|  |  |
| --- | --- |
| **Изображение выглядит как текст  Автоматически созданное описание** | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | |
| Институт информационных технологий | |
| Кафедра прикладной математики  **КУРСОВАЯ РАБОТА**  по дисциплине  Языки программирования для статистической обработки данных.  **Тема курсовой работы**: Изучение алгоритма дерева решений на примере распределения продуктов питания   |  |  | | --- | --- | | Студент группы ИМБО-01-19 | Денисова Виктория Михайловна | |  |  | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) | |  |  | | Руководитель курсовой работы | Митина Ольга Алексеевна | |  |  | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись руководителя) | |  |  | | Работа представлена к защите | «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. | |  |  | | Допущен к защите | « » \_\_\_\_\_\_\_\_\_\_\_\_2021 г. |   Москва 2021 | |

|  |  |
| --- | --- |
| **Изображение выглядит как текст  Автоматически созданное описание** | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | |
| Институт информационных технологий | |

Кафедра прикладной математики

**ЗАДАНИЕ**

**на выполнение курсовой работы по дисциплине**

Технологии организации, обработки и хранения статистических данных

|  |  |
| --- | --- |
| Студент Денисова Виктория Михайловна | Группа ИМБО-01-19 |

**Тема работы:** Изучение алгоритма дерева решений на примере распределения продуктов питания

**Исходные данные:** языка программирования высокого уровня R, методическое пособие

**Перечень вопросов, подлежащих разработке, и обязательного графического материала:**

1. Код алгоритма решающего дерева в языке программирования R.
2. Рисунок дерева решений.
3. Анализ обработки данных решающего дерева.

|  |  |  |  |
| --- | --- | --- | --- |
| **Срок представления к защите курсовой работы:** | | до «\_\_\_» \_\_\_\_\_\_\_\_\_\_2021 г. | |
|  | |  | |
| **Задание на курсовую работу выдал** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | (Митина О.А.) |
|  | | «\_\_\_» \_\_\_\_\_\_\_\_\_\_2021 г. | |
| **Задание на курсовую работу получил** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | (Денисова В.М.) |

ОГЛАВЛЕНИЕ

[Введение 4](#_Toc14194)

[Теоретическая часть 5](#_Toc12634)

[Понятие алгоритма дерева решений 5](#_Toc9232)

[Метрики качества классификации 9](#_Toc5201)

[Практическая часть 12](#_Toc27987)

[Построение решающего дерева 12](#_Toc7216)

[Заключение 19](#_Toc31836)

[Список используемой литературы 20](#_Toc1596)

[Приложение 21](#_Toc22370)

# Введение

В современном мире технологии стремительно развиваются. Необходимость автоматизировать процесс принятия решений возрастает с каждым годом. Одним из самых главных преимуществ алгоритма решающего дерева по сравнению с другими классификаторами является его легкость интерпретации человеком.

Область применения дерева принятия решений довольно обширна. Алгоритм не только позволяет классифицировать по обучающим данным, но и оценивать, например, кредитоспособность клиента для займа, контроль качеством товаров, а также диагностировать предположительное заболевание по симптомам.

Дерево решений позволяет найти значимые закономерности, которые не сможет найти человек. Алгоритм дает возможность принять более выгодное и правильное решение в любой сфере.

Поэтому данная тема курсовой работы является актуальной.

Цель курсовой работы – изучение алгоритма решающего дерева с помощью средств языка программирования высокого уровня R на примере распределения продуктов питания.

Задачи, решаемые в данной курсовой работе:

* изучение научной и методической литературы по дереву решений, метрикам качества алгоритма;
* использование знаний математической статистики и теории вероятности с использованием современных средств обработки данных: языка программирования высокого уровня R;
* решение задачи распределения продуктов питания (вин) с применением решающего дерева;
* оценка качества модели по метрикам качества;
* обучение качественному оформлению документации.

# Теоретическая часть

## Понятие алгоритма дерева решений

Современный подход к принятию решений одна из главных задач автоматизации любой сферы деятельности. Неправильно принятые решения могут привести к снижению дохода компании, увеличению издержек. Чтобы автоматизировать процессы принятия решений, используется машинное обучение. Машинное обучение – обширная сфера искусственного интеллекта, которая изучает методы построения алгоритмов, способных обучаться [5].

В машинном обучении существует несколько типов алгоритмов:

* обучение с учителем, когда изначально даны признаки множества и ответ, по которым необходимо выявить зависимость;
* обучение без учителя, когда изначально не даны ответы, а необходимо выявлять зависимости между признаками множества;
* частичное обучение, т.е. некоторые множества содержат ответ, некоторые нет, и поэтому обучение происходит как совокупность обучения с учителем и без него;
* мета-обучение, когда модель на раннее решенных задачах обучения постоянно автоматически совершенствуется с течением времени.

Одной из задач обучения с учителем является классификация. Классификация – это решение задачи машинного обучения, которая заключается в разделении заданного множества с признаками на отдельные категории (подмножества) с использованием обучающегося алгоритма [5]. Классификация разделяет множество на подмножества в зависимости от их общих и различных признаков.

В настоящее время классификация занимает важное место в сфере деятельности по автоматизации, упорядочению и нормализации данных. В условиях неопределенности ситуации, решающее дерево помогает принять последовательные выборы, которые приведут к лучшему результату.

Дерево решений – это метод представления принятия решений в виде иерархической структуры, который состоит из узлов и листьев [1]. В узлах находятся вопросы для классификации, и происходит проверка условия по какому-либо атрибуту. С помощью проверки мы разбиваем множество на два подмножества, каждое из которых удовлетворяет или не удовлетворяет условию в узле.

