СОДЕРЖАНИЕ

ВВЕДЕНИЕ..............................................................................................................3

1. Постановка задачи...............................................................................................5

2. Обзор и выбор инструментов для анализа данных..........................................6

3. Описание алгоритма анализа и визуализации данных....................................9

4. Реализация проекта...........................................................................................11

4.1Пример для работы с диаграммами......................................................12

4.2 Пример для работы с графами.............................................................17

ЗАКЛЮЧЕНИЕ.....................................................................................................25

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ...................................................26

ПРИЛОЖЕНИE А Листинг программы..............................................................27

# **ВВЕДЕНИЕ**

Одним из основных инструментов познания мира является обработка данных, получаемых человеком из различных источников [2]. Современный статистический анализ заключается в исследовании и визуализации получаемой информации.

В последнее время методы анализа и визуализации данных сильно изменились. С появлением ЭВМ и Интернета объёмы данных увеличились. Коммерческие и государственные организации обладают огромнейшими хранилищами данных. И соответственно, с увеличением объёма данных, представление этой информации в легкодоступном виде становится более сложной.

Развитие наук, посвященных анализу данных не отстает от роста объема данных. До эпохи ЭВМ и Интернета новые методы изучались учёными, и публиковались в журналах, газетах. И только спустя годы они доходили до программистов для статистической обработки данных. А сейчас новые методы появляются каждый день. Статисты постоянно публикуют в свободном доступе новые методы анализа с готовым программным кодом, который реализует эти методы.

Появление ЭВМ изменило подход к анализу данных еще одним образом [2]. Когда анализ данных проводился в вычислительных центрах, аналитики заранее устанавливали все параметры для анализа. А когда анализ завершался, вывод результата занимал сотни страниц. Аналитику приходилось оставлять нужное и отсеивать лишнее. Многие статистические программы и сейчас придерживаются этого алгоритма.

С появлением дешевого и легкого доступа к анализу данных, предоставляемого ЭВМ, произошла смена парадигмы. В плане предварительной установки параметров анализа данных процесс стал в значительно удобнее. При этом результат каждого этапа анализа служит данными для следующего этапа. На любом этапе анализа может быть произведено редактирование загруженных показателей, таких как добавление, замена или удаление значений, после чего процесс продолжается. Завершается этот процесс тогда, когда аналитик считает, что он полностью исследовал данные и ответил на все относящиеся к делу вопросы, на которые можно было ответить.

Очень многие системы и явления представимы в виде набора объектов и связей между ними. Это наглядный инструмент анализа и визуализации данных. Можно отобразить важность того или иного объекта, указать ключевые группы элементов, выделить их и подчеркнуть связи между ними. Главная задача анализа и визуализации — показать главную информацию о свойствах системы или явления максимально легким для восприятия способом. В идеальном случае анализ системы и визуализацию его результатов можно сделать в рамках одного инструмента. Например R с его обширным набором пакетов позволяет это.

В данном курсовом проекте поставлена задача анализа и визуализации данных с веб-сайтов, которая позволила бы пользователю наглядно отследить статистику просмотров и комментариев каких-либо постов.

# **1.Постановка задачи**

Цель**:** Освоить средства и методы сбора и визуализации данных с помощью языка R.

Необходимо разработать программный код на языке R, который позволит собирать данные с сайта о названии, рейтинге, количестве комментариев и реализовывать визуализацию этих данных для упрощения дальнейшего процесса аналитики.

Проанализировав поставленную задачу, я пришёл к выводу, что для лучшей читабельности и поддержки целесообразно написать две функции.

Первая функция - извлечение статистических данных с веб-страницы:

GetStat (Значение1),

где Значение1 - является узлом HTML кода, который содержит либо название статьи, либо рейтинг статьи, либо количество комментариев.

Вторая функция - визуализация данных:

SetView(Значение2)

где Значение2 - это переменная, которая содержит тип визуализации:

1. диаграмма (круговая, либо столбиковая);
2. граф.

2. Обзор и выбор инструментов для анализа данных

Перед началом выполнения поставленной задачи были проанализированы зарубежные и отечественные источники на тему: «Популярность языков анализа данных». В соответствии с веб-сайтом американской компании по бизнес-аналитике проводились голосования на выбор программного обеспечения для анализа данных в 2012, 2013, 2014 годах. Топ-3 голосов составили языки R, SAS, Python. И на первом месте лидировал язык R. Дело в том, что в отличие от своих конкурентов язык R имеет более профильную специализацию, так как его используют в основном для статистической обработки данных и построения графиков. Это делает язык R основным инструментом, который применяется специалистами по анализу данных, и одновременно ограничивает сферу применения.

Таблица 1. Сравнительная характеристика средств анализа данных[8]

|  |  |
| --- | --- |
| **What programming/statistics languages you used for an analytics / data mining / data science work in 2014?** | |
| **Language used** | http://www.kdnuggets.com/polls/graph/g0.gif % voters in 2014 (719 total)  http://www.kdnuggets.com/polls/graph/g4.gif % voters in 2013 (713 total)  http://www.kdnuggets.com/polls/graph/g8.gif % voters in 2012 (579 total) |
| R (352 voters in 2014) | http://www.kdnuggets.com/polls/graph/g0.gif 49.0% http://www.kdnuggets.com/polls/graph/g4.gif 60.9% http://www.kdnuggets.com/polls/graph/g8.gif 52.5% |
| SAS (262) | http://www.kdnuggets.com/polls/graph/g0.gif 36.4% http://www.kdnuggets.com/polls/graph/g4.gif 20.8% http://www.kdnuggets.com/polls/graph/g8.gif 19.7% |
| Python (252) | http://www.kdnuggets.com/polls/graph/g0.gif 35.0% http://www.kdnuggets.com/polls/graph/g4.gif 38.8% http://www.kdnuggets.com/polls/graph/g8.gif 36.1% |
| SQL (220) | http://www.kdnuggets.com/polls/graph/g0.gif 30.6% http://www.kdnuggets.com/polls/graph/g4.gif 36.6% http://www.kdnuggets.com/polls/graph/g8.gif 32.1% |
| Java (89) | http://www.kdnuggets.com/polls/graph/g0.gif 12.4% http://www.kdnuggets.com/polls/graph/g4.gif 16.5% http://www.kdnuggets.com/polls/graph/g8.gif 21.2% |

