КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

D. Skachko

INFORMATION TECHNOLOGY IN DECISION MAKING BASED ON CLASSIFICATION AND INDOOR MOVEMENT ANALYSIS

In this article key elements for data analysis are considered, which are obtained via Wi-Fi signal. Based on user movements inside buildings and its analysis it is possible to do user classification.
Key words: decision tree, CART, indoor navigation, decision making.

Розглядаються ключові моменти аналізу даних, зібрані за допомогою технології Wi-Fi. Завдяки аналізу даних стає можливим робити класифікацію відвідувачів, грунтуючись на їх переміщеннях по приміщенням.
Ключові слова: дерево рішень, алгоритм CART, навігація, прийняття рішень.

Рассматриваются ключевые моменты анализа данных, собранные с помощью коммуникационной технологии Wi-Fi. Благодаря анализу данных становится возможным делать классификацию посетителей, на основании их перемещения по помещениям.
Ключевые слова: дерево решений, алгоритм CART, навигация, принятие решений.

УДК 004.8

Д.А. СКАЧКО

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ ПРИНЯТИЯ РЕШЕНИЙ НА ОСНОВЕ КЛАССИФИКАЦИИ И АНАЛИЗА ПЕРЕМЕЩЕНИЙ ВНУТРИ ПРОСТРАНСТВЕННЫХ ОБЪЕКТОВ

Введение. Сегодня навигационные сервисы используются в различных отраслях науки, экономики, образования. Навигация на дорогах, в открытом пространстве с помощью мобильных сервисов стала нормой жизни. Для торговых центров большой интерес представляет навигация внутри помещения. Это позволит во многом автоматизировать принятие решений в управлении и увеличить прибыль. В настоящее время, когда у многих людей есть мобильные устройства, а торговые центры имеют все необходимое для организации внутренней навигации, разработка сервиса для мобильных устройств представляется актуальной задачей. Механизм навигации, предлагаемый в статье, разработан для программно-аппаратного комплекса «Shopуу», т. е. навигации внутри помещений, и основываются на коммуникационной технологии Wi-Fi. Для работы сервиса требуется развернутая сеть беспроводного доступа и поддерживающее ее клиентское мобильное устройство на платформе Android или iOS.

Полученные данные позволят классифицировать посетителей, что дает возможность делать целевые акции для нужной аудитории. Понимание перемещения людей, а так же целей их посещений открывает огромные возможности по оптимизации принятия решений при управлении огромными торговыми центрами.

Общая часть. Принцип работы сервиса заключается в следующем. Когда клиентское устройство находится между несколькими

Wi-Fi-источниками, по относительному уровню получаемого от них сигнала можно с приемлемой точностью определить его местоположение. Причем сами Wi-Fi-точки могут быть закрыты: требуется только знать относительный уровень сигнала, полученного пользователем, и сравнивать его с радиообстановкой в контрольных точках зданий. Чем больше плотность покрытия области точками, тем выше точность навигации. Например, проводится позиционирование, приложение отображает поэтажные планы здания с указанием местоположения мобильного устройства. Координаты мобильного устройства отслеживаются непрерывно. Пользователь может воспользоваться функцией прокладывания оптимального маршрута до места интереса (магазина, офисного помещения, выхода и т. п.), ведением по маршруту. Выбор интересующего объекта осуществляется либо непосредственно на плане, либо через встроенную поисковую систему. Представленный сервис навигации использует самый простой и доступный тип карт. Фактически это изображение местности, к которому привязываются географические координаты [1, 2].

Анализ данных. Система собирает все данные о перемещении пользователя, а также фиксирует места, где пользователь проводит больше времени, места интереса. Предварительный анализ данных выполнялся с помощью деревьев решений. Построение дерева решений осуществляется на основе алгоритма CART [3]. Алгоритм строит бинарные деревья решений, содержащие только два потомка в каждом узле. В процессе работы происходит рекурсивное разбиение примеров обучающего множества на подмножества, записи в которых имеют одинаковые значения целевой переменной.

$$\mathrm{Q(s \mid t)} = 2P_L P_R \sum_{j=1}^{N} (P(j \mid t_L) - P(j \mid t_R)),$$

где s – идентификатор разбиения, t – идентификатор узла, t_L и t_R – левый и правый потомки узла t соответственно, P_L и P_R – отношение числа примеров в левом и правом потомках к их общему числу в обучающем множестве, $P(_j|t_L)$ и $P(_j|t_R)$ – отношение числа примеров класса j в левом и правом потомках к их общему числу в каждом из них. Процесс построения регрессионных деревьев решений в основном аналогичен классификационным, но вместо меток классов в листьях будут располагаться числовые значения (рис. 1). Фактически при этом реализуется кусочно-постоянная функция входных переменных [4].

