УДК 004.85:519.6

DOI 10.31651/2076-5886-2020-1-69-77

PACS 02.70.Wz, 07.05.Kf, 07.05.Mh, 07.05.Tp

МОТОРНА Ярослава Сергіївна,
магістрантка спеціальності «Прикладна математика» Черкаського національного університету імені Богдана Хмельницького

КРАСНОШЛИК Наталія Олександрівна,
кандидат технічних наук, доцент,
доцент кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: wlik007@ukr.net
ORCID 0000-0003-4661-6997

ПІСКУН Олександр Варфоломійович,
кандидат технічних наук, доцент,
завідувач кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: piskun@ukr.net
ORCID 0000-0001-5334-6337

РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ АЛГОРИТМУ RANDOM FOREST ДЛЯ РОЗВ'ЯЗУВАННЯ ЗАДАЧ КЛАСИФІКАЦІЇ

У роботі проведено оцінку можливості застосування алгоритму випадкового лісу до розв'язання задач класифікації даних. Подано загальну постановку задачі класифікації даних, описано поняття дерева рішень та наведено основні алгоритми побудови таких дерев. Детально розглянуто алгоритм CART та описано розрахунок основного індексу, на основі якого проводиться побудова дерева для розв'язання задачі класифікації. Розглянуто використання алгоритму CART у ансамблевих методах, на основі яких будується випадковий ліс. Описано основні поняття випадкового лісу, реалізацію випадкового лісу та набір тестових даних, на основі яких проводилась перевірка реалізованого програмного продукту. Результати роботи реалізованого алгоритму випадкового лісу показали дієвість застосованого підходу до розв'язання задач класифікації.

Ключові слова: *машинне навчання, задача класифікації, дерево рішень, Random Forest.*

Вступ

Дерева рішення є одним з досить часто використовуваних способів побудови алгоритмів розпізнавання в сучасних системах обробки інформації у зв'язку з їх численними перевагами, а саме: забезпеченням можливості роботи як з суттєво значними ознаками, так і з категоріальним ознаками, гарну стійкість до аномальних значень спостережень, можливістю наочної інтерпретації одержуваного розв'язувального правила і пояснення досліджуваної закономірності у даних для їх застосування у експертних системах. однією з задач, що розв'язуються за допомогою експертних систем, є задача класифікації, яка полягає у визначенні класу об'єкта, тобто, співставленні аналізованого об'єкта з одним з відомих експертній системі класів.

Можливості застосування окремого виду ансамблю дерев рішень – випадковому лісу – для розв'язування задачі класифікації й присвячена дана стаття.

Метою статті є реалізація і дослідження алгоритму Random Forest та його застосування до розв'язування задач класифікації.

Виклад основного матеріалу

1. Постановка задачі класифікації даних

Задачі класифікації досить часто зустрічаються у найрізноманітніших областях діяльності людини. Прикладами таких задач є проблеми медичного діагностування, оцінювання кредитного ризику, визначення тенденцій на фінансових ринках тощо.

Задача класифікації належить до задач машинного навчання з учителем. Її метою є поділ деякої множини об'єктів на заздалегідь визначену кількість класів. При цьому для підмножини об'єктів, яку називають навчальною вибіркою, клас об'єкта відомий заздалегідь.

Нехай X – множина характеристик об'єктів, Y – множина номерів (або найменувань) класів. Існує невідома "цільова залежність" – відображення, значення якого відомі тільки на об'єктах навчальної вибірки $X^m = \{(x_1, y_1),...,(x_m, y_m)\}$. Для розв'язання задачі потрібно побудувати алгоритм $\alpha : X \to Y$, що апроксимує цільову залежність [1].

2. Дерева рішень і алгоритми їх побудови

Дерева рішень (також можуть називатися деревами класифікації або регресійними деревами) використовуються в області машинного навчання, статистики та аналізу даних у якості прогнозуючих моделей [2].

