Rattle: GUI анализа данных для *R*

Москва

2014

Rattle: A Data Mining GUI for R

by Graham J Williams

The R Journal Vol. 1/2, December 2009

Перевод с английского А.А.Фоменко

Предисловие

Анализ данных поставляет понимание, образцы и описательные и прогнозирующие модели от больших объемов данных, доступных сегодня во многих организациях. Шахтер данных получит в большой степени на методологиях, методах и алгоритмах от статистики, обучения машины и информатики. R все более и более обеспечивает мощную платформу для анализа данных. Однако, сценарии и программирование иногда - проблема для аналитиков данных, перемещающихся в анализ данных. Пакет Rattle обеспечивает графический интерфейс пользователя определенно для анализа данных, используя R. Это также обеспечивает стартовую площадку к использованию R как язык программирования для анализа данных.

Введение

Анализ данных комбинирует понятия, инструменты и алгоритмы из машинного обучения и статистики для анализа очень больших наборов данных, чтобы изучить, понять и преобразовать в действие.

Средства анализа данных с закрытым исходным кодом облегчили понимание анализа данных во многих организациях. Эти средства предлагают стандартную простоту в употреблении, которая делает их привлекательными для многих новых аналитиков данных на рынке, отчаянно ища высокие уровни аналитических навыков.

R идеально подходит для многих сложных задач, связанных с анализом данных. R предлагает ширину и глубину в статистических вычислениях не доступных в коммерческих средствах с закрытым исходным кодом. Все же R остается, прежде всего, языком программирования для высококвалифицированного статистика вне досягаемости для многих.

Rattle (Аналитический Инструмент R для простого обучения) является применением для графического анализа данных, написанный на R (Уильямс, 2009b). Он разрабатывался определенно, чтобы упростить переход от основ анализа исходных данных, как обязательно предлагающийся GUI, к сложным вариантам анализам данных, используя мощный статистический язык.

Rattle объединяет множество пакетов R, которые важны для аналитика данных, но часто не легки для использования новичком. Понимание R не требуется, чтобы начать с Rattle - это будет постепенно расти, поскольку мы прибавляем изощренность к нашему анализу данных. Пользовательский интерфейс Rattle обеспечивает вход в мощь R как инструмента анализа данных.

Rattle используется для преподавания анализа данных в многочисленных университетах и в обиходе консультантами и командами анализа данных по всему миру. Он также доступен как продукт в пределах информационной Сети Разработчиков - комплект анализа бизнес-данных Фокуса как RStat.

Анализ данных

Rattle определенно использует простую концепцию, основанную на вкладках, для пользовательского интерфейса (рисунок 1), получая поток операций посредством процесса анализа данных с вкладкой для каждого этапа. Типичный поток операций следует с левой вкладки (вкладка **Data**) направо (вкладка **Log**). Для любой вкладки пользователь должен сконфигурировать доступные параметры и затем нажать кнопку **Execute** (или F2), чтобы выполнить соответствующую задачу. Строка состояния в внизу окна укажет завершение действия.

Общий поток операций для проекта анализа данных может быть получен в итоге как:

1. Загрузите **Набор данных** и выберите переменные;
2. **Исследуйте** данные, чтобы понять распределения;
3. Протестируйте распределения;
4. **Преобразуйте** данные, чтобы удовлетворить моделированию;
5. Создайте **Модели;**
6. **Оцените** модели и отметьте наборы данных;
7. Рассмотрите Журнал процесса анализа данных.

Базовый код R, созданный и выполняемый Rattle, записан на вкладке **Log**, вместе с поучительными комментариями. Это позволяет пользователю рассматривать фактические команды R. Фрагменты кода R могут также быть скопированы как текст (или **Экспортируемые** в файл) от вкладки **Log** и вставили в пульт R и выполнились. Это позволяет Rattle быть развернутым для основных задач, и всё же получать доступ к полной мощности R как необходимый (например, чтобы подстроить опции моделирования, которые не представлены в интерфейсе).

Аналитик данных задокументирует их действие, поскольку они выполняются через Rattle, редактируя журнал, который также автоматически заполнен, поскольку моделирование продолжается. Простая и автоматическая обработка может тогда превратить журнал в отформатированный отчет, который также воплощает фактический код, которым можно также исполнить, чтобы тиражировать действие.

Используя связанный Tangle процессор позволяет экспортировать журнал как файл скрипта R, записывать предпринятые меры. Затем можно независимо исполнить скрипт в более позднее время (или вставиться в консоль R).

Воспроизводимость важна и в научном исследовании, и в коммерческой практике.

Данные

Если набор данных не был предоставлен Rattle, то нажимаем кнопку **Execute** (например, стартуем Rattle, и сразу щелкаем **по Execute**), и предоставляется опция для загрузки одного из наборов данных примеров Rattle из файла CSV.

Rattle может загрузить данные из различных источников. Он непосредственно поддерживает CSV (запятая как разделитель данных), TXT (табуляция как разделитель данных), ARFF (общий формат наборов данных анализа данных, который добавляет тип информации к файлам CSV), и соединения ODBC (позволяет соединение со многими источниками данных включая MySQL, SQLite, Postgress, MS/Excel, MS/Доступ, SQL-сервер, Oracle, IBM DB2, Netezza и Teradata). Фреймы данных R, присоединенные к текущему сеансу R и наборам данных, доступным из пакетов, установленных в библиотеках R, также доступны через интерфейс Rattle.

Чтобы исследовать использование Rattle как инструмента анализа данных, мы считаем набора данных примера аудита, предоставленную пакетом Rattle. Данные искусственные, но отражают набор данных реального мира, используемый для рассмотрения результатов исторических финансовых аудитов. Изображение, например, получения налоговиком налога на доходы, основаны на информации, предоставленной налогоплательщиком. Моно выполнить тысячи случайных аудитов, а результаты покажут, требуется ли корректировка предоставленной информации, приводящая к изменению ответственности налогоплательщика.

Контрольный набор данных предоставлен и как набор данных R и как файл CSV. Набор данных состоит 2,000 вымышленных налогоплательщиков, которые контролировались для выполнения налоговых требований. Для каждого случая результат после аудита записан (должны ли финансовые требования были быть скорректированы или нет). Фактическая сумма в долларах корректировки, которая закончилась, также записана (заметим, что корректировки могут пойти в любом направлении).

Контрольный набор данных содержит 13 переменных (или столбцов), первым из которых является уникальным клиентским идентификатором.

