Лабораторна робота №2

Інтелектуальний аналіз даних за допомогою програмного пакета WEKA: Класифікація і кластеризація

Вступ

В попередній лабораторній роботі ми отримали загальне уявлення про основні принципи інтелектуального аналізу даних і познайомилися з програмним пакетом **Waikato Environment for Knowledge Analysis (WEKA)**, що дозволяє аналізувати значні обсяги інформації та визначати існуючі в них тренди і залежності. Крім того, ми ознайомилися з одним з методів інтелектуального аналізу даних, а саме **регресійним аналізом,** за допомогою якого можна спрогнозувати числове значення залежної величини на підставі набору показників незалежних величин.

Метод регресійного аналізу - найбільш простий і найменш ефективний спосіб інтелектуального аналізу даних, проте, побудова моделі регресійного аналізу є дуже зручним прикладом для початку роботи з **WEKA**, а сам метод наочно демонструє можливості інтелектуального аналізу з витягання придатних до подальшого використання відомостей з великих наборів «сирих» даних.

У цій лабораторній роботі ми ознайомимося з двома іншими методами інтелектуального аналізу даних. За своєю структурою ці методи значно складніші, ніж регресійний аналіз, проте їх використання дозволяє досягти значних результатів. Якщо при використанні регресійного аналізу ви просто отримуєте якесь числове значення, залежне від вхідних параметрів, то застосування двох інших методів дозволить вам по-новому інтерпретувати отримані дані. Завдання інтелектуального аналізу - визначити і побудувати модель, відповідну нашим даним. Наприклад, ми можемо зібрати велику колекцію даних про клієнтів, але якщо у нас немає правильної моделі для інтерпретації цих даних, то вся зібрана нами інформація нічого не варта.

Поглянемо на це питання по-іншому: якби всі використовували тільки регресійний аналіз, то як той чи інший інтернет-магазин зміг би сказати нам: «Покупці, які купили X, також купили і Y»? Числової функції, яка вміла б визначати подібну інформацію, просто не існує. Тому ознайомимося з двома новими методами аналізу і розберемося, яку інформацію ми можемо отримати з їх допомогою.

Тум ми будемо досить часто зустрічати посилання на метод найближчих сусідів без пояснень суті цього методу. Ми більш детально розглянемо його в майбутньому, а поки будемо використовувати його для порівняння при описі методів, розглянутих у цій лабораторній роботі. Порівняльний аналіз цих методів допоможе нам краще зрозуміти їхні можливості.

Класифікація, кластеризація і метод найближчих сусідів

Перш ніж ми перейдемо до обговорення конкретних деталей кожного з розглянутих методів та їх реалізації в WEKA, давайте розберемося, для чого використовується цей метод, якими даними він оперує і яку інформацію можна отримати з його допомогою . При обговоренні можливостей кожного з методів ми будемо використовувати нашу модель регресійного аналізу для порівняння можливості нових методів з результатами використання вже знайомого нам підходу.

Для порівняльного аналізу методів ми скористаємося реальними прикладами, які використовують дані дилерського центру BMW і його способи підвищення продажів. Дилерський центр зберігає всі дані про своїх клієнтів, включаючи тих, хто купив BMW, цікавився чи просто зайшов в демонстраційній зал BMW. Мета дилерського центру - підвищити рівень продажів, спираючись на результати інтелектуального аналізу отриманих даних.

Регресійний аналіз

Запитання: «За якою ціною ми повинні продавати новий BMW M5?» Регресійна модель в змозі відповісти на питання про конкретне числове значення. **Модель регресійного аналізу** в якості **незалежних параметрів** буде враховувати дані проданих автомобілів BMW і параметри моделі M5, а в якості **залежного параметра** - вартість автомобілів, проданих дилерським центром. Таким чином, підставивши дані нової моделі BMW в отриману залежність, дилерський центр зможе визначити ціну автомобіля.

Приклад:
Selling Price = $25,000 + ($2900 * Liters in Engine) + ($9000 * isSedan) + ($11,000 * isConvertible) + ($100 * inches of car) + ($22,000 * isM).

Класифікація

Запитання: «Яка ймовірність того, що хтось купить останню модель BMW M5?» Створення класифікаційної моделі (дерева рішень) дозволяє оцінити ймовірність того, що якась людина купить автомобіль моделі M5. В якості вузлів дерева можуть використовуватися такі показники, як вік, рівень доходу, кількість вже наявних машин, сімейний стан, діти, наявність власного будинку або оренда будинку. Параметри конкретної людини будуть використовуватися для проходження по дереву з метою визначення ймовірності покупки M5.

Кластеризація

Запитання: «Який вік покупців, що віддають перевагу сріблястий BMW M5?». Необхідні дані можуть бути отримані при аналізі віку покупців і кольору обраних ними машин. Далі, на підставі отриманих даних, можна зробити висновок, що певна вікова група (наприклад, люди у віці від 22 до 30 років) віддає перевагу певному кольру BMW M5 (75% купують автомобілі синього кольору). Точно так, можна визначити, що інша вікова група (наприклад, люди у віці від 55 до 62 років) віддає перевагу сріблястому BMW (65% купують сріблясті автомобілі, а 20% купують сірі). Аналіз даних дозволить створити кластери за віковими групами і кольорами і визначити залежності між даними.

