УДК 004.62

ОЦІНКА ВИБОРУ ПРОДУКТУ ЗА ДОПОМОГОЮ СИСТЕМ МАШИННОГО НАВЧАННЯ

Сотников А. Д., Пронін С.В.

Харківський національний автомобільно-дорожній університет, Харків

Можливість на основі загальнодоступних даних створювати програми для оцінки переваг окремих користувачів може бути використано, зокрема, в сфері інтернет-комерції — знання про те, який товар краще для покупця, допомагають ефективніше організувати контекстні пропозиції товарів, що в свою чергу веде в збільшення ефективності бізнесу в цілому.

Технологія машинного навчання на основі аналізу даних бере початок в 1950 році, коли почали розробляти перші програми для гри в шашки. За минулі десятиліття загальний принцип не змінився. Зате завдяки бурхливому зростанню обчислювальних потужностей комп'ютерів багаторазово ускладнилися закономірності і прогнози, створювані ними, і розширилося коло проблем і завдань, що вирішуються з використанням машинного навчання [1].

Щоб запустити процес машинного навчання, для початку необхідно зібрати інформацію і сформувати інформаційний масив придатний для алгоритму який буде обробляти запити. Процес навчання триває і після виданих прогнозів, чим більше даних ми проаналізували програмою, тим більш точно вона розпізнає потрібні зображення.

Таким чином основна ідея машинного навчання полягає в тому, щоб навчити комп'ютер "вчитися", тобто виокремлювати з будь-яких даних корисні знання.

У зв'язку з швидким зростанням обсягу інформації виникає гостра необхідність в обробці та структуруванні цієї інформації для її подальшого використання в навчанні моделі, яка на виході буде давати результат. Першим етапом побудови моделі класифікації є збір і попередня обробка даних – цей

процес знаходиться на стику таких розділів як: Великі данні(Big data) [2] та інтелектуальний аналіз даних (Data Mining) [3].

Огляд методів та алгоритмів класифікації

Вирішити задачу оцінки вибору користувачем того або іншого продукту можливо за допомогою різних методів класифікації.

Побудова класифікатора дає можливість оцінити переваги вибору на основі аналізу інформації о перевагах користувачів.

У машинному навчанні завдання класифікації вирішується з використанням навчання з учителем, оскільки класи визначаються заздалегідь і для прикладів навчальної множини мітки класів задані. Аналітичні моделі, вирішуючи задачу класифікації, називаються класифікаторами.

До числа поширених методів вирішення задачі класифікації відносяться:

- нейронні мережі;
- логістична і пробіт-регресія;
- дерева рішень;
- метод найближчого сусіда;
- метод опорних векторів;
- дискримінантний аналіз;
- байєсівська(наївна) класифікація.

Для побудови класифікатору використаємо штучну нейрону мережу.

Нейронні мережі – це один з напрямків досліджень в області штучного інтелекту, засноване на спробах відтворити нервову систему людини. А саме: здатність нервової системи навчатися і виправляти помилки, що має дозволити змоделювати, хоча і досить грубо, роботу людського мозку.

Здатність до моделювання нелінійних процесів, роботи з зашумленними даними і адаптивність дають можливості застосовувати нейронні мережі для вирішення широкого класу завдань. В останні кілька років на основі нейронних мереж було розроблено багато програмних систем які охоплюють найрізноманітніші галузі: розпізнавання образів, обробка зашумленних даних,

доповнення образів, асоціативний пошук, класифікація, оптимізація, прогноз, діагностика, обробка сигналів, абстрагування, управління процесами, сегментація даних, стиснення інформації, складні зображення, моделювання складних процесів, машинний зір, розпізнавання мови [3].