При загрузке данных в Rattle могут использоваться определенные специальные префиксы к именам переменных для указания роли переменной по умолчанию. Например, если имя переменной начинается с ‘ID _’, то переменная отмечена как имеющая роль идентификатора. Другие префиксы включают, ‘IGNORE_’, ‘RISK_’, ‘IMP _’ (для оценоки) и ‘TARGET _’. Примеры контрольных данных включают IGNORE_Accounts и TARGET_Adjusted.

Опция CSV вкладки **Data** предоставляет самый простой подход к загрузке данных в Rattle. Если вкладка **Data Выполняется** без имени файла CSV, то Rattle предлагает опцию для загрузки набора данных примера. Щелчок по полю **Filename** перечислит другие доступные наборы данных выборки, включая ‘audit.csv’.

Как только Rattle загрузит набор данных, текстовое окно будет содержать список доступных переменных и их ролей по умолчанию (как в рисунке 2).

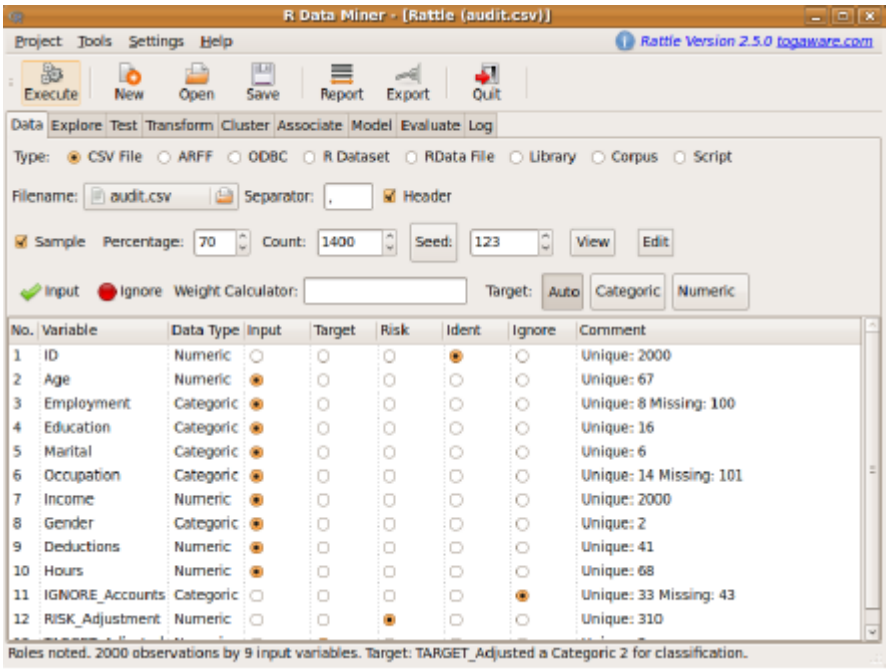

Рисунок 2: переменный ролевой экран Rattle.

По умолчанию у большинства переменных есть роль **Input** для моделирования. Можно указать одну переменную как **Целевую (Target)** переменную, и дополнительно указать другую переменную как переменную **Риска (risk)** (которая является мерой размера "целей"). Другие роли **Ident** и **Ignore.**

Rattle использует простую эвристику для присвоения ролей, особенно для цели и игнорируемых переменных. Поскольку, например, любая числовая переменная, у которой есть единственное значение для каждого наблюдения, автоматически указывается как идентификатор.

Rattle, по умолчанию, разделит набор данных на обучающий и тестовый наборы данных. Этот вид выборки полезен в исследовательских целях, когда данные достаточно большие. Его основная цель, тем не менее, состоит в том, чтобы выбрать 70%-ую выборку для обучения моделей, обеспечивая 30%-ое множество для тестирования.

Исследование

Исследовательский анализ данных важен в понимании данных. Вкладка **Explore** обеспечивает многочисленные числовые и графические инструменты для исследования данных. Еще раз, есть значительная уверенность во многих других пакетах R.

Сводка

Опция **Summary** использует команду R *summary*() для предоставления основной одномерной сводки. К ней добавлены *contents* и *describe* из пакета **Hmisc** (Harrell, 2009). Расширенные сводки включают дополнительную статистику, предоставленную пакетом **fBasics** пакетом (Wuertz, 2009), эксцесс и асимметрия, так же как сводка отсутствующих значений, используя функциональность пропущенных значений из пакета **mice** (ван Буурен и Грузуис-Удшурн, 2009).

Распределения

Опция **Распределений** обеспечивает доступ к многочисленным типам рисунка. Это всегда хорошая идея рассмотреть распределения значений каждой из переменных до анализа данных. Хотя вышеупомянутые сводки помогают, визуальные исследования могут часто быть вполне разоблачающими (Cook и Swayne, 2007).

Обширный массив инструментов доступен в пределах R для представления данных визуально, и тема затронута подробно во многих книгах включая Кливленд (1993). Rattle обеспечивает простой интерфейс для базовой функциональности в R для того, чтобы получить некоторые общие рисунки. Текущая реализация прежде всего полагается на основную графику, обеспеченную R, но может перейти на более сложную **lattice** (Sarkar, 2008) или **ggplot2** (Wickham, 2009).

Некоторые консервированные рисунки иллюстрированы в рисунке 3. По часовой стрелке мы можем видеть диаграмму, гистограмму, мозаичный рисунок и упорядоченный рисунок Бенфорда. Идентифицировав целевую переменную (на вкладке **Data**) рисунки включают распределения для каждого подмножества наблюдений, присоединенных с каждым значением целевой переменной, везде, где это имеет смысл делать так.

GGobi и Latticist

Rattle обеспечивает доступ к двум сложным инструментам для интерактивного анализа графических данных: GGobi и Latticist.

GGobi (Cook и Swayne, 2007) к инструменту визуализации получают доступ через пакет **rggobi** (Wickham и др., 2008). GGobi должен быть установлен в системе отдельно и исполняется под GNU/Linux, OS/X и MS/Windows. Он доступен для скачивания с http://www.ggobi.org/.

Ggobi полезен для исследования высоко-мерных данных через очень динамическую и интерактивную графику, особенно с турами, рисунками разброса, столбиковыми диаграммами и параллельными координатными рисунками. Рисунки интерактивные и связаны с рисованием кистью и идентификацией. Доступная функциональность обширна, и поддерживает панорамирование, изменение масштаба и вращения.