Структура дерева містить "листя" і "гілки". На ребрах (гілках) дерева рішень записані атрибути, від яких залежить цільова функція, у листі записані значення цільової функції, а в інших вузлах – атрибути, за якими розрізняються об'єкти. Кожен внутрішній вузол відповідає одній з вхідних змінних. Щоб класифікувати новий об'єкт, треба спуститися по дереву до листа і отримати відповідне значення. Приклад дерева рішень наведено на рис. 1.

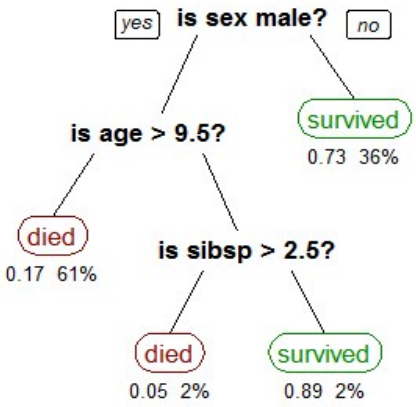

Рис. 1. Приклад дерева рішень

Дерево рішень отримується поділом вхідних наборів змінних на підмножини на основі тестування значень атрибутів. Цей процес повторюється на кожній з отриманих підмножин. Рекурсія завершується тоді, коли підмножина у вузлі має ті ж значення

цільової змінної. Такий процес, що йде зверху вниз, називається індукцією дерев рішень (TDIDT), і є прикладом жадібного алгоритму, що на сьогоднішній день є найбільш поширеною стратегією побудови дерев рішень для даних.

Алгоритми побудови дерева рішень

Загальна схема застосування дерева рішень для класифікації об'єктів тестової вибірки має наступний вигляд.

Листами дерева рішень є класи. Щоб класифікувати об'єкт за допомогою дерева рішень, потрібно послідовно спускатися по дереву, вибираючи напрямок розгалудження, опираючись при цьому на значення предикатів, що містяться у вузлах дерева, шляхом застосування їх до об'єктів, які потрібно класифікувати. Шлях від кореня дерева до листа можна трактувати як пояснення того, чому той чи інший об'єкт віднесений до якого-небудь класу.

Ключовою особливістю алгоритмів побудови дерева рішень є спосіб вибору чергового атрибуту. Існують наступні алгоритми [3]:

– алгоритм ID3, де вибір атрибута відбувається на підставі приросту або на підставі індексу *Gini*;
– алгоритм C4.5 (покращена версія ID3), де вибір атрибута відбувається на підставі нормалізованого приросту інформації;
– алгоритм CART і його модифікації – IndCART, DB-CART.

У загальному вигляді алгоритм побудови дерева рішень подано у лістингу псевдокоду 1.

Псевдокод 1

обчислюємо ентропію вхідної множини s_0

якщо s_0 ==0, то:

1) всі об'єкти вхідного набору, належать до одного класу;
2) зберігаємо цей клас як лист дерева;

якщо s_0 !=0, то:

1) шукаємо предикат, який розбиває дану множину таким чином, щоб зменшилося середнє значення ентропії;
2) знайдений предикат є частиною дерева рішень, зберігаємо його;
3) розбиваємо вхідну множину на підмножини, відповідно до даного предикату;
4) повторюємо процедуру рекурсивно для кожної підмножини.

Щоб знайти предикат для розбиття на підмножини, потрібно для кожного елемента перебирати всі його атрибути. Для кожного елемента генеруємо предикат, який розбиває вхідну множину на дві підмножини. Потім розраховуємо середнє значення ентропії та обчислюємо ΔS. Обираємо предикат з найбільшим значенням ΔS.

У найпростішому випадку можна використовувати предикати, які відносяться тільки до значення якого-небудь атрибута (наприклад «x ≥ 12», або «колір == жовтий»).

Одне з питань, яке виникає у алгоритмі побудови дерева рішень, – це оптимальний розмір кінцевого дерева. Так, невелике дерево може не охопити ту чи іншу важливу інформацію про об'єкти вибірки. При цьому важко визначити, коли алгоритм повинен зупинитися, тому що неможливо спрогнозувати, додавання якого вузла дозволить значно зменшити загальну похибку. Виникає потреба у регулюванні глибини отримуваного дерева рішень.

