Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Московский физико-технический институт (государственный университет)»
Факультет управления и прикладной математики
Кафедра теоретической и прикладной информатики

Работа допущена к защите
зав. кафедрой

_____ Тормасов А. Г.

«____» _____ 2014 г.

Выпускная квалификационная работа бакалавра

Тема: **Суррогат файла: статический анализ файловой системы**

Направление: 010900 – Прикладные математика и физика

Выполнил студент гр. 073 _____ Вялый Е. Ю.

Научный руководитель,
проф.,д.ф.м.н. _____ Тормасов А. Г.

Москва – 2014

Содержание

Аннотация

В работе ставится и решается задача классификации файлов пользователей. Используются алгоритмы решающих деревьев, случайного леса и градиентного бустинга над деревьями. Вводится признаковое описание файла, проводится вычислительный эксперимент по классификации. Получены приемлемые значения качества классификации.

Ключевые слова: *классификация, признаковое описание, решающее дерево, случайный лес, бустинг.*

Глава 1

Введение

В задачах облачного хранения файлов часто бывает полезно выделять некие семантические группы файлов. Далее, к этим группам можно применять ралич-ные политики хранения и доступа. Чтобы формализовать задачу, формируется признаковое описание файла - числовой вектор. Два наиболее распространенных подхода к выделению таких групп - кластеризация и классификация.

В первом подходе группы объектов выделяются таким образом, чтобы близкие в какой-либо метрике объекты принадлежали одной группе, а расстояние между группами было существенно больше расстояний внутри групп. При кластеризации группы могут быть заранее неизвестны. В терминах машинного обучения, кластеризация - обучение без учителя.

При классификации, наоборот, изначально известно множество классов, на которые необходимо разделить объекты. Также задана обучающая выборка - множество объектов, для которых известны метки классов. Классификатор настраивается на обучающей выборке, и затем может предсказывать класс произвольного нового объекта.

В работе ставится и решается задача классификации. В разделе "Постановка задачи"формально ставится задача классификации. Далее, в разделе "Известные результаты"описываются следующие алгоритмы, основанные на деревьях принятия решений: собственно решающее дерево, случайный лес и бустинг над деревьями. Также приводятся их достоинства и недостатки. В разделе "Результаты"формируется признаковое описание файлов, описывается способ оценки качества классификации и проводится вычислительный эксперимент по классификации реальных файлов. Полученное качество классификации существенно лучше такового для случайного угадывания, то есть данную задачу можно решать методами машиннного обучения.

Глава 2

Постановка задачи

2.1. Формальная постановка задачи

Пусть X - множество описаний объектов, $C = \{c_1, \ldots, c_k\}$ - конечное множество классов. Существует неизвестное отображение

$$y^*\colon F \to C,$$

причем его значения известны только на элементах конечной совокупности $T = \{(x_1, c_1), \ldots, (x_l, c_l)\} \subset X \times C$. Требуется построить алгоритм $a\colon F \to C$, способный классифицировать произвольный объект $x \in X$.[6]

2.2. Признаковое пространство

Признаком называется отображение $f\colon X \to D_f$, где X – множество объектов, D_f — множество допустимых значений признака. Если заданы признаки $f_1, \ldots, f_n$, то вектор $\mathbf{x} = (f_1(x), \ldots, f_n(x))$ называется признаковым описанием объекта $x \in X$. Признаковые описания допустимо отождествлять с самими объектами. При этом множество $X = D_{f_1} \times \cdots \times D_{f_n}$ называют признаковым пространством. В зависимости от множества D_f признаки делятся на следующие типы:

- бинарный признак: $D_f = \{0, 1\}$;
- номинальный признак: D_f — конечное множество;
- порядковый признак: D_f — конечное упорядоченное множество;
- количественный признак: D_f — множество действительных чисел.[6]

Глава 3

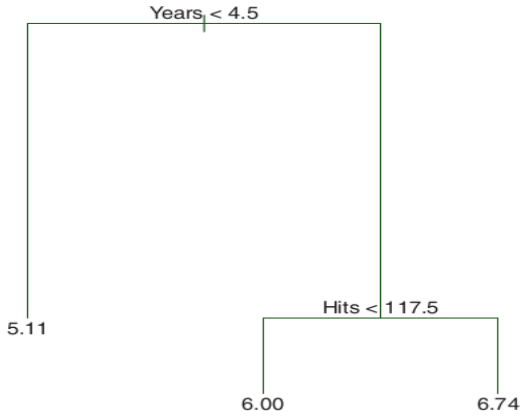
Известные результаты

3.1. Деревья принятия решений

Описание

Деревья принятия решений(Decision trees) - красивая и легко интерпретируемая модель для регрессии и классификации. В регрессионной постановке структура дерева представляет собой следующее: листы, внутренние узлы и ребра. На ребрах дерева решения записаны атрибуты, от которых зависит целевая функция, в листах записаны значения целевой функции, а в остальных узлах — атрибуты, по которым различаются случаи. Чтобы вычислить значение функции в новой точке, надо спуститься по дереву до листа и выдать соответствующую метку. Использование деревьев для классификации аналогично, в качестве целевой функции - метка класса.