Далее каждое подмножество делится снова с помощью узла до тех пор, пока не будет выполнено условие остановки алгоритма. В последнем листе происходит не разбиение, а деление на классы (сегменты) которые называются листьями. Лист показывает путь решения для каждого примера, находящегося в нем. В листе находятся множества с одинаковыми признаками, которые подошли под все условия, которые находятся выше.

Для наглядного представления работы алгоритма рассмотрим задачу кредитного скоринга. Клиент приходит в банк, чтобы взять кредит. По данным о клиенте решается вопрос выдать ему заем или нет. Решение о выдачи займа решается на основе данных о прошлых клиентов. На рисунке 1 представлен один из вариантов решающего дерева задачи.

В задаче скоринга решается задача бинарной классификации, потому что варианта ответа два: кредит одобрен или отказ. Главное превосходство дерева решений над остальными алгоритмами – это простота для понимания.

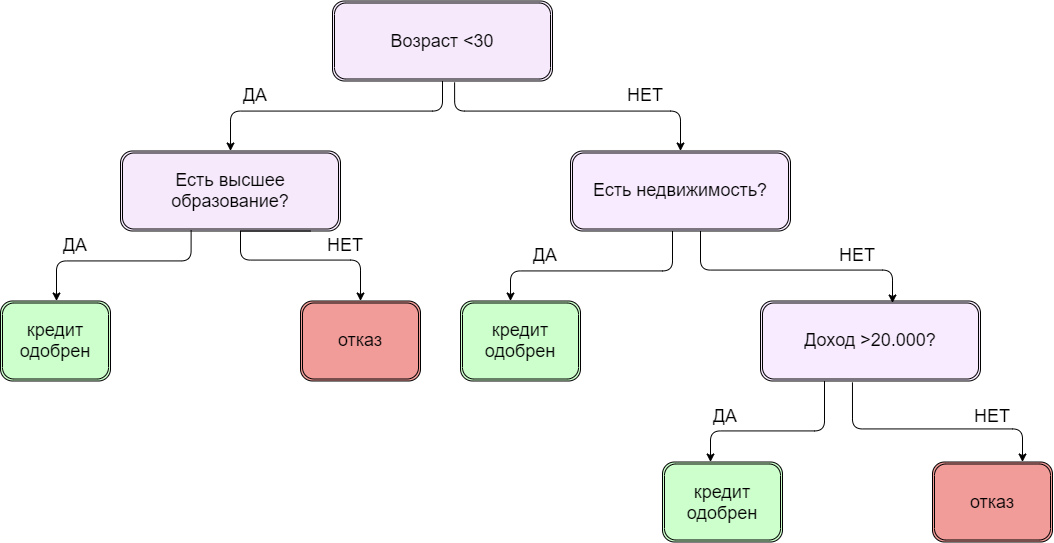


Рисунок 1 Пример решающего дерева

Итак, мы можем объяснить заемщику, почему ему отказали в кредите, анализируя модель решающего дерева. В дереве решений нет понятия «черного ящика», когда нельзя определить точно, почему принято то, или иное решение. Именно поэтому данный алгоритм получил широкий отклик у людей.

Существует множество алгоритмов построения дерева. Наиболее популярными являются ID3 (Iterative Dichotomiser 3) и C4.5.

C4.5 является усовершенствованной моделью ID3, в которой можно работать с непрерывными данными, обучаться на данных, в которых содержатся пропуски, а также производить отсечение ветвей. Подрезание дерева в C4.производится путем построения полного дерева. Затем размер дерева уменьшается за счет превращения узлов в листья. Приведем шаги построения дерева C4.5:

1. Разделение данных на обучающую и тестовую (проверочную) выборки.
2. Выращивание дерева на данных обучающей выборки.
3. Подрезание дерева.
4. Проверка на данных тестовой выборки.

Главный вопрос построения дерева заключается в том, какой признак выбрать раньше. Для решения этого вопроса введем понятие информационная энтропия. Это средняя скорость генерирования значений некоторым случайным источником данных [5]. Энтропия – это количество неизвестной информации о системе. В данном случае под системой понимается какой-либо опыт (испытание), которое имеет различные исходы, а значит и количество информации.

Энтропия вычисляется по следующей формуле:

|  |  |
| --- | --- |
| , | (1) |

где – вероятность нахождения системы в i-ом состоянии.

Чем ниже значение энтропии, тем больше информации нам известно о системе. Следовательно, лучше результаты классификации. Для улучшения качества работы дерева решений, сначала высчитывается энтропия каждого признака (атрибута) по которому будет происходить классификация. Уменьшение энтропии по-другому называют информационным приростом [9]. Высчитывается IG (information gain) по формуле:

|  |  |
| --- | --- |
| *,* | (2) |

где – число групп после разбиения, – число элементов выборки, у которых имеет -ое значение.

Затем каждый прирост информации сравнивается и выбирается тот признак, который дал наибольший IG. Тем самым классификация получается более точная и быстрая.

Решающее дерево можно построить до такой глубины, чтобы в каждом листе остался только один объект. Но таким образом мы переобучим дерево, что оно будет хорошо работать только на обучающей выборке, но не сможет качественно прогнозировать на других данных. Существует два пути решения переобучения дерева:

* искусственная остановка построения дерева путем выбора максимального числа объектов в листе или ограничение глубины;
* стрижка дерева.

Стрижка дерева заключается в том, что сначала строится дерево до максимальной глубины, а затем происходит снизу вверх «стрижка». Узел удаляется, если качество построенного дерева увеличивается. Для вычисления качества построенного дерева, введем понятие метрик.

Решающие деревья позволяют автоматизировать процесс принятия решений в различных задачах, таких как бинарная или мульти-классовая классификация.

Бинарная классификация – задача, суть которой определить объекты в один из двух классов.