Регулювання глибини дерева – це техніка, яка дозволяє зменшувати розмір дерева рішень, видаляючи ділянки дерева, які мають невелику вагу. Тобто, загальна стратегія обмеження дерева реалізується шляхом видалення вузлів, якщо вони не несуть додаткової інформації.

Скорочення дерева може здійснюватися зверху вниз або знизу вгору. Зверху вниз – обрізка починається з кореня, а знизу угору – скорочується кількість листів дерева. Один з найпростіших методів регулювання – починаючи з листів, кожен вузол замінюється на найпопулярніший клас. Якщо така заміна не впливає на точність прогнозування, то вона приймається.

Алгоритм побудови дерева рішень CART

Алгоритм CART (Classification and Regression Tree) призначений для розв'язування задач класифікації та регресії. Він розроблений у 1974-1984 роках чотирма професорами статистики: Лео Брейманом (Leo Breiman, Берклі), Джеромом Фрідманом (Jerome H. Friedman, Стенфорд), Чарлзом Стоуном (Charles Stone, Берклі) і Річардом Олшеном (Richard A. Olshen, Стенфорд) [4].

У алгоритмі CART кожен вузол дерева рішень має два нащадки. На кожному кроці побудови дерева правило, сформоване у вузлі, ділить задану множину об'єктів (навчальну вибірку) на дві частини: частина, у якій виконується правило (правий нащадок – right), і частина, у якій правило не виконується (лівий нащадок – left). Для вибору оптимального правила використовується функція оцінки якості розбиття.

Оціночна функція, яка використовується алгоритмом CART, базується на інтуїтивній ідеї зменшення невизначеності у вузлі. Мається на увазі таке розбиття, при якому у вузлі буде якомога більше прикладів одного класу та як можна менше усіх інших. Це поняття близьке до ентропії, але тут використовується інша міра невизначеності, для якої використовується термін "не чистий вузол".

У алгоритмі CART ідея "не чистого вузла" формалізована в індексі *Gini* [5]. Якщо набір даних T містить дані n класів, тоді індекс *Gini* визначається як

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

де параметр p_i – ймовірність класу *i* в *T*.

Якщо набір *T* розбивається на дві частини, T_1 і T_2 з кількістю прикладів у кожній N_1 і N_2 відповідно, то показник якості розбиття буде дорівнювати

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

Найкращим вважається те розбиття, для якого $Gini_{split}(T)$ мінімальне.

Позначимо через N кількість прикладів у вузлі, через L і R – кількість прикладів у лівому і правому нащадках відповідно, l_i і r_i – кількість екземплярів *i*-го класу у лівому/правому нащадку. Тоді якість розбиття оцінюється за наступною формулою:

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

Щоб зменшити обсяг обчислень, формулу для $Gini_{split}$ можна переписати інакше:

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

Так як множення на константу не відіграє ролі при мінімізації, то

$$Gini_{split} = L - \frac{1}{L}\sum_{i=1}^{n} l_i^2 + R - \frac{1}{R}\sum_{i=1}^{n} r_i^2 \to min,$$

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

$$\widetilde{G}_{split} = \frac{1}{L}\sum_{i=1}^{n} l_i^2 + \frac{1}{R}\sum_{i=1}^{n} r_i^2 \to max.$$

Отже, найкращим буде те розбиття, для якого величина $\widetilde{G}_{split}$ максимальна. Рідше в алгоритмі CART використовуються інші критерії розбиття: *Twoing*, *Symmetric Gini* та ін. [4].

