DOI: https://doi.org/10.26565/2304-6201-2023-59-07

УДК (UDC) 004.67:004.8

Узлов Дмитро Юрійович
к.т.н., доцент закладу вищої освіти кафедри теоретичної та прикладної інформатики
Харківський національний університет імені В .Н. Каразіна, майдан Свободи, 4, Харків, Україна, 61022
e-mail: dmytro.uzlov@karazin.ua
https://orcid.org/0000-0003-3308-424X

Морозова Анастасія Геннадіївна
к.т.н., старший викладач закладу вищої освіти кафедри теоретичної та прикладної інформатики
Харківський національний університет імені В .Н. Каразіна, майдан Свободи, 4, Харків, Україна, 61022
e-mail: a.morozova@karazin.ua
https://orcid.org/0000-0003-2143-7992

Кузнєцова Вікторія Олександрівна
к.ф-м.н., доцент закладу вищої освіти кафедри вищої математики та інформатики
Харківський національний університет імені В .Н. Каразіна, майдан Свободи, 4, Харків, Україна, 61022
e-mail: vkuznietcova@karazin.ua
https://orcid.org/0000-0003-3882-1333

Руккас Кирило Маркович
д.т.н, доцент, професор закладу вищої освіти кафедри теоретичної та прикладної інформатики
Харківський національний університет імені В .Н. Каразіна, майдан Свободи, 4, Харків, Україна, 61022
e-mail: rukkas@karazin.ua
https://orcid.org/0000-0002-7614-0793

Використання нейронних мереж для масштабування табличних даних тренувальних dataset

У роботі запропоновано метод збільшення табличних даних тренувальних dataset за допомогою нейронних мереж, описано архітектуру таких мереж.
Актуальність. На даний час існує проблема недостатньої кількості вихідних даних для навчання моделей штучного інтелекту, що призводить до значної похибки моделювання. Робота присвячено розробці підходів до генерації штучних табличних даних, які можна використовувати надалі для моделей штучного інтелекту.
Мета. Метою роботи було проаналізувати методи та алгоритми для збільшення training dataset для табличних даних за допомогою нейронних мереж.
Методи дослідження. Основним методом дослідження є процес підбору параметрів алгоритму генерації штучних даних та вибір оптимальних параметрів архітектури нейронної мережі.
Результати. Використання нейронних мереж для масштабування табличних даних тренувальних dataset підтвердило працездатність запропонованого підходу. Результати налаштування алгоритму та вибір оптимальних параметрів нейронної мережі показали, що згенеровані штучні дані найбільше нагадують початкові по критеріям середнього значення, максимального, мінімального та залежності між даними.
Висновки. Вирішено задачу масштабування табличних даних тренувальних dataset за допомогою нейронних мереж. Такий підхід дозволяє значно спростити процес навчання нейронних мереж. Наукова новизна даної роботи полягає в розробці підходів і методів збільшення табличних даних з використанням штучного інтелекту та deep learning.
Ключові слова: *нейронні мережі, database, табличні дані, data augmentation, training dataset, штучний інтелект, deep learning.*

Як цитувати: Узлов Д. Ю., Морозова А. Г., Кузнєцова В. О., Руккас К. М. Використання нейронних мереж для масштабування табличних даних тренувальних dataset. *Вісник Харківського національного університету імені В. Н. Каразіна, серія Математичне моделювання. Інформаційні технології. Автоматизовані системи управління*. 2023. вип. 59. С.63-71. https://doi.org/10.26565/2304-6201-2023-59-07

How to quote: D. Uzlov, A. Morozova, V. Kuznietcova, K. Rukkas, “Scaling tabular data of training datasets with neural networks” *Bulletin of V. N. Karazin Kharkiv National University, series*

© Узлов Д. Ю., Морозова А. Г., Кузнєцова В. О., Руккас К. М., 2023
This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0.

Mathematical modelling. Information technology. Automated control systems, vol. 59, pp.63-71, 2023. https://doi.org/10.26565/2304-6201-2023-59-07

1 Вступ

Data augmentation (збільшення даних) – це техніка штучного створення нових даних з наявних даних і значного збільшення різноманітності даних, доступних для навчання моделей. Це робиться шляхом застосування предметно-орієнтованих методів до підмножини навчальних даних. Оскільки продуктивність моделі сильно залежить від якості та кількості набору даних, використання синтетично згенерованих даних може певною мірою допомогти покращити продуктивність моделі [1]. Data augmentation техніки широко використовуються для графічних даних (Image Augmentation) [2], а також текстових даних (Text Data Augmentation) [3]. Для збільшення SQL даних або табличних даних не існує стандартних технік та методів. Сьогодні прийняття рішення на основі статистики є суттєвою для ряду задач, тому використання методів збільшення табличних даних та використання Neural Networks та Machine Learning для таких даних є актуальною задачею.

Використання штучно згенерованих даних вирішує такі проблеми як:

1. Необхідність великої кількості досліджень та збору даних (опитування користувачів сервісу, проведення тестових випробувань, тощо).
2. Аналіз зібраних даних.
3. Відкидання неправдивих даних (навмання заповнені бюлетені, помилки через технічні причини, тощо).
4. Оцифрування даних.

Техніка Data augmentation може бути використана в будь-якій компанії, яка застосовує у своїх дослідженнях штучний інтелект та табличні дані. Вона має зменшити витрати на збір даних тим самим пришвидшити впровадження нових змін в проекті, а також покращити якість навчання моделі.

Техніка Data augmentation може бути корисним в будь-якому проекті, що спеціалізується на роботі з табличними даними та штучним інтелектом, та особливо у випадках, коли є необхідність у великій кількості досліджень та збору даних, але їх за якоїсь причини важко зібрати у необхідній кількості.

Метод масштабування табличних даних повинен генерувати нові дані базуючись на вхідному dataset. Він може працювати з невеликою кількістю вхідних рядків та генерувати задану кількість нових. Нові рядки базуються здебільшого на середньому значенні вхідних рядків та копіюють їх тип розподілу відносно інших стовпчиків таблиці. Тим самим згенеровані дані «візуально» здаються схожими на ті, на яких проводилось тренування моделі.

2 Постановка задачі

Штучний інтелект набуває все більшої популярності останнім часом та прогнозується збільшення популярності у майбутньому. Через це питання збільшення training dataset для табличних даних є актуальним у сьогодення, основним завданням якого є аналіз вхідного dataset та генерація нових, схожих табличних даних. Для досягнення поставленої мети, були сформульовані наступні задачі:

1. Проаналізувати наявні бібліотеки для збільшення табличних даних.
2. Проаналізувати методи роботи бібліотеки з даними та за можливості покращити якість роботи алгоритму з ними, підібравши найкращі параметри для роботи алгоритму. А саме:
    a. скалери;
    b. алгоритми оптимізації для моделі глибокого навчання;
    c. топологія нейронної мережі.
3. Вдосконалити роботу бібліотеки для роботи з типами даних string та int.
4. Протестувати роботу бібліотеки на dataset з різними видами розподілу. У якості dataset використовувати реальні дані, а не згенеровані.

Вибір бібліотеки для збільшення табличних даних

В роботі розглянуто можливість використання різних бібліотеки для генерування нових табличних даних із вхідного dataset. Критерієм вибору бібліотеки була задача максимально

зберегти початкові характеристики даних, а саме математичне очікування, дисперсію та залежність між стовпцями.

Бібліотека deep_tabular_augmentation надає абсолютну свободу користувачу щодо налаштування вхідних параметрів, що використовуються для генерування нових даних [9]. Бібліотека дозволяє самостійно обирати, досліджувати та змінювати вхідні параметри для отримання бажаного результату.

Бібліотека RandomForestClassifier з sklearn дозволяє вказувати тільки вхідний dataset без можливості самостійно впливати на генерацію даних [10].

Пакет ydata_synthetic надає значно більше свободи користувачу у порівнянні з RandomForestClassifier [11]. Він надає більше можливостей впливати на зміну даних, але все ж таки менше ніж при використанні deep_tabular_augmentation.

Для генерації табличних даних тренувальних dataset було обрано бібліотеку deep_tabular_augmentation.

3 Алгорит збільшення табличних даних

Налаштування бібліотеки deep_tabular_augmentation для масштабування табличних даних тренувальних dataset складається з декількох етапів:

1. Підготовка та масштабування ознак.
2. Розбиття даних.
3. Визначення топології нейронної мережі
4. Вибір оптимізатора
5. Визначення кількості епох для навчання моделі

Масштабування ознак.

Перший етап – це Масштабування ознак (або Нормалізація даних). Перш за все dataset необхідно підготувати для роботи з ним. Для цього використовують масштабування ознак. Так як значення у даних можуть сильно різнитися між собою та мати різні діапазони, модель може давати хибні результати. Тому дані потрібно нормалізувати. Для цього використовують різні скалери з sklearn. В залежності від типу розподілу даних необхідно обрати відповідний скалер. Наприклад, MinMaxScaler та StandardScaler гарно працюють з числовими даними. Водночас StandardScaler використовують для нормального розподілу, MinMaxScaler за відсутності нормального розподілу та коли варто вказати на чітку відстань між значеннями. Для більшості dataset якості роботи StandardScaler достатньо. Варто також зазначити, що деякі скалери, наприклад Normalizer неможливо використовувати з бібліотекою deep_tabular_augmentation через те, що розробник не впровадив необхідні для цього зміни в свій модуль. Порівняння скалерів наведено у Таблиці 1.

Таблиця 1. Порівняння скалерів

Скалер	Стійкий до викидів	З чіткою межею даних	Межа невідома
StandardScaler	-	-	+
MinMaxScaler	-	+	-
MaxAbsScaler	+	+	-
RobustScaler	+	-	+
PowerTransformer	+-	+-	+

Розбиття даних.

Наступним етап є Розбиття даних. Дані розбиваються на дві частини: для тренування та валідації моделі. Попередньо рядки перемішують між собою. Таким чином дані для валідації використовуються для того, щоб зрозуміти наскільки навчена модель, а також це допомагає виявити проблеми Underfitting та Overfitting. Зазвичай для розбиття даних використовується train_test_split() з бібліотеки sklearn.model_selection, але для більш специфічних завдань можна звернути увагу на split_df() з mlprepare. Низькі значення функції втрат можуть вказувати на те, що модель гарно навчилась генерувати нові дані, або на те, що вона перенавчена. Було протестовано наступні варіанти розбиття даних валідацію: 5%, 8%, 10%, 12% та 15%.

В більшості випадків найнижчі значення функції втрат, при яких зберігається залежність між даними, та модель не перенавчається, досягаються при виділенні 10% даних на валідацію.

Топологія нейронної мережі.

Після масштабування ознак необхідно вказати, за допомогою якої топології нейронної мережі будуть опрацьовуватися дані. Ця топологія вказує на зв'язки між вузлами (нейронами) в мережі. Для вирішення задачі найбільш цікавими є наступні три типи мережі, топології яких наведено на рисунку 3.1.:

- Auto Encoder (AE)
- Variational AE (VAE)
- Sparse AE (SAE)

Загалом вони використовуються для класифікації та кластеризації ознак. VAE на відміну від AE приділяє більше уваги на зв'язок між даними у той час, коли AE намагається їх узагальнити. SAE схожий на VAE, але також здатний знаходити приховані шаблони групування даних. На практиці це виявляється у тому, що SAE виділяє значно більше даних, що знаходяться значно далі від основного скупчення. Наприклад, якщо взяти нормальний розподіл, то SAE буде також виділяти точки, що знаходяться біля 0, у порівнянні з AE та VAE, котрі виділяють лише дані близькі до середнього значення. Тож було вирішено зупинитися на SAE тому, що для задачі масштабування табличних даних важливо вказати всі дані, а не лише близькі до середніх значень.

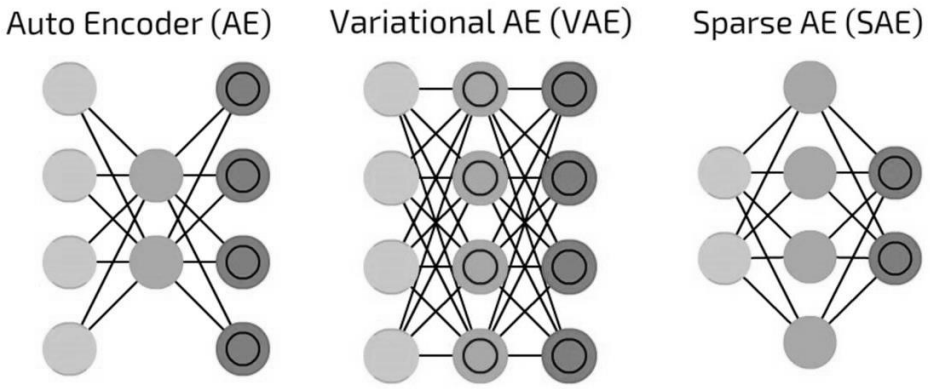

Рис.3.1. Топології нейронної мережі

Найближчі до початкових даних результати роботи алгоритму було отримано, коли слої топології виглядають наступним чином: 6, 20, 6. Тобто 6 нейронів у вхідному і вихідному слої та 20 у прихованому. При цьому, ці зміни майже не впливають на зміну середнього значення чи дисперсії даних, а змінюється здебільшого лише залежність даних між собою та вірогідність отримати мінімальних та максимальних значень одночасно в декількох стовпчиках таблиці.

Також кількість нейронів має сенс змінювати в залежності від вхідних даних. Емпірично було виявлено, що збільшення 1 та 3 слою підвищує «влучність» алгоритму, згенеровані дані стають ближчими до середніх значень розподілу. Збільшення або зменшення прихованого (другого) слою відповідно збільшує або зменшення кількість даних, що знаходяться на значному віддаленні від середнього значення. А ось зменшення першого та третього слою майже не має практичного сенсу, у більшості випадків на різних даних алгоритм все частіше дає похибки та помилково визначає залежність між розподілом даних. Наприклад, коли розподіл даних схожий на графік функції $y = a^x$ алгоритм може надати $y = \frac{1}{x}$.

Оптимізатор навчання нейронних мереж.

Наступний етап підготовки до data augmentation – це обрання алгоритму оптимізації навчання нейронної мережі, який використовується для налаштування ваги нейронної мережі в процесі її навчання. Він визначає, які значення ваги потрібно використовувати для мінімізації функції втрат, яка вимірює помилку передбачення моделі на навчальному наборі даних. Функція втрат – це функція, яка характеризує втрати при неправильному прийнятті рішень на основі спостережених даних. Тобто це метод оцінки того, наскільки добре алгоритм моделює вказаний набір даних, наскільки гарно алгоритм працює з заданим набором. [6]

Найкраще у дослідженнях себе проявили оптимізатори Adam (Адаптивне оцінювання моментів) та RMSProp (Пропагація кореня середньоквадратичного значення), та зовсім погано SGD (Стохастичний градієнтний спуск), незважаючи на його високу популярність. Хоча й Adam має більшу обчислювальну складність, ніж RMSprop, через необхідність обчислення додаткових моментів градієнта, але дані, що генеруються з його допомогою, у більшості випадках більш схожі на початкові у порівнянні з RMSProp. А саме min, max значення ближче до вхідних даних, залежність між даними більш схожа на залежність вхідних даних та менша ймовірність помилково вказати хибну залежність між даними.

Кількість епох при навчанні моделі.

Визначення цього параметру значною мірою залежить від самого датасету. У всіх розглянутих випадках найкращі результати отримувались при значенні 100-300 епох. При збільшенні цього значення модель починає перенавчатися та згенеровані нею дані все більше дублюють середнє значення початкового dataset, при зменшенні зростає ймовірність похибки алгоритму та неправильно виявленої залежності між даними.

Вдосконалення алгоритму при роботі з типами даних string та integer.

Кластеризація, або кластерний аналіз — це статистична процедура, задача якої полягає в розбитті вибірки об'єктів на підмножини, що не перетинаються і називаються кластерами [7]. Отже типовою задачею кластеризації є розбиття даних на основі їх подібності. Бібліотека deep_tabular_augmentation не вміє працювати зі строковими типами даних, хоча якщо строкові дані розбити на невелику кількість кластерів, то виходить, що генерувати нові дані спираючись на порядковий номер кластеру має сенс. Звісно, що такими діями не вийде згенерувати новий текст, а можна лише використовувати старий, а також такі дії будуть мати сенс лише у випадках, коли кількість даних більша за кількість кластерів. Тож було вирішено додати до алгоритму можливість генерувати дані зі строковим типом.

Також deep_tabular_augmentation завжди генерує дані з плаваючою точкою, навіть якщо вхідні дані мають цілий тип (integer). Цю особливість також було виправлено.

Види розподілу початкових даних та результати роботи алгоритму

Для тестування методів бібліотеки deep_tabular_augmentation та data augmentation були використані різні dataset, отримані із реальних даних на сайті kaggle.com. Нижче наведено результати роботи методів бібліотеки для різних видів розподілів.

Приклад 1. Нормальний розподіл.

На рисунку 3.2. зображені вхідні дані, а на рисунку 3.3. – згенеровані.

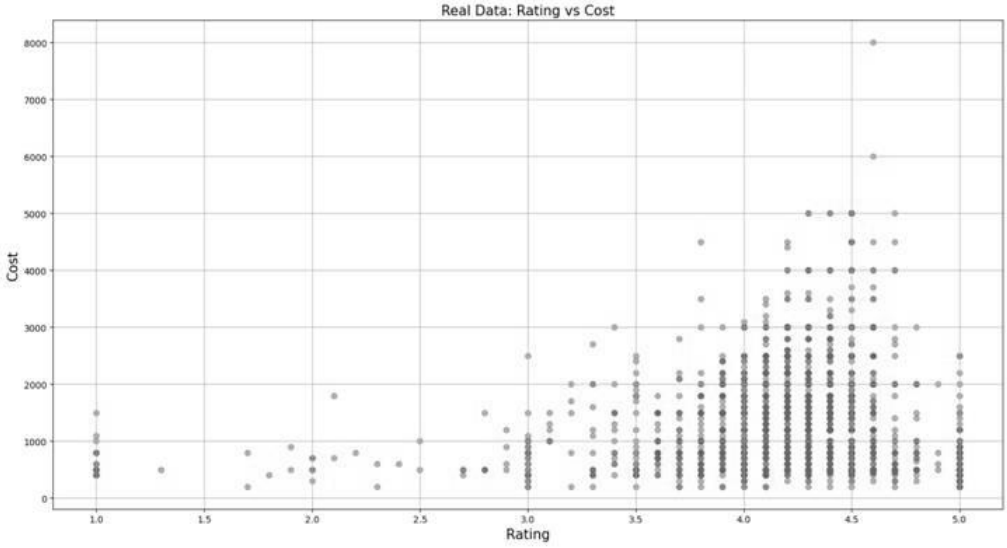

Рис.3.2. Вхідні дані

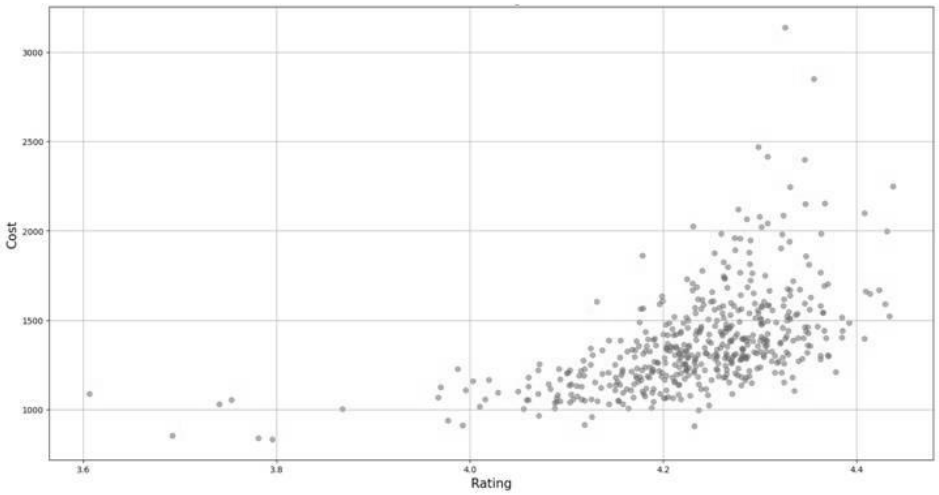

Рис.3.3. Згенеровані дані

На рисунках видно, що зберігається загальна форма розподілу та середнє значення. Варто зазначити, що у вхідних даних наявний шаг по осі X (Rating), тобто дані розташовані на певному віддалені один від одного, при цьому цей шаг не зберігається у згенерованих даних. Дані були згенеровані з використанням StandardScaler. Також, було застосовано інший вид скайлера, а саме MinMaxScaler, який застосовують для вказання чіткої відстані між даними. Але його застосування у даному випадку не виправдало себе, бо відстань він вказував недоречно. Тому було вирішено вказувати шаг між даними вже після генерації, оброблюючи дані, а не з застосуванням скалеру.

Приклад 2. Лінійний розподіл.

Наступний тип розподілу – лінійний. На рисунку 3.4. зображені вхідні дані, які один відносно одного розташовані по прямій лінії. На рисунку 3.5. – результат роботи алгоритму. Видно, як алгоритм доповнив пустоти. Також між даними у вхідному dataset зберігається математична залежність: по осі X – кількість років, по осі Y – кількість місяців, тому Y завжди у 12 разів більше за X, але у згенерованих даних ця математична точність втрачається.

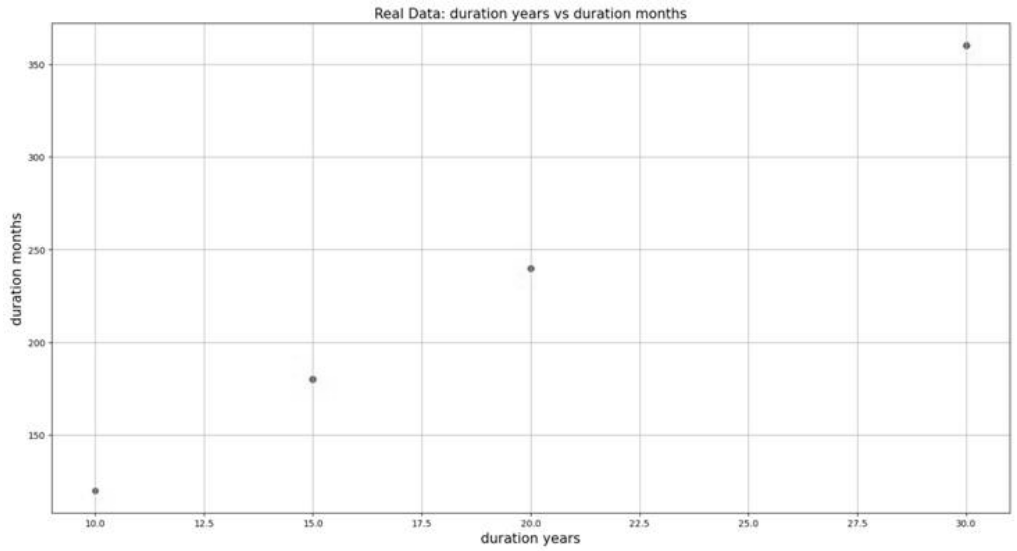

Рис.3.4. Вхідні дані

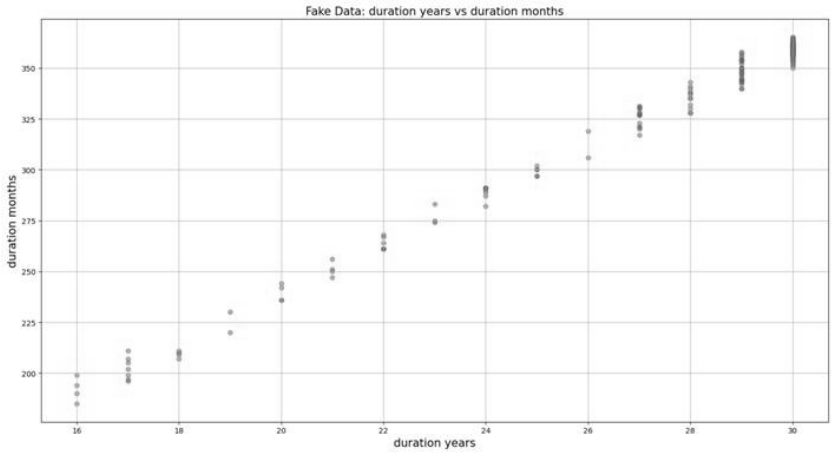

Рис.3.5. Згенеровані дані

Приклад 3.

Ще один приклад розподілу наведено на рисунку 3.6. та відповідні згенеровані дані на рисунку 3.7.

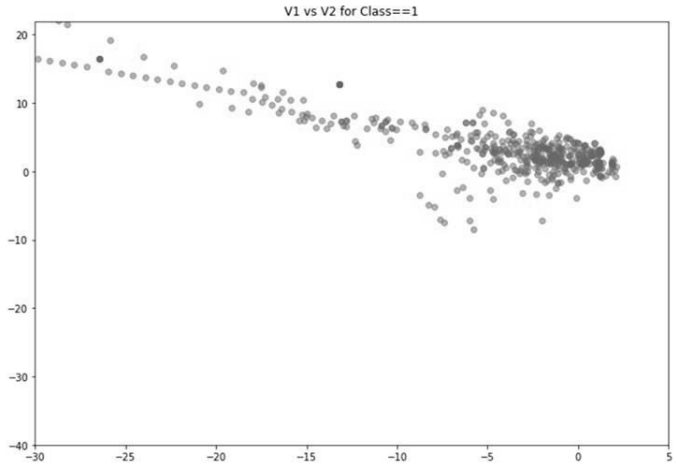

Рис.3.6. Вхідні дані

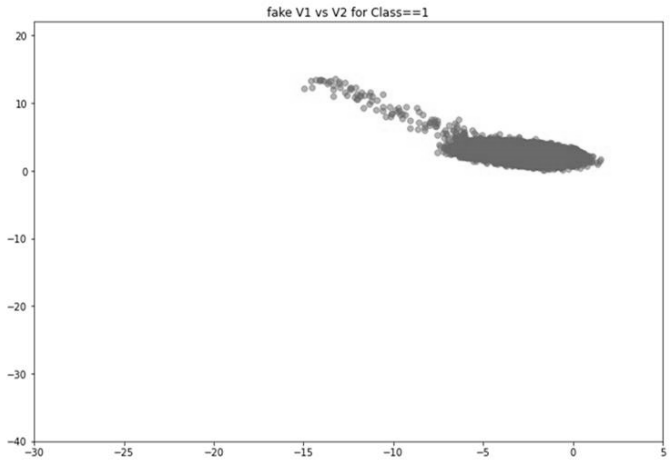

Рис.3.7. Згенеровані дані

Недоліки бібліотеки deep_tabular_augmentation.

Найбільш суттєвим недоліком є майже повна відсутність дисперсії у згенерованих даних. Дані дійсно мають схоже середнє значення, але наприклад, коли вхідні дані мають нормальний

розподіл, згенеровані візуально більше схожі на криву лінію, чи в деяких випадках дві криві лінії, що перетинаються. Приклад вхідних та згенерованих даних наведено на рисунку 3.8.

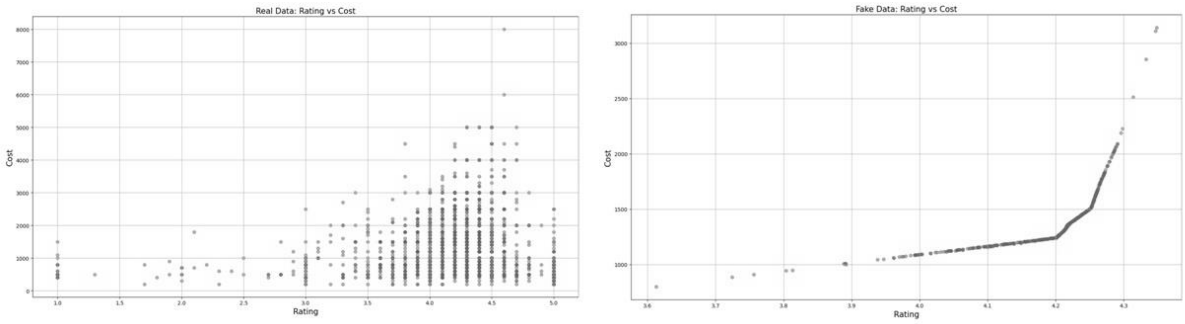

Рис.3.8.Ліворуч – вхідні дані, праворуч – згенеровані дані

Для вирішення цієї проблеми розробник радить трохи зміщувати дані, на відстань рівну 10% від дисперсії. У результаті змішування можна отримати приблизно наступний графік, наведений на рисунку 3.9.

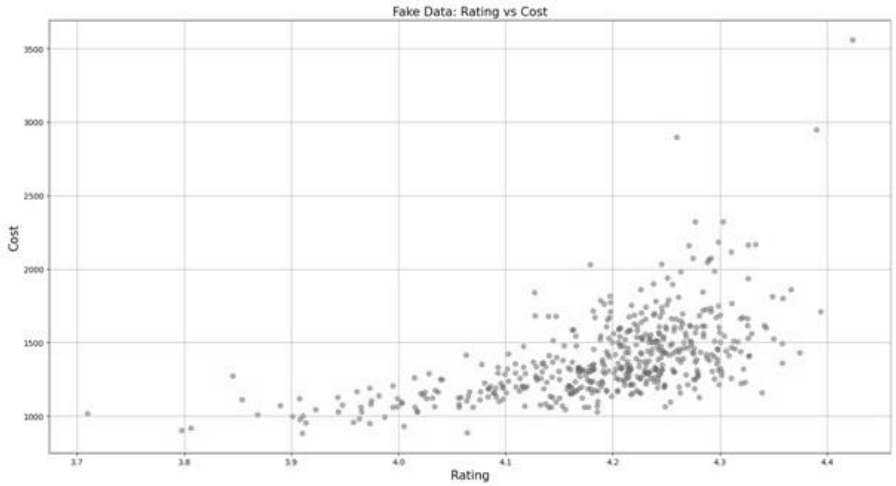

Рис.3.9. Згенеровані дані зі зміщенням 10% від дисперсії

Звісно, що в результаті ми ніколи не матимемо однакові дисперсії, а в спробі змістить дані на відстань рівну дисперсії ми можемо отримати у чисельному вигляді майже однакові середні значення та дисперсію, у порівнянні з початковими даними, але візуально це вже зовсім не схоже на нормальний розподіл, тобто залежність між даними втрачається.