Мульти-классовая классификация строится на базе бинарной, и ее задача заключается в определении объектов в некоторое количество классов. Общая схема работы мульти-классового классификатора заключается в использовании нескольких бинарных классификаторов, которые обучены прогнозировать различные признаки объектов по группам. Затем происходит сопоставление данных прогнозирования каждой бинарной группы и выявляется конечный результат.

Таким образом, решающие деревья позволяют решать задачу как бинарной, так и мульти-классовой классификации. Чтобы оценить качество построенного дерева, необходимо рассчитать соответствующие метрики качества классификации.

## Метрики качества классификации

Для того, чтобы определить качество построенной модели, выбрать лучший алгоритм работы, необходимо провести оценку модели алгоритма. В случае низких показателей предсказания необходимо внести коррективы в работу модели или выбрать другой алгоритм решения задачи.

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются метрики [4].

Метрика – это численный показатель характеристики модели машинного обучения [4]. Метрики показывают корректность работы построенной модели. Существует множество методов оценки качества классификации. Они являются инструментами для количественной оценки результатов классификации.

Выделяют внешние и внутренние методы оценки качества классификации. Внешний метод оценивает классификацию на основе сравнения с уже известным разбиением на классы. Внутренний метод оценивает качество модели по информации в данных.

Подробнее рассмотрим внешние методы оценки качества классификации. Для этого введем понятие матрицы ошибок (англ. confusion matrix). Матрица ошибок показана на рисунке 2.

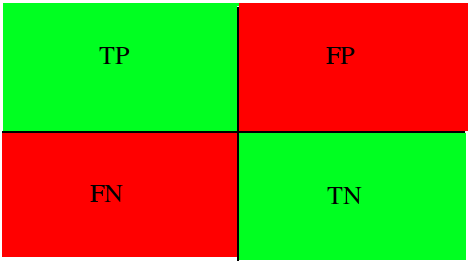


Рисунок 2 Матрица ошибок

Введем обозначения, использующиеся в матрице:

* Positive – решающее дерево отнесло объект к «положительному объекту», согласно условию;
* Negative – решающее дерево отнесло объект к «отрицательному объекту», согласно условию.

Перед этим добавляется слова True (верно) и False (неверно), которые указывают, правильно ли классифицирована модель. В итоге получаются следующие прогнозы модели:

* (True Positive) – истинно-положительные исходы, т.е. модель определила объект как положительный и определила верно;
* (False Positive) – ложно-положительные исходы, т.е. модель определила объект как положительный и определила неверно;
* (True Negative)– истинно-отрицательные исходы, т.е. модель определила объект как отрицательный и определила верно;
* (False Negative) – ложно-отрицательные исходы, т.е. модель определила объект как отрицательный и определила неверно.

С помощью приведенных выше прогнозов модели рассчитываются необходимые метрики для оценки качества классификации алгоритма.

Наиболее распространённой и простой метрикой является аккуратность (англ. Accuracy**)**. Она показывает отношение меток, которые дерево решений определило верно, ко всем меткам. Формула имеет вид:

|  |  |
| --- | --- |
|  | (3) |

Аккуратность не является самой точной метрикой, так как она не учитывает неверно определенные метки, а в некоторых задачах классификации это очень важно.

Еще одной популярной метрикой является точность (англ. Precision). Она указывает количество верно «положительно-определенных» меток ко всем «положительно-определенным» меткам. Метрика показывает, насколько модель способна отличать классы друг от друга. Формула точности:

|  |  |
| --- | --- |
|  | (4) |

Следующая метрика – полнота или чувствительность (англ. Recall). Она вычисляет отношение верно-определенных «положительных» меток ко всем меткам класса, которые алгоритм определил как «положительные». Вычисляется по формуле:

|  |  |
| --- | --- |
|  | (5) |

Специалисты по машинному обучению используют приведенные выше метрики, чтобы определить качество построенной модели для бинарной классификации. Если хотя бы одна из метрик ниже 0.75, то построенное решающее дерево будет плохо классифицировать на новых данных, следовательно, прогноз будет неправильным.

Таким образом, задача бинарной классификации в настоящее время упрощает и автоматизирует работу специалистов. Существует множество алгоритмов классификации, но так как решающее дерево является легко-интерпретируемым, то оно популярно в информационных кругах.

В данной работе мы используем решающее дерево, чтобы разделить объекты на две категории. Чтобы определить качество построенного дерева, воспользуемся метриками – аккуратность, точность, чувствительность. Они помогут оценить то, насколько прогнозы модели точны и верны.

# Практическая часть

## Построение решающего дерева

Рассмотрим бинарную задачу, в которой необходимо определить белое и красное вино. Данные о показателях вин взяты с открытого источника kaggle [11]. Рассмотрим признаки вин:

* type – тип вина (красное/белое), строкового типа данных;
* fixed acidity – фиксированная кислотность, вещественного типа данных;
* volatile acidity – летучая кислотность, вещественного типа данных;
* citric acid – лимонная кислота, вещественного типа данных
* residual sugar – остаточный сахар, вещественного типа данных;
* chlorides – хлориды, вещественного типа данных;

Для решения задачи подключим пакеты, необходимые для построения дерева решений.

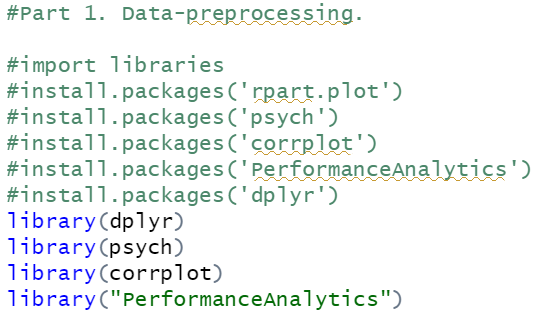


Рисунок 3 Импорт пакетов в R

На рисунке 3 изображены пакеты, которые будут использованы. Пакет «rpat.plot» необходим для отрисовки графиков и изображений. Пакет «dplyr» нужен для правильного считывания данных из файла .csv. Остальные пакеты помогут автоматизировать работу программы и сделать ее проще, используя функции пакетов.