Правила розбиття

Вектор предикторних змінних, що подається на вхід дереву, може містити як числові (порядкові), так і категоріальні змінні. У будь-якому випадку, у кожному вузлі розбиття здійснюється лише по одній змінній. Якщо змінна має числовий тип, то у вузлі формується правило виду $x_i \le c$, де c – деякий поріг, який найчастіше вибирається як середнє арифметичне двох сусідніх упорядкованих значень змінної навчальної вибірки. Якщо змінна має категоріальний тип, то у вузлі формується правило $x_i \in V(x_i)$, де $V(x_i)$ – деяка непорожня підмножина множин значень змінної x_i у навчальній вибірці. Отже, для *n* значень числового атрибуту алгоритм порівнює $n-1$ розбиття, а для категоріального – $\left(2^{n-1}-1\right)$. На кожному кроці побудови дерева алгоритм послідовно порівнює всі можливі розбиття для всіх атрибутів і вибирає найкращий атрибут і найкращу розбивку для нього [4].

3. Алгоритм Random Forest

Ансамбль алгоритмів – це певна сукупність алгоритмів, які об'єднуються для розв'язання однієї спільної задачі і утворюють єдине ціле.

Беггінг (Bagging від Bootstrap aggregation) – це один з перших і найпростіших видів ансамблів [6]. Він базується на статистичному методі бутстрепа.

Метод бутстрепа полягає у наступному. Нехай є вибірка *X* розміру *N*. Рівномірно візьмемо з вибірки *N* об'єктів з поверненням. Це означає, що ми будемо *N* разів вибирати довільний об'єкт вибірки (вважаємо, що кожен об'єкт "досягається" з однаковою ймовірністю $1/N$), причому кожного разу ми обираємо з усіх початкових *N* об'єктів. Відзначимо, що внаслідок наявності повернень серед них можуть бути повтори. Позначимо нову вибірку через X_1. Повторюючи процедуру *M* разів, згенеруємо *M* підвибірок $X_1, \ldots, X_M$.

Для побудови ансамблю алгоритмів на основі беггінгу за допомогою бутстрепа генерують вибірки, на кожній з яких навчають свій класифікатор $a_i(x)$. Результуючий класифікатор буде усереднювати відповіді усіх алгоритмів (у разі класифікації це відповідає голосуванню):

$$a(x) = \frac{1}{M}\sum_{i=1}^{M} a_i(x).$$

Беггінг дозволяє знизити дисперсію класифікатора і запобігає перенавчанню. Ефективність беггінга досягається завдяки тому, що базові алгоритми, які навчались на різних підвибірках, отримуються досить різними, і їх помилки взаємно компенсуються при голосуванні. Крім того, об'єкти-викиди можуть не потрапити до деяких навчальних підвибірок.

Random Forest (випадковий ліс) – один з найбільш ефективних алгоритмів машинного навчання, запропонований Лео Брейманом [7]. Він являє собою множину дерев рішень. У задачі регресії їх відповіді усереднюють, а у задачі класифікації приймається рішення голосуванням за більшістю.

Всі дерева ансамблю будуються незалежно один від одного за наступною процедурою:

– згенерувати випадкову підвибірку розміром n з навчальної вибірки;
– побудувати дерево рішень, причому в ході створення чергового вузла дерева розглядають не всі ознаки, а лише m випадково обраних ознак, на основі яких буде проводиться розбиття;
– дерево будується до повного вичерпання об'єктів підвибірки і не піддається процедурі відсікання гілок.

Алгоритм Random Forest може бути використаний у задачі оцінки важливості ознак. Для цього необхідно навчити алгоритм на вибірці і під час побудови моделі для кожного елемента навчальної вибірки порахувати out-of-bag-помилку. Потім для кожного об'єкта така помилка усереднюється по всьому випадковому лісі. Щоб оцінити важливість ознаки, його значення перемішуються для усіх об'єктів навчальної вибірки і out-of-bag-помилка обчислюється знову. Важливість ознаки оцінюється шляхом усереднення різниці показників out-of-bag-помилок по всіх деревах до і після перемішування значень. При цьому значення таких помилок нормалізуються на стандартне відхилення.

4. Реалізація і дослідження алгоритму Random Forest

Будемо розглядати задачу класифікації. Це формалізована задача, в якій задана множина об'єктів, розділених деяким чином на класи. Задана кінцева множина об'єктів, для яких відомо, до яких класів вони відносяться. Ця множина називається вибіркою. Класова належність інших об'єктів не відома. Потрібно побудувати алгоритм, здатний класифікувати довільний об'єкт з початкової множини. Класифікувати об'єкт – означає вказати номер (або назву) класу, до якого відноситься даний об'єкт [1].