Простой пример дерева принятия решений для следующих данных: логарифм зарплаты футболистов в зависимости от количества лет, сыгранных в премьер-лиге(Years) и количества забитых мячей в прошлом году(Hits)[1]:



Достоинства

Естественный учет зависимостей признаков - в случае сложных взаимодействий предикторов другие модели могут давать намного худшие результаты.
Гибкость - категориальные и числовые признаки учитываются одинаково.
Легкость интерпретации - результат классификации можно представить в виде цепочки правил вида "если A то B"

Недостатки

Точность прогноза - например, в случае данных с линейной зависимостью, линейная регрессия дает значительно лучшие результаты. Это следствие общности модели деревьев, они не учитывают специфику данных. Однако, точность прогноза можно существенно улучшить используя такие методы, как случайный лес, бустинг.

3.2. Случайный лес

Описание

Случайный лес(англ. Random forest) - алгоритм машинного обучения, заключающийся в использовании комитета(ансамбля) решающих деревьев [3]. Алгоритм сочетает в себе две основные идеи: метод бэггинга, и метод случайных подпространств.

Бэггинг(англ. Bootstrap AGgregrating, bagging) был предложен Л. Брейманом в 1996 году [2] и работает следующим образом. Пусть дана обучающая выборка D размера n. Генерируется m новых выборок D_i размера n', выбором из D случайно с возвращением. Некоторые наблюдения могут попасть в выборку несколько раз, некоторые могут не попасть вообще. Если $n' = n$ и n велико, то доля различных наблюдений в D_i будет $(1 - 1/e) \approx 63.2\%$. Далее, обучается m

классификаторов на каждой выборке D_i. При классификации новой точки, эти классификаторы голосуют и относят точку к классу, за который проголосовало большинство. В методе случайных подпространств (random subspace method, RSM) классификаторы обучаются на различных подмножествах признакового описания, которые также выделяются случайным образом.

Рассмотрим алгоритм построения случайного леса. Пусть обучающая выборка состоит из N примеров, размерность пространства признаков равна M, и задан параметр m (в задачах классификации обычно $m \approx \sqrt{M}$).

Все деревья комитета строятся независимо друг от друга по следующей процедуре:

1. Сгенерируем случайную подвыборку с повторением размером n из обучающей выборки. (Таким образом, некоторые примеры попадут в неё несколько раз, а примерно $N/3$ примеров не войдут в неё вообще)

2. Построим решающее дерево, классифицирующее примеры данной подвыборки, причём в ходе создания очередного узла дерева будем выбирать признак, на основе которого производится разбиение, не из всех M признаков, а лишь из m случайно выбранных. Выбор наилучшего из этих m признаков может осуществляться различными способами. В оригинальном коде Бреймана используется критерий Гини. В некоторых реализациях алгоритма вместо него используется критерий прироста информации.

3. Дерево строится до полного исчерпания подвыборки.

Классификация объектов проводится путём голосования: каждое дерево комитета относит классифицируемый объект к одному из классов, и побеждает класс, за который проголосовало наибольшее число деревьев.

Достоинства

- Высокое качество получаемых моделей, сравнимое с SVM и бустингом, и лучшее, чем у нейронных сетей [4].
- Способность эффективно обрабатывать данные с большим числом признаков и классов.
- Нечувствительность к масштабированию (и вообще к любым монотонным преобразованиям) значений признаков.
- Одинаково хорошо обрабатываются как непрерывные, так и дискретные признаки. Существуют методы построения деревьев по данным с пропущенными значениями признаков.
- Существует методы оценивания значимости отдельных признаков в модели.
- Внутренняя оценка способности модели к обобщению (тест out-of-bag).
- Высокая параллелизуемость и масштабируемость.

Недостатки

- Алгоритм склонен к переобучению на некоторых задачах, особенно на зашумленных задачах [5].
- Большой размер получающихся моделей. Требуется $O(NK)$ памяти для хранения модели, где N – размер обучающей выборки, K – число деревьев.

3.3. Градиентный бустинг

Описание

Бустинг(англ. Boosting) - объединение ансамбля слабых классификаторов с целью получить сильный классификатор и уменьшить смещение. Здесь сла-

бым классификатором называется классификатор, дающий лишь слегка лучший результат, чем случайное угадывание(его предсказания слабо коррелированы с истинным распределением классов). Предсказания же сильного классификатора сильно коррелированы с истинным распределением.