Следующим шагом считаем данные о винах.

Вторая строчка на рисунке 4 считывает данные из файла winequalityN.csv, переводя их в таблицу. Функции str и head мы используем, чтобы рассмотреть, что считалось и в каком виде.

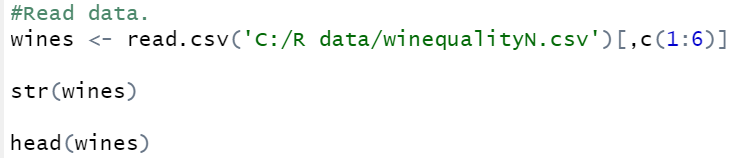


Рисунок 4 Считывание данных в R

На рисунке 5 показана командная строка и результат выполнения этих команд.

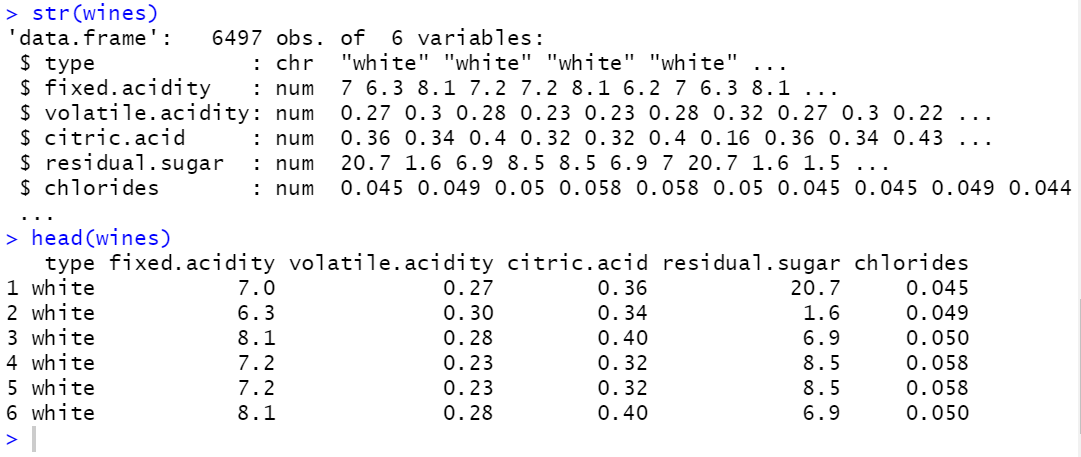


Рисунок 5 Результат выполнения команд

Далее делаем проверку, есть ли в данных NA, т.е. пропущенные значения. Если есть, то их необходимо удалить, так как они будут мешать верному прогнозированию данных. Удалим строчки данных, в которых есть NA, используя команду na.omit. Затем снова делаем проверку на пропущенные значения.

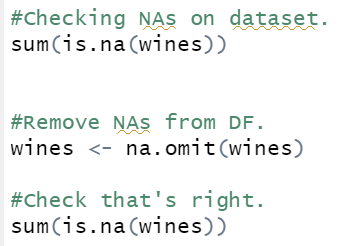


Рисунок 6 Проверка и удаление NA

На рисунке 7 отображен результат выполнения проверки и удаления пропущенных значений.

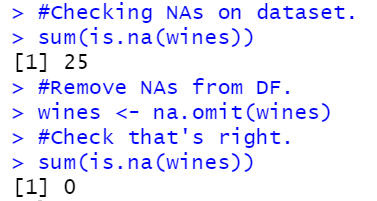


Рисунок 7 Результат удаления NA

В исходном файле о признаках вин в первых строчках содержится информация только о белых винах. Так как модель будет обучаться на первых строчках данных, то перемешаем их, чтобы модель обучалась распознавать не только белые, но и красные вина. Иначе предсказания алгоритма будут не точными. Код отображен на рисунке 8.

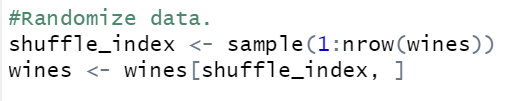


Рисунок 8 Перемешивание данных

Создадим вектор с числами от 1 до количества строчек в данных wines и перемешиваем их по вектору. Затем изменим данные wines, сопоставляя число в векторе строчке с данными.

Проверим данные на корреляцию, то есть зависимость одной переменной от другой. Для этого используем команду chart.Correlation и рассмотрим полученную матрицу. Корреляционная матрица представлена на рис. 9.