Алгоритм Random Forest був реалізований у середовищі Octave. Реалізацію дерева рішень здійснювали за алгоритмом CART [8]. Реалізовані програми дозволяють розв'язувати як задачу бінарної (двокласової класифікації), так і багатокласової класифікації.

Для дослідження алгоритму Random Forest розглядали наступні набори даних [9]:

– бінарна класифікація:
  - Titanic (класифікація пасажирів);
  - Seeds (класифікація ядра пшениці);
  - Telecom_churn (класифікація відтоку клієнтів телеком оператора);
  - CreditScoring (класифікація видачі кредитів);
– багатокласова класифікація:
  - Iris (класифікація ірисів);
  - Wine Quality (класифікація винних виробів).

Наведемо опис розглянутих наборів даних.

Набір даних Titanic (класифікація пасажирів) містить дані про справжніх пасажирів на кораблі “Титанік”. Кожний рядок представляє одну особу. Колонки описують різні атрибути про особу: вік (age), клас (pclass), номер квитка (ticket), вартість проїзду, яку заплатили (fare). Пасижири класифікуються за двома класами: 1 – вижили, 2 – не вижили.

Набір даних Seeds (класифікація ядра пшениці) містить дані про ядра, які належать до двох різних сортів пшениці: “Елегія”, “Олеся”. Для побудови даних були виміряні шість геометричних параметрів ядра пшениці: периметр, компактність, довжина ядра, ширина ядра, коефіцієнт несиметрії, довжина канавки ядра.

Набір даних Telecom_churn містить дані про клієнтів телеком оператора, які належать до двох різних класів: втрата клієнта (відтік), клієнт залишився. Для побудови класифікатора застосовувались такі ознаки: загальна кількість дзвінків вдень, загальна кількість дзвінків вночі, загальна кількість дзвінків увечері, загальна кількість міжнародних дзвінків, загальна сума оплати за міжнародні дзвінки, число звернень у сервісний центр. Клієнти класифікуються за двома класами: 1 – втрата клієнта, 2 – клієнт залишився

CreditScoring (класифікація видачі кредитів). Наявні дані про клієнтів банку, які класифікуються на два класи: банк видасть кредит, банк не видасть кредит. Для побудови даних були взяті наступні ознаки: вік клієнта, термін занятості, кількість дітей, заробітня плата, чи має клієнт нерухомість, скільки активних кредитів. Клієнти класифікуються за двома класами: 1 – видати кредит, 2 – не видати.

Iris (класифікація ірисів). Набір містить дані про види ірисів. Він включає в себе три види ірисів, по 50 кожного виду, а також деякі властивості кожної квітки: довжина пелюстки, ширина пелюстки, довжина чашолистика, ширина чашолистика. Іриси класифікуються за трьома класами: 1 – Iris-setosa, 2 – Iris-versicolor, 3 – Iris-virginica.

Wine Quality (класифікація винних виробів). Наявні дані про різноманітні сорти білих вин, які належать до трьох різних класифікацій якості: хороша, середня та найгірша. Для побудови даних були виміряні властивості вина: фіксована кислотність, залишковий цукор, pH, сульфати, летюча кислотність, вміст алкоголю. Сорти класифікуються за трьома класами: 1 – хороша якість, 2 – середня якість, 3 – найгірша якість

Якість роботи алгоритму оцінювалась за часткою правильних відповідей на відкладеній вибірці. Для набору даних Titanic отримано правильних відповідей 89.04%, для Seeds – 94.11%, для Telecom_churn – 91.66%, для CreditScoring – 90.47%, для Iris – 95.94%, для Wine Quality – 93.65%. На рис. 2 наведено приклади виклику алгоритму Random Forest для наборів даних Titanic і CreditScoring.