В результате в каждом листе должны оказаться примеры с похожими значениями выходной переменной. Чем ближе они будут, тем меньше станет их дисперсия. Поэтому она является хорошей мерой «чистоты» узла. Тогда наилучшим разбиением в узле является то, которое обеспечит максимальное уменьшение дисперсии выходной переменной в нем. В процессе роста дерева алгоритм CART проводит для каждого узла полный перебор всех атрибутов, на основе которых может быть построено разбиение, и выбирает тот, который максимизирует значение показателя. На каждом шаге построения дерева алгоритм последовательно сравнивает все возможные разбиения для всех атрибутов, выбирает

наилучший атрибут (интерес пользователя) и наилучшее разбиение для него. Полученные результаты показывают, где пользователь мобильного устройства находился больше времени, какие объемы проходил без задержки и в какие возвращался. Таким образом, рассмотренная модель содержит объекты пространства и полученные с сенсоров мобильных устройств характеристики сигналов, которые описываются скрытыми и наблюдаемыми состояниями соответственно. Кроме того, данная модель имеет параметры: переходные вероятности между скрытыми состояниями, значения которых определяются из графовой модели пространства, и вероятности перехода от скрытых состояний к наблюдаемым. Следовательно, применительно к задаче определения текущего местоположения, имеются известные параметры (вероятности переходов) и наблюдаемая последовательность $V = \{v_1, v_2, \ldots, v_p\}$, требуется подобрать последовательность состояний системы $Q = (q_1, \ldots, q_p)$, которая лучше всего соответствует наблюдаемой последовательности [5].

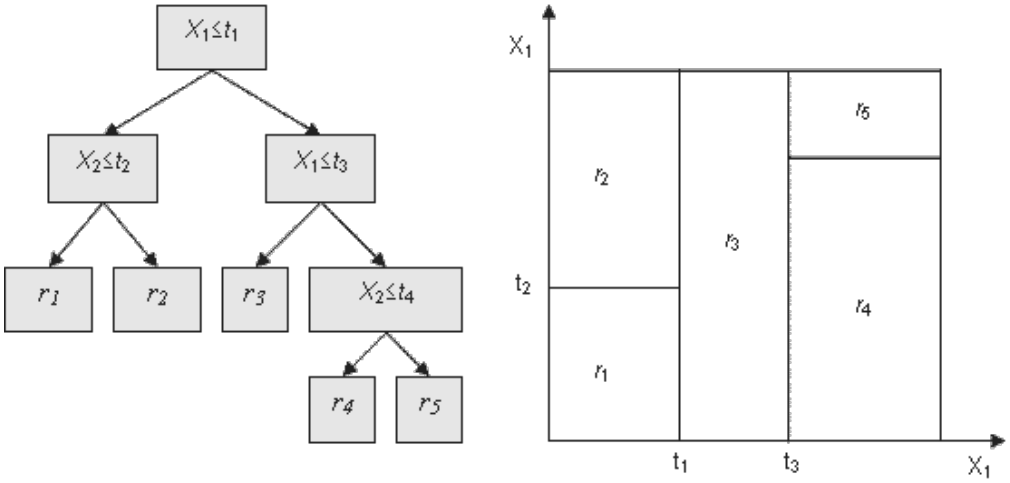

РИС. 1. Построение регрессионного дерева решений

Описание алгоритма CART. CART (Classification And Regression Tree), переводится как "Дерево Классификации и регрессии" – алгоритм бинарного дерева решений, впервые опубликованный Бриманом в 1984 году. Алгоритм предназначен для решения задач классификации и регрессии [6]. Обучение дерева решений относится к классу обучения с учителем, т. е. обучающая и тестовая выборки содержат *классифицированный* набор примеров. Оценочная функция, используемая алгоритмом CART, базируется на интуитивной идее уменьшения нечистоты (неопределённости) в узле. Узел имеет максимальную "нечистоту". В алгоритме CART идея "нечистоты" формализована в индексе Gini. Если набор данных *T* содержит данные *n* классов, тогда индекс Gini определяется как:

$$\text{Gini}(T) = 1 - \sum_{i=1}^{n} p_i^2,$$

где p_i – вероятность (относительная частота) класса i в T. Если набор T разбивается на две части T_1 и T_2 с числом примеров в каждом N_1 и N_2 соответственно, тогда показатель качества разбиения будет равен:

$$\text{Gini}_{\text{split}}(T) = \frac{N_1}{N}\text{Gini}(T_1) + \frac{N_2}{N}\text{Gini}(T_2).$$