Багатошаровий перцептрон (MLP). Нейронна мережа на основі MLP являє собою систему з пов’язаних між собою шарів нейронів. Кожен нейрон характеризується функцією активації, яка перетворює вхідний сигнал нейрона у вихідний. Зв’язки нейронів з іншими нейронами характеризуються коефіцієнтами – так званими вагами зв’язку. Важливим фактором у навчанні нейронної мережі є вид вхідних даних. Для досягнення найкращих результатів необхідно попередньо провести відображення даних на діапазон

[-1;1] за допомогою операцій центрування і масштабування (1).

$$x_n = \frac{(x - m_x)}{m_x}. \quad (1)$$

Процес навчання є ітеративною послідовністю операцій розрахунку вихідного сигналу мережі та подальшого коректування ваг зв’язків. В якості алгоритму коригування ваг в мережах на основі MLP зазвичай застосовується алгоритм зворотного поширення помилки.

Алгоритм зворотного поширення помилки (Back propagation algorithm). Алгоритм зворотного поширення помилки – популярний алгоритм навчання 227лоско шарових нейронних мереж прямого поширення (багатошарових персептронов). Відноситься до методів навчання з учителем, тому вимагає, щоб в навчальних прикладах були задані цільові значення. Також є одним з найбільш відомих алгоритмів машинного навчання.

Даний алгоритм належить до класу градієнтних алгоритмів, т. е. зміни ваг зв’язків виробляються в напрямку мінімізації градієнта помилки. Помилка прогнозу при навчанні дорівнює різниці сигналу на виході мережі і еталонного значення виходу, відповідного вхідним даними (2).

$$e_i = (y_i - d_i). \quad (2)$$

Навчання мережі необхідно виконувати до тих пір, поки середня величина помилки за одну епоху навчання зменшується. Подальше навчання зазвичай приводить до погіршення аналітичних можливостей нейронної мережі. Навчання мережі проводилося за допомогою алгоритму зворотного поширення помилки та алгоритму градієнтного спуску.

В основі ідеї алгоритму лежить використання вихідний помилки нейронної мережі (3) для обчислення величин корекції ваг нейронів в її прихованих шарах:

$$\mathrm{E} = \frac{1}{2} \sum_{\mathrm{i}=1}^{\mathrm{k}} (\mathrm{y} - \mathrm{y}')^2, \quad (3)$$

де k - число вихідних нейронів мережі, y- цільове значення;

y ' - фактичне вихідне значення.

Алгоритм є ітеративним і використовує принцип навчання «по кроках» (навчання в режимі on-line), коли ваги нейронів мережі коригуються після подачі на її вхід одного навчального прикладу.

На кожній ітерації відбувається два проходи мережі - прямий і зворотний. На прямому вхідний вектор поширюється від входів мережі до її виходів і формує певний вихідний вектор, що відповідає поточному (фактичного) стану ваг. Потім обчислюється помилка нейронної мережі, як різниця між фактичним і цільовим значеннями. На зворотному проході ця помилка поширюється від виходу мережі до її входів, і проводиться корекція ваг нейронів відповідно за формулою 4:

$$\Delta \mathrm{w}_{\mathrm{j,i}}(\mathrm{n}) = \frac{-\eta \, \partial \mathrm{E}_{\mathrm{av}}}{\partial \mathrm{w}_{\mathrm{ij}}}, \quad (4)$$

де $w_{j,i}$ - вага і-й зв'язку j-го нейрона;

η - параметр швидкості навчання, який дозволяє додатково керувати величиною кроку корекції;

Δw_{ji} з метою більш точного налаштування на мінімум помилки і підбирається експериментально в процесі навчання (змінюється в інтервалі від 0 до 1).

Стохастичний градієнтний спуск. Цей алгоритм відноситься до оптимізаційних алгоритмів та використовується для налаштування параметрів моделі машинного навчання. При стандартному (або «пакетному», «batch») градієнтному спуску для коригування параметрів моделі використовується градієнт. Градієнт зазвичай вважається як сума градієнтів, викликаних кожним елементом навчання. Вектор параметрів змінюється в напрямку антиградієнта з заданим кроком. Тому стандартному градієнтному спуску потрібно один прохід по навчальних даних до того, як він зможе змінювати параметри. При стохастичному (або «оперативному») градієнтному спуску значення градієнта апроксимується градієнтом функції вартості, обчисленому тільки на одному елементі навчання. Потім параметри змінюються пропорційно наближеному градієнту. Таким чином параметри моделі змінюються після кожного об'єкта навчання. Для великих масивів даних стохастичний градієнтний спуск може дати значну перевагу в швидкості в порівнянні зі стандартним градієнтним спуском. Між цими двома видами градієнтного спуску існує компроміс, званий іноді «mini-batch». В цьому випадку градієнт апроксимується сумою для невеликої кількості навчальних зразків.