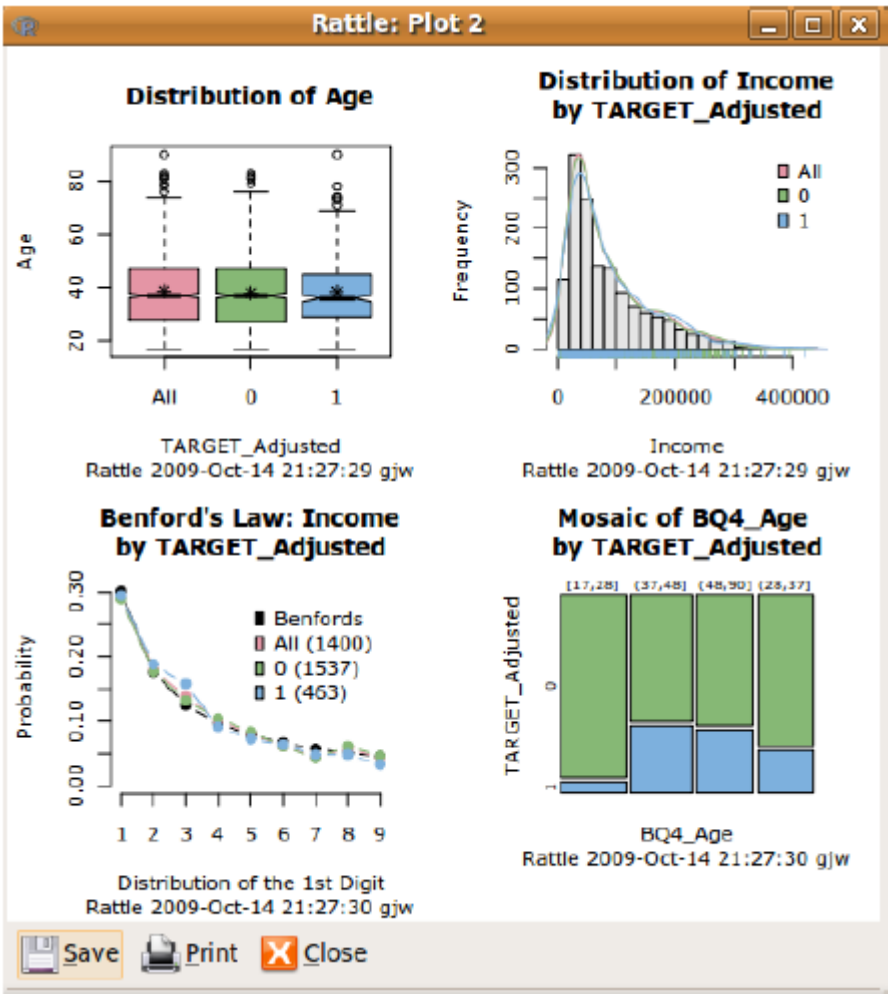

Рисунок 3: Исследование переменных распределений.

Рисунок 4 выводит на экран рисунок разброса Возраста против (оставленного) Поступившего и матрица рисунка разброса через четыре переменные в одно время (право). Рисование кистью используется, чтобы отличить класс каждого наблюдения.

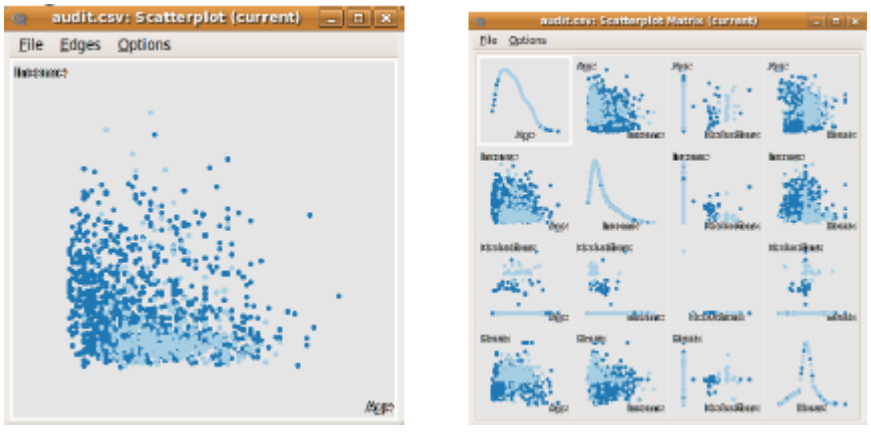

Рисунок 4: Пример GGobi, использующего **rggobi**, чтобы соединиться.

Более свежее дополнение к комплекту R пакетов - пакеты **latticist** и **playwith** (Эндрюс, 2008), которые используют **lattice** графику в пределах графического интерфейса для исследования данных в интерактивном режиме. Инструмент поддерживает различные рисунки, выбор данных и подмножество и поддержку рисования кистью и аннотаций. Рисунок 5 иллюстрирует рисунок по умолчанию при выводе из Rattle.

Тест

Вкладка Test обеспечивает доступ ко многим параметрическим и непараметрическим статистическим тестам распределений. Это более свежее расширение Rattle продолжает получать внимание (и, следовательно, будет изменяться в течение долгого времени). В контексте анализа данных часто примененные к задаче двоичной классификации, текущие тесты, прежде всего, осуществляют статистическое тестирование двух выборок.

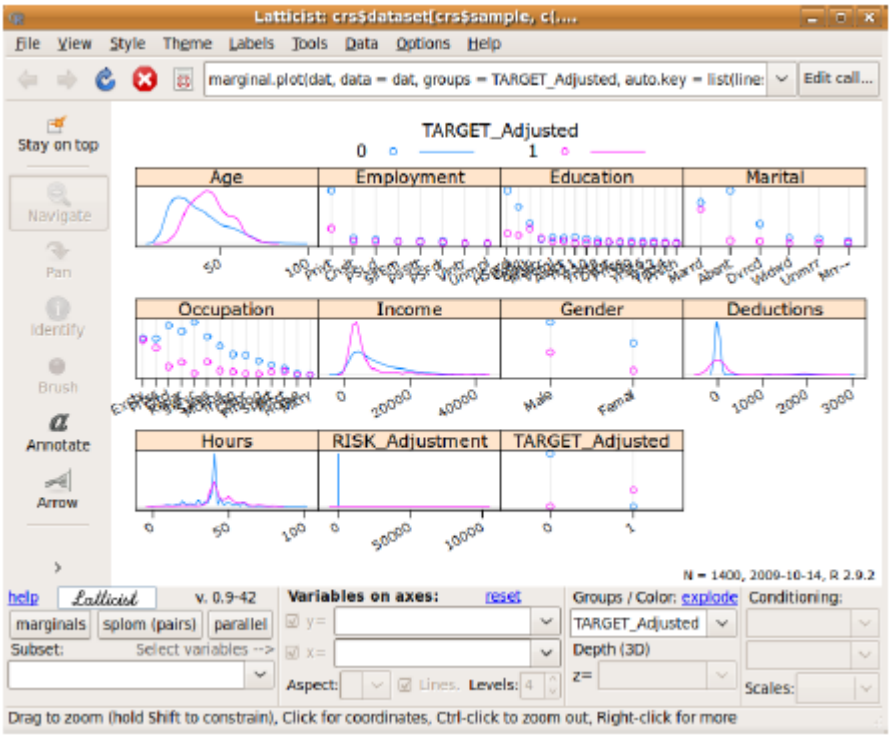

Рисунок 5: Latticist, выводящий на экран контрольные данные.