Наилучшим считается то разбиение, для которого $\text{Gini}_{\text{split}}(T)$ минимально. Обозначим N – число примеров в узле – предке, L, R – число примеров соответственно в левом и правом потомке, l_i и r_i – число экземпляров i-го класса в левом/правом потомке. Тогда качество разбиения оценивается по следующей формуле:

$$\text{Gini}_{\text{split}} = \frac{L}{N}\left(1 - \sum_{i=1}^{n}\left(\frac{l_i}{L}\right)^2\right) + \frac{R}{N}\left(1 - \sum_{i=1}^{n}\left(\frac{r_i}{R}\right)^2\right) \to \min.$$

Чтобы уменьшить объем вычислений формулу можно преобразовать.

$$\text{Gini}_{\text{split}} = \frac{1}{N}\left(L\left(1 - \frac{1}{L^2}\sum_{i=1}^{n} l_i^2\right) + R\left(1 - \frac{1}{R^2}\sum_{i=1}^{n} r_l^2\right)\right) \to \min.$$

Так как умножение на константу не играет роли при минимизации:

$$\text{Gini}_{\text{split}} = L - \frac{1}{L}\sum_{i=1}^{n} l_l^2 + R - \frac{1}{R}\sum_{i=1}^{n} r_l^2 \to \min,$$

$$\text{Gini}_{\text{split}} = N - \left(\frac{1}{L}\sum_{i=1}^{n} l_l^2 + \frac{1}{R}\sum_{i=1}^{n} r_l^2\right) \to \min,$$

$$G_{\text{split}} = \frac{1}{L}\sum_{i=1}^{n} l_l^2 + \frac{1}{R}\sum_{i=1}^{n} r_l^2 \to \max.$$

В итоге, лучшим будет то разбиение, для которого величина *максимальна.*

Правила разбиения. Вектор предикторных переменных, подаваемый на вход дерева может содержать как числовые (порядковые), так и категориальные переменные. В любом случае в каждом узле разбиение идет только по одной переменной. Если переменная числового типа, то в узле формируется правило ви-

да *xi* <= *c*, где *c* – некоторый порог, который чаще всего выбирается как среднее арифметическое двух соседних упорядоченных значений переменной *xi* обучающей выборки. Если переменная категориального типа, то в узле формируется правило *xi* *V*(*xi*), где *V*(*xi*) – некоторое непустое подмножество множества значений переменной *xi* в обучающей выборке. Следовательно, для *n* значений числового атрибута алгоритм сравнивает *n*–1 разбиений, а для категориального (2*n*–1 – 1). На каждом шаге построения дерева алгоритм последовательно сравнивает все возможные разбиения для всех атрибутов и выбирает наилучший атрибут и наилучшее разбиение для него.

Механизм отсечения дерева. Механизм отсечения дерева, оригинальное название minimal cost-complexity tree pruning, – наиболее серьезное отличие алгоритма CART от других алгоритмов построения дерева. CART рассматривает отсечение как получение компромисса между двумя проблемами: получение дерева оптимального размера и получение точной оценки вероятности ошибочной классификации. Основная проблема отсечения – большое количество всех возможных отсеченных поддеревьев для одного дерева. Более точно, если бинарное дерево имеет |*T*| – листьев, тогда существует ~[$1.5028369^{|T|}$] отсечённых поддеревьев. И если дерево имеет хотя бы 1000 листьев, тогда число отсечённых поддеревьев становится просто огромным.

