются оптимальными для определения эффектов лечения и формулирования выводов на основе полученных результатов.

Ввиду того, что объемы данных и размеры групп (выборок) могут сильно варьировать, а данные быть весьма разнообразными, возникает необходимость в более гибком подходе и выборе методов статистического анализа, подходящих конкретной задаче.

Различные статистические методы предполагают нормальность распределения данных. Такие методы называют параметрическими. Они строятся на основе параметров выборочной совокупности и представляют функции этих параметров, непараметрические - функции от вариант данной совокупности с их частотами. Непараметрические критерии применимы к распределениям самых различных форм. Последние имеют определенные преимущества по сравнению с параметрическими, благодаря меньшим требованиям к их применению, большему диапазону возможностей и, часто, большей простоте реализации. Однако нужно учитывать более низкую точность этих критериев по сравнению с параметрическими.

К сожалению, на практике в ходе сбора данных далеко не всегда получаются полностью и правильно укомплектованные наборы. Опечатки и пропуски отдельных значений являются повсеместным явлением и поэтому, прежде чем начать применять статистические методы, обрабатываемые данные следует привести к «каноническому» виду. Для этого необходимо идентифицировать такие проблемы, а в дальнейшем, либо удалить фрагменты объектов с недостающими элементами, либо заменить имеющиеся пропуски и опечатки разумными значениями. Также еще одной проблемой «сырых» данных является наличие выбросов в исходных исследовательских данных. Под «выбросом» понимается наблюдение, которое слишком велико или слишком мало по сравнению с большинством других имеющихся наблюдений. Чувствительность разных статистических методов к наличию выбросов в данных неодинакова. Так, наличие выбросов может сделать использование некоторых статистических моделей невозможным и в то же время никак не сказаться на результатах другой.

Именно по вышеописанным причинам перед проведением статистического анализа необходимо выполнить проверку начальных данных на валидность, т.к. согласно практике, предварительное исследование данных может занимать до 50% времени, затраченного на анализ.

Валидизация данных является процессом обнаружения и исправления ошибок, таких как: пропущенные данные, выход величин за определенные пределы, последовательность дат, лабораторные показатели относительно нормы, взаимосвязь данных. На данный момент валидизация данных зачастую осуществляется при помощи визуальной «ручной» проверки. При визуальной проверке происходит проверка на полноту присланной документации, содержащей информацию, подлежащую обработке (индивидуальные регистрационные карты, бланки с результатами анализов). Производится визуальная оценка качества указанной информации (некорректные исправления, опечатки, нечитаемые данные. При визуальной проверке может потребоваться медицинская экспертиза (осуществление проверки информации на валидность, требующая специальных медицинских знаний и не реализуемая программными средствами). Программная проверка существенно ускорит дальнейший анализ, а так же окажется более эффективной по сравнению с проверкой, выполняемой визуально. Во время программной валидизации данных выявляются пропущенные данные, опечатки, величины, выходящие за определенные пределы, определяется последовательность всех дат, лабораторные показатели сверяются с референтными пределами. Так же программа должна решать такую специфическую задачу, как определение нормальности распределения данных.

В ходе выполнения дипломной работы была изучена предметная область, выбрано и изучено программное средство, а именно язык R, разработаны и описаны архитектура библиотеки и алгоритмы подпрограмм, написан программный код библиотеки, которая идентифицирует потенциальные проблемы исследования данных, выполнены анализ, тестирование и отладка библиотеки, рассмотрен пример применения на реальных задачах.

1. ПОСТАНОВКА ЗАДАЧИ
   1. Оп**исание предметной области**

При статистическом анализе данных клинических исследований необходимо выполнять рутинные процедуры, поддающиеся автоматизации. Создание единого промышленного программного комплекса алгоритмов статистического анализа данных клинических исследований в параллельных группах, реализованных в виде R-библиотеки с сопутствующей документацией, позволит существенно сократить время проведения статистического анализа и поможет проводить более качественные исследования в сжатые сроки.