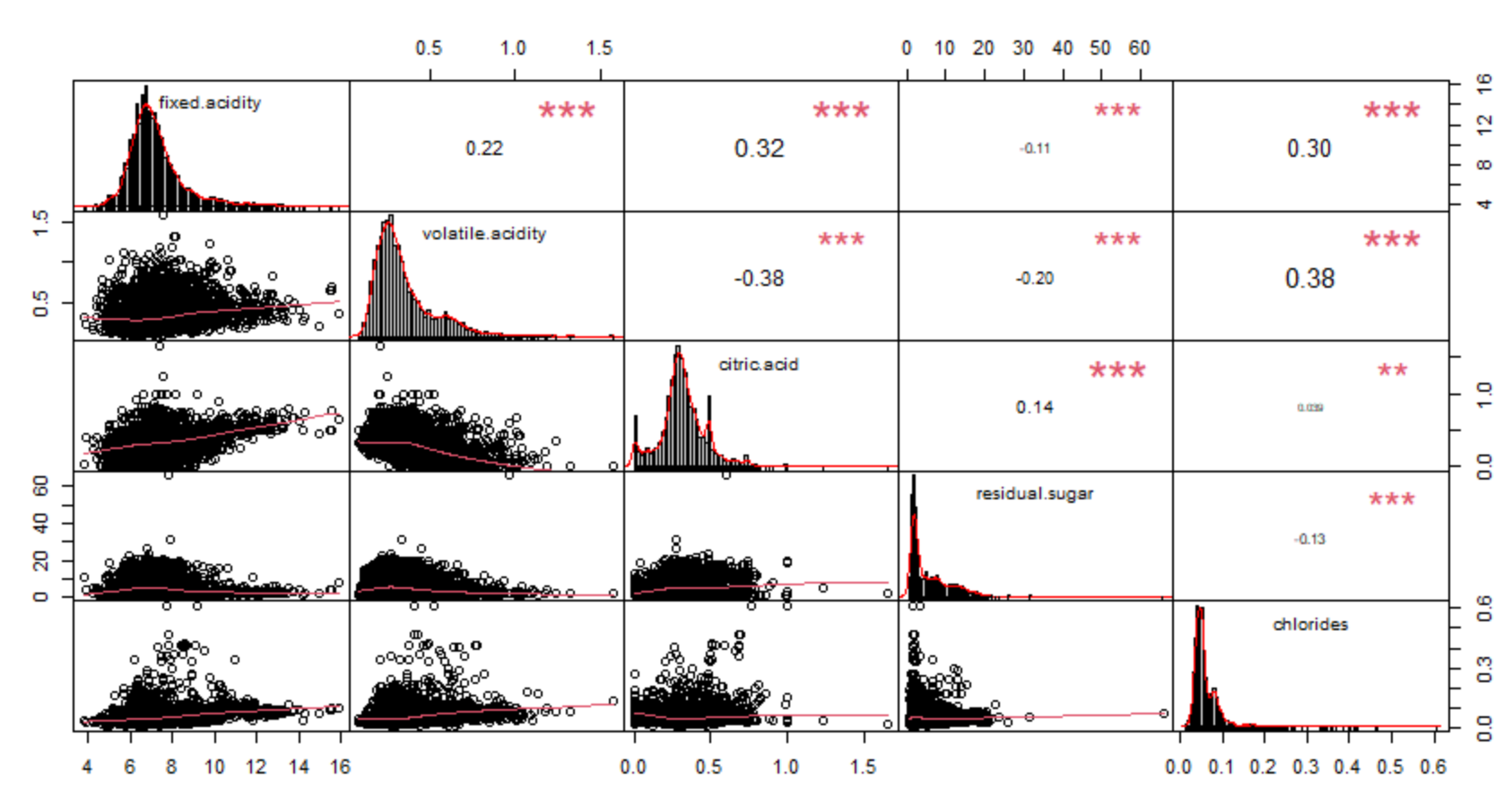


Рисунок 9 Корреляционная матрица

По диагонали расположены графики распределения каждой переменной по группам. Под главной диагональю отрисованы графики рассеяния с трендом по группам. Над главной диагональю представлены значения корреляции. Чем больше звездочек, тем выше уровень корреляции. Итак, данные не сильно коррелируют. Следовательно, нет необходимости удалять какие-либо столбцы с признаками.

Теперь, когда данные перемешаны, мы можем заменить строковые данные о типе вина. На рисунке 10 отображен код.

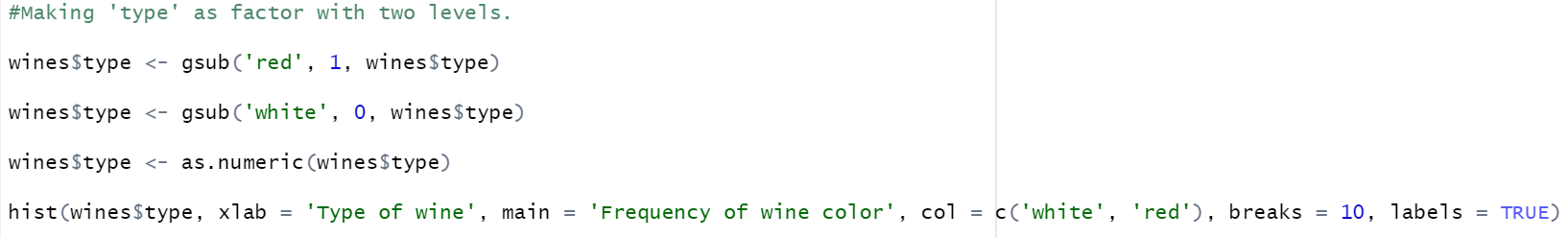


Рисунок 10 Создание «типа» как фактора с двумя уровнями

Заменяем в таблице тип вина red – на 1, white – на 0. Если переводить в обозначения матрицы ошибок, то получается, что red – Positive, white – Negative. Изменяем тип данных на числовой и нарисуем гистограмму, где распределены доли данных о красном и белом вине. Гистограмма помогает узнать, сбалансированы ли классы.