Финальный классификатор ищется в виде линейной комбинации классификаторов. Поиск оптимальных значений коэффициентов этой линейной комбинации - слишком трудоемкая задача, поэтому в градиентном бустинге используется жадный алгоритм постепенного добавления классификаторов.[9]

Достоинства

- Хорошая обобщающая способность. В реальных задачах (не всегда, но часто) удаётся строить композиции, превосходящие по качеству базовые алгоритмы. Обобщающая способность может улучшаться (в некоторых задачах) по мере увеличения числа базовых алгоритмов.
- Простота реализации.
- Собственные накладные расходы бустинга невелики. Время построения композиции практически полностью определяется временем обучения базовых алгоритмов.
- Возможность идентифицировать объекты, являющиеся шумовыми выбросами.[7]
- Устойчивость к переобучению.

Недостатки

- Жадная стратегия последовательного добавления приводит к построению неоптимального набора базовых алгоритмов. Для улучшения композиции можно периодически возвращаться к ранее построенным алгоритмам и обучать их заново. Для улучшения коэффициентов можно оптимизировать их

ещё раз по окончании процесса бустинга с помощью какого-нибудь стандартного метода построения линейной разделяющей поверхности.

- Бустинг может приводить к построению громоздких композиций, состоящих из сотен алгоритмов. Такие композиции исключают возможность содержательной интерпретации, требуют больших объёмов памяти для хранения базовых алгоритмов и существенных затрат времени на вычисление классификаций.[7]

Глава 4

Результаты

4.1. Признаковое описание

Рассматривается решение двуклассовой задачи классификации. Интересующий класс - кэш операционной системы или приложений. Здесь и далее:

- Класс 1 - кэш
- Класс 2 - все остальное

Будем говорить, что объект классифицирован положительно, если в результате классификации он отнесен к классу 1, иначе говорим, что объект классифицирован отрицательно.

Чтобы решать задачу классификации, необходимо определить пространство признаков. Были выбраны следующие признаки:
x_1 – размер файла в байтах,
x_2 – расширение файла, преобразованное в число(использован хэш md5),
x_3 – глубина вложенности файла в дереве файловой системы,
x_4 – средний размер файлов в данной папке у данного пользователя,
x_5 – средний размер файлов у данного пользователя.
Использование хэша для представления расширения файла считаю обоснованным, так как различные расширения соответствуют различным типам файлов и должны соответствовать различным точками в признаковом пространстве. Вероятностью коллизий при вычислении хэша пренебрегаем. По сути, расширение - категориальный признак.

Глубина вложенности у файлов кэшей, в среднем, больше, чем у остальных файлов(8.55 против 7.47 соответственно). Данное отличие статистически значимо, проверка выполнена с помощью t-теста Стьюдента и рангового критерия Уилкоксона(в обоих случаях $pvalue \approx 0$). Аналогично средний размер файла в папке и

средний размер по пользователю для кэшей значимо меньше, чем для остальных файлов.

Таким образом, размерность пространства признаков равна пяти, каждому файлу x^i соответствует точка
$x^i = (x_1, x_2, \ldots, x_5) \in \mathbf{R}^5_+$.

4.2. Семплирование

Стратегия семплирования: выбирается репрезентативная выборка файлов(размер выборки 2428105 файлов), с соотношением положительных примеров к общему числу примеров 0.155. Далее, выборка случайно разделяется на три равных части: обучающая, тестовая и контрольная. Доли положительных примеров в каждой части: 0.19, 0.13, 0.14. Подбор параметров осуществляется с использованием обучающей и тестовой выборок. Заключительная проверка выполняется на контрольной выборке.

4.3. Оценка качества

Чтобы иметь возможность оценивать качество классификатора и сравнивать их между собой, необходимо ввести меру качества. Ввиду несбалансированности классов в задаче, в качестве таковой была выбрана F1-мера, т.к. в нее не входит количество верно классифицированных отрицательных примеров(доля таких примеров может достигать 96-98%). F1-мера определяется так:

$$F_1 = \frac{|PredPositive \cap RealPositive|}{|PredPositive \cup RealPositive|},$$