* 1. Формулировка задачи

Целью работы являлась реализация системы автоматизированного управления установкой БНЗТ, а именно: разработка консоли оператора, сопряжение периферии с консолью и отображение данных оператору с периодом не менее раза в секунду.

Контроль управляющих узлов источниками питания должен осуществляться при помощи заложенных в них протоколах связи (например, SCMP). Диагностика состояния ускорителя осуществляется через измерение токов, напряжений и температур.

В качестве измерителей параметров были выбраны дистанционные устройства ввода-вывода, управляемые консолью оператора через Ethernet (протокол Modbus).

Так же требуется ведение журнала всех значений с периодом не менее 1 раза в секунду.

Были поставлены и решены следующие задачи:

* Изучена текущая конфигурация установки;
* Произведен анализ требуемых устройств;
* Изучены принципы работы с периферийными устройствами;
* Разработана консоль оператора;
* Налажена связь консоли оператора с устройствами управления и диагностики;
* Разработана база данных для хранения данных об эксперименте;

Программирование осуществлялось на языке R, с использованием следующих пакетов:

* base
* methods
* utils
* grDevices
* xlsx

Использовалась система контроля версий git и репозиторий на сервере GitHub.

* 1. Функциональные требования

В рамках дипломной работы были поставлены следующие задачи:

* Ввод данных должен осуществляется в виде Excel-таблиц или внутренней структуры языка R – таблиц данных (data.frame).
* Поиск пропущенных значений (незаполненных полей).
* Поиск опечаток.
* Поиск «выдающихся значений» или «выбросов».
* Исследование нормальности распределений методами Шапиро-Уилка, Андерсона-Дарлинга, Крамера-фон Мизеса, Лиллиефорса, Шапиро-Франчиа, вывод результатов в виде таблицы в .doc-файле или .csv-файле.
* Сравнение числовых показателей в 2 группах U-критерием Манна-Уитни или t-критерием Стьюдента в зависимости от типа распределений показателя или по выбору пользователя, вывод результатов в виде таблицы в .doc файле или .csv файле, построение графиков boxplot.
* Сравнение категориальных показателей в двух группах точным двусторонним критерием Фишера или критерием хи-квадрат в зависимости от типа распределений показателя, или по выбору пользователя, вывод результатов в виде таблицы в .doc-файле или .csv-файле, построение графиков в виде столбчатых диаграмм.
  1. Нефункциональные требования
* Реализация методами ООП, используя объектную модель S4 в языке R.
* Использование системы контроля версий Git в связке с сервером GitHub или Bitbucket.
* Тестирование созданной библиотеки.
* Использование встроенной графической системы ggplot2.
  1. Характеристики выбранных технических средств

Для написания проекта был выбран язык R и среда разработки RStudio.

R – язык программирования высокого уровня, предназначенный для статистической обработки данных с упором на визуализацию и воспроизводимость. Так же является свободной кроссплатформенной программной средой вычислений с открытым исходным кодом в рамках проекта GNU. R – интерпретируемый язык с интерфейсом командной строки. R сочетает в себе функциональное, процедурное, объектно-ориентированное и рефлексивное программирование, т.е. является мультипарадигменным языком.

Был разработан на основе языка программирования S. С 2011 года поддерживается и развивается организацией R Foundation. Сегодня R является безусловным лидером среди свободно распространяемых систем статистического анализа. R обладает хорошей расширяемостью с помощью пакетов. Каждый такой пакет представляет собой библиотеку, содержащую набор специфических функций. В среде R реализованы многие статистические методы: линейные и нелинейные модели, проверка статистических гипотез, анализ временных рядов, классификация, кластеризация, графическая визуализация. Так же, одной из особенностей языка является поддержка графических возможностей, которая позволяет визуализировать данные в виде различных графиков и диаграмм.

R имеет обширное сообщество и развитую систему поддержки, включающую обновление компонентов среды, интерактивную помощь и различные образовательные ресурсы.