Базовая идея метода – не рассматривать все возможные поддеревья, ограничившись только "лучшими представителями" согласно приведённой далее оценке. Обозначим |*T*| – число листьев дерева, *R*(*T*) – ошибка классификации дерева, равная отношению числа неправильно классифицированных примеров к числу примеров в обучающей выборке. Определим $C\alpha(T)$ – полную стоимость (оценку/показатель затраты-сложность) дерева T как: $C\alpha(T) = R(T) + \alpha * |T|$, где |*T*| – число листьев (терминальных узлов) дерева, – некоторый параметр, изменяющийся от 0 до $+\infty$. Полная стоимость дерева состоит из двух компонент – ошибки классификации дерева и штрафа за его сложность. Если ошибка классификации дерева неизменна, тогда с увеличением полная стоимость дерева будет увеличиваться. Тогда менее ветвистое дерево, дающее большую ошибку классификации может стоить меньше, чем дающее меньшую ошибку, но более ветвистое. Определим $T_{\max}$ – максимальное по размеру дерево, которое предстоит обрезать. Если мы зафиксируем значение α, тогда существует наименьшее минимизируемое поддерево α, которое выполняет следующие условия:

$$C\alpha(T(\alpha)) = \min_{T <= T_{\max}} C\alpha(T), \tag{1}$$

$$if C\alpha(T) = C\alpha(T(\alpha)) \text{then} T(\alpha) <= T. \tag{2}$$

В условии (1) видно, что не существует такого поддерева дерева $T_{\max}$, которое имело бы меньшую стоимость, чем $T(\alpha)$ при этом значении α. В условии (2) – если существует более одного поддерева, имеющего данную полную стои-

мость, тогда мы выбираем наименьшее дерево. Можно показать, что для любого значения существует такое наименьшее минимизируемое поддерево. Но эта задача не тривиальна. Что она говорит – что не может быть такого, когда два дерева достигают минимума полной стоимости и они несравнимы, т. е. ни одно из них не является поддеревом другого. Мы не будем доказывать этот результат. Хотя имеет бесконечное число значений, существует конечное число поддеревьев дерева T_{max}. Можно построить последовательность уменьшающихся поддеревьев дерева T_{max}: $T_1 > T_2 > T_3 > ... > \{t_1\}$ (где t_1 – корневой узел дерева) такую, что T_k – наименьшее минимизируемое поддерево для $[\alpha_k, \alpha_{k+1}]$. Это важный результат, так как он означает, что мы можем получить следующее дерево в последовательности, применив отсечение к текущему дереву. Это позволяет разработать эффективный алгоритм поиска наименьшего минимизируемого поддерева при различных значениях α. Первое дерево в этой последовательности – наименьшее поддерево дерева T_{max} имеющее такую же ошибку классификации, как и T_{max}, т. е. $T_1 = T(\alpha = 0)$. Пояснение: если разбиение идет до тех пор, пока в каждом узле останется только один класс, то $T_1 = T_{max}$, но, так как часто применяются методы ранней остановки (prepruning), тогда может существовать поддерево дерева T_{max} , имеющее такую же ошибку классификации. Для того чтобы получить следующее дерево в последовательности и соответствующее значение α, тогда обозначим T_t – ветвь дерева T с корневым узлом t. Если мы отсечём в узле t, тогда его вклад в полную стоимость дерева $T - T_t$ станет $C_\alpha(\{t\}) = R(t) + \alpha$, где $R(t) = r(t)* p(t)$, $r(t)$ – это ошибка классификации узла t и $p(t)$ – пропорция случаев, которые "прошли" через узел t. Альтернативный вариант: $R(t)= m/n$, где m – число примеров классифицированных некорректно, а n – общее число классифицируемых примеров для всего дерева. Вклад T_t в полную стоимость дерева T составит

$$C_\alpha(T_t) = R(T_t) + \alpha |T_t|,$$

где

$$R(T_t) = \sum t' \in T_t R(t').$$

Дерево $T - T_t$ будет лучше, чем T, когда $C_\alpha(\{t\}) = C_\alpha(T_t)$, потому что при этой величине они имеют одинаковую стоимость, но $T - T_t$ наименьшее из двух. Когда $C_\alpha(\{t\}) = C_\alpha(T_t)$ получаем:

$$R(T_t) + \alpha|T_t| = R(t) + \alpha$$

решая для α, получаем:

$$\alpha = \frac{R(t) - R(T_t)}{|T_t| - 1}.$$

Так для любого узла t в T_1 , если увеличиваем α, тогда когда

$$\alpha = \frac{R(t) - R(T_{1,t})}{\left|T_{1,t}\right| - 1},$$

дерево, полученное отсечением в узле t, будет лучше, чем T_1.