Продолжение таблицы 1.

|  |  |
| --- | --- |
| Unix shell/awk/sed (63) | http://www.kdnuggets.com/polls/graph/g0.gif 8.8% http://www.kdnuggets.com/polls/graph/g4.gif 11.1% http://www.kdnuggets.com/polls/graph/g8.gif 14.7% |
| Pig Latin/ Hive/ other Hadoop-based languages (61) | http://www.kdnuggets.com/polls/graph/g0.gif 8.5% http://www.kdnuggets.com/polls/graph/g4.gif 8.0% http://www.kdnuggets.com/polls/graph/g8.gif 6.7% |
| SPSS (58) | http://www.kdnuggets.com/polls/graph/g0.gif 8.1% not asked not asked |
| MATLAB (45) | http://www.kdnuggets.com/polls/graph/g0.gif 6.3% http://www.kdnuggets.com/polls/graph/g4.gif 12.5% http://www.kdnuggets.com/polls/graph/g8.gif 13.1% |
| Scala (28) | http://www.kdnuggets.com/polls/graph/g0.gif 3.9% http://www.kdnuggets.com/polls/graph/g4.gif 2.2% http://www.kdnuggets.com/polls/graph/g8.gif 2.4% |
| C/C++ (26) | http://www.kdnuggets.com/polls/graph/g0.gif 3.6% http://www.kdnuggets.com/polls/graph/g4.gif 9.3% http://www.kdnuggets.com/polls/graph/g8.gif 14.3% |

Соответственно для анализа и обработки данных был выбран язык программирования R. R – это язык программирования и среда для статистических вычислений и графического анализа, который был первоначально разработан в Bell Laboratories. Это программа для анализа данных с открытым кодом, которая поддерживается большим и активным исследовательским сообществом по всему миру. Как было сказано, что помимо R существует много распространенных программ для статистической и графической обработки данных. Почему именно язык R?

У R есть много достоинств, которые могли бы рекомендовать эту программу [3].

* многие статистические программы стоят больших денег. Среда R полностью бесплатна. Если вы студент, то выбор очевиден;
* R – это мощная статистическая программа, в которой реализованы все способы анализа данных;
* R имеет самые современные графические возможности. Если требуется визуализировать сложные данные, то в R реализованы самые разнообразные и мощные методы анализа данных из доступных;
* получение данных из разных источников. R может импортировать данные из веб-сайтов, текстовых файлов, Excel файлов, систем управления базами данных, и других хранилищ данных. R может также экспортировать данные в форматах всех этих систем;
* R не имеет аналогов для простого написания программ, которые реализуют новые статистические методы;
* R работает на многих операционных системах, включая Windows, Unix и Mac OS X. Она, возможно, запустится на любом компьютере, который у вас есть.

3. Описание алгоритма анализа и визуализации данных

Начало

Импорт данных

Подготовка данных, знакомство с ними, исправление ошибок

Подбор статистической модели

Оценка модели на новых данных

Результат

Конец

Рисунок 1 - Блок-схема процесса обработки данных

Теперь рассмотрим по пунктам:

* Импорт данных. Чтобы работать с данными, нам нужно их сначала получить. Данные считываются с веб-страниц в векторы. А затем заносятся в Excel таблицу или обрабатываются в R.
* Подготовка данных, знакомство с данными. Исправление ошибок. Здесь нужно ознакомиться с данными, правильно ли мы их считали и так далее.
* Подбор статистической модели. Если данные считаны правильно, то приступаем к подбору нужных нам моделей для визуализации данных.
* Оценка модели на новых данных. В этом пункте мы смотрим на адекватность нашей модели, а именно, полностью ли мы исследовали данные и ответили на все относящиеся к делу вопросы, на которые можно было ответить.
* Результат. Результат интерпретации программного кода.

4. Реализация проекта

Процесс накопления данных с веб-сайта начинается с извлечения всех названий статей, их рейтинга и количества комментариев под каждой статьёй. Для этого были выделены узлы страницы, которые отвечают за искомую информацию. Далее по сформированным таблицам строятся круговая и столбиковая диаграммы, которые отображают рейтинг статей или популярность статьи по количеству комментариев.

Перед тем как начать считывать данные, нужно загрузить нужный пакет для работы с HTML страницами rvest, среди возможностей которого есть функции, которые нам требуются. Это следующие функции:

* read\_httml() - выполняет парсинг целевой страницы;
* html\_nodes() - выбирает узел из HTML кода страницы;
* html\_text() - считываем текст по указанному узлу.

Для визуализации данных с помощью графов требуется загрузить пакет igraph. Будет использована функция plot() из этого пакета.

Для работы с Excel обычно используется пакет csvread.

Устанавливаем требуемые библиотеки и подключаем их.

install.packages("rvest")

install.packages("igraph")

install.packages("csvread")

library("rvest")

library("igraph")

library("csvread")

4.1 Пример для работы с диаграммами

Выполняем парсинг нужной веб-страницы.

url<-read\_html("http://habrahabr.ru/top/")

Теперь нужно определить, какие фрагменты HTML-кода нам необходимы. Веб-страница представляет собой сложную структуру, которая состоит из вложенных объектов, называемую объектной моделью документа (DOM). Соответственно, требуется выяснить, какие фрагменты DOM нужны. Можно воспользоваться расширением Google Chrome, которое называется SelectorGadjet. При наведении на нужные элементы сайта будет виден узел HTML кода нужного элемента.