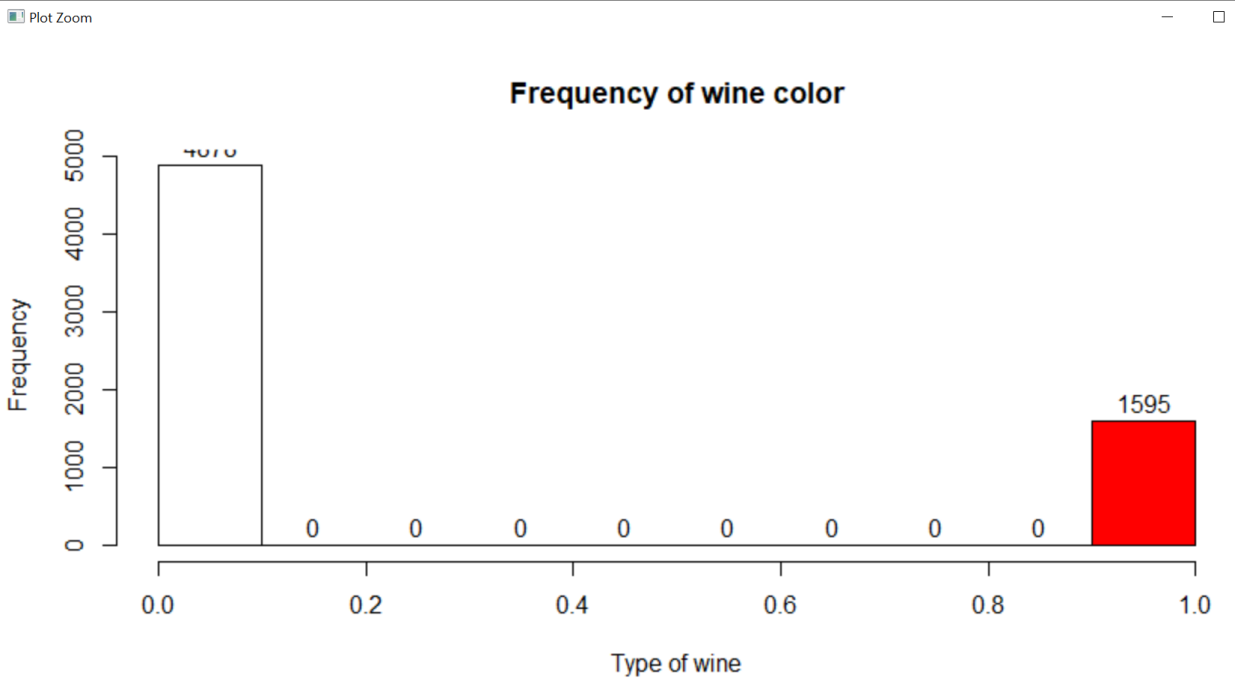


Рисунок 11 Гистограмма количества красных и белых вин

Гистограмма на рисунке 11 показывает, что информации о красном вине в данных содержится 1595, о белом – 4070, что в 2,5 раза больше. Следовательно, дерево решений может неправильно определять тип вина на тестовых данных, поэтому вычислим метрики качества модели.

Разделим данные на тестовую и обучающую выборку. Это необходимо для того, чтобы проверить корректность работы алгоритма. Если производить обучение на целом наборе данных, то протестировать дерево решений не представится возможным.

Используя разделение на тренировочную и тестовую выборки, аналитик освобождает себя от сбора дополнительных данных для тестирования модели. Если точность работы алгоритма на тренировочной выборке будет высокой, а на тестовой – низкой, это будет значить, что модель переобучилась.

Для разделения данных необходимо выбрать случайные записи из исходной таблицы и произвести обучение на данной выборке. Остальные записи попадают в тестовую выборку. На рисунке 12 показан код разделения данных.

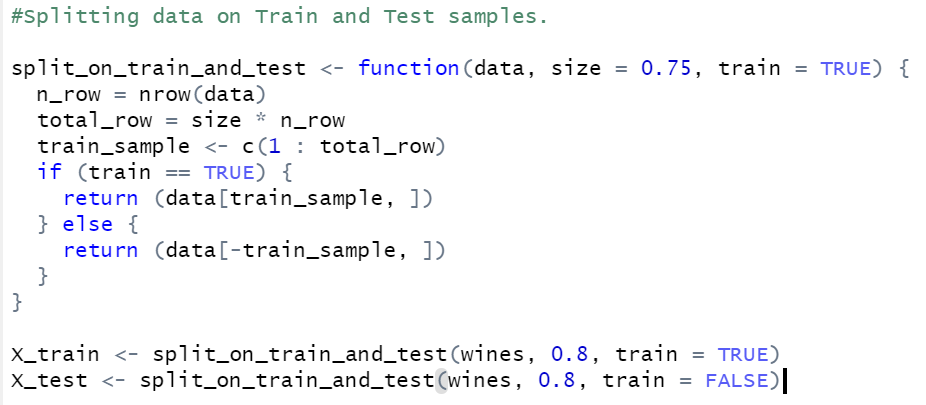


Рисунок 12 Разделение данных на тестовую и обучающую выборки

X\_train – тренировочная (обучающая) выборка, X\_test – тестовая (проверочная) выборка.

Определим количество данных для тестовой и тренировочной выборки, а также вычислим долю красного и белого вина в обучающей выборке. На рисунке 13 отображен код и доля обучающей выборки.

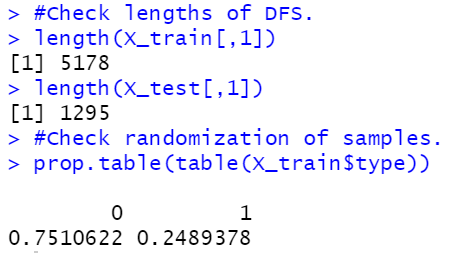


Рисунок 13 Проверка рандомизации выборок

Итак, доля белого вина больше, чем красного.

Теперь можно приступать к построению решающего дерева и его визуализации.

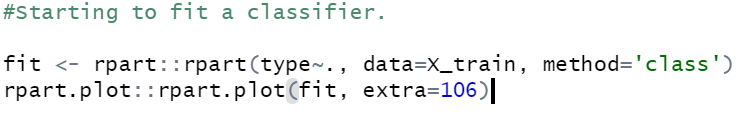


Рисунок 14 Построение дерева решений

Код на рисунке 14, визуальное представление дерева на рисунке 15.

