Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

NLP система для автоматичного рейтингування за важливістю – разом з Fuzzy Logic

Текстова частина до курсової роботи

за спеціальністю „Прикладна математика” - 113

Керівник курсової роботи
к.т.н., доц.
Гороховський К.С.

_____

(підпис)

“____” _____ 2021 р.

Виконав студент факультету інформатики

Спеціальності «Прикладна математика» - 3 курс

Кольчик М.О.

“____” _____ 2021 р.

Київ 2021

Зміст

Анотація.....2
Вступ.....3
Розділ 1. Deep Learning.....4
1.1 Перші згадки про нейронні мережі.....4
1.2 Вступ в нейронні мережі.....4
1.3 Нейрон.....5
1.4 Навчання нейронної мережі.....6
1.5 Рекурентні нейронні мережі (Recurrent Neural Networks - RNN).....6
1.6 LSTM.....7
Розділ 2. Нечітка Логіка (Fuzzy Logic).....10
2.1 Вступ до Fuzzy Logic.....10
2.2 Нечіткі множини.....11
2.3 Функція належності (Membership function).....11
2.4 База правил і Нечіткий вивід (Fuzzy rules / Fuzzy inference).....13
2.5 Алгоритм нечіткого виводу Мамдані.....14
2.6 Методи приведення до чіткості (defuzzification).....15
Розділ 3. Створення метрики «Важливість» для новин фінансового ринку.16
3.1 З чого береться «важливість».....16
3.2 Як саме Fuzzy Logic покращує оцінку настроїв.....17
3.3 Як порахувати «важливість».....19
Розділ 4. Реалізація програми на мові Python.....20
4.1 Пояснення з кодом.....20
4.2 Експеримент: як впливає новина на ціну акцій.....22
Висновок.....24
Список літератури.....25

Анотація

У даній роботі розглядається програма для сортування новин за важливістю за допомогою нечіткої логіки і нейронних мереж. Розроблена архітектура програми, визначені основні недоліки і переваги. Проведений аналіз ефективності програми. Розглянуто її застосування на цінах акцій. Програма була написана на Python3.

Вступ

В останні роки, був зроблений великий прогрес в обробці природної мови. Особливо, більшість State-of-the-art рішень стосуються машинного навчання з вчителем. Також було зроблено нестандартні кроки в напрямі Sentiment analysis, тобто визначення тональності речень. Один із таких експериментів – поєднання натренованих моделей машинного навчанні із fuzzy logic – нечіткою логікою. За допомогою такого поєднання дослідникам вдалося покращити точність оцінок, бо саме нечітка логіка вміє добре працювати з розпливчатими даними. Я вирішив також проекспериментувати з такою комбінацією і поєднав нейронні мережі з нечіткою логікою для обчислення «важливості» для новин фінансового ринку. Одразу варто зазначити, що така модель не претендує на об’єктивність, бо по-перше, важливість новин для кожної людини є суб’єктивною і по-друге, це всього лиш математична модель, яка теж може помилятися.

Така модель буде актуальна для аналітиків фінансових ринків, адже для швидкого фундаментального аналізу аналітики будуть менше часу витрачати, шукаючи найважливіші новини.

Тому, метою роботи є створення математичної моделі для рейтингування новин фінансового ринку за допомогою Fuzzy Logic і нейронних мереж.

Розділ 1. Нейронні мережі

1.1 Перші згадки про нейронні мережі

Вперше за авторством У. Маккалока і У. Піттса в 1943 році була випущена робота, у якій було формалізовано поняття нейронної мережі.

В 1958 Році Френком Розенблатом було створено нейрокомп'ютер – пристрій, який зміг вирішувати задачі класифікації.

1986 рік – з'явилися рекурентні нейронні мережі, які наразі використовуються в машинному перекладі , обробці відео і розпізнаванні мови.

1989 Ян Лекун запропонував згорткову нейронну мережу, яка здатна класифікувати зображення.

1.2 Вступ в нейронні мережі

Намагаючись створити систему, яка вчиться подібно людині, вчені надихнулися саме структурою людського мозку на створення нейронних мереж. Саме тому багато базової термінології було взято з біологічних наук.

Подібно нейронам, з яких складається людський мозок, архітектура нейронних мереж теж містить обчислювальну одиницю, яка називається перцептрон. Нейрон передає електричні імпульси через нервову систему, а перцептрон отримує на вхід деякі сигнали і перетворює їх на вихідні сигнали.

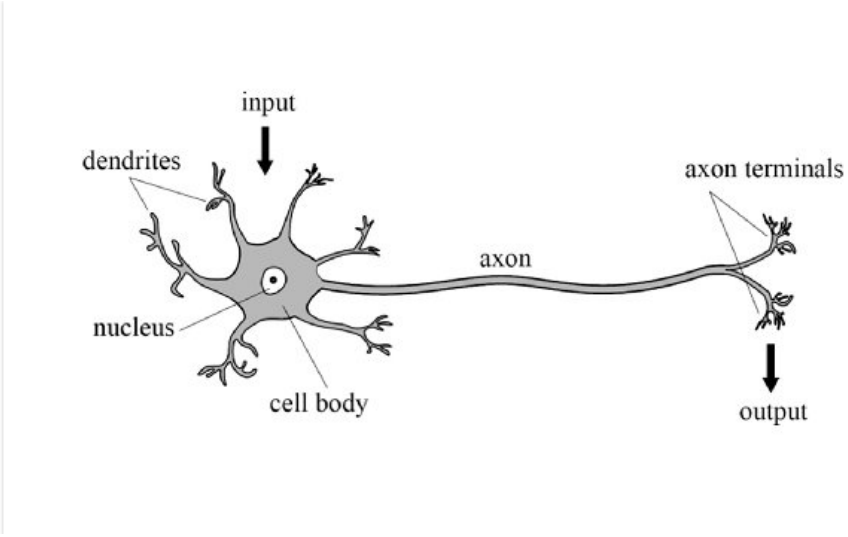

Рис. 1.1 Структура нейрону головного мозку

Нейронна мережа складається з шарів, які є набором перцептронів, кожен з яких виконує просту обчислювальну функцію. Вона намагається імітувати

людський мозок для того, щоб знайти закономірності у вхідних даних, використовуючи техніки математичного аналізу і лінійної алгебри.