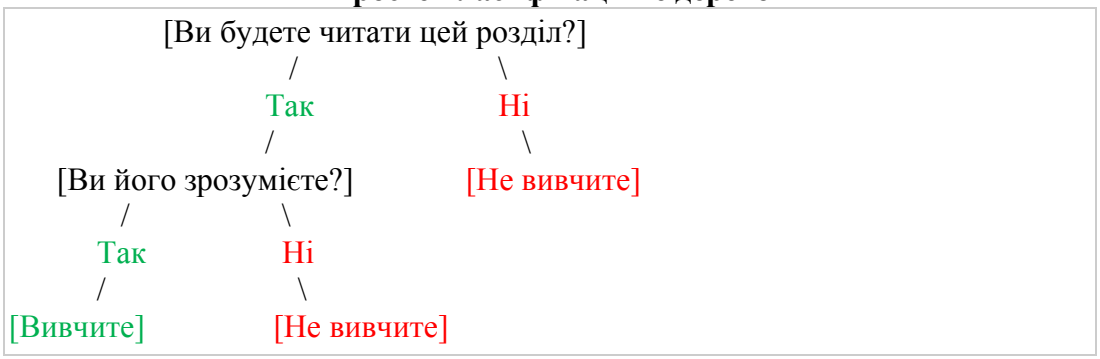
Метод найближчих сусідів

Питання : «Якщо людина купує автомобіль BMW M5, які ще товари вона може придбати?» Як свідчать статистичні дані, якщо людина купує BMW M5, то досить часто вона бере відповідну за кольором барсетку (подібні дослідження називаються «аналіз ринкового кошика»). Використовуючи подібні дані, дилерський центр може розмістити рекламу відповідних товарів або подати на друк оголошення про знижки на барсетки і сумки відповідного кольору або їх безкоштовну пропозицію всім покупцям M5 в якості одного із способів підвищення кількості продажів.

Класифікація

Метод класифікації (також відомий як метод класифікаційних дерев або дерев прийняття рішень) - це алгоритм аналізу даних, який визначає покроковий спосіб прийняття рішення в залежності від значень конкретних параметрів. Дерево цього методу має наступний вигляд: кожен вузол являє собою точку прийняття рішення на підставі вхідних параметрів. Залежно від конкретного значення параметра ви переходите до наступного вузла, від нього - до наступного вузла, і так далі, поки не дійдете до листка, який і дає вам остаточне рішення. Звучить досить заплутано, але насправді метод досить прямолінійний. Давайте звернемося до конкретного прикладу.

Просте класифікаційне дерево



Це просте класифікаційне дерево визначає відповідь на питання «Чи вивчите ви принцип побудови класифікаційного дерева?» У кожному вузлі ви відповідаєте на відповідне питання і переходьте за відповідній гілці до наступного вузла, до тих пір, поки не дійдете до листа з остаточною відповіддю «так» чи «ні». Ця модель застосовна до будь-яких сутностей, і ви зможете відповісти, чи в змозі ці сутності вивчити класифікаційні дерева, за допомогою двох простих питань. У цьому і полягає основна перевага класифікаційних дерев - вони не вимагають надмірної кількості інформації для створення досить точного і інформативного дерева рішень.

Модель класифікаційного дерева використовує підхід, аналогічний тому, який ми застосовували при створенні моделі регресійного аналізу, а саме створення моделі на основі навчальної послідовності (training set). Цей підхід **використовує відомі дані** для аналізу впливу **вхідних значень** на **результат** і будує модель на основі отриманих залежностей. **Таким чином, коли у нас є новий набір даних з невідомим результатом, ми підставляємо наші дані в модель і отримуємо очікуваний висновок.**

Поки що новий метод працює так само, як регресійна модель. Проте, існують і відмінності. Метод класифікаційного аналізу використовує покращений підхід, суть якого полягає в тому, що набір відомих даних ділиться на дві частини. **Перша частина** (60-80% даних) використовується в якості навчального набору для створення моделі. Після цього дані, що залишилися проганяються через цю модель, і змодельовані результати порівнюються з реальними. Такий підхід дозволяє оцінити точність аналітичної моделі.

Для чого потрібен додатковий крок перевірки моделі? Він дозволяє уникнути *зайвої підгонки* моделі під зібрані дані. Якщо ми будемо використовувати *занадто багато* даних для розробки моделі, то отримана модель буде ідеально відповідати зібраними даними і тільки їм. Наше завдання - створити модель для прогнозування невідомих нам даних, а не для прогнозування даних, які і так вже відомі. Для перевірки цього і використовується тестовий набір. З його допомогою ми визначаємо, що точність моделі на тестовому наборі не зменшується. Таким чином, ми гарантуємо, що наша модель зможе з досить високою ймовірністю визначити невідомі значення. З допомогою **WEKA** ми розглянемо, як це працює на конкретному прикладі.

Питання надмірної кількості даних приводить нас до обговорення ще одного важливого принципу побудови класифікаційних дерев - принципу відсікання гілок. *Відсікання гілок*, як випливає з назви, має на увазі видалення гілок з класифікаційного дерева. Але навіщо видаляти інформацію з дерева рішень? Знову ж, для того, щоб уникнути зайвої підгонки дерева під відомі дані. У міру зростання даних зростає і кількість атрибутів, так що наше дерево може стати надмірно складним і розгалуженим. Теоретично, можна створити дерево з кількістю листків leaves = (rows * attributes) , проте наскільки корисне таке дерево? Воно навряд чи зможе допомогти нам визначити невідомий результат, так як воно всього лише визначає вже відомі дані. Необхідно досягти певного балансу між точністю аналізу і простотою моделі. Нам

потрібна аналітична модель з мінімальною кількістю вузлів і листків, яка, тим не менш, відрізнялася б високою точністю. Тут, як ми бачимо, необхідний певний компроміс.