Основная идея состоит в следующем: вычислим это значение для каждого узла в дереве T_1, и затем выберем "слабые связи" (их может быть больше чем одна), т. е. узлы для которых величина является наименьшей.

$$g(t) = \frac{R(t) - R(T_{1,t})}{\left|T_{1,t}\right| - 1}.$$

Мы отсекаем T_1 в этих узлах, чтобы получить T_2 – следующее дерево в последовательности. Затем мы продолжаем этот процесс для полученного дерева и так пока мы не получим корневой узел (дерево в котором только один узел).

Выбор финального дерева. Итак, мы имеем последовательность деревьев, нам необходимо выбрать лучшее дерево из неё. То, которое будем использовать в дальнейшем. Наиболее очевидным является выбор финального дерева через тестирование на тестовой выборке. Дерево, давшее минимальную ошибку классификации, и будет лучшим. Однако, это не единственно возможный путь. Также есть процесс построения дерева и процесс получения последовательности уменьшающихся поддеревьев методом cost-complexity pruning. В последовательности деревьев, необходимо выбрать лучшее дерево из неё. То, которое будем использовать в дальнейшем. Отметим, что отсечение дерева не использовало никаких других данных, кроме тех, на которых строилось первоначальное дерево. Даже не сами данные требовались, а количество примеров каждого класса, которое «прошло» через узел. Наиболее очевидным и возможно наиболее эффективным является выбор финального дерева посредством тестирования на тестовой выборке. Естественно, качество тестирования во многом зависит от объема тестовой выборки и «равномерности» данных, которые попали в обучающую и тестовую выборки. Часто можно наблюдать, что последовательность деревьев дает ошибки близкие друг к другу. Случай показан на рис. 2.

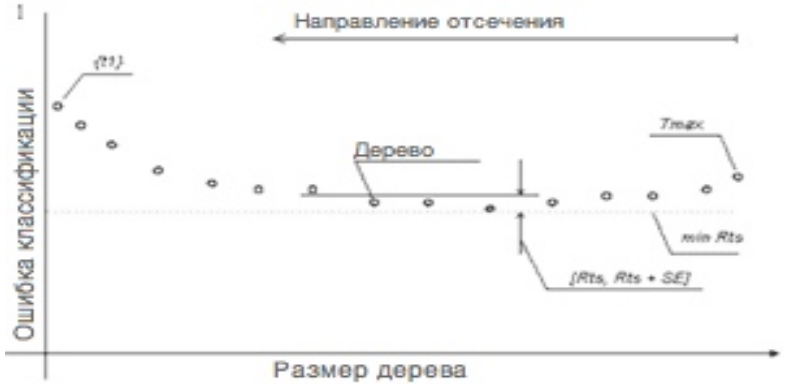

РИС. 2. Зависимость ошибки от размера дерева

Данная длинная плоская последовательность очень чувствительна к данным, которые будут выбраны в качестве тестовой выборки. Чтобы уменьшить данную нестабильность CART использует 1-*SE* правило: выберите минимальное по размеру дерево с Rts в пределах интервала [min R^{ts}, min min R^{ts} + *SE*]. R^{ts} – ошибка классификации дерева. *SE* – стандартная ошибка, являющаяся оценкой реальной ошибки.

$$SE(R^{ts}) = \sqrt{\frac{R^{ts}(1-R^{ts})}{n_{\text{test}}}},$$

где n_{test} – число примеров в тестовой выборке [7, 8].

Перекрестная проверка – самая оригинальная и сложная часть алгоритма CART. Этот путь выбора финального дерева используется, когда набор данных для обучения мал или каждая запись в нем по своему "уникальна" так, что мы не можем выделить выборку для обучения и выборку для тестирования.

В таком случае строим дерево на всех данных, вычисляем α_1, α_2, …, α_k и $T_1 > T_2 > \ldots > T_N$. Обозначим T_k – наименьшее минимизируемое поддерево для $[\alpha_k, \alpha_{k+1}]$. Теперь нужно выбрать дерево из последовательности, но уже использовали все имеющиеся данные. Хитрость в том, что вычисляем ошибку дерева T_k из последовательности косвенным путем.