Нам требуются: название статьи, рейтинг и количество комментариев. Заносим узлы HTML кода в переменные.

selector\_count<-".js-favs\_count"

selector\_post<-".post\_title"

selector\_comment<-".post-comments\_\_link\_all"

Далее с помощью html\_nodes() выбираем узлы из HTML страницы и извлекаем в вектор fpost название статьи, в вектор fcount рейтинг и в вектор fcomment количество комментариев с помощью функции html\_text(). Оператор вида "%>%" представляет собой оператор THEN.

fcount<-html\_nodes(url, selector\_count) %>%

html\_text()

fpost<-html\_nodes(url, selector\_post) %>%

html\_text()

fcomment<-html\_nodes(url, selector\_comment) %>%

html\_text()

Данные уже в R! Вывести на экран их довольно просто:

fpost

fcount

fcomment

Так как были считаны названия статей с русских веб-страниц, они поместились в вектор в непонятной кодировке CP1251. Чтобы конвертировать в кодировку UTF-8, необходимо использовать функцию iconv().

fpost <- iconv(fpost, "UTF-8","CP1251")

Теперь нужно подготовить данные для построения моделей(диаграмм). Рейтинг и количество комментариев считались в вектор как строки. Преобразуем их в числовой формат с помощью функции as.numeric().

fcount.num<-as.numeric(fcount)

fcomment.num<-as.numeric(fcomment)

Занесём в вектор нужные значения и дадим каждому элементу вектора название.

pie.rate <- c((fcount.num[1]), (fcount.num[2]), (fcount.num[3]), (fcount.num[4]), (fcount.num[5]), (fcount.num[6]))

names(pie.rate) <- c(fpost[1], fpost[2], fpost[3],fpost[4], fpost[5], fpost[6])

Данные полностью готовы для построения диаграмм. Построим для рейтинга статей круговую диаграмму, а для количества комментариев статей столбиковую диаграмму.

Строим круговую диаграмму с помощью функции pie(). Функция pie() имеет несколько аргументов (можно посмотреть подробнее командой ? pie). Основными из них являются следующие:

* x - вектор из положительных чисел, на основе которых строится диаграмма;
* labels - текстовый вектор, содержащий подписи секторов диаграммы; если значения x уже имеют атрибут names (имена), то аргумент labels указывать не обязательно;
* radius - изменяет размер квадрата, внутри которого строится диаграмма; в
* случаях, когда подписи секторов диаграммы слишком длинные, размер этого
* квадрата можно уменьшить (возможные значения: от -1 до 1);
* init.angle - угол поворота диаграммы;
* col - вектор (числовой или текстовый), содержащий коды цветов для заливки секторов диаграммы;
* main - текстовый вектор, содержащий заголовок диаграммы;

Строим:

pie(pie.rate)

Получаем следующий результат:

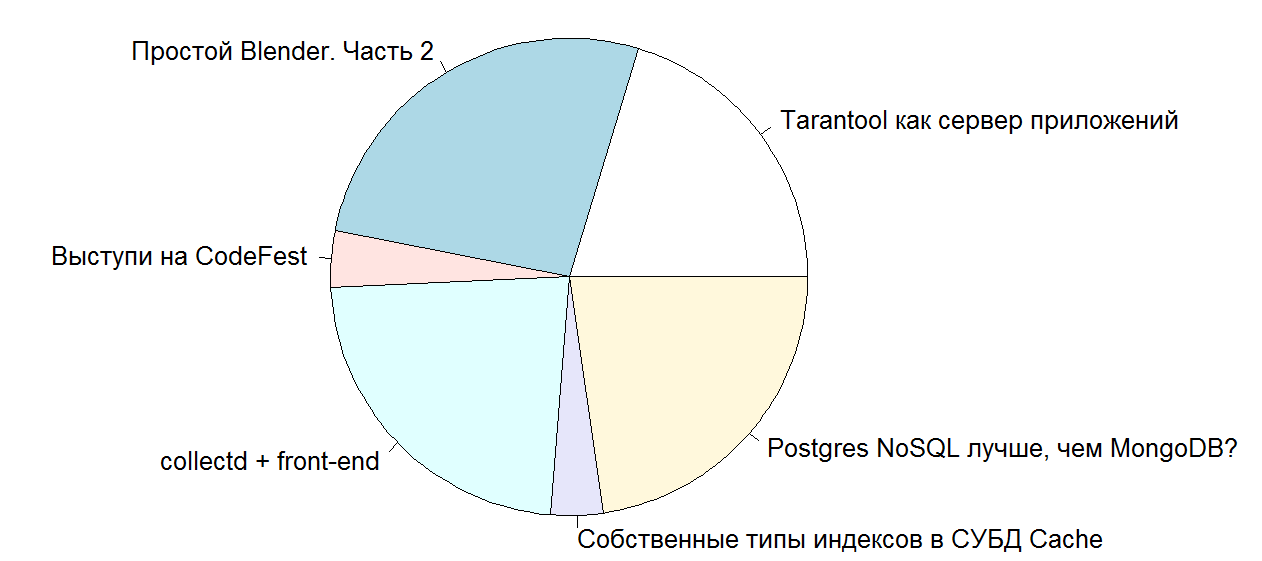
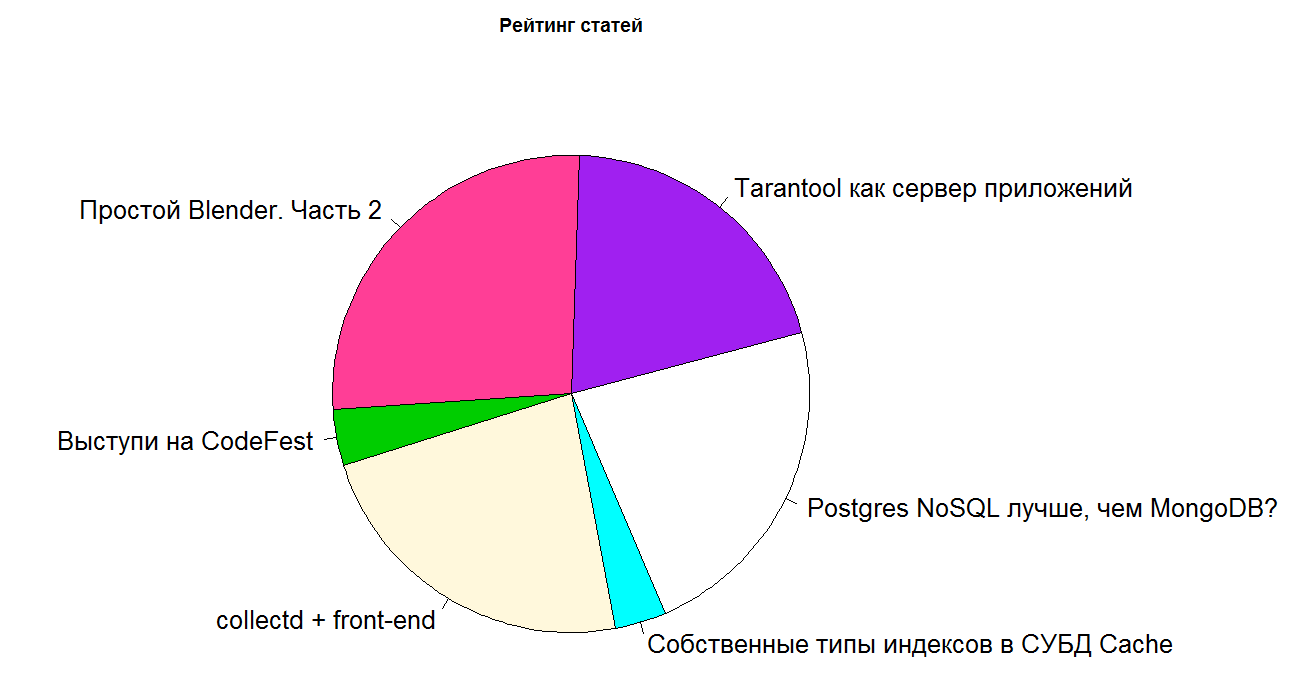