4 Висновки

В роботі було проаналізовано методи та алгоритми для збільшення training dataset для табличних даних. Для досягнення цієї мети біло обрано бібліотеку deep_tabular_augmentation та проаналізував функції, що в ній використовуються. Під час аналізу було підібрано діапазони значень вхідних параметрів, при яких досягається найвища точність роботи алгоритму та згенеровані дані найбільше нагадують початкові по критеріям середнього значення, максимального, мінімального та спостерігається залежність між даними. Були підібрані наступні параметри:

- функція втрат;
- скалери;
- топологія нейронної мережі;
- оптимізатор навчання нейронних мереж;
- оптимальна кількість епох.

Також були помічені та проаналізовані недоліки модулю та покращена якість роботи алгоритму при роботі з типами даних string та integer.

СПИСОК ЛІТЕРАТУРИ

1. Abinaya Mahendiran, Vedanth Subramaniam. Data Augmentation Techniques for Tabular Data. Mphasis. https://www.mphasis.com/content/dam/mphasis-com/global/en/home/innovation/next-lab/Mphasis_Data-Augmentation-for-Tabular-Data_Whitepaper.pdf
2. Luis Perez, Jason Wang. The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv:1712.04621, 2017. https://arxiv.org/pdf/1712.04621
3. Shorten, C., Khoshgoftaar, T.M. & Furht, B. Text Data Augmentation for Deep Learning. J Big Data 8, 101 (2021). https://doi.org/10.1186/s40537-021-00492-0
4. Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, Quoc V. Le. Unsupervised Data Augmentation for Consistency Training. arXiv:1904.12848v6, 2020. https://arxiv.org/pdf/1904.12848v6
5. E. Jannik Bjerrum. SMILES Enumeration as Data Augmentation for Neural Network Modeling of Molecules. ArXive-prints, Mar. 2017
6. Alhassan Mumuni, Fuseini Mumuni. Data augmentation: A comprehensive survey of modern approaches. https://doi.org/10.1016/j.array.2022.100258
7. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. International conference on machine learning, PMLR (2015), pp. 448-456
8. Agnieszka Mikolajczyk, Michal Grochowski. Data augmentation for improving deep learning in image classification problem. 2018 International Interdisciplinary PhD Workshop (IIPhDW). DOI:10.1109/IIPHDW.2018.8388338
9. https://github.com/lschmiddey/deep_tabular_augmentation
10. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
11. https://docs.synthetic.ydata.ai/1.4

Uzlov Dmytro	*Doctor of Philosophy, Associate professor of theoretical and applied computer science department, V. N. Karazin Kharkiv National University, Svobody Sq., 4, Kharkiv, Ukraine, 61022*
Morozova Anastasiia	*Doctor of Philosophy, Senior lecturer of theoretical and applied computer science department, V. N. Karazin Kharkiv National University, Svobody Sq., 4, Kharkiv, Ukraine, 61022*
Kuznietcova Victoriya	*Doctor of Philosophy, Associate professor of higher mathematics and computer sciences department, V. N. Karazin Kharkiv National University, Svobody Sq., 4, Kharkiv, Ukraine, 61022*
Rukkas Kyrylo	*Doctor of Technical Sciences, Associate professor, Professor of theoretical and applied computer science department, V. N. Karazin Kharkiv National University, Svobody Sq., 4, Kharkiv, Ukraine, 61022*

Scaling tabular data of training datasets with neural networks

The paper proposes a method of scaling the tabular data of the training dataset using neural networks, describes the architecture of such networks.

Relevance. Presently, there is a problem of insufficient amount of raw data for training artificial intelligence models, which leads to significant modeling error. The work is devoted to the development of approaches to the generation of artificial tabular data, which can be used in the future for artificial intelligence models.

Goal. The purpose of the work was to analyze methods and algorithms for scaling the training dataset for tabular data using neural networks.

Research methods. The main research method is the process of selecting the parameters of the artificial data generation algorithm and choosing the optimal parameters of the neural network architecture.

The results. Using neural networks for scaling the tabular data of the training dataset confirmed the efficiency of the proposed approach. The results of the algorithm adjustment and the selection of the optimal parameters of the neural network showed that the generated artificial data most resemble the initial ones in terms of the criteria of average value, maximum, minimum and dependence between data.

Conclusions. The task of scaling the tabular data of the training dataset using neural networks has been solved. This approach makes it possible to significantly simplify the process of learning neural networks. The scientific novelty of this work lies in the development of approaches and methods for increasing tabular data using artificial intelligence and deep learning.

Keywords: *neural networks, database, tabular data, data augmentation, training dataset, artificial intelligence, deep learning.*

Надійшла у першій редакції 09.07.2023, в останній - 17.08.2023.
The first version has been received on 09.07.2023, the final version - on 17.08.2023.