1. РЕАЛИЗАЦИЯ
   1. ООП модель

Поставленные задачи были реализованы с использованием ООП модели S4 языка R.

Созданная логическая структура модели состоит из:

* Файлы
* Колонки
* Ошибки

Ниже, на рисунке представлена диаграмма классов ...

Класс File.

Методы getInputFilePath() и getOutputFilePath() устанавливают значение полей path\_in и path\_out, в которых хранятся полные пути фалов-таблиц для ввода и вывода результата. Соответственно методы setInputFilePath() и setOutputFilePath() необходимы для получения текущих значений этих полей.

Структура данных таблицы, с которой работает библиотека, может состоять из четырех типов значений: дата, непрерывные, дискретные и категориальные (номинальные). Непрерывные переменные (continuous variables) могут принимать любые численные значения, которые естественным образом упорядочены на числовой оси (например, рост, вес). Дискретные переменные (discrete variables) могут принимать счётное множество упорядоченных значений, которые могут просто обозначать целочисленные данные или ранжировать данные по степени проявления на упорядоченной ранговой шкале (клиническая стадия опухоли, тяжесть состояния пациента). Категориальные переменные (categorial variables) являются неупорядоченными и используются для качественной классификации (пол, цвет глаз, место жительства); в частности, они могут быть бинарными (дихотомическими) и иметь категорические значения: 1/0, да/нет, имеется/отсутствует. Поэтому для каждый из четырех типов значений описывает свой класс и для каждого из них реализован свой метод поиска ошибок.



Родительский класс Column имеет поле column\_index, в котором хранится номер колонки в исследуемой таблице, и два метода geColumnIndex() setColumnIndex() для задания нового значения этого поля и получения текущего. Данные методы доступны во всех четырех дочерних классах.

Объекты класса Continuous описывают колонки таблицы с непрерывными значениями. Метод FindErrors() состоит из вызова метода FindMisprints() для поиска опечаток и метода FindOutliers() для поиска выбросов.

Объекты класса Discrete описывают такие колонки таблицы, значения которых представляют собой дискретные переменные. Для реализации не только поиска, но и исправления опечаток была разработана структура словаря, состоящая из множества ключей и соответствующего им одного значения. Поля объекта Discrete, key (ключ) и value (значение), являются переменными R-типа list, т.е. они имеют вложенную структуру, представляющую из себя массив массивов, и могут содержать в себе сочетания любых типов данных. Это позволяет эффективно, т.е. в одном объекте, хранить разнородную информацию. Более подробное описание применения словаря описано ниже, в п.

Для класса Discrete были созданы методы задания новых ключей и значений словаря, а также, методы получения текущих. Пользователь может сам задать новые значения этих полей объекта, используя созданные методы setValue() и setKey(), а также получить их текущие значения при помощи getValue() и getKey() соответственно. Эти методы также доступны в дочернем классе Binary(), который уже имеет более узкую специализацию и описывает бинарные категориальные переменные. Во время инициализации класса Binary() значения для поля value устанавливаются по умолчанию: 0 и 1. При необходимости, значения по умолчанию могут быть изменены при помощи родительского метода setValue().

Объекты класса Dates нужны для описания элементов таблицы, содержащих даты.

На рисунке изображена логическая структура опечаток.



Как было сказано ранее, процесс валидации сырых входных данных должен выявлять следующие типы ошибок:

* Опечатки.
* Непоследовательные даты.
* Выбросы.
* Пропущенные значения.

Для каждой из перечисленных ошибок был создан класс, все вместе они объединены в единую иерархическую структуру

Класс-родитель Error имеет следующие поля: indices, которое хранит индексы ячеек таблицы, содержащих ошибки, поле style используется для хранения названия стиля, который должен быть применен для раскраски неправильно заполненных ячеек, поле title содержит строку с названием типа ошибки, являющуюся, по сути, легендой результирующей таблицы, а col\_index\_legend хранит позицию ячейки таблицы, в которой будет размещена легенда.