Рисунок2 - Круговая диаграмма, построенная по считанным данным

Добавим несколько аргументов в функцию pie(), а именно название диаграммы, цвета кусков диаграммы, повернём её на несколько градусов и зададим радиус.

pie(pie.rate, radius=0.8,init.angle = 15, col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "white"),main = "Рейтинг статей")



**Рейтинг статей**

Рисунок 3 - Отредактированная круговая диаграмма, построенная по считанным данным

Из диаграммы отчётливо видно, какие статьи более популярны, а какие совсем утратили свою популярность.

Теперь для количества комментариев к каждой статье построим столбиковую диаграмму с помощью функции barplot(). У этой функции имеется большое количество аргументов (можно посмотреть подробнее командой ? barplot), к основным из которых относятся:

* height ("высота") - числовой вектор или матрица со значениями, используемыми для построения диаграммы. Если аргумент hight указан в виде вектора, то строится график из последовательно расположенных столбцов, высоты которых соответствуют значениям этого вектора. Если hight указан в виде матрицы и аргумент beside = FALSE, то будет построена столбчатая диаграмма с накоплением. Если же hight указан в виде матрицы и аргумент beside = TRUE, то столбцы диаграммы будут сгруппированы в соответствии со столбцами матрицы.
* col - вектор (числовой или текстовый), содержащий коды цветов для заливки секторов диаграммы;
* xlab - подпись вдоль оси X;
* ylab - подпись вдоль оси Y;

Строим столбиковую диаграмму и добавляем легенды:

barplot(height=c(fcomment.num [1], fcomment.num[2], fcomment.num[3],

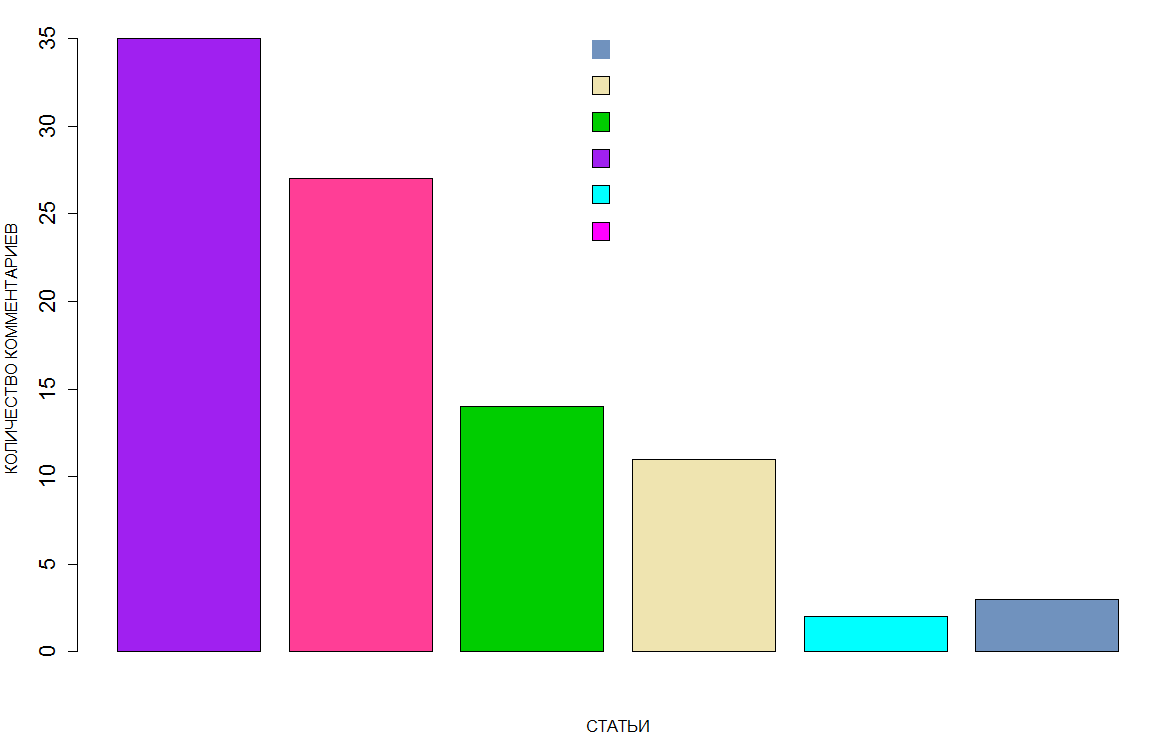
fcomment.num[4], fcomment.num[5], fcomment.num[6]),