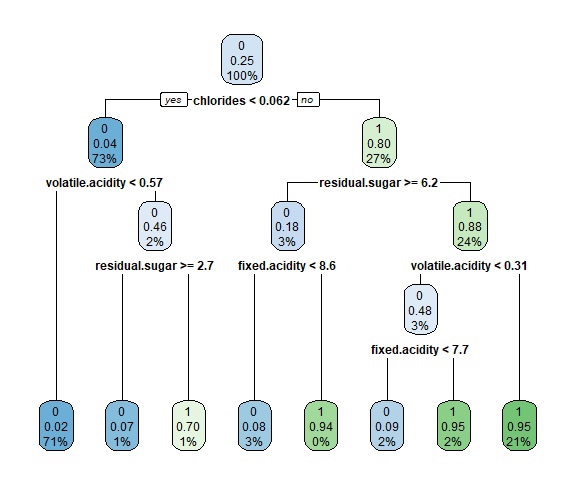


Рисунок 15 Решающее дерево

Рассмотрим рисунок 15. На узлах отображены три числа. Первое – это номер класса (0 –белое вино, 1 – красное вино). Второе – это энтропия (или информационный прирост). Чем ниже значение энтропии, тем больше информации нам дало это разделение. Третье число – это доля класса от общего числа наблюдений.

На рисунке 16 показано прогнозирование данных.



Рисунок 16 Прогнозирование данных

Теперь, когда у нас есть решающее дерево, спрогнозированные данные по тестовой выборке, оценим качество построенной модели.

На рисунке 17 показано построение матрицы ошибок.

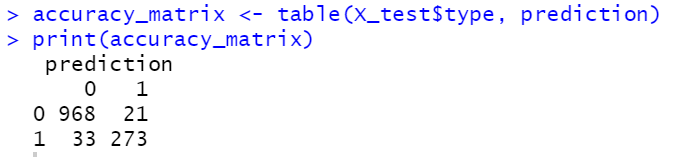


Рисунок 17 Матрица ошибок

Столбцы означают предсказание модели, т.е. отнесла она объект к красному или к белому вину. Строки показывают реальные значения вина. В названии столбцов и строк используются следующие названия: 0 – белое вино, 1 – красное вино. Верно определенные вина расположены на диагонали матрицы, т.е. (0;0) и (1;1). Вина, которые модель определила неверно, расположены в матрице в ячейках (0;1) и (1;0)

Модель верно спрогнозировала 968 раз белое вино, 273 раза красное вино. Модель 21 раз неверно определила белое вино, предположив, что это красное и 33 раза неверно определила красное вино, предположив, что оно белое.

Далее вычислим формулы аккуратности, точности и полноты.

На рисунке 18 показан код вычисления.

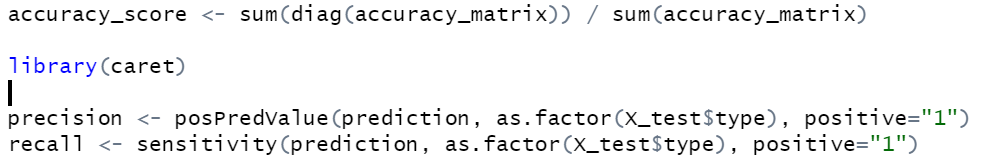


Рисунок 18 Вычисление аккуратности, точности и полноты

На рисунке 19 вывод результата вычисления метрик качества.

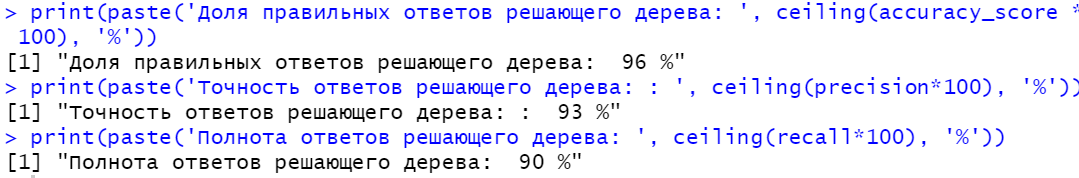


Рисунок 19 Результаты вычисления метрик качества

Полный программный код представлен в приложении 1.

Таким образом, построенная модель имеет высокий уровень прогнозирования. Доля правильных ответов составляет 96%. Точность ответов 93%. Полнота ответов 90%. Следовательно, наша модель хорошо обучена.

Исследование дерева решений показало, что данная модель хорошо справляется с решением задачи бинарной классификации и является одним из самых простых и популярных методов решения бинарных задач.

# Заключение

С каждым годом технологии развиваются. Чтобы не отставать от конкурентов, необходимо идти в ногу со временем и автоматизировать процессы насколько это возможно. Это не только более качественная классификация, чем оценка человеком, но и более быстрая.

Алгоритм решающего дерева подходит для задач бинарной классификации. Само дерево легко интерпретировать любому человеку, даже который первый раз видит данную модель. Тем самым данная модель является наиболее популярной среди людей.

В результате данного курсового проекта проведено исследование решающего дерева на языке программирования высокого уровня R,а также решена задача бинарной классификации на примере распределения продуктов питания (вин).

Качество полученной модели зависит от обучающей выборки. Необходимо, чтобы данные не сильно коррелировали между собой, были разнообразны и не содержали пропущенных значений. Тогда дерево решений будет построено максимально точно.

Цель – изучение алгоритма решающего дерева с помощью средств языка программирования высокого уровня R на примере распределения продуктов питания – достигнута.