Значения для полей style, title, col\_index\_legend устанавливаются во время инициализации каждого из дочерних классов Error. Для этого используется созданный метод initialize(), а не стандартный конструктор. Значение поля indices изменяется по мере вызовов метода FindErrors() для каждой из колонок таблицы.

SetColor()

AddTableLegend()

PrintReport()

* + 1. Описание алгоритма поиска опечаток

Поиск опечаток, а также, по возможности, их исправление, осуществляется при помощи метода FindMisprints(). У него существует три реализации, для каждого из четырех типов значений, описанных выше, которые может содержать исследуемая таблица.

* + - 1. Реализация метода FindMisprints() для дискретных и категориальных значений

Осуществляется доступ к элементам определенной колонки в исходной таблице данных по индексу, который был передан при вызове метода. Полученные значения элементов присваиваются новой переменной, представляющей собой вектор, или в более привычном понимании, одномерный массив. В цикле, последовательно проверяется каждый элемент этой переменной на предмет различных ошибок.

В первую очередь осуществляется поиск пропущенных значений. Если проверка выявила существование пустого элемента в колонке, то его индекс передается полю объекта класса missingValue для дальнейшей раскраски, и вызывается метод PrintReport(), который производит запись сообщения о найденной ошибке и ее координатах в пользовательский текстовый файл-отчет. Строка сообщения выглядит следующим образом: Тип ошибки в номер строки строке номер столбца столбце. Тип ошибки определяется автоматически, путем получения экземпляра класса.



Если результат поверки на заполнение оказался успешным и значение ячейки отлично от пустой строки, то начинается проверка на совпадение этого элемента с одним из значений словаря. Для корректности сравнения, значение словаря и значение элемента колонки приводятся к верхнему регистру, во избежание ошибок.

Если значение колонки не совпало ни с одним значение словаря, проверяется совпадение элемента с одним из ключей. Оба значения так же приводятся к верхнему регистру. Если совпадение найдено, то по нужному индексу в итоговой таблице производится замена ключа на соответствующее ему значение в словаре элемента передаются в переменную-слот объекта класса Misprint для раскраски итоговой таблицы. Происходит вызов метода PrintReport(). Данное значение в итоговой таблице будет выделено как исправленная опечатка.

В случае, когда элемент колонки не совпал ни с одним из ключей, индексы этого элемента в таблице (номер строки и номер колонки) передаются в переменную-слот объекта класса UnsolvedMisprint, и производится запись сообщения об ошибке в файл при помощи PrintReport().

* + - 1. Реализация метода FindMisprints() для непрерывных значений

Так же, как и в методе, реализованном для категориальных переменных, в первую очередь внутри метода производится доступ к определенной колонке и проверка ее элементов на пропуски.

Далее значение элемента колонки сравнивается с шаблоном, заданным регулярным выражением, которое имеет тип: начало строки одно или более чисел, далее может быть (разделитель точка или запятая (если значение – это десятичная дробь) одно число и более), конец строки. Если значение элемента не соответствует шаблону, это может означать, что оно содержит буквы, или любые другие символы. Следовательно, индексы элемента передаются слоту объекта класса UnsolvedMisprint, который, как было сказано ранее, хранит в себе индексы элементов таблицы, которые будут раскрашены как неисправленные опечатки. Далее производится запись сообщения о найденной ошибке в файл при помощи PrintReport().

* + - 1. Реализация метода FindMisprints() для значений, содержащих даты

Данная реализация схожа с предыдущей, но отличается шаблоном регулярного выражения, применяемым для сравнения со значением элемента. Он имеет вид:.

Если значение элемента не соответствует шаблону, то производится запись сообщения о найденной опечатке в пользовательский файл, а индексы этого элемента передаются слоту объекта класса UnsolvedMisprint.

