|  |  |
| --- | --- |
| Добрий день, мене звати Антон Мишенін. Сьогодні я представляю свою дипломну роботу на ступінь бакалавра на тему «Реалізація веб-застосунку для візуалізації багатовимірних даних з застосуванням алгоритму t-SNE». | Мета цієї дипломної роботи полягає в створенні веб-застосунку для візуалізації багатовимірних даних за допомогою алгоритму зниження їх розмірності t-distributed stochastic neighbor embedding (t-SNE). |
| На сам перед, ознайомимось з таким методом дослідження даних, як візуалізація. Цей метод значно спрощує аналіз даних, що можна побачити в нашому наступному прикладі. З лівого богу, ви бачити таблицю цін акцій компанії Motorola Solution Inc. в період з 13 березня по 13 квітня 2018 року. Без сумніву, ця таблиця містить в собі всі необхідні дані, але чи є вона інформативна? Що можна сказати про становище компанії просто дивлячись на цю таблицю? Майже нічого. На противагу їй, на правій стороні слайду ви бачите ті ж самі данні, але представлені у вигляді графіку. | Незважаючи на те, що ціни ніяким чином на ньому не представлені, ми можемо стверджувати, що десь до 20ого березня, в компанії був період відносної стабільності, з 20 березня до 6 квітня – період загального спаду з тимчасовими підйомами, що не перевищують значень в період до 20 березня, з 6 квітня ми бачимо ріст і початок періоду стабільності, однак, ціна не змогла повернутись на до кризисні позначки. |
| Однак не для всіх наборів даних візуалізація настільки тривіальна. В цьому випадку у нас було лише дві ознаки – дата та ціна, з яких з легкістю можна побудувати графік. Але що робити тоді, коли природа описуваних об’єктів набагато складніша, і нам необхідно зобразити більше двох, і навіть, більше трьох ознак? | Для вирішення цього завдання Laurens van der Maaten і Geoffrey Hinton розробили алгоритм t-SNE, що розшифровують як t-distributed stochastic neighbor embedding. |
| t-SNE – це алгоритм машинного навчання, призначений для зниження розмірності багатовимірних даних з метою їх подальшої візуалізації. Він складається з двох етапів, етап І ­– побудувати розподіл ймовірностей для кожної пари багатовимірних об’єктів, таким чином, що схожі об’єкти мають високу вірогідність бути згрупованими, а несхожі – малу. На другому етапі, будують подібний розподіл ймовірностей на відповідній низько вимірній мапі, а між самими розподілами мінімізується Відстань Кульбака — Лейблера, враховуючи місцезнаходження точок. | Зважаючи на складність алгоритму, найкращій спосіб його пояснити – це пояснити на прикладі. Припустимо, маємо двовимірний простір, тобто площину, на якій зображені точки. Кожна точка являє собою сукупність двох величин, так само як в прикладі з цінами на акції Motorola Solution Inc. Наша задача – перевести двовимірні точки в одновимірні, тобто точку на прямій. |
| Як я вже говорив раніше, на першому етапі треба побудувати розподіл ймовірностей для кожної пари багатовимірних об’єктів, для цього:  визначимо точку, в якій ми зацікавлені\*  визначимо відстань між точкою, якою ми зацікавлені, і всіма іншими\*  підставити відстань в якості аргументу функції нормального розподілу Гауса, отримані значення прийняти за непромаштабований коефіцієнт схожості\*  промаштабувати ці коефіцієнти, як відношення непромаштабованого коефіцієнта схожості для пари точок і суми всіх коефіцієнтів схожості для точки зацікавлення. | Ці дії повторяти доти, доки всі точки не побувають у ролі точки зацікавлення. З отриманих чисел будуємо матрицю схожості, схожість точки самої на себе приймаємо як 0. На цьому ми завершуємо перший етап. |
| На другому етапі, будуємо карту на низькому вимірі з випадково розставленими точками. Так само як і минулого разу формуємо матрицю схожостей, але використовуємо не нормальний розподіл Гауса, а т-розподіл Стюарда. | Далі ми пересуваємо точки на прямій і відповідні комірки в другій матриці таким чином, зоб перша і друга матриці були тотожні. Звичайно, це дещо спрощена інтерпретація алгоритму, але вона максимально розкриває суть того, що відбувається. |
| Також алгоритм t-SNE знайшов своє застосування в різних системах штучного інтелекту. Це пов’язано з великою кількістю проблем під час роботи системи штучного інтелекту з багатовимірними даними: від збільшення навантаження на ресурси комп’ютера, зокрема пам’ять та процесор, до звичайних збоїв в роботі алгоритмів, що пристосовані до дво- чи тривимірних просторів. Цей комплекс проблем називають прокляттям багатовимірності, або англійською сurse of dimensionality. | В кожній галузі обробки та аналізу даних, ця проблема представлена своїм явищем.  В алгоритмах побудованих з використанням принципів комбінаторики, відбувається те, що називають комбінаторним вибухом. Зі зростанням кількості вимірів, зростає часова складність алгоритму, оскільки кількість комбінацій даних зростає екпоненційно.  При обчисленні евклідової відстані між точками, можна помітити, що при збільшенні розмірності простору, збільшується відстань між точками, що може призвести до її виходу з допустимих числових меж. |
| Про всі вияви прокляття багатовимірності можна говорити безкінечно, але як я говорив раніше, всі вони мають загальну особливість, всі вони пов’язані зі збільшенням навантаження на ресурси комп’ютера, зокрема пам’ять та процесор, звісно, через збільшення кількості даних, а також звичайними збоями в роботі алгоритмів, що пристосовані до дво- чи тривимірних просторів. | Оскільки алгоритм t-SNE призначений для обробки даних, настав час поговорити про ту множину даних, розмірність якої ми будемо зменшувати. Як таку, ми обрали відомості про погоду для певної місцевості у заданий період. До ознак цієї множини відносяться наступне:   * температура * швидкість вітру * напрям вітру у градусах * вологість * видимість * тиск * хмарність   Разом з часом окремого спостереження, отримуємо набір 8-вимірних даних. |
| Ми отримаємо ці дані з так званого історичного API, наданий компанією World Weather API в режимі пробної версії. Це дозволяє нам безкоштовно робити запити до сервісу і отримувати відомості, однак пробна версія має декілька обмежень, в основному пов’язаних з кількістю запитів на добу. | Зрузумівши, які дані ми оброблюємо, ми можемо продовжити ознайомлення з програмою, яку я назвав «Кліматичний аналізатор». На слайді перед вами скріншот програмного інтерфейсу. Він складається з трьох елементів: полів вводу дат, мапи, на якій можна вибрати місцевість, і кнопки пуск, що починає обчислення.  Користувач обирає місце на гугл-мапі, при необхідності, змінюючи масштаб мапи та / або переміщуючи її. Потім за допомогою полів вводу дат, він обирає періоди, що його цікавлять і користувач натискає кнопку «Пуск». |
| Після запуску обчислень, користувач переходить на іншу сторінку з їх графічним відображенням. В нашому випадку, ми бачимо результати обчислень з 26 березня по 28 березня 2017 і 2018 року відповідно з Берліна. Обидві картинки схожі, це значить, що зміни клімату для цього періоду в Берліні не значні. | Розглянемо архутектуру програми. Як і кожен клієнт-серверний застосунок, Кліматичний аналізатор складається з двох частин – бекенд і фронтенд. Фронтенд написано за допомогою бібліотек React/Redux та інших з цієї екосистеми. В якості хостінга, я використовую AWS S3 Bucket з відповідними налаштуваннями.  Бекенд представляє собою сукупність трьох AWS Lambda функцій, котрі викликають запити до відповідної API адреси. Таким чином, особливістю цієї архутектури є її безсерверність, що дозволяє не дбати про налаштування серверу з однієї сторони, і робить всю конструкцію безкоштовною з іншої. На жаль, подібна архутектура має свої обмеження: |
| * через сладність обчислень, кожен з періодів не може перевищувати 14 днів * через те, що я використовую безкоштовну версію World Weather API, в день дозволено не більше 500, при цьому кожен період рахують к окремий запит * через помилки в роботі World Weather API, перший запит в день повертає помилку | Але кожний недолік, ще можна назвати напрямком розвитку, як такі ми виділили:   * впровадження автоматичних тестів та системи continuous integration, що допоможе в подальшій розробці, коли проект буде рости. Для написання тестів буде використано фреймворк Cucumber.js для бекенду і Protractor / Selenium WebDriver для фронтенду. Continuous Integration буде реалізована за допомогою сервісу CircleCI. |
| * запровадити систему «замовлення результатів», це означає, що користувач повинен не миттєво отримувати результат, а лише посилання на цей результат, коли обчислення будуть завершені. Такий підхід пов’язаний з тим, що максимальний час очікування відповіді від AWS Lambda складає 30 секунд, що є недостатнім для об’єму даних більш ніж 14днів в одному періоді. * запровадити аналіз даних за допомогою систем штучного інтелекту * покращити фронтенд, перейти до Angular 5 | Метою дипломної роботи було створення веб-застосунку, що дозволить користувачу візуалізувати введені ним багатовимірні дані. Щоб досягти цю мету, ми визначили наступні завдання дипломної роботи:   1. Ознайомитись з методиками візуалізації багатовимірних даних, зокрема алгоритмом T-SNE. 2. Визначити досліджувану множину даних. 3. Розробити архітектуру застосунку. 4. Обрати технології, за допомогою яких буде реалізовано застосунок. 5. Реалізувати застосунок.   Кожне з яких було нам зроблено в ході виконання дипломної роботи, а отже і мета загалом була досягнута. |