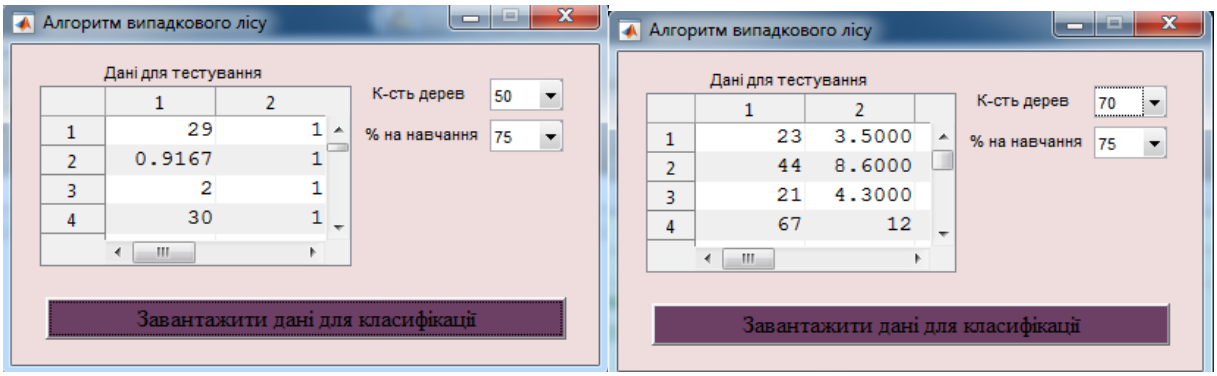

Рис. 2. Приклад застосування алгоритму Random Forest до класифікації даних

Як видно з поданого вище, отримані результати показують дієвість застосування ансамблю дерева рішень до задач класифікації.

Висновки

У результаті проведеної роботи отримано підтвердження дієвості застосування ансамблевих методів на основі дерев рішень, які будуються за допомогою алгоритму CART. Тестування показало точність результатів, більшу за 90%, що свідчить про високу якість роботи алгоритму. Цілком зрозуміло, що така якість отримується завдяки використанню кількох різних дерев рішень, результати роботи яких узагальнюються при виведенні остаточної оцінки.

Список використаної літератури:

1. Классификация [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Классификация.
2. Chelliah, I. Understanding Decision Trees in Machine Learning [Електронний ресурс]. – Режим доступу: https://medium.com/better-programming/understanding-decision-trees-in-machine-learning-86d750e0a38f
3. Деревья решений: общие принципы [Електронний ресурс]. – Режим доступу: https://loginom.ru/blog/decision-tree-p1
4. Breiman, L. Classification and Regression Trees / L. Breiman, J. H. Friedman, R. A. Olshen, C. T. Stone. – Chapman and Hall/CRC, 1984. – 368 p.
5. Андреев, И. Деревья решений – CART математический аппарат. Часть 1. [Електронний ресурс]. – https://basegroup.ru/community/articles/math-cart-part1
6. Rocca, J. Ensemble methods: bagging, boosting and stacking [Електронний ресурс]. – https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205
7. Breiman, L. Random Forests / Leo Breiman // Machine Learning. – 2001. – Vol. 45. - № 1. – P. 5 – 32.
8. Patel, F. Decision Tree the CART Algorithm. [Електронний ресурс]. – https://medium.com/analytics-vidhya/decision-tree-the-cart-algorithm-28c481d28813
9. Datasets [Електронний ресурс]. – https://www.kaggle.com/datasets

References:

8. Klassifikatsiya [Classification]. (2011). Retrieved from http://www.machinelearning.ru/wiki/index.php?title= Classification. [in Russian]
9. Chelliah, Indhumathy. (2020). Understanding Decision Trees in Machine Learning. Retrieved from: https://medium.com/better-programming/understanding-decision-trees-in-machine-learning-86d750e0a38f
10. Derevia resheniy: obshchiye printsipy [Decision trees: general principles]. (2019). Retrieved from https://loginom.ru/blog/decision-tree-p1. [in Russian]
11. Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. T. (1984). Classification and Regression Trees. Chapman and Hall/CRC.
12. Andreev, I. (2005). Derevia resheniy – CART matematicheskiy apparat. Chast 1. [Decision trees – CART mathematical apparatus. Part 1]. Retrieved from https://basegroup.ru/community/articles/math-cart-part1. [in Russian]
13. Rocca, J. (2019). Ensemble methods: bagging, boosting and stacking. Retrieved from https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205
14. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32.
15. Patel F. (2020). Decision Tree the CART Algorithm. Retrieved from https://medium.com/analytics-vidhya/decision-tree-the-cart-algorithm-28c481d28813
16. Datasets. (2021). Retrieved from https://www.kaggle.com/datasets

MOTORNA Yaroslava,
student, The Bohdan Khmelnytsky National University of Cherkasy, Ukraine
KRASNOSHLYK Natalia,
Candidate of Technical Sciences, Associate Professor, The Bohdan Khmelnytsky National University of Cherkasy, Ukraine
PISKUN Oleksandr,
Candidate of Technical Sciences, Associate Professor, Head of Department of Applied Mathematics and Informatics, Bohdan Khmelnytsky National University of Cherkasy, Ukraine

IMPLEMENTATION AND RESEARCH OF THE RANDOM FOREST ALGORITHM TO SOLVE CLASSIFICATION PROBLEMS

Summary. Introduction. *Solution trees are one of the most commonly used ways to build recognition algorithms in modern information processing systems due to their many advantages, in particular, the possibility of their application in expert systems. One of the problems solved by expert systems is the classification problem, which is to determine the class of an object from a set of classes known to the expert system. This article is devoted to the possibility of using one type of decision tree ensemble - a random forest - to solve the classification problem.*

The purpose *of this paper is to implement and study the Random Forest algorithm and its application to solving classification problems.*

Results. *The problem of classification by means of a random forest is considered in the work. This is a formalized task that specifies a set of objects that are divided into classes in some way. A finite set of objects is specified for which classes they are known. The Random Forest algorithm is implemented in the Octave environment. The decision tree is implemented according to the CART algorithm. Implemented programs allow us to solve both the problem of binary (two-class classification) and multi-class classification.*

To study the Random Forest algorithm, the following data sets were considered: passenger classification, seed classification, mobile operator customer outflow classification, credit issuance classification, iris classification, wine classification. The quality of the algorithm was evaluated by the share of correct answers in the deferred sample: 89.04% received correct answers for the Titanic data set, 94.11% for Seeds, 91.66% for Telecom_churn, 90.47% for CreditScoring, 95.94% for Iris, and 93.65% for Wine Quality.

Conclusion. *As a result of this work, the effectiveness of the application of ensemble methods based on decision trees, which are built using the CART algorithm, was confirmed. Testing showed the accuracy of the results, more than 90%, which indicates the high quality of the algorithm. It is clear that this quality is obtained through the use of several different decision trees, the results of which are summarized in the derivation of the final assessment.*

Keywords: *machine learning, classification problem, decision tree, Random Forest.*

Одержано редакцією 20.12.2019 р.
Прийнято до публікації 24.02.2020 р.

УДК 378:517

DOI 10.31651/2076-5886-2020-1-77-86

PACS 02.60.-x

ТАРАСЕНКОВА Ніна Анатоліївна,
доктор педагогічних наук, професор,
завідувач кафедри математики та
методики навчання математики,
Черкаський національний університет
імені Богдана Хмельницького
e-mail: ntaras7@ukr.net
ORCID: 0000-0002-6418-6380

СЕРДЮК Зоя Олексіївна,
кандидат педагогічних наук, доцент,
доцент кафедри математики та методики
навчання математики,
Черкаський національний університет
імені Богдана Хмельницького
e-mail: serdyuk_z@ukr.net
ORCID: 0000-0002-9376-4346

ОСОБЛИВОСТІ ВИКЛАДАННЯ КУРСУ МАТЕМАТИЧНОГО АНАЛІЗУ ДЛЯ ФАХІВЦІВ З АНАЛІЗУ ДАНИХ