Нейронна мережа складається з таких шарів:

1- Вхідний шар (Input Layer)

2- Прихований шар (Hidden Layer)

3- Вихідний шар (Output Layer)

Кожен попередній шар з'єднаний з наступним за допомогою спеціальних зв'язків, які називаються вагами (weights). Вхідний шар це вхідні дані, які подаються до мережі. Прихований шар розглядає комбінації вхідного шару і вирішує, яка з комбінацій важлива, придаючи їй деяке значення важливості за допомогою вищезгаданих вагів.

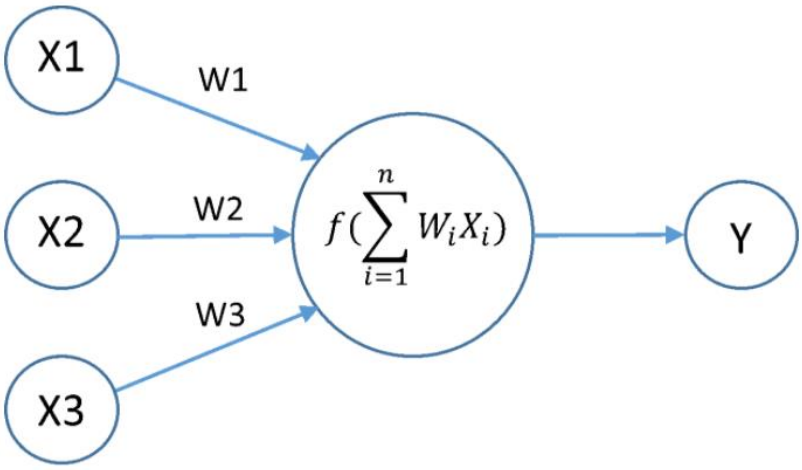

Рис. 1.2 Структура нейронної мережі

1.3 Нейрон

Кожен нейрон має алгоритм обчислення вихідного значення. Розіб'ємо його на кілька кроків:

Першим кроком рахується зважена сума вхідних значень, тобто, маючи n вхідних елементів, маємо:

$$\sum_{i=1}^{n} W_i X_i + b$$

b- зміщення

Другий крок – порахована сума проходить через функцію активації. Функція активації дозволяє нам перетворювати вихідні дані у бажаний нам формат. Наприклад, якщо ми хочемо, щоб мережа давала на виході імовірності, то можемо використати сигмоїдальну функцію, яка видає значення в проміжку [0,1]. Тобто, обертаючи в суму в функцію активації, маємо:

$$f(\sum_{i=1}^{n} W_i X_i)$$

1.4 Навчання нейронної мережі

Процес передавання вхідних значень в нейромережу і отримання з неї вихідних даних називається пряме поширення (forward propagation). Після того, як він завершився, вихідний шар порівнює результат з істинними значенням і адаптує ваги, вважаючи на різницю між порахованим значенням та істинним. Процес, який це робить, називається метод зворотнього поширення помилки (backpropagation). Його суть полягає в тому, що за допомогою функції втрат (loss function) ми обчислюємо помилку, тобто порівнюємо прогнозоване значення з істинним і після цього рахуємо похідну від значення помилки по кожній вазі окремо. Процес зведення вагів до оптимуму називається градієнтним спуском.

1.5 Рекурентні нейронні мережі (Recurrent Neural Networks - RNN)

РНН використовуються для аналізу послідовних даних, таких як часові ряди, тексти, аудіо та відеоряди. Більшість традиційних проблем машинного навчання припускають, що минулі вхідні значення є незалежними і не пов’язані з теперішніми вхідними значеннями. Але якщо подивитись на часові ряди, тексти, аудіо та відеоряди, то можна побачити, що змінні пов’язані з минулими значеннями. Наприклад, наступне слово пов’язане з попереднім або наступне значення ціни криптовалюти пов’язане з попередніми. РНН дають нам змогу для обробки таких послідовностей і обчислення передбачень. Отже, РНН мають деяку “пам’ять” , яка враховує попередню інформацію. РНН мають таку структуру:

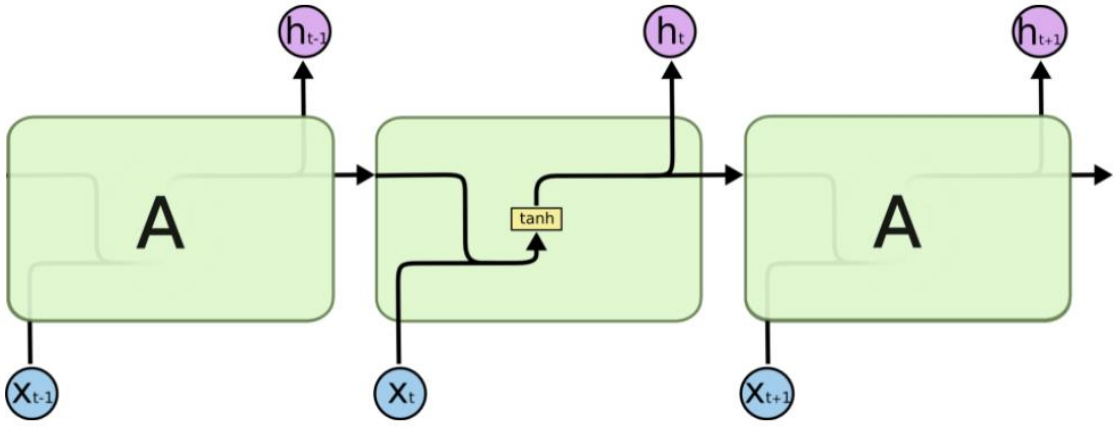

Рис. 1.3 Структура РНН

На рисунку 1.3 можемо побачити розгортку РНН, і ми можемо виявити, що інформація в кожному моменті передається в наступний момент. X_t – вхідне значення для даного моменту часу, A – комірка нейронної мережі, h_t – вихідне значення, яке разом з X_t слугує вхідним значенням для X_{t+1} . Але,

працюючи з великими проміжками часу, РНН можуть втрачати здатність зв’язувати інформацію і страждають від проблеми градієнтного затухання.