В ходе данной курсовой работы выполнены следующие задачи:

* изучена научная и методическая литературы по дереву решений, метрикам качества алгоритма;
* использованы знания математической статистики и теории вероятности с использованием современных средств обработки данных: языка программирования высокого уровня R;
* решена задача распределения продуктов питания (вин) с применением решающего дерева;
* оценено качество модели с помощью метрик качества;
* пройдено обучение качественному оформлению документации.

# Список используемой литературы

1. Жуков, Д.А. Анализ критериев классификации при диагностике функционирования технического объекта / Д.А. Жуков // Mathematical modelling. – 2018. – С. 13
2. Полное учебное руководство по моделированию на основе деревьев с нуля на Python и R / . – : Analytics Vidya Content Team, 2016. – 20-25 с.
3. Therneau, T. M. Введение в рекурсивное сегментирование с использованием подпрограмм RPART / T. M. Therneau, E. J. Atkinson. – : Mayo Foundation, 2015. – 55 с.
4. [Просветов, Г. И.. Финансовый менеджмент: Задачи и решения: Учебно-методическое пособие. М.: Издательство РДЛ,. – 2005. – 376 с.](https://laws.studio/menedjment-finansovyiy/finansovyiy-menedjment-zadachi-resheniya.html)
5. Айвазян, С.А. Прикладная статистика: основы моделирования и первичная обработка данных / С.А. Айвазян, И.С. Енюков, Л.Д. Мешалкин. – : М.: Финансы и статистика, 1983. – 13–18 с.
6. Книпович, Н.М. Классификация в науке / Н.М. Книпович, И.С. Енюков, Л.Д. Мешалкин. – СПб : Энциклопедический словарь Брокгауза и Ефрона. – 1890–1907 с.
7. Svetnik, V. Случайный лес: Инструмент классификации и регрессии для классификации соединений и моделирования QSAR = Random Forest:  A Classification and Regression Tool for Compound Classification and QSAR Modeling / V. Svetnik, A. Liaw, J. C. Culberson. – : Journal of Chemical Information and Computer Sciences, 2003. – 1947–1958 с.
8. Webiomed [Электронный ресурс]. – Режим доступа: https://webiomed.ai/blog/osnovnye-metriki-zadach-klassifikatsii-v-mashinnom-obuchenii/ – Дата доступа: 20.04.2021.
9. Wiki.Loginom [Электронный ресурс]. – Режим доступа: <https://wiki.loginom.ru/articles/inform-entropy.html?_ga=2.258547011.811363888.1619112797-1416488438.1619112797> – Дата доступа: 20.04.2021.
10. Habr [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/ods/blog/322534/. – Дата доступа: 20.04.2021.
11. Kaggle [Электронный ресурс]. – Режим доступа: <https://www.kaggle.com/danielpanizzo/wine-quality>. – Дата доступа: 13.05.2021.

# Приложение

Листинг программы:

#THEME: Decision tree.

#TASK: Classify wines on RED and WHITE.

#Part 1. Data-preprocessing.

#import libraries

#install.packages('rpart.plot')

#install.packages('psych')

#install.packages('corrplot')

#install.packages('PerformanceAnalytics')

#install.packages('dplyr')

library(dplyr)

library(psych)

library(corrplot)

library("PerformanceAnalytics")

#Read data.

wines <- read.csv('C:/R data/winequalityN.csv')[,c(1:6)]

str(wines)

head(wines)

#Checking NAs on dataset.

sum(is.na(wines))

#Remove NAs from DF.

wines <- na.omit(wines)

#Check that's right.

sum(is.na(wines))

#Randomize data.

shuffle\_index <- sample(1:nrow(wines))

wines <- wines[shuffle\_index, ]

chart.Correlation(wines[, 2:length(wines)], histogram=TRUE, pch=20)

#Remove 'quality' from DF.

head(wines)

#Making 'type' as factor with two levels.

wines$type <- gsub('red', 1, wines$type)

wines$type <- gsub('white', 0, wines$type)

wines$type <- as.numeric(wines$type)

hist(wines$type, xlab = 'Type of wine', main = 'Frequency of wine color', col = c('white', 'red'), breaks = 10, labels = TRUE)

#Splitting data on Train and Test samples.

split\_on\_train\_and\_test <- function(data, size = 0.75, train = TRUE) {

n\_row = nrow(data)

total\_row = size \* n\_row

train\_sample <- c(1 : total\_row)

if (train == TRUE) {

return (data[train\_sample, ])

} else {

return (data[-train\_sample, ])

}

}

X\_train <- split\_on\_train\_and\_test(wines, 0.8, train = TRUE)

X\_test <- split\_on\_train\_and\_test(wines, 0.8, train = FALSE)

#Check lengths of DFS.

length(X\_train[,1])

length(X\_test[,1])

#Check randomization of samples.

prop.table(table(X\_train$type))

#PART 2. Fitting a classification model.

#Starting to fit a classifier.

fit <- rpart::rpart(type~., data=X\_train, method='class')

rpart.plot::rpart.plot(fit, extra=106)

#PART 3. Predict a class of wine.

prediction <- predict(fit, X\_test, type='class')

#PART 4. Check score of classifier.

accuracy\_matrix <- table(X\_test$type, prediction)

print(accuracy\_matrix)

accuracy\_score <- sum(diag(accuracy\_matrix)) / sum(accuracy\_matrix)

library(caret)

precision <- posPredValue(prediction, as.factor(X\_test$type), positive="1")

recall <- sensitivity(prediction, as.factor(X\_test$type), positive="1")

print(paste('Доля правильных ответов решающего дерева: ', ceiling(accuracy\_score \* 100), '%'))

print(paste('Точность ответов решающего дерева: : ', ceiling(precision\*100), '%'))

print(paste('Полнота ответов решающего дерева: ', ceiling(recall\*100), '%'))