Все значения элементов колонки, которые подошли под указанный выше шаблон, являются датами, но существует возможность того, что они записаны неверно. Поэтому производится дополнительная проверка разделителей между числами в записи даты и замена их на точку, если существует ошибка. Далее, вызванный метод PrintReport() сообщает о найденной и исправленной опечатке и добавляет индексы таких элементов слоту объекта класса Misprint.

* + 1. Описание алгоритма поиска выбросов

Поиск выбросов осуществляется только среди значений элементов колонок, которые описывает класс Continuous, т.е. непрерывных переменных. Анализ возможен лишь в том случае, когда среди них нет неисправленных опечаток. В противном случае подобная оценка будет ошибочной и, следовательно, бесполезной.

Выбросы в столбце определяются при помощи функции boxplot.stats(), производной графической функции высокого уровня boxplot(), которая служит для построения диаграмм размахов. Boxplot.stats(), в свою очередь, используется для сбора статистики, необходимой для создания диаграмм размахов.

Диаграммы размахов, или "ящики с усами" (англ. box-whisker plots), получили свое название за характерный вид: точку или линию, соответствующую медиане или средней арифметической, окружает прямоугольник ("ящик"), длина которого соответствует одному из показателей разброса или точности оценки генерального параметра. Дополнительно от этого прямоугольника отходят "усы", также соответствующие по длине одному из показателей разброса или точности. Графики этого типа очень популярны, поскольку позволяют дать очень полную статистическую характеристику анализируемой совокупности. Кроме того, диаграммы размаха можно использовать для визуальной экспресс-оценки разницы между двумя и более группами (например, между датами отбора проб, экспериментальными группами, участками пространства, и т.п.). Строение получаемых при помощи этой функции "ящиков с усами" представлено ниже на рисунке 1:

  [](http://3.bp.blogspot.com/-sqSGopnp0lo/Uvu_wl_dPQI/AAAAAAAAAgs/F2DBOSdfiU4/s1600/boxplot.PNG)

В R при построении диаграмм размахов используются устойчивые (робастные) оценки центральной тенденции (медиана) и разброса (интерквартильный размах, далее ИКР). Верхний "ус" простирается от верхней границы "ящика" до наибольшего выборочного значения, находящегося в пределах заданного расстояния, чаще всего используется 1.5 х ИКР от этой границы. Аналогично, нижний "ус" простирается от нижней границы "ящика" до наименьшего выборочного значения, находящегося в пределах расстояния 1.5 х ИКР от этой границы. Наблюдения, находящиеся за пределами "усов", потенциально могут быть выбросами.

В первую очередь выполняется проверка значения счетчика неисправленных опечаток. Если он равен нулю, то это означает, что можно приступить к анализу значений элементов колонки. Иначе, происходит вызов метода печати в файл PrintReport() и запись предупреждения о невозможности проведения анализа, т.к. среди значений определенной колонки присутствуют опечатки, а так же сообщается номер колонки.

Далее осуществляется доступ к элементам колонки по переданному индексу и поиск среди них пропущенных значений.  
Функция boxplot.stats() работает только с численными значениями. Опечатки в колонке таблицы, содержащей непрерывные переменные (например, пробел после запятой в десятичной дроби или разные разделители между числами), препятствуют правильному распознаванию их значений системой, как численные, вместо этого они представляются как значения строкового типа. Именно поэтому, для того, чтобы включить эти значения в анализ выбросов, был использован шаблон, заданный при помощи регулярного выражения: начало строки одно или более чисел разделитель точка или запятая одно и более чисел конец строки. Все значения элементов колонки сравниваются с данным шаблоном и при совпадении сохраняются в новую переменную, используемую для анализа. Далее, среди значений этой переменной производится замена разделителей в десятичных дробях на точку. После замены используется приведение типов каждого элемента переменной к численному, и выполняется поиск выбросов с помощью boxplot.stats(), которая анализирует сазу весь массив. Результат работы функции хранится в виде сложной структуры массива массивов. Значения (не их индексы) из переданного массива, потенциально являющимися выбросами, содержатся в элементе out. Если он не пустой, то полученные из него значения сопоставляются с исходными значениями колонки, для того, чтобы определить позицию уже найденных выбросов в таблице. Для сравнения разделители в исходных элементах, все еще являющихся строковыми, тоже преобразуются. Во время вызова метода PrintReport() происходит печать сообщения о найденных выбросах и их позиции в таблице (номера строки и столбца).