1.6 LSTM

Для вирішення проблеми градієнтного затухання була створена мережа LSTM (Long Short Term Memory), яка фактично побудована на базисі РНН, але містить в собі деякі переваги. Одна з таких переваг є те, що замість одного шару нейронної мережі вона має чотири, які взаємодіють між собою особливим чином.[5]

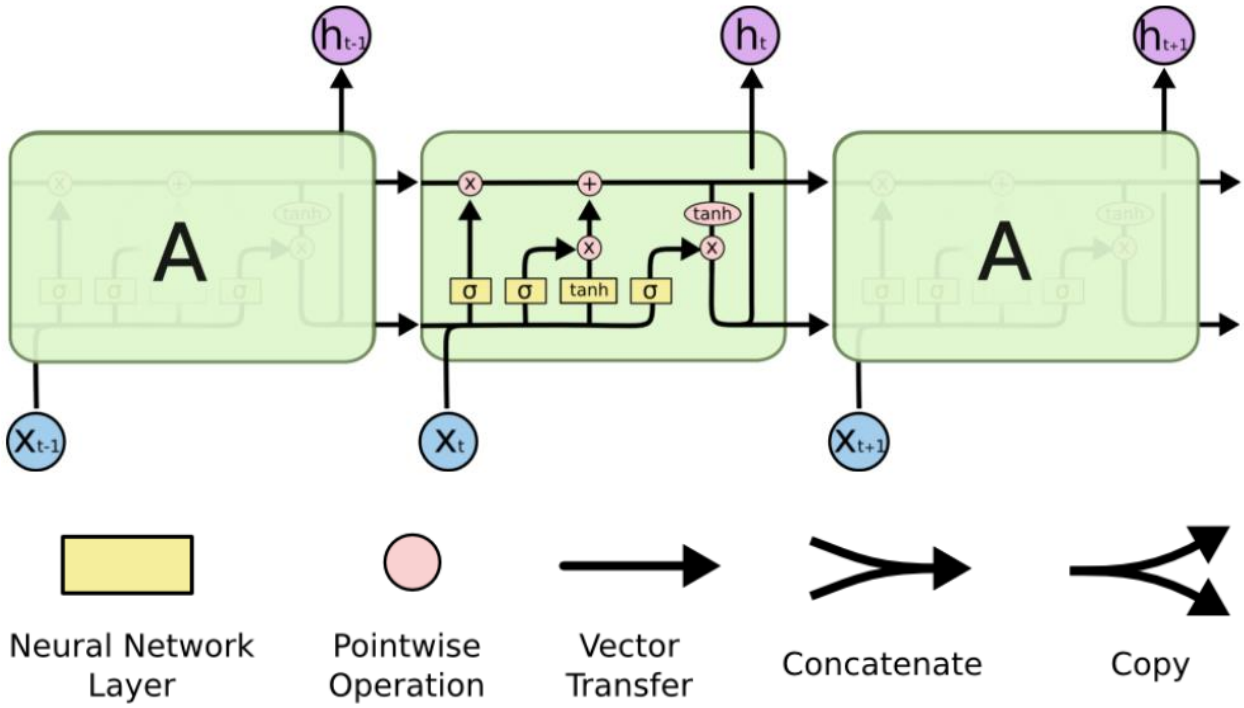

Рис. 1.4 Структура LSTM

Перший шар – сигмоїдальний, який називається *forget gate*. Він визначає, яку інформацію можна викинути із стану комірки. Шар дивиться на h_t і X_t і якщо шар повертає 0, це означає повністю викинути інформацію, а 1 – повністю залишити. І так для кожного стану C_{t-1} .

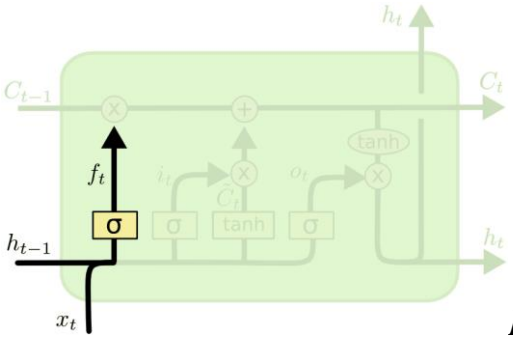

Рис 1.5 forget gate

Далі, щоб оновити значення комірки, ми маємо шар *input gate*. Спочатку вхідні дані передаються в сигмоїдальний шар, який вирішує, які дані необхідно оновити, перетворюючи значення від 0 до 1.

Потім прихований стан передається в *input modulation gate*, де будується вектор нових значень за допомогою функції *tanh*, які можуть буди додані в новий стан.

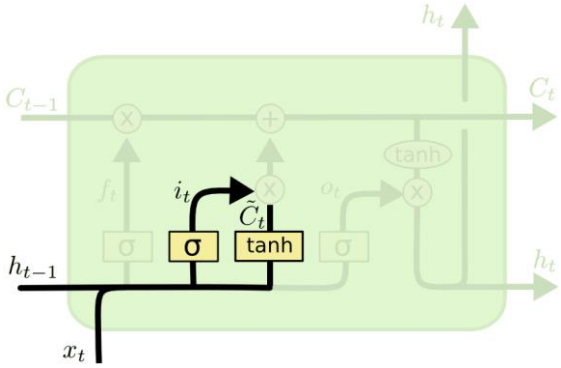

Рис 1.6 input and modulation gate

Нарешті, останній шар *output gate* вирішує, яким повинен бути наступний стан. Спочатку значення проходять через сигмоїдальний шар, який рахує, яка інформація буде виводитися, а потім стани проходять через tanh шар, щоб отримати значення в проміжку [-1,1]. І в кінці перемножитися з отриманими значеннями з проміжку [0,1].

На відміну від звичайних RNN, де присутній тільки *hidden state*, у LSTM також присутній *cell state,* який грає роль довгострокової пам’яті. Вона створена для того, щоб саме вирішувати проблему зникаючого градієнта. *Cell state* модифікується *forget* гейтом, розташованим нижче *cell state* і також регулюється *input modulation gate*:

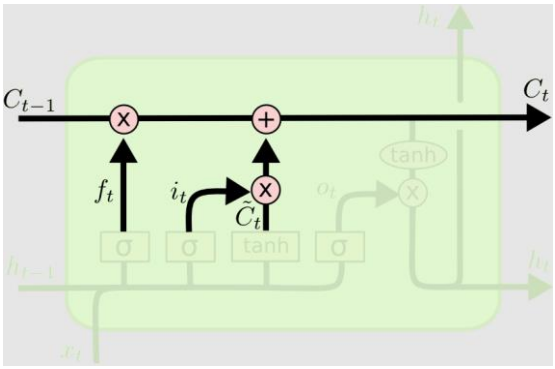

Рис 1.6 cell state

Отже, для обчислення наступного прихованого стану і наступного вихідного значення, використовуються такі формули:

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

Де f_t – значення після проходження forget gate,

I_t – значення після проходження *input layer gate*

o_t – значення після проходження output gate

c_t – новий стан

h_t – нове вихідне значення

Розділ 2. Нечітка логіка (Fuzzy Logic)

2.1 Вступ до Fuzzy Logic

В 1965 році в журналі «Information and Control» була опублікована стаття “Fuzzy sets” за авторства Л. Заде. Мотивом для написання статті, в якій описуються нечіткі множини, стала необхідність опису таких понять, які мають неточний характер. Математичні методи, які використовували звичайну логіку, не дозволяли вирішувати проблеми нечітких типів.

Коли дані неточні, наприклад, повні наближень або просто неможливо застосувати звичайну логіку, неточні (*soft*)обчислення є найкращим підходом. Наприклад, під час водіння автомобіля, ми маємо комбінацію двох подій: натискання педалі газу і педалі гальма. Якщо казати про самокеровані автомобілі, обидві події повинні керуватися одночасно і без втручання людини. Розглядаючи звичайну логіку, ми знаємо, що вона керується звичайними множинами (*crisp sets*). В таких множинах значення дорівнюють 0 або 1. Тоді функціонування автомобіля виглядатиме таким чином:

Дія	Результат
Натиснути газ	1
Натиснути гальма	1
Відпустити газ	0
Відпустити гальма	0

Але маємо проблему: 1 означає повністю натиснути педаль, а 0 – повністю відпустити, і немає проміжного значення.

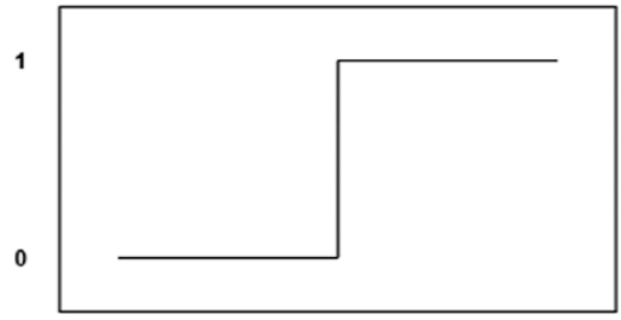

Рис. 2.1 Дії автомобіля в розрізі звичайної логіки

1 – повне натискання педалі

0 – повне відпускання педалі

2.2 Нечіткі множини

Нечіткою множиною A в деякому непорожньому просторі X, така, що $A \subseteq X$, називається множина пар

$$A = \{(x, \mu_A(x); x \in X)\}, \text{де}$$

$$\mu_A: X \to [0,1]$$

-функція належності нечіткої множини А. Ця функція належності надає кожному елементу $x \in X$ степінь його належності до нечіткої множини A, тоді:

1) $\mu_A(x) = 1$ - повна належність елементу x до нечіткої множини A

2) $\mu_A(x) = 0$ - повна відсутність належності елементу x до нечіткої множини A

3) $0 < \mu_A(x) < 1$ - часткова належність елементу x до нечіткої множини A

Нечітка множина записується у вигляді

$$A = \frac{\mu_A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \cdots + \frac{\mu_A(x_n)}{x_n}\{(x, \mu_A(x); x \in X)\}, \text{ де}$$

Риска означає не «поділити», а надання елементам $x_1 \ldots x_n$ степені належності. [7,46-47]

2.3 Функція належності (Membership function)

Замість того, щоб представляти значення 0 або 1, можна кожен елемент множини подати як значення з проміжку [0,1]. Кожне значення буде називатися степенем належності до нечіткої множини (degree of membership), і тоді всі значення будуть давати криву, яка називається функцією належності (*Membership function*). Тоді степінь натискання педалі можна зобразити так:

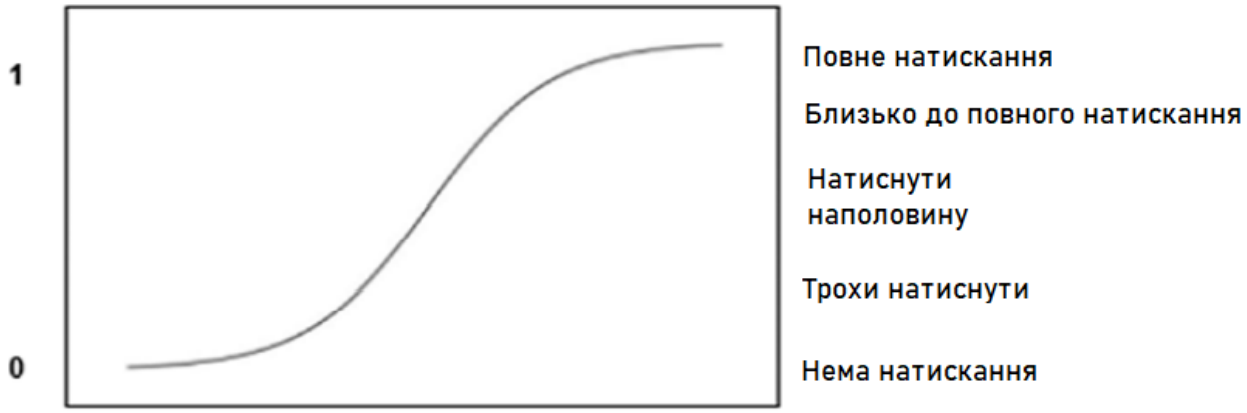

Рис. 2.2 Степінь натискання педалі

Існує декілька найпоширеніших функцій належності.

1) Трикутна функція належності

Має три параметри: a, b, c , де a – нижня межа, b - центр, c- верхня межа.

Трикутна функція належності визначається формулою:

$$f(x;a,b,c) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x \end{cases}$$

І має такий графік:

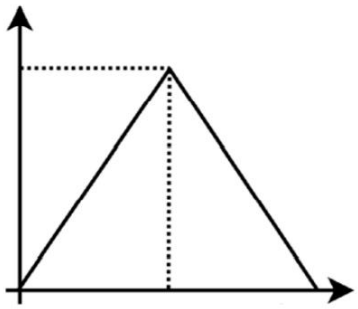

Рис. 2.3 Графік трикутної функції належності

2) Трапецоїдальна функція належності

Має чотири параметри : a, b, c, d , причому $b < c < d$

Формула для визначення Трапецоїдальної функції:

$$f(x;a,b,c,d) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \dfrac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases}$$

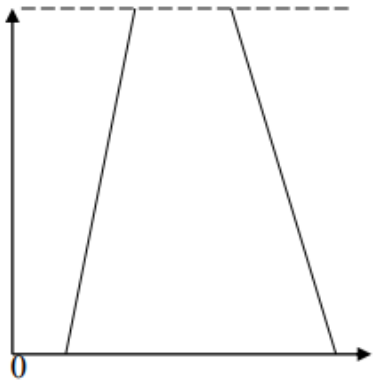

Рис. 2.4 Трапецоїдальна функція належності

3) Гаусова функція належності

Визначається за формулою:

$$\mu_A(x, c, s, m) = e^{\frac{-1}{2} * |\frac{x-c}{s}|^m}$$

c– середнє, s – стандартне відхилення, m – фактор нечіткості

Вона виглядає так:

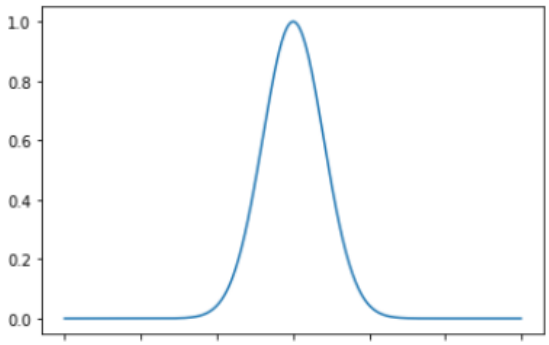

Рис. 2.5 Гаусова функція належності

2.4 База правил і Нечіткий вивід (Fuzzy rules / Fuzzy inference)

Основою для операцій нечіткого виводу є база нечітких правил, які мають форму :

$$\boldsymbol{IF}\ x_1\ is\ A_1\ \boldsymbol{AND}\ x_2\ is\ A_2\ \boldsymbol{AND}\ \dots\ \boldsymbol{AND}\ x_n\ is\ A_n\ \boldsymbol{THEN}\ y\ is\ B_1, \text{де}$$

$$x_{1\dots n} - \text{вхідні змінні}$$

$$y - \text{вихідні змінні}$$

$$\mathrm{A}_{1\dots n}, B - \text{нечіткі множини}$$

База правил має деякі аксіоми

1) Логічний наслідок : $x \in A\ , A \subset B \Rightarrow x \in B$

2) Кон’юнкція : $x \in A, x \in B => x \in A \cap B$

3) Диз’юнкція : $x \in A, x \in B => x \in A \cup B$

4) Заперечення: $not(x \in A) => x \in \neg A$

5) Generalized Modus ponens: $x \in A', \boldsymbol{IF}\ x\ \in A\ \boldsymbol{THEN}\ y\ \in B => y \in B'$

6) Generalized Modus tollens: $y \in B', \boldsymbol{IF}\ x\ \in A\ \boldsymbol{THEN}\ y\ \in B => x \in A'$

Логічний вивід має 4 етапи: приведення до нечіткості, нечіткий вивід, композиція, приведення до чіткості. Схема виглядає так:

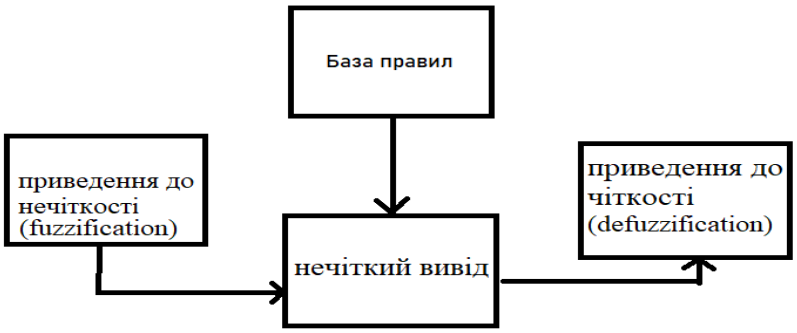

Рис. 2.6 Схема етапів механізму нечіткого виводу

2.5 Алгоритм нечіткого виводу Мамдані

Найпопулярнішим алгоритмом нечіткого виводу є алгоритм Мамдані. В ньому використовується мінімаксна композиція нечітких множин. Нехай маємо m правил з A_{ik} – заданими нечіткими множинами з функціями належності, $i=1..m$, $k=1..n$

1) Fuzzification : визначаємо значення функцій належності для лівих частин кожного правила, де $A_{ik}(x_k)$ – степені істинності

2) Знаходимо рівні відсічення для лівої частини кожного із правил

$$alpha_i = \min(A_{ik}(x_k))$$

3) Знаходимо відсічені функції належності:

$$B_i^*(y) = \min(alpha_i, B_i(y))$$

4) Знаходимо композицію отриманих відсічених функцій, використовуючи максимальне об’єднання нечітких множин

$$MF(y) = max(B_i^*(y))$$

Маємо $MF(y)$ - функція належності кінцевої нечіткої множини.

5) Defuzzification : отримуємо значення з чіткої множини за допомогою методів дефаззифікації, наприклад центроїдного методу.

$$\bar{y} = \frac{\sum y * \mu(y)}{\sum \mu(y)}$$

Геометричний сенс $\bar{y}$- центр тяжіння кривої $MF(y)$.

Переваги такого методу :

1) Він інтуїтивно зрозумілий

2) Добре інтерпретована база правил

3) Широко відомий

Недоліки методу Мамдані:

1) При зростанні кількості змінних у вводі, кількість правил зростає експоненційно

2) При великій кількості змінних важко знайти зв'язок між вводом і виводом

Існують також ще декілька відомих методів виводу, таких, як алгоритми Tsukamoto, Sugeno , Larsen

2.6 Методи приведення до чіткості (defuzzification)

1) Centroid method, [6,103]

$$\hat{y} = \frac{\sum y * \mu(y)}{\sum \mu(y)}$$

2) Weighted Average Method, [6,105]

$$\hat{y} = \frac{\sum \bar{y} * \mu(\bar{y})}{\sum \mu(\bar{y})}$$

3) Center of Sum Method[6,106]

$$\hat{y} = \frac{\int \underline{y} \sum \mu(y)}{\int \sum \mu(y)}$$

$\underline{y}$ – відстань від центроїда до функції належності

Розділ 3. Створення метрики «Важливість» для новин фінансового ринку.

3.1 З чого береться «важливість»

Що таке «важливість»? Очевидно, що це абсолютно суб'єктивна метрика і не існує якогось єдиного критерію, за яким ми можемо визначити «важливість» для всіх людей разом. Тому перед нами стоїть задача – зробити «важливість» максимально незалежною від людських емоцій. Тому я вирішив, чому б не створити власну метрику, яка б визначала «важливість» новин так, щоб вона була найбільш універсальною.

Також, слід зазначити, що вся робота пов'язана не на аналізі самих новин, а на аналізі їх заголовків через обмежені обчислювальні можливості і з метою підвищення швидкості роботи моделі.

Створення такої метрики було розділено на кілька етапів. Перший етап – знайти розмічені дані для заголовків, які б мали хоч якісь додаткові дані про новину, наприклад тип тексту (стаття, звичайне повідомлення і тд.), або релевантність. Другий етап – якимось чином використати ці дані для нових новин за допомогою нейронних мереж. Третій етап – використати Fuzzy Logic в комбінації з новими даними про новини.

Тому, нова метрика «важливості» була створена з двох частин.

Перша частина: на сайті kaggle.com був взятий датасет під назвою "Two Sigma: Using News to Predict Stock Movements". Він має більше 1 млн спостережень і близько 20 змінних. Нас цікавить чотири із них: текст заголовку (headline), тип тексту (urgency), тобто стаття або повідомлення, релевантність (relevance) і речення, в якому вперше було згадано заданий актив (firstmentioned). Потім вони будуть натреновані за допомогою нейронних мереж і будуть використані в якості параметрів для того, щоб порахувати «важливість».

Друга частина - Аналіз настроїв у заголовку за допомогою бібліотеки TextBlob і Fuzzy Logic. Алгоритм аналізу настроїв виглядає так: спочатку завантажуємо заголовки, потім знаходимо 2 параметри: *polarity*- «настрій» заголовків (позитивний, негативний, нейтральний) за допомогою бібліотеки TextBlob – 1 якщо позитивний, 0 якщо нейтральний і -1 якщо негативний. Причому, число може бути будь- яким на проміжку [-1,1], наприклад 0.3 або -0.74, що позначатиме відповідну належність класу «настроїв». Останній крок – застосувати Fuzzy Logic до «настроїв», тим самим покращивши якість належності до класу; *subjectivity* – емоційне забарвлення речення з проміжку [0;1], де 0 – емоційно слабке речення, 1 – емоційно сильно забарвлене.

Отже, з одного боку, ми маємо моделі, які будуть вказувати на тип тексту, його релевантність до активу і речення, в якому актив вперше згаданий, а з іншого боку, матимемо Fuzzy Sentiment оцінки заголовка. З цього і будуватимемо метрику «важливості».

Отже, метрика «важливості» загалом, виглядає таким чином:

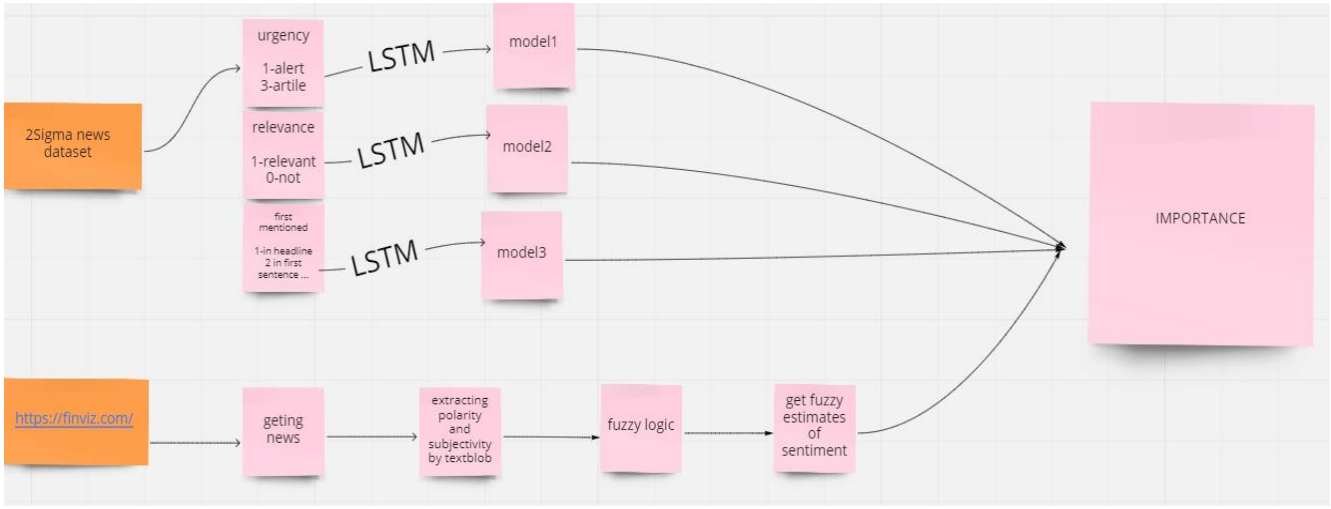

Рис. 3.1 Модель для обчислення важливості

3.2 Як саме Fuzzy Logic покращує оцінку настроїв

Прототип системи перетворення звичайних оцінок в нечіткі був взяти із роботи [1]. Автори аналізували настрої твітів із датасету, в якому кожному твіту було призначено оцінку його настрою. Новація їхнього підходу була в тому, автори створили систему із 9 нечітких правил для обчислення настроїв речення і запропонували unsupervised підхід, який підходить для будь- якого датасету. Також було проведено порівняння із state-of-the-art алгоритмами, які рахують тональність речень. Дані правила були отримані з перетину попарно двох змінних – позитивних і негативних. Кожна змінна має по три нечітких множини і активує тільки одне правило.

Rule	Positive Score	Negative Score	Sentiment
R1	Low	Low	Neutral
R2	Medium	Low	Positive
R3	High	Low	Positive
R4	Low	Medium	Negative
R5	Medium	Medium	Neutral
R6	High	Medium	Positive
R7	Low	High	Negative
R8	Medium	High	Negative
R9	High	High	Neutral

Рис. 3.2 Дев'ять Нечітких правил

Далі ці правила були перетворені у відповідні числа, які відображали силу даного правила:

$$w_{r1} = positive_{low} \wedge negative_{low}$$

$$w_{r2} = positive_{medium} \wedge negative_{low}$$

$$w_{r3} = positive_{high} \wedge negative_{low}$$

$$w_{r4} = positive_{low} \wedge negative_{medium}$$

$$w_{r5} = positive_{medium} \wedge negative_{medium}$$

$$w_{r6} = positive_{high} \wedge negative_{medium}$$

$$w_{r7} = positive_{low} \wedge negative_{high}$$

$$w_{r8} = positive_{medium} \wedge negative_{high}$$

$$w_{r9} = positive_{high} \wedge negative_{high}$$

Змінні positive_low, pos_medium і positive_high складають першу частину нечітких правил, і вони відображають низький, середній та високий нечіткі набори для позитивного балу твіта. Подібним чином negative_low, negative _med та negative _high складають другу частину нечітких правил, і вони відображають низький, середній та високий нечіткі набори для негативного балу твіта.

Складання правил відбувається за такими формулами:

$$w_{negative} = w_{r4} \vee w_{r7} \vee w_{r8}$$

$$w_{neutral} = w_{r1} \vee w_{r5} \vee w_{r9}$$

$$w_{positive} = w_{r2} \vee w_{r3} \vee w_{r6}$$

Де $w_{negative}, w_{neutral,} w_{positive}$- степені виконання нечітких правил певних емоцій.

Далі ми повинні порахувати наслідки функції належності шляхом проекції на початкові функції належності:

$$Negative = \{0,0,5\}$$

$$Neutral = \{0,5,10\}$$

$$Positive = \{5,10,10\}$$

Маємо формули:

$$activation_{low} = w_{negative} \ \wedge \ \ Negative$$

$$activation_{medium} = w_{neutral} \ \wedge \ \ Neutral$$

$$activation_{high} = w_{positive} \ \wedge \ \ Positive$$

Вивід загальної функції належності рахується так:

$$y = \ activation_{low} \ \cup activation_{medium} \ \cup \ activation_{high}$$

Останній крок – дефуззифікація:

$$output = \frac{\sum y * \mu(y)}{\sum \mu(y)}$$

Якщо $0 < output < 3.33$, то речення негативне, якщо $3.34 < output < 6.66$, то речення нейтральне і якщо $6.67 < output < 10$, то речення позитивне.

Автори роботи порівняли даний алгоритм звичайним Sentiment Analysis на різних датасетах і отримали такі результати:

Lexicons	Methods	SemEval 2017[51]		SemEval 2016[42]		SemEval 2015[50]		Gilbert Tweets[16]	
		F1-Micro	F1-Macro	F1-Micro	F1-Macro	F1-Micro	F1-Macro	F1-Micro	F1-Macro
SentiWordNet[4]	Cavalcanti[9]	0.358	0.334	0.436	0.314	0.372	0.309	0.549	0.406
	Ortega[46]	0.473	0.419	0.255	0.253	0.467	0.428	0.363	0.332
	Fuzzy Rules	0.485	0.231	0.326	0.227	0.478	0.221	0.346	0.223
AFINN[44]	Simple SA[44]	0.558	0.515	0.308	0.185	0.618	0.594	0.079	0.073
	Fuzzy Rules	**0.686**	0.308	**0.457**	**0.419**	0.484	0.236	0.44	0.426
VADER[16]	Simple SA[16]	0.528	0.526	0.475	0.428	**0.604**	0.585	1	1
	Fuzzy Rules	0.525	0.381	0.34	0.232	0.524	0.319	**0.865**	**0.772**

Рис. 3.3 Порівняння роботи алгоритмів

За показником F-score в більшості випадках нечітка логіка показала кращі результати.

3.3 Як порахувати «важливість»

Оскільки критерії для оцінки важливості я створював самостійно, то і формулу для важливості я створив самостійно. Ми маємо дві групи оцінок: Перша - чим оцінка вища, тим краще і друга - чим оцінка менше, тим краще. Очевидно, що перша група має стояти в чисельнику, а друга в знаменнику. Тому можемо скласти формулу «важливості» з таких змінних : *urgency, relevance, first_mentioned, fuzzy_sentiment, polarity, subjectivity*, яка виглядатиме так:

$$IMPORTANCE = \frac{e^{FuzzySentiment} * e^{polarity} * e^{subjectivity}}{urgency} + e^{\frac{relevance}{first\_mentioned}}$$

Використовуємо експоненту, щоб позбавитися нулів

Розділ 4. Реалізація програми на мові Python

4.1 Пояснення з кодом

В датасеті two-sigma news є багато колонок, але нас цікавлять тільки декілька з них: ‘headline’, ‘urgency’, ‘relevance’, ‘firstMentionSentence’

headline	urgency	relevance	firstMentionSentence
Seoul antitrust body forms team on Qualcomm-report	3	0.13484	12
Stiller, Smith achieve box office milestones	3	0.204124	21
Korea Hot Stocks-LG Elec, LG.Philips, Banks, KEPCO	3	0.078811	23
TABLE-HK short selling turnover HK$1.18 bln by lunch	3	0.5	5

Рис. 4.1 Обираємо потрібні колонки

Очищуємо заголовки за допомогою функції clean_text

```
def clean_text(text):
    text = re.sub(r'<.*?>', '', text)
    text = re.sub(r"\\", "", text)
    text = re.sub(r"\'", "", text)
    text = re.sub(r"\"", "", text)
    text = text.strip().lower()
    filters='!"\'#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n'
    translate_dict = dict((c, " ") for c in filters)
    translate_map = str.maketrans(translate_dict)
    text = text.translate(translate_map)
    return text

clean_news_bef = []
for new in news:
    clean = clean_text(new)
    clean_news_bef.append(clean)
```

Рис. 4.2 функція clean_text

Наступним кроком буде створення LSTM моделей для кожної змінної за допомогою бібліотеки *keras*, де за змінну *X* буде виступати заголовок, перетворений у вектор чисел, а за *y* – кожна змінна ‘urgency’, ‘relevance’, ‘firstMentionSentence’ почергово.

Спочатку токенізуємо слова у заголовках за допомогою класу Tokenizer :

```
max_words = 15000
max_s_length = 36
EMBEDDING_DIM = 30

tokenizer = Tokenizer(num_words=max_words, filters='!"#$%&()*+,-./:;<=>?@[\]^_`{|}~', lower=True)
tokenizer.fit_on_texts(clean_news_bef)
word_index = tokenizer.word_index
```

Рис. 4.3 Tokenizer

За допомогою pad_sequences перетворюємо токенізовані заголовки в числовий вектор:

```
X = tokenizer.texts_to_sequences(clean_news_bef)
X = pad_sequences(X, maxlen=max_s_length)
```

Рис. 4.4 Перетворення речення у вектор

За допомогою функції train_test_split() розділяємо на тренувальну і тестову вибірки:

```
y1 = dff['urgency']

X_train1,X_test1,y_train1,y_test1 = train_test_split(X, y1)
plt.hist(y_train1)

y2 = dff['relevance']
X_train2,X_test2,y_train2,y_test2 = train_test_split(X, y2)

y3 = dff['firstMentionSentence']

X_train3,X_test3,y_train3,y_test3 = train_test_split(X, y3)
```