col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "white"),

xlab = "Статьи", ylab = "Количество комментариев")

legend("topright", legend = c(fpost[1], fpost[2], fpost[3],fpost[4], fpost[5], fpost[6]), fill = 1:6, cex = 0.55)

Получим следующий результат:



Tarantool как сервер приложений

Простой Blender. Часть 2

Выступи на CodeFest

collectd + front-end

Типы индексов в Cache

Postgres NoSQL лучше, чем Mongo DB

Рисунок 4 - Столбиковая диаграмма, построенная по считанным данным

4.2 Пример для работы с графами

В следующем примере я покажу работу с графами на основе считанных данных с веб-сайтов. Где вершины будут соответствовать статьям сайта, их размер рейтингу страницы, а направление рёбер - похожим статьям в конце каждой статьи, из рассматриваемых.

Данные будут храниться в двух Excel файлах. В одном - вершины(название статей), их рейтинг и любые значения, которые будут использоваться для цвета вершин. В другом - направление рёбер, т.е в каких статях есть ссылки на рассматриваемые статьи.

Для работы с графами требуется загрузить пакет. И соответственно пакет для работы с HTML. Так как данные будут храниться в Excel, нужно подключить пакет для работы с Excel.

Считываем данные таким же способом, как и в прошлом примере, только с каждой веб-страницы статьи: название статей, их рейтинг и похожие статьи в каждой из статьей.

url1<-read\_html("http://habrahabr.ru/post/108443/")

selector\_similar<-".post-preview\_\_link"

frate1<-html\_nodes(url1, selector\_rate) %>%

html\_text()

Подготавливаем данные для занесения в Excel.

id<-c('s1',"s2","s3","s4","s5","s6")

media<-c(ftitle1,ftitle2,ftitle3,ftitle4,ftitle5,ftitle6)

media.type<-c(1,2,3,4,5,6)

size<-c(frate1,frate2,frate3,frate4,frate5,frate6)

audience.size<-as.numeric(size)

Формируем таблицу в R:

DF <- data.frame(id, media, media.type, audience.size)

И записываем эту таблицу в Excel.

write.table(DF,"nodes.csv",sep=',')

Разберёмся со вторым файлом. Здесь нам требуются направления рёбер. Перебираем все возможные варианты с добавлением значений в вектор с помощью цикла for(). И записываем вектор в Excel.

for (i in 1:6)

{

for (k in 1:3)

{

if (media[i]==fsimilar1[k]) s<-append(s,paste0("s",i,",","s",k))

}

write.table(s,"edge.csv",sep=',')

Функция append() добавляет новое значение в вектор, а функция paste0() "склеивает" строки, указанные в аргументе

Данные полностью готовы для визуализации и выглядят они следующим образом:

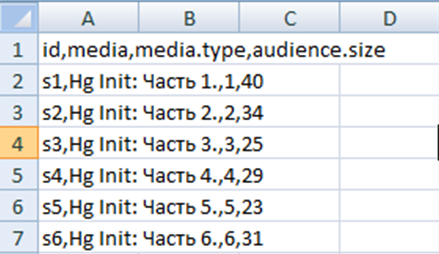


Рисунок 5 - Входные данные 1, считанные с веб-страниц

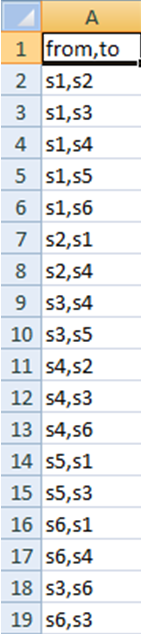


Рисунок 6 - Входные данные 2, считанные с веб-страниц

Считываем данные для графа с Excel.

nodes <- read.csv("nodes.csv", header=T)

links <- read.csv("edges.csv", header=T)

Связываем вершины и рёбра графа функцией graph.data.frame(), указываем в аргументе directed, для направленности рёбер. И довольно просто изображаем граф функцией plot().

net <- graph.data.frame(links, nodes, directed=T)

plot(net)

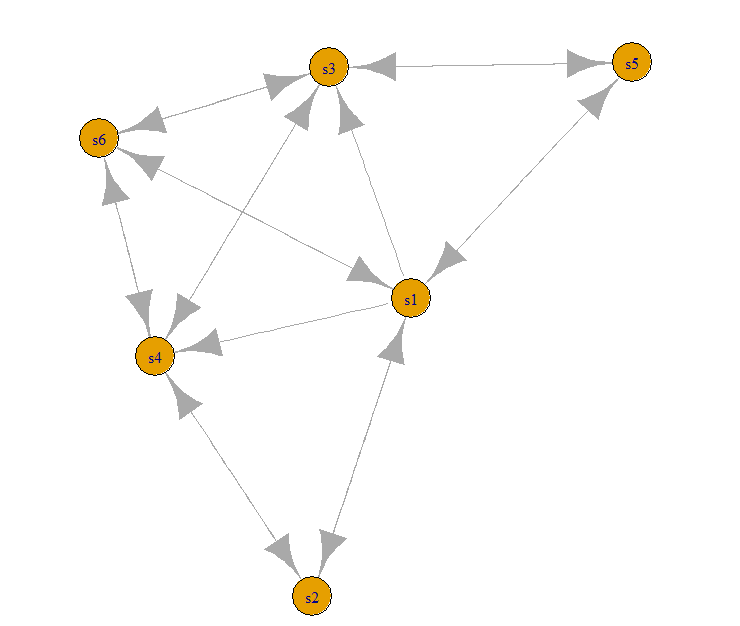


Рисунок 7 - Граф 1, построенный по считанным данным

Получилось не очень красиво. Приведём в порядок наш граф, изображённый на рисунке 7. Заменим вершины на название статей с помощью аргумента vertex.lable, зададим размер стрелок, с помощью аргумента edge.arrow.size. Поменяем цвет названия статей(vertex.label.color), вершин(vertex.color), рёбер(edge.color) и контура вершин(vertex.frame.color).