У якості інтерпретації результату в мережі застосовується функція softmax та Relu. Softmax - це логістична функція для багатовимірного випадку. Функцію застосовують ні до окремого значення, а до вектора. Наприклад, її можна використовувати в тому випадку, коли стоїть завдання багато класової класифікації. Для такої класифікації мережу будують таким чином, що на останньому шарі кількість нейронів виявляється рівною кількості шуканих класів. При цьому кожен нейрон має видавати значення ймовірності

приналежності об'єкта до класу (5), тобто значення між нулем і одиницею, а все нейрони в сумі повинні дати одиницю.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{k} e^{z_k}}. \quad (5)$$

Relu - Rectified linear unit (ReLU) або «випрямляч» (rectifier, за аналогією з однопівперіодним випрямлячем в електротехніці) є найбільш часто використовуваної функцією активації з 2015 року. Це проста умова і має переваги перед іншими функціями. Функція визначається наступною формулою:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (6)$$

На даний момент існує ряд мов програмування, за допомогою яких можна розроблювати додатки для машинного навчання та аналізу даних. Для побудови класифікатора застосуємо мову Python.

Одна з основних причин, чому Python використовується для машинного навчання полягає в тому, що у нього є безліч фреймворків, які спрощують процес написання коду і скорочують час на розробку. Обговорімо, які саме бібліотеки і фреймворки Python використовуються в машинному навчанні. У наукових розрахунках використовується Numpy, SciPy, в добуванні і аналізі даних - SciKit-Learn. Ці бібліотеки часто використовують разом з TensorFlow, CNTK і Apache Spark за допомогою яких проектуються нейронні мережі. Крім того простий синтаксис мови Python допомагає розробнику тестувати складні алгоритми з мінімальною витратою часу на їх реалізацію.

Для рішення задачі достатньо фреймворку Scikit-learn [4].

Бібліотека scikit-learn складається з 35 модулей, які можна поділити на модулі кластеризації, модулі оцінки моделі і кількісного визначення якості прогнозів, модулі роботи з наборами даних (предобробка, нормалізація),

модулі роботи з ознаками (витяг і виявлення найбільш значущих), модулі, що реалізують різні алгоритми вирішення задач класифікації і регресії.

Scikit-learn побудована на основі стека SciPy (Scientific Python), який включає в себе:

- NumPy додає підтримку великих багатовимірних масивів і матриць, а також бібліотеку високорівневих математичних функцій для операцій з ними.

- SciPy - відкрита бібліотека високоякіс-кількісний наукових інструментів для мови програмування Python.

- Matplotlib - бібліотека для візуалізації двовимірної і тривимірної графіки.

- Pandas реалізує різні структури даних і аналіз.

У Scikit-learn класифікатор на основі штучної нейронної мережі задається у вигляді:

model = sklearn.neural_network.MLPClassifier (activation='logistic', max_iter=100, hidden_layer_sizes=(2,), solver='lbfgs')

де activation='logistic' – функція активації;

max_iter=100 – кількість циклів навчання;

hidden_layer_sizes=(2,) – кількість слоїв мережі;

solver='lbfgs' – алгоритм навчання.

Список використаних джерел

[1] Weka: Data Mining [Он-лайн]. Доступно: https://www.cs.waikato.ac.nz/~ml/weka/.

[2] Big Data [Он-лайн]. Доступно: https://www.it.ua/ru/knowledge-base/technology-innovation/big-data-bolshie-dannye.

[3] А. А. Барсегян, М. С. Куприянов, Методы и модели анализа даннях. С.-Петербург: БХВ Петербург, 2004 г.

[4] Scikit-learn. Machine Learning in Python. [Он-лайн]. Доступно: http://scikit-learn.org.