І останнє питання, на якому ми хотіли б зупинитися, перш ніж приступити до створення конкретних моделей в WEKA, це **проблема помилкового розпізнавання** (false positive і false negative). Основний сенс проблеми помилкового розпізнавання полягає в тому, що модель розглядає який-небудь атрибут який має вплив на кінцевий результат, в той час як насправді цей атрибут ніякого впливу на результат не надає. І навпаки, істотний атрибут трактується моделлю як несуттєвий.

Хибне розпізнавання свідчить про помилковість моделі і невірну класифікацію даних. Безумовно, певна похибка в класифікації припустима, і завдання розробника моделі - визначити допустимий відсоток помилок. Наприклад, при розробці моделі оцінки кардіомоніторів для лікарень від вас будуть потрібні виключно низькі показники помилковості результатів. Навпаки, якщо ви розробляєте навчальну модель для ілюстрації статті про інтелектуальний аналізі даних, ви можете дозволити собі достатньо високий рівень похибки моделі. Розвиваючи цю тему, корисно визначити прийнятне процентне відношення хибно-негативного розпізнавання до хибно-позитивного. Найбільш очевидний приклад - модель визначення спаму. Хибно-позитивне розпізнавання (потрібний лист позначається як спам), швидше за все, буде мати більше негативних наслідків, ніж хибно-негативні (лист-спам потрапить в розряд потрібних листів). У цьому випадку може вважатися прийнятним співвідношення хибно-негативних розпізнавань до хибно-позитивних як 100:1.

Тепер візьмемо реальні дані і пропустимо їх через аналітичний механізм **WEKA**.

Набір даних для WEKA

Набір даних, який ми будемо використовувати для прикладу класифікаційного аналізу, містить інформацію, зібрану дилерським центром BMW. Центр починає рекламну кампанію, пропонуючи розширену дворічну гарантію своїм постійним клієнтам. Подібні кампанії вже проводилися, так що дилерський центр має 4500 показники щодо попередніх продажів з розширеною гарантією. Цей набір даних має такі атрибути:

- Розподіл за доходами
  [0=$0-$30k,
  1=$31k-$40k,
  2=$41k-$60k,
  3=$61k-$75k,
  4=$76k-$100k,
  5=$101k-$150k,
  6=$151k-$500k,
  7=$501k+]
- Рік / місяць покупки першого автомобіля BMW
- Рік / місяць покупки останнього автомобіля BMW
- Чи скористався клієнт розширеною гарантією

Файл даних у форматі Attribute-Relation File Format (ARFF) буде виглядати наступним чином:

```
@attribute IncomeBracket {0,1,2,3,4,5,6,7}
@attribute FirstPurchase numeric
@attribute LastPurchase numeric
@attribute responded {1,0}

@data

4,200210,200601,0
5,200301,200601,1
...
```

Класифікація в WEKA

Завантажте файл **bmw-training.arff** в програмний пакет **WEKA**, використовуючи ті ж кроки, які ми виконали для завантаження даних у випадку регресійного аналізу.

Зауваження: *в пропонованому файлі містяться 3000 з наявних 4500 записів. Ми розділили набір даних так, щоб частина їх використовувалася для створення моделі, а частина - для перевірки її точності, щоб переконатися, що модель не є підігнаною під конкретний набір даних.*

Після завантаження даних вікно WEKA має виглядати так, як показано на рис. 1.

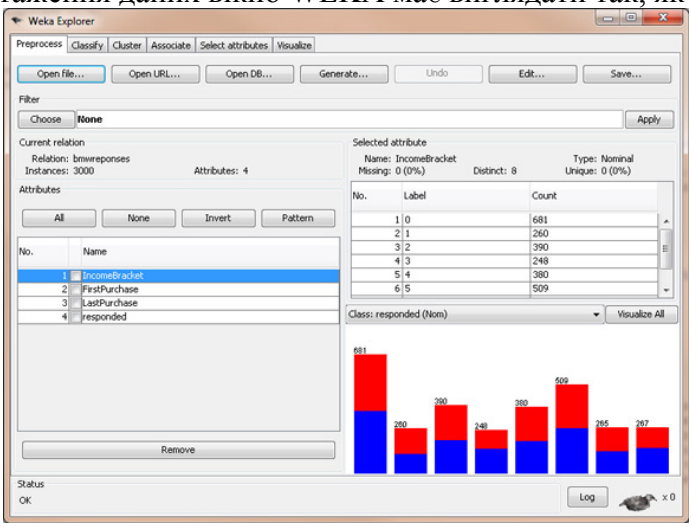


Рис. 1. Дані дилерського центру BMW для класифікації в WEKA

Аналогічно тому, як ми вибирали тип моделі для регресійного аналізу в попередній лабораторній роботі, тепер нам слід вибрати модель для класифікації: відкрийте закладку **Classify** , виберіть опцію **trees**, а потім опцію **J48**.