Однако всегда следует внимательно относиться подобным нестандартным наблюдениям - они вполне могут оказаться "нормальными" для исследуемой совокупности, и поэтому не должны удаляться из анализа без дополнительного расследования причин их появления.

* + 1. Описание алгоритма поиска непоследовательных дат

Метод FindDateMisprints() реализует поиск непоследовательных дат внутри таблицы. Он принимает два объекта класса Dates, которые описывают колонки с разными датами, которые предстоит проверить. Например, это могут быть даты первичных и повторных замеров или даты поступления и выписки. Они сортируются по индексу колонки. Затем применяется преобразование каждого элемента обеих колонок к типу данных date с форматом представления месяц.день.год с указанием века и присваивание полученных значений двум новым переменным. Далее эти переменные поэлементно сравниваются между собой. Если какое-либо из значений первой колонки дат оказалось больше значения второй колонки дат в этой же строке, то его индексы строки и столбца добавляются в поле indices объекта DateMisprint. В результирующей таблице будут выделены обе непоследовательные в строке.

Описание функционала

Данный функционал необходим для того, чтобы перед началом проверки сырых данных таблицы на валидность, пользователь получил представление о значениях, которые она содержит и, исходя из этого, имел возможность верно сопоставить… а так же задать ключи и значения словарей, которые будут использоваться для поиска и исправления опечаток.

Реализацией данного функционала является метод TableColumnForExcel(). Новой переменной присваиваются названия колонок исходной таблицы, полученные при помощи функции colnames(). Создается новая пустая таблица данных. Далее, в цикле к элементам каждой колонки таблицы применяется функция table(). Ее результат содержит все различные значения столбца и частоту встречаемости каждого из этих значений. Производится преобразование этого результата и приведение его к нужному формату, путем приведения типов и транспонирования матрицы. Добавляется в созданную ранее таблицу данных полученный результат в виде новых строк. Т.к. количество различных значений в разных столбцах таблицы разное (для сравнения колонка пол содержит четыре различных значения, а колонка возраст…), то это создает преграды для создания нужной таблицы обычным путем, т.к. строки должны быть одной длины. Поэтому для решения этой проблемы использовалась функция rbind.fill(), которая заменяет недостающие столбцы NA. Если количество различных значений в одной колонке больше десяти, то для отображения в итоговой таблице берутся только первые десять значений, о чем указывается в шапке файла.

Вставка названия колонки исходной таблицы перед каждой новой строкой, описывающей содержимое столбца в результирующей таблице данных.

Работа с библиотекой xlsx, создание новой рабочей книги, добавление таблицы данных. Установление стилей, объединение ячеек. Сохранение.

1. РАЗРАБОТКА БИБЛИОТЕКИ
2. ОТЛАДКА И ТЕСТИРОВАНИЕ

ЗАКЛЮЧЕНИЕ

В ходе работы были проведен обзор средств работы с большими данными и на их основе разработана система в рамках лямбда архитектуры. Система удовлетворяет всем поставленным требованиям:

• Легко горизонтально масштабируема

• Обладает высокой отказоустойчивостью

• Обладает высоким быстродействием

• Легко расширяема на разные источники данных

• Реализует требуемый функционал

В рамках работы была доказана целесообразность применения технологий работы с большими данными, применительно к области улучшения безопасности дорожного движения. На основании результатов, полученных с помощью этой системы, могут быть предприняты мероприятия по снижению аварийности дорог. Также система эффективна как мониторинговая.

Система обладает большими возможностями и потенциалом к дальнейшему развитию и улучшению. Таким образом, все поставленные задачи в ходе работы были выполнены, а цель работы можно считать достигнутой.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЯