Практична робота 2

Фільтрація спаму на основі наївного байєсівського класифікатора. Використання метода knn.

Байєсова фільтрація спаму - метод для фільтрації спаму, заснований на застосуванні наївного байєсівського класифікатора, в основі якого лежить застосування теореми Байєса. При навчанні фільтра для кожного зустрінутого в листах слова вираховується і зберігається його «вага» - оцінка ймовірності того, що лист з цим словом - спам. У найпростішому випадку в якості оцінки використовується частота: «появ в спамі / появ у тексті».

У більш складних випадках можлива попередня обробка тексту: приведення слів в початкову форму, видалення службових слів, обчислення «ваги» для цілих фраз, транслітерація та інше. При перевірці знову прийшов листи ймовірність «спамових» обчислюється за вказаною вище формулою для безлічі гіпотез. В даному випадку «гіпотези» - це слова, і для кожного слова «достовірність гіпотези»

$$P(A_i) = N_{word_i}/N_{words\ total}$$

- частка цього слова в листі, а «залежність події від гіпотези» $P(B|A_i)$ - обчислена раніше «вага» слова. Тобто «вага» листа в даному випадку - усереднена «вага» всіх його слів. Віднесення листи до «спаму» чи «не-спаму» проводиться за тим, чи перевищує його «вага» якусь планку, задану користувачем (зазвичай беруть 60-80%). Після прийняття рішення по листу в базі даних оновлюються «ваги» для ввійшли в нього слів.

Математичні основи

Байєсовські фільтри грунтуються на теоремі Байеса, яка буде використовуватись кілька разів в контексті спаму: в перший раз для обчислення ймовірності, що повідомлення - спам, знаючи, що дане слово з'являється в цьому повідомленні; вдруге - щоб обчислити ймовірність, що повідомлення - спам, враховуючи всі його слова (або відповідні їх підмножини); іноді тричі - коли зустрічаються повідомлення з рідкісними словами.

Формула для визначення належності листа до спаму, отримана з теореми Байеса і формули повної ймовірності:

$$\Pr(S \mid W) = \frac{\Pr(W \mid S) \cdot \Pr(S)}{\Pr(W)} = \frac{\Pr(W \mid S) \cdot \Pr(S)}{\Pr(W \mid S) \cdot \Pr(S) + \Pr(W \mid H) \cdot \Pr(H)}$$

$\Pr(S \mid W)$ - умовна ймовірність того, що повідомлення – спам, за умови, що ключове слово знаходиться в ньому;

$\Pr(S)$ - повна ймовірність того, що довільне повідомлення – спам;

$\Pr(W \mid S)$ - умовна ймовірність того, що слово ключове слово з'являється в повідомленнях, якщо вони є спамом;

$\Pr(H)$ - повна ймовірність того, що довільне повідомлення не спам;

$\Pr(W \mid H)$ - умовна ймовірність того, що слово ключове слово з'являється в повідомленнях, якщо вони є не є спамом.

Задача – реалізувати фільтр спаму на основі класифікатора Байєса з використанням датасету із списку матеріалів (або будь-якого зручного для вас, або навіть на прикладі власної поштової скриньки).

Допоміжні матеріали:

1. https://habr.com/ru/post/415963/ – приклад реалізації на R
2. https://randerson112358.medium.com/email-spam-detection-using-python-machine-learning-abe38c889855 - Email Spam Detection Using Python & Machine Learning, приклад реалізації на мові Python (зверніть увагу на посилання в кінці статті, там наведені інші приклади реалізації). інші приклади фільтрів спаму на основі класифікатора Байєса:
3. https://www.kaggle.com/astandrik/simple-spam-filter-using-naive-bayes
4. http://oaji.net/articles/2017/3603-1524458638.pdf - Naive Bayes Spam detection
5. https://towardsdatascience.com/spam-detection-in-emails-de0398ea3b48

   Датасет пошта+спам: https://archive.ics.uci.edu/ml/datasets/spambase

Метод k-найближчих сусідів (KNN) - це один з найпростіших, хоч і точних алгоритмів машинного навчання. KNN - це непараметричний алгоритм, який не робить жодних припущень щодо структури даних, модель не потребує навчання, весь набір тренувань зберігається.

KNN можна використовувати як для класифікації, так і для регресії. В обох випадках прогнозування базується на екземплярах навчального процесу, найближчих до екземпляру вводу. У регресійній задачі вихідним буде значення властивості, яке, як правило, є середнім значенням k найближчих сусідів

Для пошуку найближчих сусідів використовуються різні методи вимірювання відстані. Популярні включають відстань Геммінга, Манхеттенську метрику, відстань Мінковського:

відстань Геммінга: $d_{ij} = \sum_{k=1}^{p} |x_{ik} - x_{jk}|$

Манхеттенська метріка: $d_1(p,q) = ||p-q||_1 = \sum_{i=1}^{n} |p_i - q_i|$

відстань Мінковського $= \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$

Найбільш використовуваним методом для безперервних змінних є, як правило, Евклідова відстань, яка визначається за формулами нижче:

$$\text{Евклідова відстань} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Евклідова відстань є доцільною для проблем, яка включає ознаки одного типу. Для задач класифікації вихід також може бути представлений як набір

імовірностей екземпляра, що належить класу. Наприклад, для двійкових задач вірогідність може бути розрахована як $P(0) = \frac{N_0}{N_0+N_1}$, де P (0) - це ймовірність членства класу 0, N_0, N_1- числа сусідів, що відносяться до класів 0 і 1, відповідно. Значення k відіграє вирішальну роль у точності прогнозування алгоритму. Проте, вибір значення k є нетривіальною задачею. Менші значення k, швидше за все, призведуть до зниження точності, особливо в наборах даних із великим шумом, оскільки кожен примірник набору тренувань тепер має більшу вагу в процесі прийняття рішення. Більші величини k знижують продуктивність алгоритму. На додаток до цього, якщо значення занадто високе, модель може перенавчитись, що зробить межі класу менш чіткими і знову призводить до зниження точності. Як загальний підхід рекомендується вибрати k, використовуючи формулу нижче:

$$k = \sqrt{n}$$

Для класифікації завдань з рівною кількістю класів рекомендується вибрати непарну k, оскільки це призведе до виключення можливості встановлення нічиєї під час підрахунку більшості голосів. Недоліком алгоритму KNN є погана продуктивність на нерівномірно розподілених наборі даних.

Задача – реалізувати класифікатор спамуіз використанням датасету із списку матеріалів (або будь-якого зручного для вас, або навіть на прикладі власної поштової скриньки) на основі методу knn.

Зауваження: так як важко підібрати датасет потрібного вигляду, можна згенерувати його самостійно.

Допоміжні матеріали:

1. https://habr.com/ru/post/149693/ - опис методу
2. http://datascientist.one/k-nearest-neighbors-algorithm/ - опис методу + в кінці є приклад на мові R та інші приклади реалізації
3. https://tproger.ru/translations/top-10-data-mining-algorithms/ - опис методу і приклади на Python та R

окрім цього:

4. https://habr.com/ru/company/iticapital/blog/262155/ - опис поширених методів класифікації, деякі будуть потрібні пізніше
5. https://tproger.ru/translations/top-machine-learning-algorithms/ - так само
6. https://tema.spbstu.ru/userfiles/files/courses/2018-machine-learning/Machine_Learning_LTU_2.pdf - Байєс + knn з поясненням та прикладами.