plot(net, edge.arrow.size=.5, edge.color="orange",

vertex.color="orange", vertex.frame.color="#FF0000",

vertex.label=V(net)$media, vertex.label.color="black")

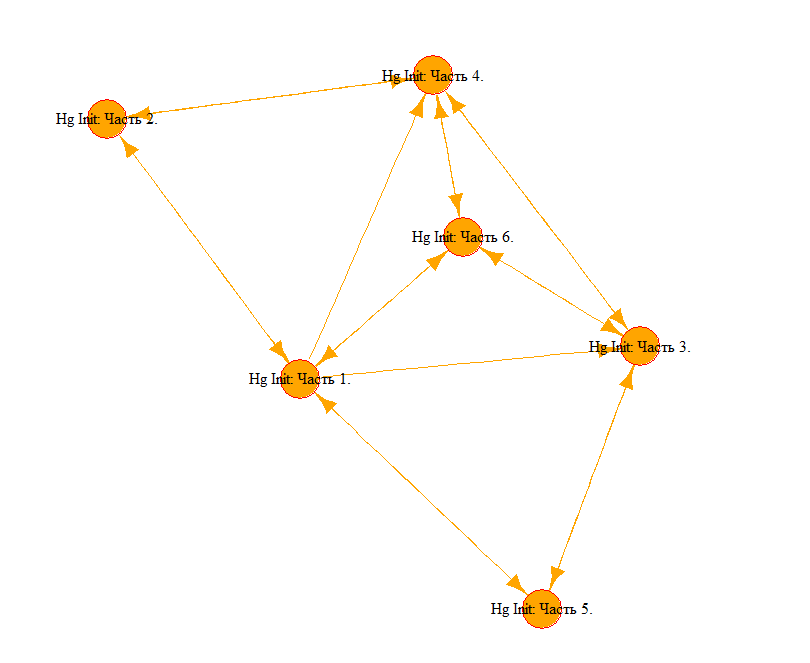


Рисунок 8 - Граф 2, построенный по считанным данным

Получили полноценный граф, изображённый на рисунке 8, где вершины - это статьи, считанные с веб-страниц, а направления рёбер - это ссылки на похожие статьи между ними.

Теперь разукрасим все вершины разными цветами, зададим размер вершин, в зависимости от рейтинга статей и размер ребра. И для наглядности уберём названия вершин. Чтоб разукрасить вершины, для удобства сразу создадим вектор с цветами.

colrs <- c("gray50", "tomato", "gold","chocolate3","green","lightblue")

plot(net, edge.arrow.size=.5, edge.color="orange", vertex.frame.color="#FF0000", vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1)

Соответственно получим граф, изображённый на рисунке 9.

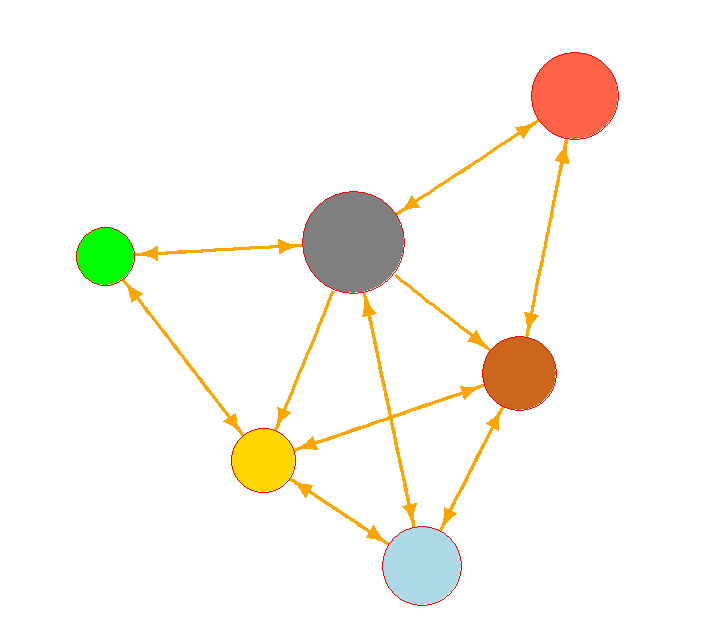


Рисунок 9 - Граф 3, построенный по считанным данным

Так как направления рёбер наложены друг на друга в некоторых местах, сделаем их дугами. И добавим легенды, чтобы было понятно, какая вершина относится к какой статье.

plot(net, edge.arrow.size=.5,

edge.color="orange", vertex.frame.color="#FF0000",

vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1,

edge.curved=.3)

legend(x=1.6, y=0.5, V(net)$media, pch=21,

col="#777777", pt.bg=colrs, pt.cex=3, cex=1.4, bty="n")

И получим следующий граф с легендами, изображённый на рисунке 10.