Тесты распределения данных включают тесты Коломогоров-Смирнова и ранговый знаковый тест Вилкоксона. Для тестирования расположения средней имеются тесты Суммы ранга t-тест Wilcoxon. F-тест и корреляция Пирсона также доступны.

Преобразование

Очистка данных и создание новых средств (получение переменных) занимают намного больше времени для аналитика данных. Есть много подходов к очистке данных, и язык программирования, подобный R, поддерживает их. Вкладка **Transform** Rattle (рисунок 6) предоставляет многие общие возможности для преобразования, включая перемасштабирование, уменьшение асимметрии, заполнение пропущенных значений, превращение числовых переменных в категорические переменные, и наоборот, работа с выбросами, и удаление переменных или наблюдений с пропущенными значениями. Здесь рассмотрим некоторые из преобразований.

Перемасштабирование

Опция **Перемасштабирование - Rescale** предлагает ряд операций масштабирования, используя команду масштаба scalt основы и команду перемасштабирования rescale из пакета **reshape** (Wickham, 2007). Перемасштабирование включает центрирование вокруг нуля **(Перецентр-Recenter)**, масштабирование в промежутке 0-1 (**Scale [0,1]**), преобразование в ранговое упорядочение **(Ранг-Rank)**, устойчивое перемасштабирование около нуля, используя медиану (**-Медиана/MAD**), и повторно масштабирование, основанное на группах в данных.

Для любого преобразования исходная переменная не меняется. Новая переменная создается с префиксом, добавленным к имени переменной, чтобы указать на преобразование. Префиксы включают ‘RRC _’, ‘R01 _’, ‘RRK _’, ‘RMD _’, и ‘RBG _’, соответственно.

Эффект переперемасштабирования может быть исследован, используя вкладку **Explore** (рисунок 7). Заметьте, что Rattle накладывает столбиковые диаграммы с рисунком плотности по умолчанию.

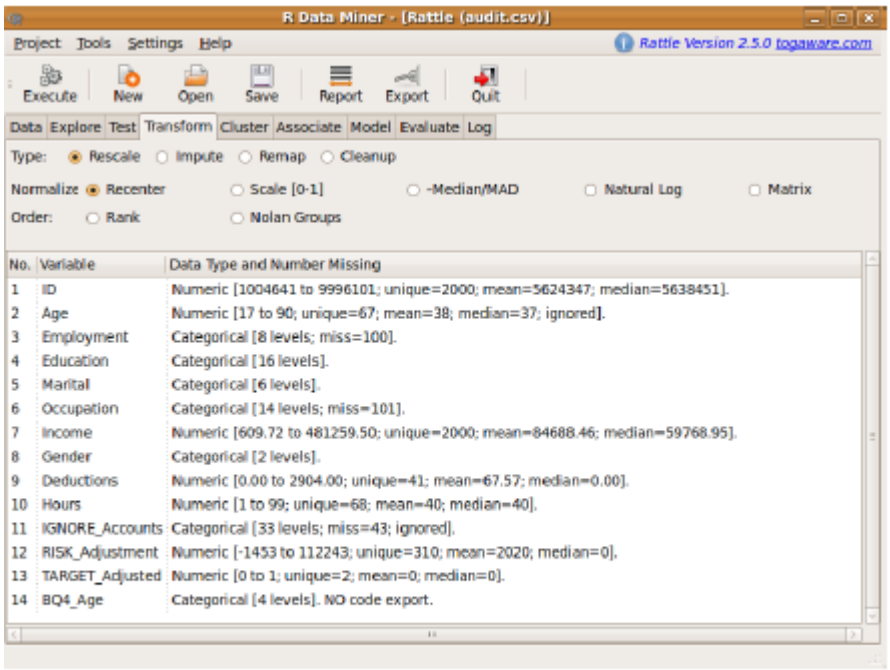

Figure 6: Transform options.

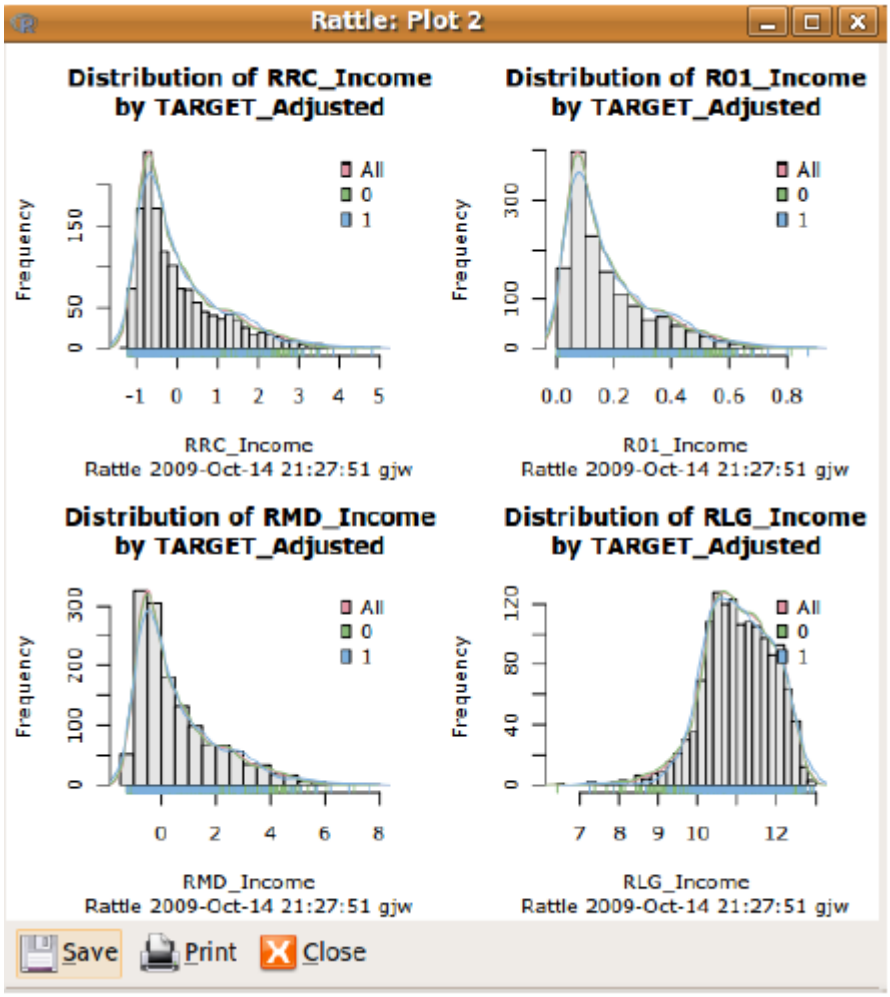

Рисунок 7: Четыре повторно определенных масштаб версии Дохода.