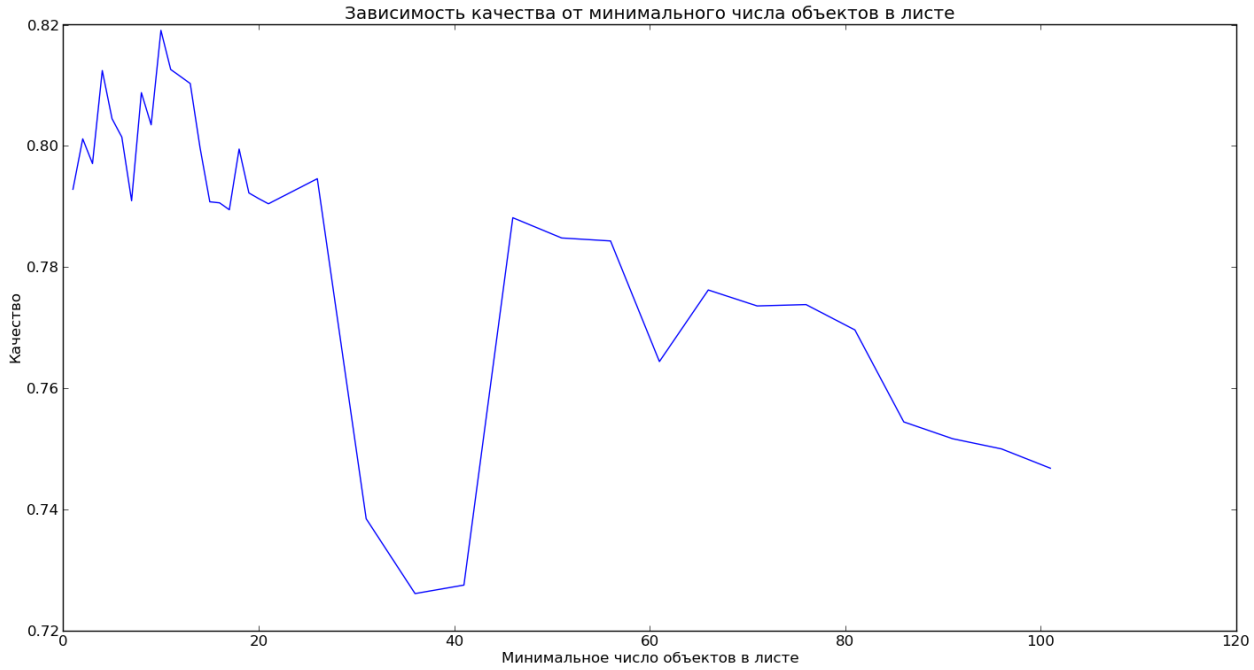
где $PredPositive$ - множество объектов, классифицированных положительно, $RealPositive$ - множество объектов, на самом деле принадлежащих классу 1. Максимальное значение меры равно 1, чем ее значение меньше тем больше классификатор пропускает положительных примеров и неверно классифицирует отрицательных.

4.4. Выбор классификатора

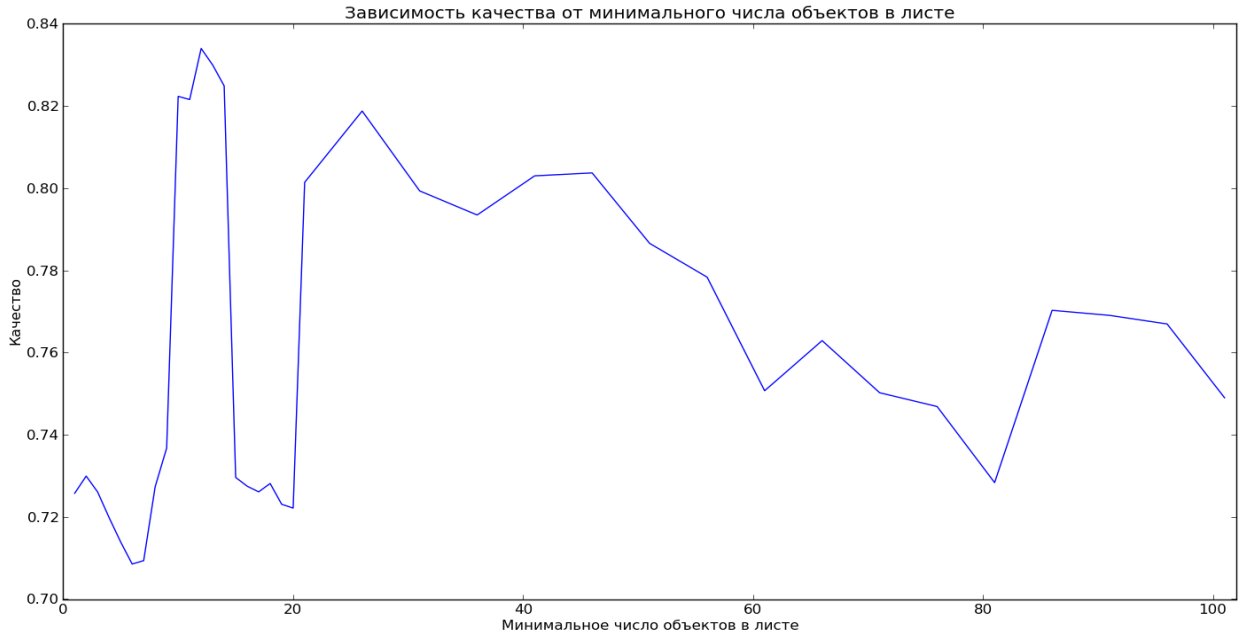
Основа всех нижеприведенных алгоритмов - решающие деревья, так как они легко интерпретируемы и хорошо обрабатывают категориальные и числовые признаки одновременно.