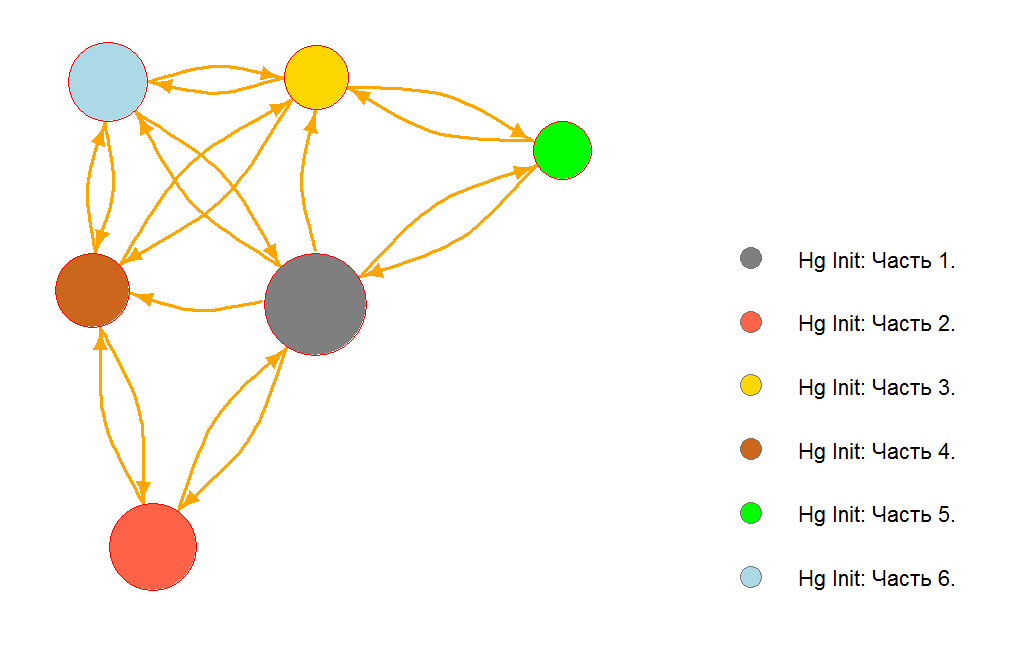


Рисунок 10 - Граф 4, построенный по считанным данным

В итоге у нас имеется граф, где вершины - это названия статей, считанные с веб-страниц, а рейтинг зависит от размера радиуса вершины, направления рёбер - это ссылки на похожие статьи между ними. И размер вершин зависит от рейтинга каждой статьи.

ЗАКЛЮЧЕНИЕ

Итак, анализ и визуализация данных необходима всегда, когда результат неочевиден, и часто даже тогда, когда он кажется очевидным.

Можно проводить сравнения между разными данными. Например, можно выяснить, у какой статьи больший или меньший рейтинг, больше или меньше комментариев при помощи диаграмм и даже графов, т.е какая статья вызывает больше интересов посвящать свои статьи более востребованным темам. Кроме того, модератор сайта может отфильтровывать потенциально не интересные статьи.

В ходе исследования курсового проекта я узнал как извлекать данные с интернет-страниц. Научился подготавливать данные для дальнейшего исследования. Рассмотрел несколько способов визуализации на основе этих данных. И выполнил их обработку и анализ с помощью средств языка R.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. А.Б.Шипунов. Е.М. Балдин. П.А. Волкова. [и др.] Наглядная статистика. Используем R! // М.: ДМК-Пресс. 2014.— C. 293.
2. Мастицкий С.Э., Шитков В.К. Статистический анализ и вмзуализация данных с помощью R. // М.: ДМК. 2014.— C.188.
3. Роберт И. Кабаков. R в действии. Анализ и визуализация данных в программе R / пер. с англ. Полины А. Волковой. // М.: ДМК Пресс, 2014. – C. 588.
4. Общая документация [Электронный ресурс] Статья, URL: http://cran.r-project.org/ [Дата обращения: 5 ноября 2015].
5. Визуализация статических и динамических сетей на R. [Электронный ресурс] Статья, URL:http://[habrahabr.ru/company/infopulse/blog/262079](file:///C:\\Users\\Алексей\\AppData\\Roaming\\Microsoft\\Word\\www.habrahabr.ru\\company\\infopulse\\blog\\262079) [Дата обращения: 26 ноября 2015].
6. R с нуля [Электронный ресурс] Статья, URL:http://[kstera.ru/R/index.htm](file:///C:\\Users\\Алексей\\AppData\\Roaming\\Microsoft\\Word\\www.kstera.ru\\R\\index.html)l [Дата обращения: 5 ноября 2015].
7. How to Webscrape in R, the Rvest and pipeR way. [Электронный ресурс] Статья, URL: http://[asbcllc.com/blog/2014/november/creating\_bref\_scraper](http://asbcllc.com/blog/2014/november/creating_bref_scraper/) [Дата обращения: 15 ноября 2015].
8. Four main languages for Analytics, Data Mining, Data Science. [Электронный ресурс] Статья, URL:http://[kdnuggets.com/2014/08/four-main-languages-analytics-data-mining-data-science.html](http://www.kdnuggets.com/2014/08/four-main-languages-analytics-data-mining-data-science.html) [Дата обращения: 1 декабря 2015].

ПРИЛОЖЕНИЕ А

Листинг программы

library(igraph)

library(rvest)

library(csvread)

GetStat<-function(selector)

{

url<-read\_html("http://habrahabr.ru/top/")

f<-html\_nodes(url, selector) %>%

html\_text()

}

SetView<-function(visual)

{

if (visual=="pie")

{

pie.rate <- c((fcount.num[1]), (fcount.num[2]), (fcount.num[3]), (fcount.num[4]), (fcount.num[5]), (fcount.num[6]))

names(pie.rate) <- c(fpost[1], fpost[2], fpost[3],fpost[4], fpost[5], fpost[6])

pie(pie.rate, cex=1.6)

pie(pie.rate, radius=0.8, cex=1.6, init.angle = 15, col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "white"),main = "Рейтинг статей")

}

if (visual=="barplot")

{

barplot(height=c(fcomment.num [1], fcomment.num[2], fcomment.num[3],

fcomment.num[4], fcomment.num[5], fcomment.num[6]),

col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "white"),

xlab = "Статьи", ylab = "Количество комментариев")

legend("topright", legend = c(fpost[1], fpost[2], fpost[3],fpost[4], fpost[5], fpost[6]), fill = 1:6, cex = 0.55)

}

if (visual=="graph")

{

net <- graph.data.frame(links, nodes, directed=T)

plot(net)

plot(net, edge.arrow.size=.5, edge.color="orange",

vertex.color="orange", vertex.frame.color="#FF0000",

vertex.label=V(net)$media, vertex.label.color="black")

colrs <- c("gray50", "tomato", "gold","chocolate3","green","lightblue")

plot(net, edge.arrow.size=.5, edge.color="orange", vertex.frame.color="#FF0000", vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1)

plot(net, edge.arrow.size=.5,

edge.color="orange", vertex.frame.color="#FF0000",

vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1,

edge.curved=.3)

legend(x=1.6, y=0.5, V(net)$media, pch=21,

col="#777777", pt.bg=colrs, pt.cex=3, cex=1.4, bty="n")

}

}

#Пример 1

fpost<-GetStat(".post\_title")

fcount<-GetStat(".js-favs\_count")

fcomment<-GetStat(".post-comments\_\_link\_all")

fpost <- iconv(fpost, "UTF-8","CP1251")

fcount.num<-as.numeric(fcount)

fcomment.num<-as.numeric(fcomment)

fcount

fpost

fcomment

s<-SetView("pie")

s<-SetView("barplot")url1<-read\_html("http://habrahabr.ru/post/108443/")

#Пример 2

url2<-read\_html("http://habrahabr.ru/post/108658/")

url3<-read\_html("http://habrahabr.ru/post/108904/")

url4<-read\_html("http://habrahabr.ru/post/109074/")

url5<-read\_html("http://habrahabr.ru/post/109203/")

url6<-read\_html("http://habrahabr.ru/post/109428/")

selector\_rate<-".js-favs\_count"

selector\_title<-".post\_title"

selector\_similar<-".post-preview\_\_link"

frate1<-html\_nodes(url1, selector\_rate) %>%

html\_text()

frate2<-html\_nodes(url2, selector\_rate) %>%

html\_text()

frate3<-html\_nodes(url3, selector\_rate) %>%

html\_text()

frate4<-html\_nodes(url4, selector\_rate) %>%

html\_text()

frate5<-html\_nodes(url5, selector\_rate) %>%

html\_text()

frate6<-html\_nodes(url6, selector\_rate) %>%

html\_text()

ftitle1<-html\_nodes(url1, selector\_title) %>%

html\_text()

ftitle2<-html\_nodes(url2, selector\_title) %>%

html\_text()

ftitle3<-html\_nodes(url3, selector\_title) %>%

html\_text()

ftitle4<-html\_nodes(url4, selector\_title) %>%

html\_text()

ftitle5<-html\_nodes(url5, selector\_title) %>%

html\_text()

ftitle6<-html\_nodes(url6, selector\_title) %>%

html\_text()

fsimilar1<-html\_nodes(url1, selector\_similar) %>%

html\_text()

fsimilar2<-html\_nodes(url2, selector\_similar) %>%

html\_text()

fsimilar3<-html\_nodes(url3, selector\_similar) %>%

html\_text()

fsimilar4<-html\_nodes(url4, selector\_similar) %>%

html\_text()

fsimilar5<-html\_nodes(url5, selector\_similar) %>%

html\_text()

fsimilar6<-html\_nodes(url6, selector\_similar) %>%

html\_text()

ftitle1 <- iconv(ftitle1, "UTF-8", "CP1251")

id<-c('s1',"s2","s3","s4","s5","s6")

media<-c(ftitle1,ftitle2,ftitle3,ftitle4,ftitle5,ftitle6)

media.type<-c(1,2,3,4,5,6)

size<-c(frate1,frate2,frate3,frate4,frate5,frate6)

audience.size<-as.numeric(size)

DF <- data.frame(id, media, media.type, audience.size)

write.table(DF,"f1.csv",sep=',')

s<-''

for (i in 1:6)

{

for (k in 1:3)

{

if (media[i]==fsimilar1[k]) s<-append(s,paste0("s",i,",","s",k))

if (media[i]==fsimilar2[k]) s<-append(s,paste0("s",i,",","s",k))

if (media[i]==fsimilar3[k]) s<-append(s,paste0("s",i,",","s",k))

if (media[i]==fsimilar4[k]) s<-append(s,paste0("s",i,",","s",k))

if (media[i]==fsimilar5[k]) s<-append(s,paste0("s",i,",","s",k))

if (media[i]==fsimilar6[k]) s<-append(s,paste0("s",i,",","s",k))

}

}

write.table(s,"f2.csv",sep=',')

nodes <- read.csv("nodes.csv", header=T)

links <- read.csv("edges.csv", header=T)

head(nodes)

head(links)

net <- graph.data.frame(links, nodes, directed=T)

plot(net)

plot(net, edge.arrow.size=.5, edge.color="orange",

vertex.color="orange", vertex.frame.color="#FF0000",

vertex.label=V(net)$media, vertex.label.color="black")

colrs <- c("gray50", "tomato", "gold","chocolate3","green","lightblue")

plot(net, edge.arrow.size=.5, edge.color="orange", vertex.frame.color="#FF0000", vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1)

plot(net, edge.arrow.size=.5,

edge.color="orange", vertex.frame.color="#FF0000",

vertex.label=NA, vertex.size=V(net)$audience.size\*1.1,

vertex.color=colrs[V(net)$media.type], edge.width=3.1,

edge.curved=.3)

legend(x=1.6, y=0.5, V(net)$media, pch=21,

col="#777777", pt.bg=colrs, pt.cex=3, cex=1.4, bty="n")