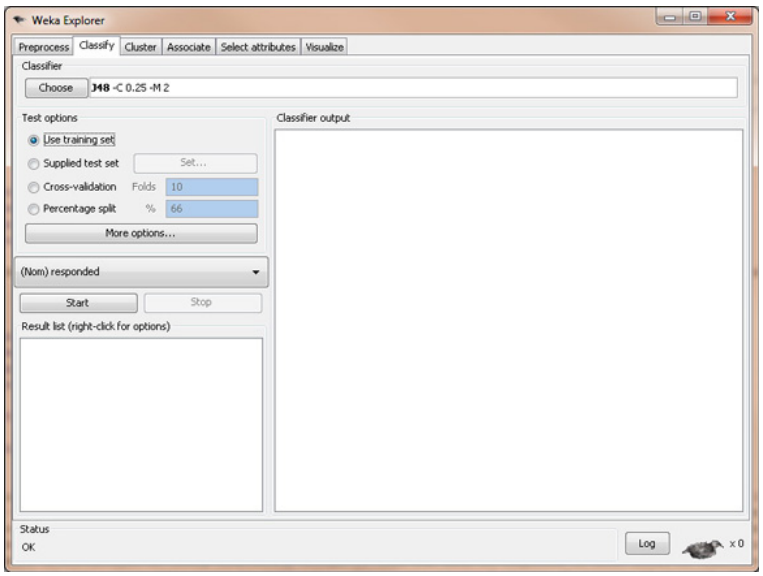


Рис. 2. Алгоритм класифікації даних BMW

Тепер ми готові приступити до створення конкретної моделі аналізу засобами пакета WEKA. Переконайтеся, що обрана опція **Use training set** , щоб пакет WEKA при створенні моделі використовував саме ті дані, які ми тільки що завантажили у вигляді файлу. Натисніть кнопку **Start** і надайте WEKA можливість попрацювати з нашими даними. Результуюча модель повинна виглядати так, як показано нижче:

Результат роботи класифікаційної моделі WEKA

```
Number of Leaves  :         28

Size of the tree :    43


Time taken to build model: 0.18 seconds

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances        1774               59.1333 %
Incorrectly Classified Instances      1226               40.8667 %
Kappa statistic                          0.1807
Mean absolute error                      0.4773
Root mean squared error                  0.4885
Relative absolute error                 95.4768 %
Root relative squared error             97.7122 %
Total Number of Instances             3000

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.662    0.481      0.587     0.662     0.622      0.616    1
                 0.519    0.338      0.597     0.519     0.555      0.616    0
Weighted Avg.    0.591    0.411      0.592     0.591     0.589      0.616

=== Confusion Matrix ===
```

```
   a    b   <-- classified as
1009  516 |    a = 1
 710   765 |    b = 0
```

Розберемося що означають всі ці числа. Як нам зрозуміти, наскільки хороша отримана модель? І де, власне, це так зване «дерево», яке мало вийти в результаті? Цілком закономірні питання. Давайте відповімо на кожне з них по черзі:

- **Що означають всі ці числа?** Найбільш суттєві дані - це показники класифікації "**Correctly Classified Instances**" (59.1%) і "**Incorrectly Classified Instances**" (40.9%). Крім того, слід звернути увагу на число в першому рядку стовпця **ROC Area** (0.616). Трохи пізніше ми докладніше обговоримо ці значення, поки ж просто запам'ятайте їх. Нарешті, таблиця **Confusion Matrix** показує кількість **хибно-позитивних** (516) і **хибно-негативних** (710) розпізнавань.
- **Як зрозуміти, наскільки хороша отримана модель?** Оскільки показник точності нашої моделі - 59,1%, то в первісному розгляді її не можна назвати досить хорошою.
- **Де це так зване дерево?** Ви зможете побачити дерево, якщо клацнете правою кнопкою миші в панелі результуючої моделі. У контекстному меню виберіть опцію **Visualize tree**. На екрані відобразиться візуальне уявлення класифікаційного дерева нашої моделі (рис. 3), проте в даному випадку картинка мало чим нам допоможе. Ще один спосіб побачити дерево моделі - прокрутити вгору висновок у вікні **Classifier Output**, там ви знайдете текстовий опис дерева з вузлами і листками.

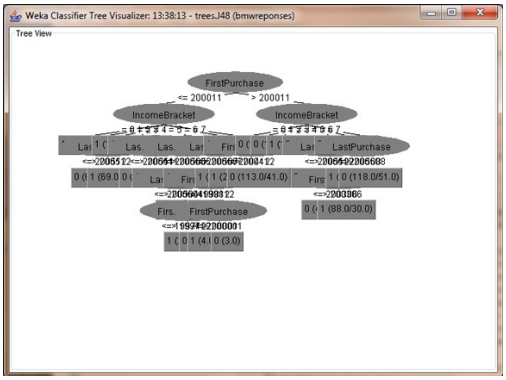


Рис. 3. Візуальне подання дерева класифікації

Залишився останній етап перевірки класифікаційного дерева: нам треба пропустити набір даних, що залишився через отриману модель і перевірити, наскільки результати класифікації будуть відрізнятися від реальних даних. Для цього в секції **Test options** виберіть опцію **Supplied test set** і натисніть на кнопку **Set**. Вкажіть файл **bmw-test.arff**, що містить решту 1500 даних, які не були включені в навчальний набір. При натисканні на кнопку **Start** WEKA пропустить тестові дані через модель і покаже результат роботи моделі. Давайте натиснемо на **Start** і перевіримо, що у нас вийшло.

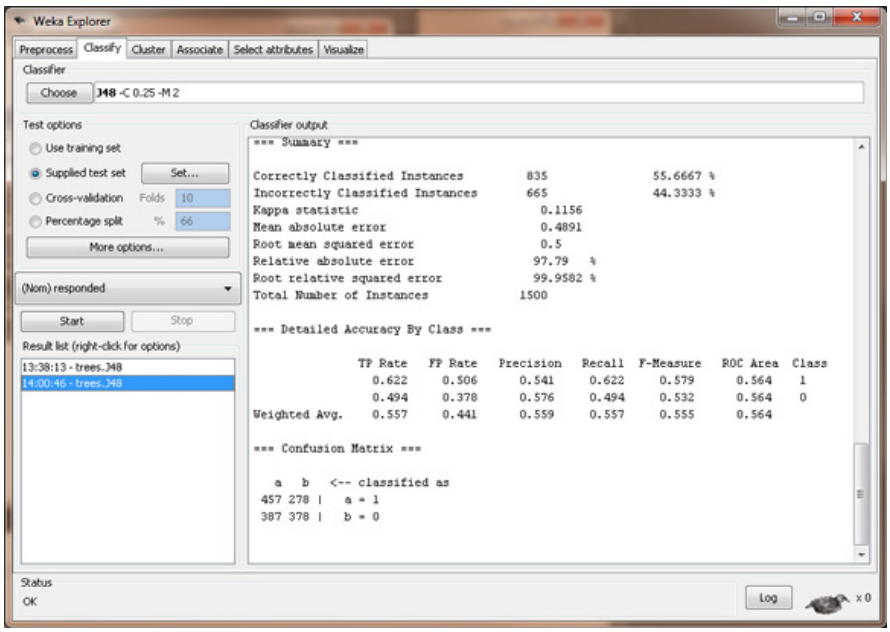


Рис 4. Перевірка класифікаційного дерева

Порівнюючи показник **Correctly Classified Instances** для тестового набору (55,7%) з цим же показником для навчального набору (59,1%), ми бачимо, що точність моделі для двох різних наборів даних приблизно однакова. Це означає, що нові дані, які будуть використовуватися в цій моделі в майбутньому, не знизять точність її роботи.

Однак, оскільки власне точність моделі досить низька (всього лише 60% даних класифіковано вірно), ми маємо повне право зупинитися і сказати: «На жаль, ця модель взагалі нікуди не годиться. Вона працює з точністю трохи вище 50%, з таким же успіхом ми можемо просто намагатися вгадати значення випадковим чином». І цей вислів буде цілком справедливим. Це приводить нас до розуміння дуже важливого факту: існують випадки, коли використання алгоритмів інтелектуального аналізу даних призводить до створення невдалої аналітичної моделі. Наш приклад - наочнй тому доказ, і саме для цього ми його розглянули.

Ми свідомо проробили всі кроки, необхідні для створення класифікаційного дерева на підставі даних, здавалося б, ідеально відповідних для класифікаційної моделі. Однак, результат, отриманий WEKA, вказує на помилковість наших міркувань. Класифікаційна модель не підходить для аналізу наявних у нас даних. Створена нами модель не дасть нам ніяких корисних відомостей, а її використання може призвести до прийняття неправильних рішень і втрати грошей.

Чи означає це, що наші дані взагалі не підлягають ніякому аналізу? Відповідь демонструє ще одну важливу особливість інтелектуального аналізу даних: використовуючи метод «найближчих сусідів», який ми детально розглянемо в майбутньому, ми створимо іншу модель на базі цього ж набору даних, з точністю роботи 88%. Отже, завжди необхідно пам'ятати, що **для того, щоб витягти корисну інформацію з великого набору даних, вам слід вибрати відповідну модель.**

Кластеризація

Кластеризація дозволяє розбити дані на групи, кожна з яких має певні ознаки. Метод кластерного аналізу використовується в тих випадках, коли необхідно виділити деякі правила, взаємозв'язки або тенденції у великих наборах даних. Залежно від потреб, ви можете виділити кілька різних груп даних. Одна з явних переваг кластеризації в порівнянні з класифікацією полягає в тому, що для розбиття множини на групи може використовуватися будь-який атрибут (метод класифікації використовує тільки певну підмножину атрибутів). В якості основного недоліку методу кластеризації слід зазначити

той факт, що укладач моделі повинен заздалегідь вирішити, на скільки груп слід розбити дані. Для людини, яка не має жодного уявлення про конкретний набір даних, прийняти таке рішення досить важко. Нам варто створити три групи або п'ять груп? А може, нам потрібно визначити десять груп? Може знадобитися кілька повторювань проб і помилок, для того щоб визначити оптимальну кількість кластерів.

Тим не менше, для середньостатистичного користувача кластеризація може виявитися найбільш корисним методом інтелектуального аналізу даних. Цей метод дозволить вам швидко розбити ваші дані на окремі групи і зробити конкретні висновки і припущення щодо кожної групи. Математичні методи, що реалізують кластерний аналіз, досить складні і заплутані, так що в разі кластеризації ми будемо цілком покладатися на обчислювальні можливості **WEKA**.

Короткий огляд математичних методів

Далі пропонується короткий загальний опис математичних методів і алгоритмів, використовуваних методом кластеризації.

1. Кожен атрибут має бути приведений до нормального вигляду. Для цього кожен показник ділиться на різницю між найбільшим і найменшим значенням, які приймає розглянутий атрибут на конкретному наборі даних. Наприклад, якщо розглянутий атрибут - вік, і його найбільше значення - 72, а найменше - 16, то значенням 32 буде відповідати нормалізована величина 0.5714.
2. Виходячи з бажаної кількості кластерів, випадковим чином вибирається така ж кількість рядків даних. Ці рядки будуть використовуватися в якості початкових центрів мас кластерів.
3. Для кожного рядка даних визначається відстань від цього рядка до центру мас кластера (випадковим чином вибраного рядка даних) за допомогою методу найменших квадратів.
4. Кожен рядок набору даних входить у той кластер, відстань до центру мас якого виявилося найменшим.
5. У кожному кластері визначається новий центр мас як набір середніх значень по стовпцях на безлічі елементів цього кластеру.
6. Визначається відстань від кожного елемента даних до нового центру мас. Якщо при цьому розподіл елементів по кластерах не змінюється, розбиття даних на кластери закінчено, і всі групи даних визначені. Якщо склад кластерів змінюється, слід повернутися до п.3 і повторювати цей процес до тих пір, поки розбиття на кластери не стане незмінним.

Для того щоб розбити набір даних з 10 рядків на три кластери за допомогою електронних таблиць, потрібно приблизно півгодини напруженої роботи. Можна уявити, скільки часу займе розбиття на 10 кластерів набору з 100000 записів. Але комп'ютер здатний виконати подібні розрахунки за кілька секунд.

Набір даних для WEKA

Для побудови моделі кластеризації ми знову скористаємося даними дилерського центру BMW. Співробітники центру зібрали дані про усіх відвідувачів демонстраційного залу, машинах, які їх зацікавили, і про те, наскільки часто відвідувачі демонстраційного залу в підсумку купували автомобіль, який їм сподобався. Тепер дилерському центру треба проаналізувати ці дані для того, щоб виділити різні групи відвідувачів і зрозуміти, чи не можна визначити будь-які тенденції в їх поведінці. У нашому прикладі використовується 100 записів, і кожен стовпець описує певний етап, який, як правило, проходить покупець у процесі вибору та придбання автомобіля. Відповідно, значення 1 у стовпці говорить про те, що відвідувач пройшов конкретний етап, а 0 - що відвідувач цей етап не пройшов. Файл з даними у форматі ARFF приведений нижче.

Дані для кластерного аналізу засобами WEKA

```
@Attribute Dealership numeric
```

```
@Attribute Showroom numeric
@Attribute ComputerSearch numeric
@Attribute M5 numeric
@Attribute 3Series numeric
@Attribute Z4 numeric
@Attribute Financing numeric
@Attribute Purchase numeric

@Data
1,0,0,0,0,0,0,0
1,1,1,0,0,0,1,0
...
```

Кластеризація в WEKA

Завантажте файл **bmw-browsers.arff** в WEKA, виконавши ті ж кроки, які ми виконали раніше для відкриття даних у закладці **Preprocess**. Витратьте кілька хвилин на візуальний аналіз даних, зверніть увагу на атрибути даних, розподіл даних по стовпцях, і так далі. Після завантаження даних ваш екран WEKA повинен виглядати так, як показано на рис. 5.

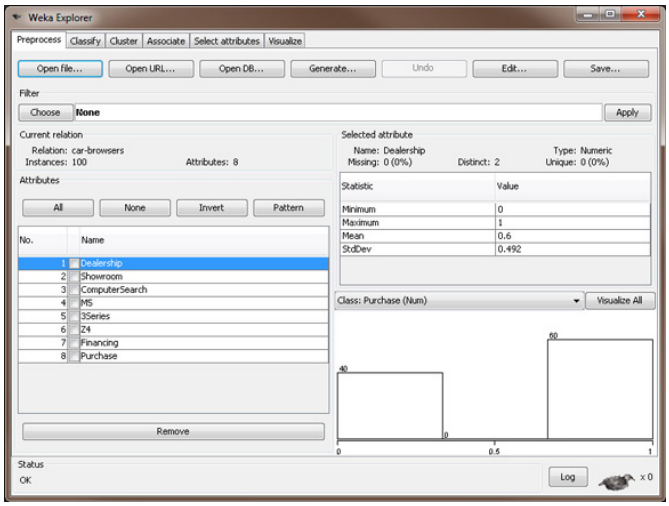


Рис. 5. Дані BMW для кластеризації

Оскільки ми хочемо розбити наявні в нас дані на кластери, замість закладки **Classify** нам буде потрібно закладка **Cluster**. Натисніть на кнопку **Choose** і в пропонованому меню виберіть опцію **SimpleKMeans** (в рамках даної роботи ми будемо користуватися цим методом кластеризації). У результаті вікно **WEKA Explorer** буде виглядати так, як показано на рис. 6.

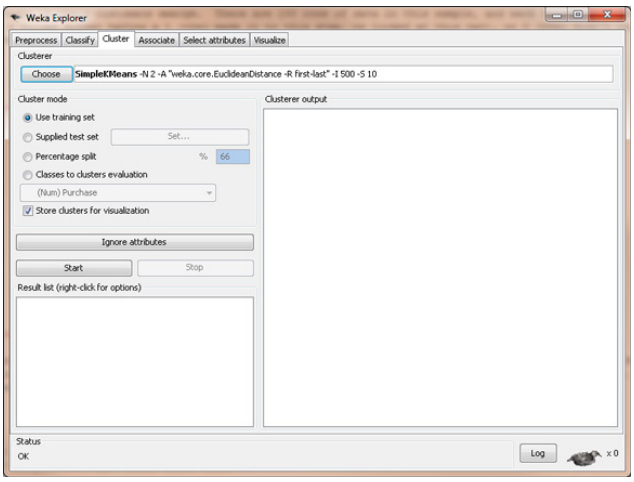


Рис. 6. Алгоритм кластеризації даних BMW

Тепер нам потрібно вибрати необхідні параметри для нашого алгоритму кластеризації. Клацніть на опції **SimpleKMeans** (дизайн користувальницького інтерфейсу залишає бажати кращого, але працювати з ним можна). Єдиний атрибут алгоритму, який нас цікавить - це поле **numClusters**, яке вказує на кількість кластерів для розбиття (нагадуємо, що це значення вам потрібно вибрати ще до створення моделі). Змінимо значення за замовчуванням (**2**) на **5**. Постарайтеся запам'ятати послідовність кроків, щоб ви змогли згодом змінити кількість кластерів. Тепер ваше вікно **WEKA Explorer** має виглядати так, як показано на рис. 7. Натисніть на кнопку **OK**, щоб зберегти вибрані параметри.

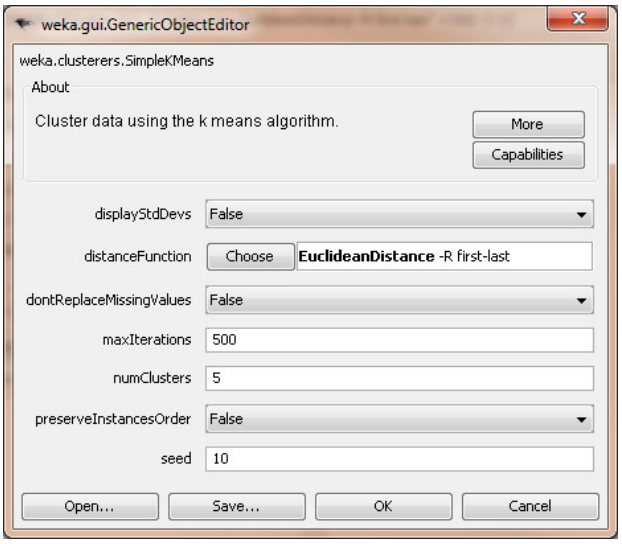


Рис. 7. Налаштування алгоритму кластеризації

Тепер ми можемо приступити до створення моделі. Як вже зазначалося вище, для розбиття 100 рядків на 5 кластерів за допомогою електронних таблиць було б потрібно декілька годин, проте WEKA видає нам результат менш ніж за секунду. Результат обробки наших даних показаний нижче.

Результати кластеризації

	Cluster#					
Attribute	Full Data	0	1	2	3	4
	(100)	(26)	(27)	(5)	(14)	(28)
Dealership	0.6	0.9615	0.6667	1	0.8571	0

Showroom	0.72	0.6923	0.6667	0	0.5714	1
ComputerSearch	0.43	0.6538	0	1	0.8571	0.3214
M5	0.53	0.4615	0.963	1	0.7143	0
3Series	0.55	0.3846	0.4444	0.8	0.0714	1
Z4	0.45	0.5385	0	0.8	0.5714	0.6786
Financing	0.61	0.4615	0.6296	0.8	1	0.5
Purchase	0.39	0	0.5185	0.4	1	0.3214

Clustered Instances

0	26 (26%)
1	27 (27%)
2	5 (5%)
3	14 (14%)
4	28 (28%)

Як нам інтерпретувати отриманий результат? Дані кластеризації показують, яким чином сформований кожен кластер: значення «1» означає, що у всіх даних в цьому кластері відповідний атрибут дорівнює 1, а значення «0» означає, що у всіх даних в цьому кластері відповідний атрибут дорівнює 0. Дані відповідають середньому значенню атрибута у кластері. Кожен кластер характеризує певний тип поведінки клієнтів, таким чином, на підставі нашого розбиття ми можемо зробити деякі корисні висновки:

- **Кластер 0** - цю групу відвідувачів можна було б назвати «мрійники». Вони бродять навколо дилерського центру, розглядаючи машини, виставлені на зовнішній парковці, але ніколи не заходять всередину, і, гірше того, ніколи нічого не купують.
- **Кластер 1** - цю групу слід було б назвати «шанувальники M5», оскільки вони відразу ж підходять до виставлених автомобілів цієї моделі, повністю ігноруючи BMW серії 3 або Z4. Тим не менш, ця група не відрізняється високими показниками покупки машин - всього 52%. Це потенційно може свідчити про недостатньо продуману стратегію продажів і про необхідність поліпшити роботу дилерського центру, наприклад, за рахунок збільшення кількості продавців у секції M5.
- **Кластер 2** - ця група настільки мала, що ми могли б назвати її вибраковуванням. Справа в тому, що дані цієї групи статистично досить розкидані, і ми не можемо зробити будь-яких певних висновків щодо поведінки відвідувачів, що потрапили в цей кластер (подібна ситуація може вказувати на те, що вам слід скоротити кількість кластерів в моделі)
- **Кластер 3** - цю групу слід було б назвати «улюбленці BMW», тому що відвідувачі, що потрапили в цей кластер, завжди купують машину і отримують необхідне фінансування. Зверніть увагу, дані цього кластеру демонструють цікаву модель поведінки цих покупців: спочатку вони оглядають виставлені на парковці машини, а потім звертаються до пошукової системи дилерського центру. Як правило, вони купують моделі M5 або Z4, але ніколи не беруть моделі третьої серії. Дані цього кластеру вказують на те, що дилерському центру слід активніше привертати увагу до пошукових комп'ютерів (можливо, винести їх на зовнішню парковку), і крім того, слід знайти якийсь спосіб виділити моделі M5 і Z4 в результатах пошуку, щоб гарантовано звернути на них увагу відвідувачів. Після того, як відвідувач, що потрапив в цей кластер, вибрав певну модель автомобіля, він гарантовано отримує необхідний кредит і здійснює покупку.
- **Кластер 4** - цю групу можна назвати «початківці власники BMW», оскільки вони завжди шукають моделі 3 серії і ніколи не цікавляться більш дорогими M5. Вони відразу ж проходять в демонстраційний зал, не витрачаючи час на огляд машин на зовнішній стоянці. Крім того, вони не користуються пошуковою системою центру. Приблизно 50%

цієї групи отримують схвалення по кредиту, тим не менш, покупку роблять всього 32% учасників. Аналізуючи дані цього кластеру, можна зробити наступний висновок: відвідувачі цієї групи хотіли б купити свій перший BMW і точно знають, яка машина їм потрібна (модель 3 серії з мінімальною конфігурацією). Однак, для того щоб купити машину, їм потрібно отримати позитивне рішення по кредиту. Щоб підвищити рівень продажів серед відвідувачів 4 кластера, дилерському центру слід було б знизити рівень вимог для отримання кредиту або знизити ціни на моделі 3 серії.

Ще один цікавий спосіб вивчення результатів кластеризації - це візуальне подання даних. Клацніть правою кнопкою мишки в секції **Result List** закладки **Cluster**. У контекстному меню виберіть опцію **Visualize Cluster Assignments**. В результаті відкриється вікно з графічним представленням результатів кластеризації, налаштування якого ви можете вибрати найбільш зручним для вас чином. Для нашого прикладу, змініть настройку осі **X** так, щоб вона відповідала кількості автомобілів M5 (**M5 (Num)**), а настройку осі **Y** - так, щоб вона показувала кількість куплених автомобілів (**Purchase (Num)**), і вкажіть виділення кожного кластера окремим кольором (для цього встановіть значення поля **Color** в **Cluster (Nom)**). Такі налаштування допоможуть нам оцінити розподіл по кластерах залежно від того, скільки людина цікавилося BMW M5, і скільки чоловік купило цю модель. Крім того, посуньте покажчик **Jitter** приблизно на три чверті у бік максимуму, це штучно збільшить розкид між групами точок, щоб нам було зручніше їх переглядати.

Чи відповідає візуальне відображення кластеризації тим висновкам, які ми зробили на підставі даних в одержаному результаті кластеризації? Як ми бачимо, поблизу точки **X = 1**, **Y = 1** (відвідувачі, які цікавилися автомобілями моделі M5 і купили їх) розташовані тільки два кластери: **1** і **3**. Аналогічно, поблизу точки **X = 0**, **Y = 0** розташовані тільки два кластери: **4** і **0**. Чи відповідає це нашим висновкам? Так, відповідає. Кластери **1** і **3** купують BMW M5, в той час як кластер **0** не купує нічого, а кластер **4** шукає BMW серії 3. На рис. 8 показано візуальне відображення кластерів нашої моделі. Пропонуємо вам самостійно попрактикуватися у виявленні інших трендів і течій, змінюючи налаштування осей **X** і **Y**.

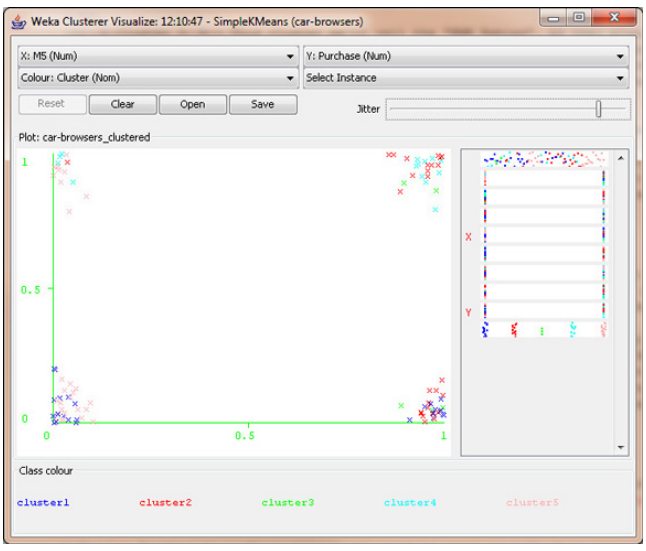


Рис. 8. Візуальне відображення кластеризації

Контрольні запитання

1. З якими методами інтелектуального аналізу даних ми ознайомилися в даній лабораторній роботі?
2. Під якими іншими назвами відомий метод класифікації?

3. У чому полягає основна перевага класифікаційних дерев?
4. Суть підходу створення моделі на основі навчальної послідовності (training set).
5. З якою метою в методі класифікаційного аналізу набір відомих даних ділиться на дві частини?
6. Принцип відсікання гілок. Для чого може бути потрібно видаляти інформацію з дерева рішень?
7. Сенс проблеми помилкового розпізнавання. Що є більш небажаним: хибно-позитивне чи хибно-негативне розпізнавання? Яке співвідношення хибно-негативних розпізнавань до хибно-позитивних може вважатися прийнятним?
8. В результаті аналізу для тестового і навчального набору даних одержана приблизно однакова точність. Як це вплине на нові дані які будуть використовуватися в цій моделі в майбутньому?
9. Коли використовується метод кластерного аналізу?

Основний недолік методу кластеризації.