Заполнение

Заполнение пропущенных значений - хитрая тема и должно делаться только с хорошим пониманием данных. Часто, наблюдательные данные (в отличие от экспериментальных данных) будут содержать отсутствующие значения, и это может вызвать проблему для алгоритмов анализа данных. Например, опция Forest (использующая **randomForest**) тихо удаляет любое наблюдение с любым отсутствующим значением! Для наборов данных с очень большим количеством переменных и разумным числом отсутствующих значений, это может хорошо привести к небольшому, нетипичному набору данных, или даже никаким данным вообще!

Есть много типов возможных заполнений, только некоторые из которых непосредственно доступны в Rattle. Далее, Rattle еще не поддерживает множественное заполнение. Образец пропущенных значений может быть просмотрен, используя кнопку **Показать пропущенные** опции Summary вкладки **Explore.**

Самое простое, и наименее рекомендуемый, заполнение обвинений включает замену всех пропущенных значений для переменной единственным значением. Это имеет смысл, когда мы знаем, что отсутствующие значения фактически имеют значение “0”, а не неизвестное. Например, в контексте налогообложения, если налогоплательщик не обеспечивает значение для определенного типа удержаний, то мы могли бы предположить, что они равны нулю. Точно так же, если число дочерних элементов в семействе не записано, то разумно предположить, что это нуль (но это могло бы одинаково хорошо средний, что число только неизвестно).

В общем, для пропущенных значений, которые, как известно, не равны нулю, следует использовать некоторое "центральное" значение переменной. Это часто среднее, медиана, или мода. Можно использовать среднее, например, если переменная нормально распределена (и в особенности имеет небольшую асимметрию). Если данные действительно имеют некоторую асимметрию (например, есть некоторое количество очень больших значений), то тогда медиана могла бы быть лучшим выбором.

Опасайтесь любого выполняемого заполнения. Это, в конце концов, изобретает новые данные! Будущая разработка Rattle может оказать больше поддержки с основанным на модели заполнениями через такие пакеты как **Amelia** (Honaker и др., 2009).

Переотображение

Опция **Переотображения-Remap** обеспечивает многочисленные операции переотображения, включая стеллажирование, логарифмирование, отношения и отбор категорических переменных в переменные индикатора для ситуации, где потроитель модели требует числовых данных. Rattle предоставляет возможности использовать **Квантиль стеллажирование**, **KMeans стеллажирование**, и **Равную Ширину стеллажирования**. Для каждой опции число стелажей по умолчанию равняется 4, но можно изменить под наши потребности. Генерируемые переменные снабжены префиксом видао ‘BQn _’, ‘BKn _’, и ‘BEn _’ соответственно, с ‘n’, указывающем число стеллажей. Таким образом мы можем создать множественный binnings для любой переменной.

Есть также опции **Присоединения к Categorics** - удобный способ расслоить набор данных, основанный на множественных категорических переменных. Преобразование **Логарифма** также доступно.

Модель

Алгоритмы анализа данных часто описываются как дескриптивные или прогнозирующие. Rattle, в настоящий момент, поддерживает два общих дескриптивных или безнадзорных подхода к построению модели: кластерный анализ и анализ зависимости. Поддерживается много построителей моделей прогноза: деревья решений, усиление, случайные леса, машины опорных векторов, обобщенные линейные модели и нейронные сети.

Прогнозирующее моделирование, а в общем задача классификации, является серцевиной анализа данных. Rattle первоначально сосредоточился на общей задаче анализа данных - двоичной классификации их (или два класса), но теперь поддерживает множественную классификацию и регрессию, так же как описательные модели.

Rattle обеспечивает прямой интерфейс для набора построителей описательных и прогнозирующих моделей, доступных в R. Для каждой модели простой набор настраивающихся параметров представлен через графический интерфейс. Где только возможно Rattle пытается представить хорошие значения по умолчанию (часто те же самые по умолчанию как выбрано автором соответствующего пакета), что позволяет пользователю просто создавать модель без настройки или с минимальной настройкой. Это может не всегда быть правильным подходом, но является, конечно, разумным местом для старта.

Мы рассмотрим моделирование в пределах Rattle через деревья решений и случайные леса.

Деревья решений

Один из классических методов обучения машины, широко развернутых в анализе данных, является индукцией дерева решений Quinlan (1986). Используя простой алгоритм и простую древовидную структуру, чтобы представить модель, подход, оказалось, был очень эффективен. Внизу, **rpart** (Therneau и др., 2009) и **сторона** (Hothorn и др., 2006) пакеты призваны, чтобы сделать работу. Показы рисунка 8 вкладка **Model** с результатами создания дерева решений, выведенного на экран дословно (обычный результат от сводной команды для "rpart" объекта).