Процесс построения дерева происходит последовательно. На первом шаге мы получаем регрессионную оценку просто как константу по всему пространству примеров. Константу считаем просто как среднее арифметическое выходной переменной в обучающей выборке. Итак, если мы обозначим все значения выходной переменной как $Y_1, Y_2, \ldots, Y_n$ тогда регрессионная оценка получается:

$$\hat{f}(x) = \left(\frac{1}{n}\sum_{i=1}^{n} Y_i\right) I_R(x),$$

где R – пространство обучающих примеров, n – число примеров, $I_r(x)$ – индикаторная функция пространства – фактически, набор правил, описывающих попадание переменной x в пространство. Мы рассматриваем пространство R как прямоугольник. На втором шаге мы делим пространство на две части. Выбирается некоторая переменная x_i и если переменная числового типа, тогда мы определяем:

$$R_1 = \{x \in R : x \le \alpha\}, R_2 = \{x \in R : x > \alpha\}.$$

Если x_i категориального типа с возможными значениями $A_1, A_2, \ldots, A_q$, тогда выбирается некоторое подмножество I $\{A_1, \ldots, A_n\}$ и мы определяем

$$R_1 = \{x \in R : x \in I\}, R_2 = \{x \in R : x \in \{A_1, A_2, ..., A_n\} \setminus I\}.$$

Регрессионная оценка принимает вид:

$$\hat{f}(x) = \left(\frac{1}{|I_1|} \sum_{I} Y_i \right) I_{R_1}(x) + \left(\frac{1}{I_2} \sum_{I_2} Y_i \right) I_{R_2}(x),$$

где $I_1 = \{i, x_i R_1\}$ и $|I_1|$ – число элементов в I_1.

Каким образом выбирается лучшее разбиение? В качестве оценки здесь служит сумма квадратов разностей:

$$E = \sum (Y_i - \hat{f}(x_i))^2.$$

Выбирается разбиение с минимальной суммой квадратов разностей. Мы продолжаем разбиение до тех пор, пока в каждом подпространстве не останется малое число примеров или сумма квадратов разностей не станет меньше некоторого порога.

Отсечение, выбор финального дерева. Происходят аналогично дереву классификации. Единственное отличие – определение ошибки ответа дерева:

$$\mathrm{R}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{f}(x_i))^2$$

или, иначе говоря, среднеквадратичная ошибка ответа.
Стоимость дерева равна:

$$C_\alpha(\hat{f}) = R(\hat{f}) + \alpha \left| \hat{f} \right|.$$

Остальные операции происходят аналогично дереву классификации.

Выводы. В статье рассмотрен и описан механизм навигации внутри помещений. А при совместном использовании с мобильными устройствами на платформе Android или iOS так же и возможность интерактивного взаимодействия с картой и элементами карты. Также предложена модель анализа данных с помощью алгоритма CART. В качестве основы для кластеризации данных были взяты данные перемещения посетителя между базовыми Wi-Fi модулями, которые включают в себя множество параметров, нетривиальных для простого изучения. Пошагово описано обучение, разбиение и выбор дерева решений. Проведенный анализ позволит вести интерактивную коммуникацию с посетителями через мобильные приложения, делать индивидуальные акции, а так же автоматизацию принятия решений в управлении ТРЦ.

1. *Bill R., Cap C., Kofahl M., Mundt T.* Indoor and Outdoor Positioning in Mobile Environments // Geographical Information Sciences. – 2004. – Vol. 10, N 2. – P. 91 – 98.
2. *Indoor* Atlas. [Электронный ресурс]. [Режим доступа: http://www.indooratlas.com/.

3. *BaseGroup* Labs- Деревья решений – CART математический аппарат. Часть2. [Электронный ресурс]. [Режим доступа: http://www.basegroup.ru/library/analysis/tree/math_cart_part2/]
4. *Kamol Kaemarungsi and Prashant Krishnamurthy*. Modeling of Indoor Positioning Systems Based on Location Fingerprinting // IEEE INFOCOM'2004. – 2004. – P. 7 – 11.
5. *Zeimpekis V., Giaglis G.M., Lek G.* A taxonomy of indoor and outdoor positioning techniques for mobile location services // SIGecom Exchange. – 2003. – P. 19 – 27.
6. *Nicola Lenihan*. A local Optimal User Position System for Indoor Wireless Devices // University of Limerick. – 2004. – P. 33 – 72.
7. *BaseGroup* Labs- Деревья решений – общие принципы работы. [Электронный ресурс]. [Режим доступа: http://www.basegroup.ru/library/analysis/tree/description/]
8. *BaseGroup* Labs- Деревья решений - CART математический аппарат. Часть1. [Электронный ресурс]. [Режим доступа: http://www.basegroup.ru/library/analysis/tree/math_cart_part1/]

Получено 10.07.2014