4.4.1. Решающее дерево

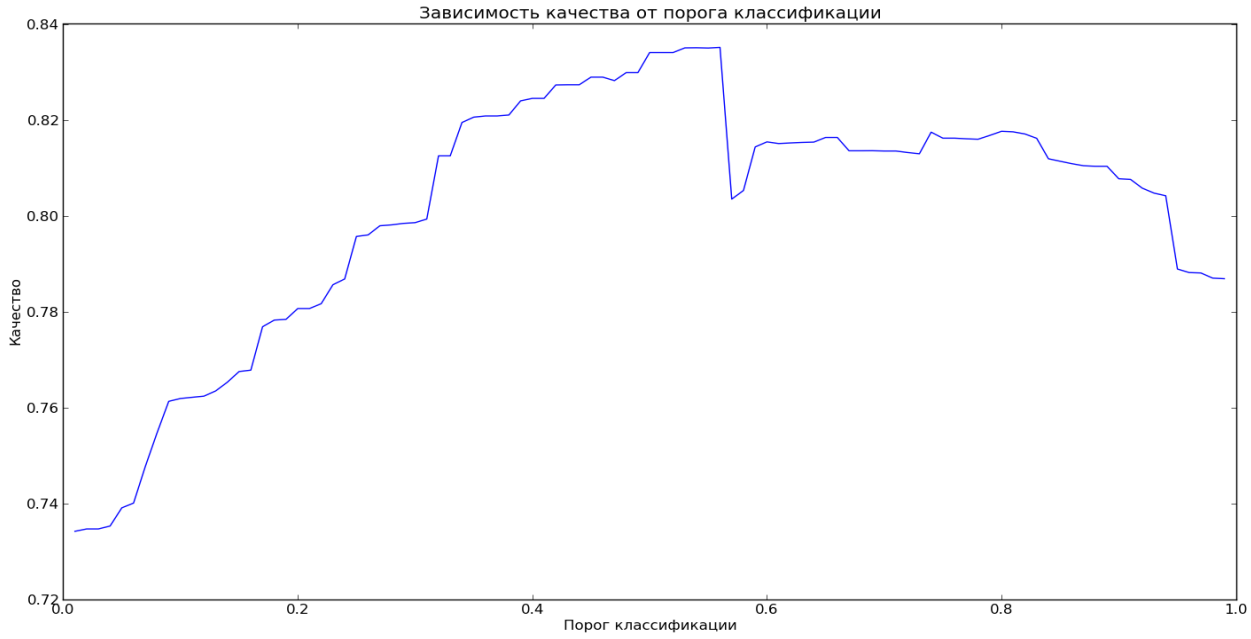
Используемый классификатор - решающее дерево. Имеется два параметра, существенно влияющих на качество классификации: число объектов в листе и порог классификации(если вероятность отнесения объекта к классу 1 больше этого порога, объект классифицируется положительно). Первый настраиваемый параметр - минимальное число объектов в листе дерева(min samples per leaf). Были исследованы значения параметра от 1 до 100. График зависимости качества классификации от настраиваемого параметра(при построении дерева использован критерий Джини):



Максимум качества достигается при минимальном числе объектов в листе равном 10. Качество при этом составляет 81.9%. Аналогичный график при использовании критерия прироста информации для построения дерева:



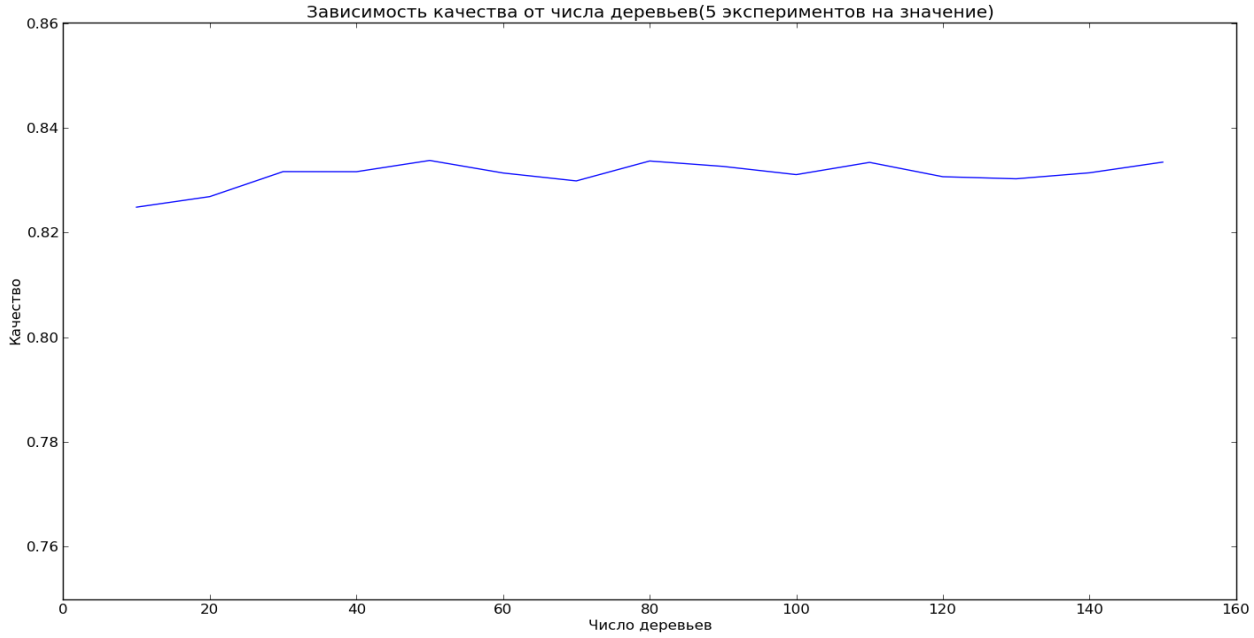
Максимум качества достигается при числе объектов в листе 12, качество при этом уже выше: 83.4%. Для дальнейшей настройки используем критерий прироста информации. Посмотрим на качество при различных порогах:

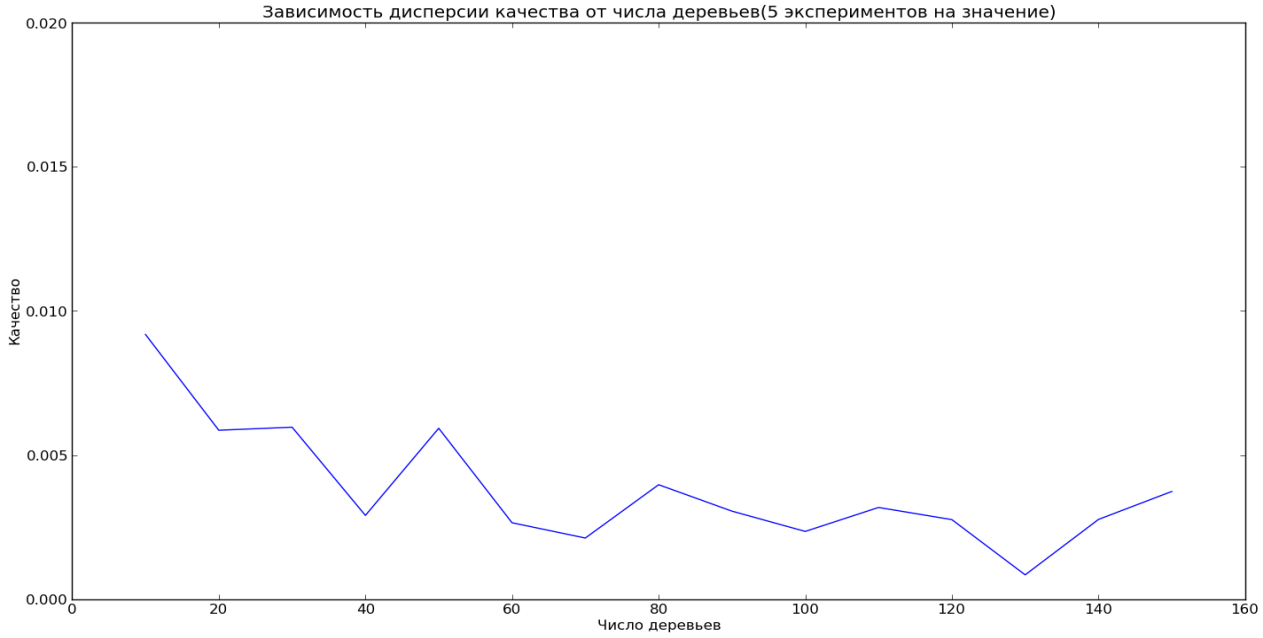


Максимум качества приходится на порог 0.50–0.56. Будем считать порог равным 0.5. Итоговые параметры классификатора: минимальное число объектов в листе дерева 12, качество на тестовой выборке 83.4%. Значение F-меры на контрольной выборке: 0.52.

4.4.2. Случайный лес

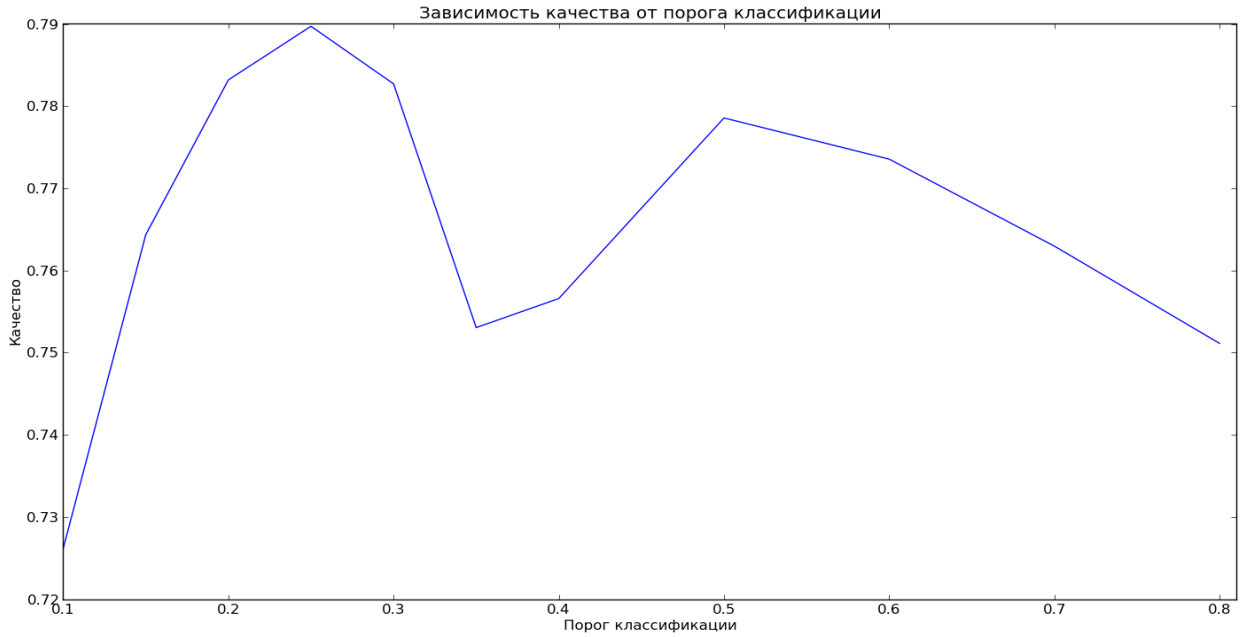
У случайного леса имеются три интересующих меня параметра: число деревьев в ансамбле, порог классификации и минимальный размер листа. Так как построение леса не детерминировано, необходимо усреднение результатов предсказаний по нескольким экспериментам. Выбираются некоторые значения параметров, строится график зависимости среднего и стандартного отклонения качества от числа деревьев в ансамбле:





Как и следовало ожидать, дисперсия качества уменьшается с ростом числа деревьев. При достаточно большом числе деревьев, проведения нескольких экспериментов для усреднения не требуется. Далее при настройке параметров используем 100 деревьев.

Далее, так как число признаков мало(равно пяти), нет необходимости в случайном выборе подмножества признаков при построении деревьев. Осталось проварьировать порог классификации:

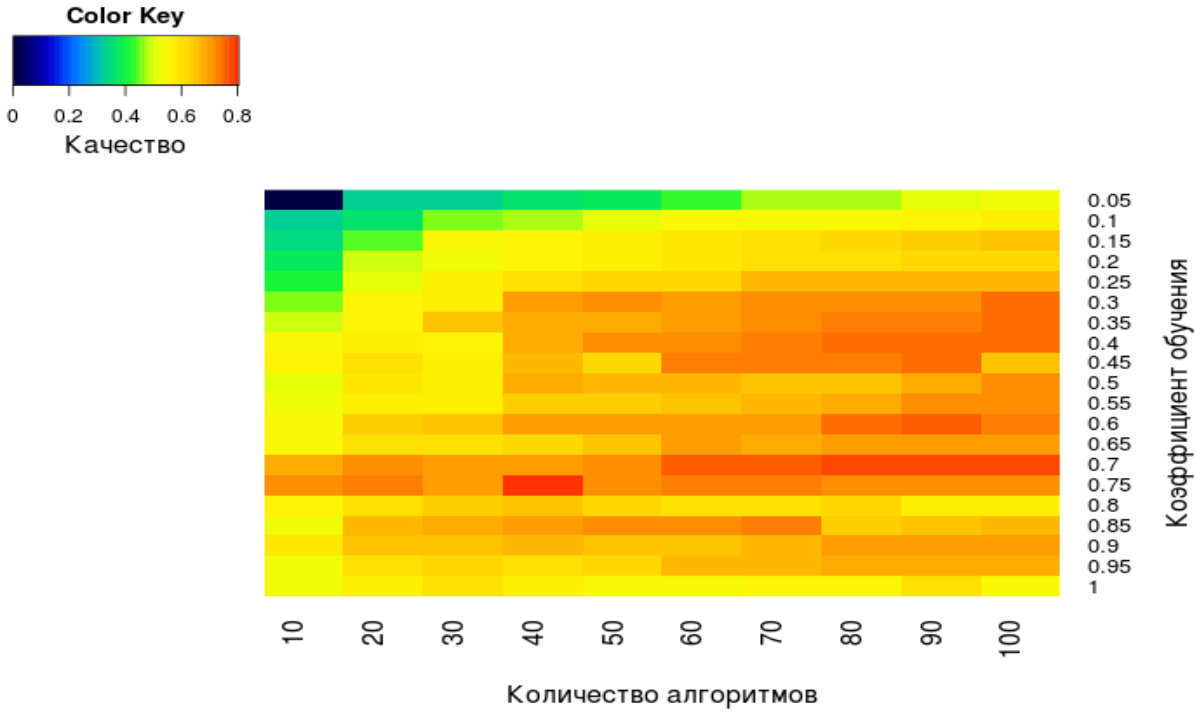


Видно, что максимум качества достигается при пороге классификации 0.25, качество при этом составляет 79%. Итоговые параметры: минимальное число объектов в листе: 12, количество деревьев: 100, порог классификации: 0.25, качество на тестовой выборке: 79%. Значение F-меры на контрольной выборке: 0.60.

4.4.3. Градиентный бустинг над решающими деревьями

Исследовался алгоритм градиентного бустинга над решающими деревьями. Настраивались два параметра: коэффициент обучения(англ. learning rate) - вклад каждого классификатора, и число деревьев. Использовать поиск по каждому из параметров последовательно не получится, так как параметры сильно взаимосвязаны. Двумерная карта зависимости качества от коэффициента обучения и от

числа деревьев, значения качества кодируются цветом:



Путем поиска по двумерной сетке этих параметров были выбраны оптимальные значения коэффициент обучения $= 0.75$, число деревьев $= 40$. Значение F-меры при этом 0.80. Значение F-меры на контрольной выборке: 0.65.

Глава 5

Заключение

5.1. Возможные улучшения

Есть обширные возможности для улучшения качества классификации для применения к реальным задачам. Далее приведены некоторые из них:

- Использование более информативных признаков, например: время создания, последнее время доступа, права доступа(категориальная переменная), количество изменений файла за определенный промежуток времени.
- Обучение классификатора на более вариативной выборке.
- Уменьшение шага подбора параметров, т.е. более точная настройка классификаторов.
- Использование смесей классификаторов(формирование результата как линейной комбинации ответов нескольких классификаторов).

5.2. Выводы

В работе поставлена и решена задача бинарной классификации файлов на реальной компьютерной системе. В качестве интересующей группы файлов была выбрана группа "кэш приложений". Было разработано признаковое описание файлов. Были использованы следующие алгоритмы классификации: деревья принятия решений, случайный лес, градиентный бустинг. Для измерения качества и сравнения алгоритмов использовалась F-мера, так как классы в задаче сильно несбалансированы. Экспериментально показано, что методы машинного обучения применимы в данной задаче и дают приемлемое качество. Лучшее качество показал алгоритм градиентного бустинга(значение F-меры на контрольной выборке:

0.65). Для сравнения, при случайном угадывании, значение F-меры составляет 0.12, т.е. исследованные алгоритмы существенно лучше случайного угадывания.

Литература

1. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. An Introduction to Statistical Learning with Applications in R // Springer, 2013.

2. Breiman L. Bagging predictors // Machine Learning, 1996, vol. 24, no. 2, pp. 123–140.

3. Leo Breiman. Random Forests // Machine Learning, October 2001, Volume 45, Issue 1, pp 5-32.

4. Caruana R., Niculescu-Mizil A. An Empirical Comparison of Supervised Learning Algorithm // Department of Computer Science, Cornell University, Ithaca, NY 14853 USA.

5. Mark R. Segal. Machine Learning Benchmarks and Random Forest Regression // Division of Biostatistics, University of California, San Francisco, CA 94143-0560, April 14, 2003.

6. К. В. Воронцов. Математические методы обучения по прецедентам(теория обучения машин) // курс лекций, МФТИ(2004).

7. К. В. Воронцов. Лекции по алгоритмическим композициям // 7 октября 2012 г.

8. К. В. Воронцов. Лекции по линейным алгоритмам классификации // 19 января 2009 г.

9. L. Mason , J. Baxter , P. Bartlett , M. Frean. Boosting Algorithms as Gradient Descent // 2000