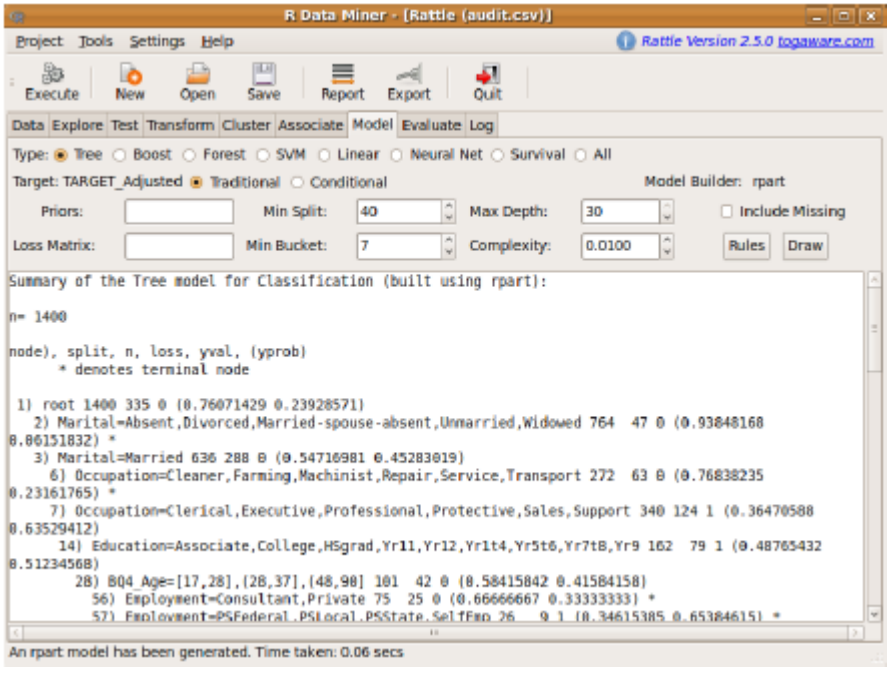

Рисунок 8: Создание дерева решений.

Rattle увеличивает стоимость основной **rpart** функциональности с дополнительными дисплеями дерева решений, как в рисунке 9 и преобразовании дерева решений в список правил (использующий кнопки **Draw и Rules** соответственно).

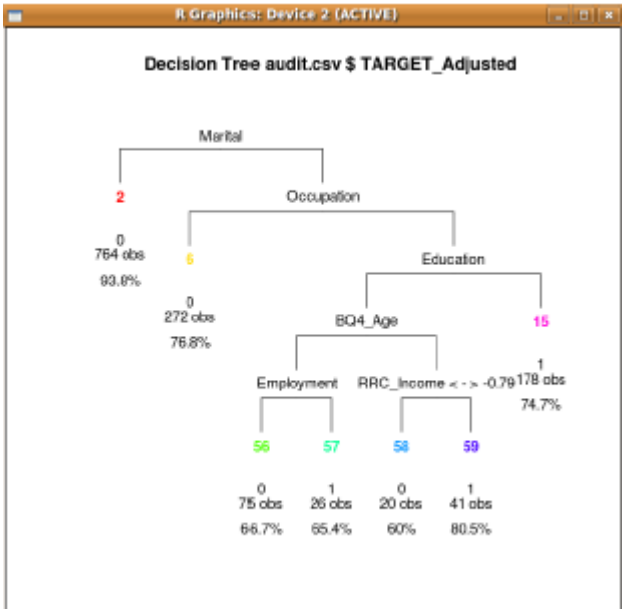

Рисунок 9: дисплей Rattle дерева решений.

Множество

Подход множества получил много интереса в последнее время. Ранняя работа (Уильямс, 1988) экспериментировала с идеей объединить набор деревьев решений. Результаты там указали на преимущество создания множественных деревьев и объединения их в единственную модель как множество.

Недавние разработки продолжают демонстрировать эффективность множеств в анализе данных с помощью усиления и случайных лесных алгоритмов. Оба поддерживаются в скрежете, и мы рассматриваем только случайный лес здесь.

Случайные леса

Случайный лес (Breiman, 2001) разрабатывает множество деревьев решений. Случайные леса часто используются, когда у нас есть очень большие обучающие наборы данных и очень большое количество входных переменных (сотни или четные тысячи входных переменных). Случайная лесная модель обычно составляется из десятков или сотен деревьев решений, каждое созданное использование случайной выборки набора данных, и создавая любое дерево, случайную выборку переменных рассматривают в каждом узле.

Случайная выборка и данных и переменных гарантирует, что даже создание 500 деревьев решений может быть эффективным. Это также достойно поставляет значительную устойчивость шуму, выбросам и сверхподгонке, когда по сравнению с единственным древовидным классификатором.

Rattle использует **randomForest** пакет (Лиоу и Вайнер, 2002), чтобы создать лес деревьев. Это - интерфейс к исходному случайному лесному коду от исходных разработчиков метода. Следовательно, хотя, у получающихся деревьев есть различная структура к стандартным "rpart" деревьям, и таким образом, некоторые из тех же самых древовидных визуализаций не легко доступны. Rattle может перечислить все правила, генерируемые для случайного леса, если требующийся. Для комплексных задач это может быть очень длинным списком действительно (тысячи правил).

Лесная опция может также вывести на экран рисунок относительной переменной значимости. Это обеспечивает понимание, какие переменные играют больше всего важную роль в предсказании результатов класса. Кнопка **Importance** выведет на экран две меры по альтернативе показа рисунков относительной значимости переменных в нашем наборе данных относительно предсказания класса.

Построение всех моделей и настройки

Опытным путем различные разработчики модели часто производят модели, которые выполняют одинаково с точки зрения порядка неверно спецификации. Таким образом, вполне поучительно использовать все построители модели для одного и того же набора данных. Опция **All** построит одну модель для каждого построителя модели.

Мы можем рассмотреть результативность каждой из созданных моделей и выбрать ту, которая лучшие удовлетворяет наши потребности. В выборе единственной модели мы можем не обязательно выбрать самую точную модель. Другие коэффициенты могут играть роль. Например, если простое дерево решений почти так же точно как эти 500 деревьев в случайном лесном множестве, то можно отказаться от сложности случайного леса для развертывания.

Эффективная альтернатива, где объяснения не требуются, и точность, требуется, должен создать модель каждого типа, а затем создавать множество, которое является линейной комбинацией этих моделей.

Оценка

Rattle обеспечивает стандартный набор инструментов для оценки и сравнения результативности моделей. Он включает матрицу ошибок (или таблицу беспорядка), диаграммы подъема, кривые ROC, и Кривые Стоимости, использующих, например, пакет **ROCR** (Пойте и др., 2009). Рисунок 10 показывает варианты

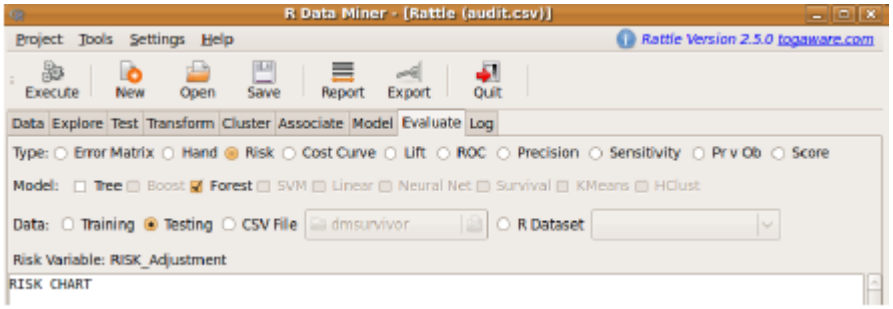

Рисунок 10: Опции для оценки.

Совокупное изменение кривой ROC реализовано в Rattle как диаграммы Риска (рисунок 11). Диаграммы риска особенно подходят для двоичных задач классификации, которые распространены в анализе данных. Цель состоит в том, чтобы эффективно вывести на экран легко понятую меру результативности модели относительно доступных ресурсов. Такие диаграммы, как оказалось, были с большей готовностью объяснимы руководителям принятия решений.

Диаграмма риска особенно полезна в контексте контрольного набора данных, и для задач анализа рисков вообще. У контрольного набора данных есть две целевых переменных класса так же как так называемая переменная риска, которая является мерой размера риска, присоединенного с каждым наблюдением. У наблюдений, у которых нет корректировки после аудита (то есть, клиенты, которые предоставили корректную информацию), конечно, будет риск нуля присоединенным с ними. Наблюдениям, у которых действительно есть корректировка, будут обычно присоединять риск с ними, и для удобства мы просто идентифицируем значение корректировки как величина риска.

Rattle использует идею диаграммы риска для оценки результативностимодели в контексте анализа рисков.

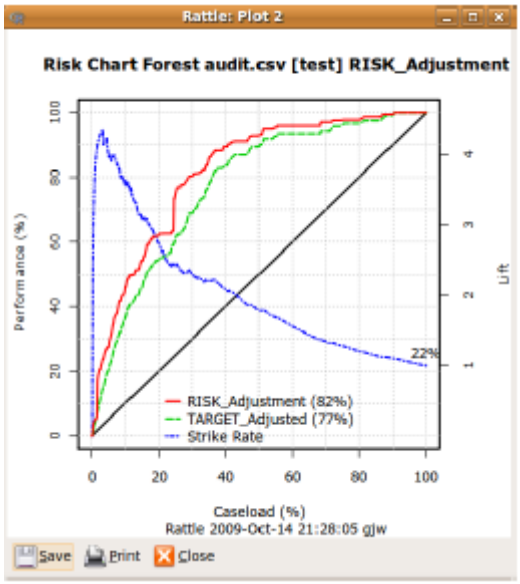

Рисунок 11: простая совокупная диаграмма риска.

Диаграмма риска рисует результативность против вариантов загрузки. Предположим, что у нас была совокупность из 100 наблюдений (или варианты аудита). Загрузка варианта - процент этих случаев, которые мы фактически попросим, чтобы наши аудиторы обработали. Остающиеся случаи не будут в дальней обсчитаны, ожидая, что они будут с низким риском, и следовательно, с ограниченными ресурсами, не требуя никакого действия.

Решение относительно того, какой процент случаев обрабатывать, соответствует оси X диаграммы риска – случаи загрузки. 100%-ая случаи загрузки указывает, что мы будем обрабатываем все контрольные случаи. 25%-ая случаи загрузки указывает, что мы будем обрабатывать всего одну четверть всех случаев.

Результативность - процент общего количества обработанных случаев, которые потребовали корректировки (или полный риск - оба графически изображены, если переменная риска идентифицирована), который мог бы быть покрыт на обрабатываемом множестве.

Диаграмма риска показывает компромисс между ресурсами и риском.

Развертывание модели

Как только мы выбрали модель, которая представляет приемлемое уточнение нашим бизнес-процессам, мы сталкиваемся с развертыванием. Развертывание может колебаться от непосредственного исполнения модели до полностью автоматической со тщательно управляемой средой развертывания.

Далее обсудим некоторые проблемы и исследуем, как Rattle поддерживает развертывание.

Сценарии R

Самый простой подход к развертыванию - это применение модели к новому набору данных. Это часто упоминается как *оценка выигрыша*. В контексте R это - не что иное как использование функции predict.

Вкладка оценки Rattle поддерживает оценку выигрыша опцией **Score**. Есть дальнейшие опции, чтобы отметить обучающий набор данных, набор данных теста или данные, загруженные из файла CSV (который должен содержать те же самые переменные).

Любое число моделей может быть выбрано, и результаты записаны файлу CSV.

Выигрыш часто выполняется некоторое время после того, как модель будет создана. В этом случае модель должна быть сохранена для более позднего использования. Понятие проекта Rattle полезно при таком обстоятельстве. Текущее состояние Rattle (включая фактические данные и модели, созданные во время сеанса), может быть сохранено к проекту, и позже загружено в новый экземпляр Rattle (работающий на том же самом узле или даже различной хост-системе и операционной системе). Новый набор данных может тогда быть отмечен, используя сохраненную модель.

Внизу, сохранение/загружение проекта Rattle требует, чтобы не больше, чем использование команд сохранения и загрузки R создало двоичное представление объектов R и сохранение их к файлу. Проект Rattle может стать вполне большим, особенно с большими наборами данных.

Большие файлы занимают больше времени для загрузки, и для развертывания модель, которую часто не обязательно сохранять с исходными данными. Поэтому, поскольку мы серьезно относимся к развертыванию, мы могли бы сохранить только подлежащую развертыванию модель. Это делается путем использования функции сохранить и знания немного о внутреннем устройстве Rattle (но не больше, чем что представлено через вкладку **Log**).

Подход сохранения модели randomForest имеет вид:

```
> myrf <- crs$rf
> save(myrf, file = "model01_090501.RData")
```

Затем позже можем загрузить модель и применить модель (использующий скрипт, основанный на командах, показанных на вкладке **Rattle Log**) к новому набору данных:

```
> library(randomForest)
> (load("model01_090501.RData"))
[1] "myrf"
> dataset <- read.csv("cases_090601.csv")
> pr <- predict(myrf, dataset,
+ type = "prob")[, 2]
> write.csv(cbind(dataset,
+ pr), file = "scores_090601.csv",
+ row.names = FALSE)
> head(cbind(Actual = dataset$TARGET_Adjusted,
+ Predicted = pr))
Actual Predicted
1 0 0.022
2 0 0.034
3 0 0.002
4 1 0.802
5 1 0.782
6 0 0.158
```

Как в стороне, мы можем видеть, что случайная лесная модель делает хорошо на этих немногих наблюдениях.

В практике (например, в австралийском Office Налогообложения), как только было одобрено развертывание модели, модель развернута в безопасную среду. Запланировано регулярно быть примененным к новым данным, используя скрипт, который очень подобен этому выше (использование **littler** пакета для GNU/Linux). Данные получены из хранилища данных, и результаты заполняют таблицу хранилища данных, которая тогда используется, чтобы автоматически генерировать единицы работы для аудиторов к действию.

Экспорт в PMML

Альтернативный подход к развертыванию должен экспортировать модель так, чтобы это могло быть импортировано в другое программное обеспечение для прогноза на новых данных.

Мы экспериментировали с экспортом случайных лесов к коду C++. Это демонстрировалось, работая на основе миллионов строк новых данных в хранилище данных в секундах.

Журнал

GUI не так полон и гибок как полный язык программирования. Rattle достаточен для многих аналитиков данных, обеспечивая среду основного-пункта-и-щелчка для быстрого и непротиворечивого анализа данных, многое получая от ширины и глубины R. Пользователь, профессиональный аналитик данных, скоро найдет потребность выйти за предположения, воплощенных в Rattle. Rattle поддерживает это через вкладку **Log**.

Как упомянуто выше, журнал команд R содержит конструкции Rattle через вкладку **Log**. Намерение состоит в том, чтобы команды R были доступны для копирования в Консоль R так, чтобы там, где Rattle только представляет ограниченное количество опций, дальнейшие опции могли быть настроены через пульт R.

Вкладка **Log** получает команды для более позднего выполнения и также для образования. Включены информативные комментарии описания использованных шагов. Намерение состоит в том, что это обеспечивает учебное введение в использование R из командной строки, где мы получим гораздо больше возможностей.

Текст, который появляется наверху вкладки **Log**, показан на рисунке 12. Тексту комментария предшествует символ комментария R (#) командами R между ними.

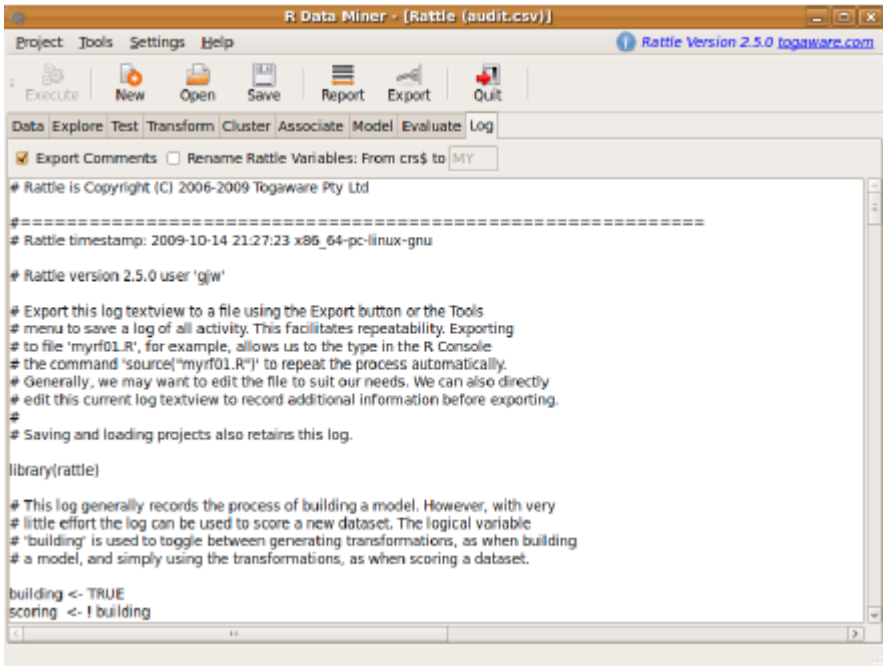

Рисунок 12: Журнал Rattle.

Весь журнал может быть экспортирован в файл скрипта (с расширением ‘.R’), а затем загружен в R или редактор скриптов R (как Emacs/ESS или Tinn-R), чтобы повторить точные шаги взаимодействий Rattle. Вообще, можно рассмотреть код и поправить его, чтобы удовлетворить нашим целям. После экспорта вкладки Log в файл с именем файла *‘myrf01.R’* получаем файл, который можно выполнить как скрипт в R:

```
> source("myrf01.R")
```

Справка

Меню помощи обеспечивает доступ к кратким описаниям функциональности Rattle, структурированного, чтобы отразить пользовательский интерфейс. Также многие сообщения справки обеспечивают прямой доступ к базовой документации для различных включенных пакетов.

Будущее

Rattle продолжает подвергаться разработке, расширяющейся в направлениях, продиктованных ее фактическим использованием в проектах анализа данных, и от предложений и кода, предлагаемого ее пользовательской совокупностью. Здесь мы упоминаем некоторые экспериментальные разработки, которые могут появиться в Rattle в течение долгого времени.

Многие более новые пакеты R обеспечивают возможности, которые могут улучшить Rattle значительно. Пакет **стороны** и присоединенные усилия объединить представление моделей дерева принятия решения через R являются захватывающей разработкой. Предложения **каре** объединенный интерфейс к исполнению множеством разработчиков модели и значимой поддержкой настройки моделей по различным установкам параметров. Эта последняя возможность - что-то, с чем провели эксперименты в Rattle, но еще хорошо разработалось.

Возможность глубокого анализа текста готовится. Текущие версии Rattle могут загрузить корпус документов, преобразовать их в пробел функции, и затем иметь доступный все возможности Rattle. Загрузка корпуса и его преобразование в пробел функции полагаются на пакет **TM** (Feinerer, 2008).

Анализ временных рядов непосредственно не поддерживается в Rattle. Такая возможность включит возможность проанализировать истории блога и наблюдения многих существ в течение долгого времени.

Пространственный анализ данных - другая область большого интереса, часто на этапе предварительной обработки анализа данных. Обширная работа, завершенная для пространственного анализа данных с R (Bivand и др., 2008), может обеспечить основание для того, чтобы оно расширило Rattle в этом направлении.

Далее сосредоточьтесь на обвинении отсутствующего значения, вероятно, с введением более нормальных подходов, включая k самых близких соседей и множественного обвинения.

Начальная поддержка автоматизированной генерации отчета с использованием пакета **odfWeave,** включенного в Rattle (через кнопку Report). Шаблоны стандартного отчета разрабатываются для каждой из вкладок. Для вкладки **Data**, например, отчет обеспечивает рисунки и распределения показа таблиц и основную статистику.

Код Rattle останется открытым исходным кодом, и другие могут способствовать. Исходный код размещен Google Code (http://code.google.com/p/rattle/). Пользовательский список рассылки Rattle (http://groups.google.com/group/rattle-users) также размещен Google. Справочник с открытым исходным кодом также доступен (Уильямс, 2009a).