Рис. 4.5 Розділюємо датасети для тренування і тестування

Створюємо LSTM моделі

```
from keras.utils import to_categorical
y_train1_categ = to_categorical(y_train1)
model1 = Sequential()
model1.add(Embedding(max_words, EMBEDDING_DIM, input_length=X.shape[1]))
model1.add(SpatialDropout1D(0.2))
model1.add(Bidirectional(GRU(100, dropout=0.2, recurrent_dropout=0.2)))
model1.add(Dense(4, activation='softmax'))
model1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])


history = model1.fit(X_train1, y_train1_categ, epochs=1, batch_size=256,validation_split=0.2,
                     callbacks=[EarlyStopping(monitor='val_accuracy', patience=3, min_delta=0.0001,restore_best_weights=True)])

y_pred1 = model1.predict(X_test1)
model1.save('model1.h5')
model1 = load_model('model1.h5')
```

```
model2 = Sequential()
model2.add(Embedding(max_words, EMBEDDING_DIM, input_length=X.shape[1]))
model2.add(SpatialDropout1D(0.2))
model2.add(GRU(100, dropout=0.2, recurrent_dropout=0.2))
model2.add(Dense(25, activation='relu'))
model2.add(Dense(1))
model2.compile(loss='mse', optimizer='adam', metrics=['mse'])
history = model2.fit(X_train2, y_train2, epochs=2, batch_size=64,validation_split=0.2,
                     callbacks=[EarlyStopping(monitor='val_loss', patience=3, min_delta=0.0001,restore_best_weights=True)])
y_pred2 = model2.predict(X_test2)
model2 = load_model('model2.h5')
model2.save('model2.h5')
```

```
model3 = Sequential()
model3.add(Embedding(max_words, EMBEDDING_DIM, input_length=X.shape[1]))
model3.add(SpatialDropout1D(0.2))
model3.add(GRU(100, dropout=0.2, recurrent_dropout=0.2))
model3.add(Dense(25, activation='relu'))
model3.add(Dense(1))
model3.compile(loss='mse', optimizer='adam', metrics=['mse'])
history = model3.fit(X_train3, np.array(y_train3), epochs=1, batch_size=256,validation_split=0.3,
                     callbacks=[EarlyStopping(monitor='val_loss', patience=3, min_delta=0.0001,restore_best_weights=True)])
y_pred3 = model3.predict(X_test3)
model3 = load_model('model3.h5')
model.save('model3.h5')
```

Рис. 4.6 LSTM моделі

Accuracy в першій моделі вийшло близько 95%, тобто з такою імовірністю модель вгадає правильно чи якийсь заголовок є статтею чи попередженням.

Наступним кроком буде завантаження заголовків із сайту finviz.com. В результаті матимемо такий датафрейм:

Index	ticker	date	time	headline
0	MSFT	May-06-21	09:00AM	New study shows digital preparedness helped organizations adapt to COVID-19
1	MSFT	May-06-21	04:46AM	Nintendo Warns Chip Crunch May Hit Switch During Gaming Boom
2	MSFT	May-06-21	04:00AM	Microsoft to allow EU customers to process, store data in the region
3	MSFT	May-06-21	04:00AM	Apples App Store Manager to Take Center Stage at Epic Trial

Рис. 4.7 Новини з сайту FinViz.com

Далі токенізуємо заголовки і вставляємо їх в наші моделі, щоб порахувати ‘urgency’, ‘relevance’, ‘firstMentionSentence’. Одна третя частина роботи готова.

Друга треть роботи полягає в тому, щоб ми порахували параметри polarity і subjectivity для кожного заголовка, щоб зрозуміти його настрій, тобто емоційне забарвлення і суб'єктивність.

headline	urg	rel	sentiment	firstment	polarity	subjectivity
Apples App Store Manager to Take Center Stage at Epic Trial	3	0.7709199	neut	4.0370407	0	0.25
Day 3 of Apple-Epic trial: Developers conti…	1	0.9895488	pos	1.9321595	0.1	0.4
Jim Cramer: FAANG's Still Got Teeth	1	0.8093996	neut	2.9418921	0	0
Dow Jones Rallies As Janet Yellen Backs Off…	1	0.90815085	neut	1.9648273	0	0

Рис. 4.8 Рахуємо полярність і суб'єктивність

Передостанній крок для побудови метрики «важливості» - перетворити polarity в «нечіткі» оцінки. Для цього ми використовуємо ідею з частини 3.2. Кожен заголовок має нечітку оцінку *estimate*, яка $0<estimate<3.33$ позначатиме негативне речення, $3.34<estimate<6.66$ позначатиме нейтральне речення, $6.67<estimate<10$ позначатиме позитивне речення. Але нас хвилює те, наскільки оцінка відрізняється від нейтральної оцінки, тому візьмемо

$$estimate = |5 - estimate|$$

headline	urg	rel	sentiment	firstment	polarity	subjectivity	FuzzySentiment
New study shows digital preparedness helped organizations adapt to COVID-19	1	0.996759	pos	1.9877038	0.0681818	0.227273	0.09
Nintendo Warns Chip Crunch May Hit Switch During Gaming Boom	1	1.0091969	neut	2.0276582	0	0	0.09
Microsoft to allow EU customers to process, store data in the region	1	1.029549	neut	3.2851017	0	0	0.09

Рис. 4.9 Результат

Останній крок – розрахувати «важливість» за формулою, вказаною у розділі 3.3 і відсортувати датасет за цією метрикою за спаданням

Зараз можемо подивитись результат роботи загальної моделі. Перші важливі десять новин і останні десять «неважливі» новини виглядають так:

headline	
Check out these stocks: 25 companies that h…	Bill and Melinda Gates announce divorce after 27-years of marriage
AMD CEO Lisa Su: 'This is a very unique time in the semiconductor market'	Bill and Melinda Gates say they are ending their marriage
Heres Why Microsoft (MSFT) Became a Top Contributor in Baron Funds Q1 Portfolio	The Zacks Analyst Blog Highlights: Google, NVIDIA, Microsoft and Facebook
10 Best Dividend Stocks to Buy According to Billionaire Philippe Laffont	The future of work according to big tech
10 Best Dividend Stocks to Buy According to Billionaire Stan Druckenmiller	Jim Cramer: FAANG's Still Got Teeth
10 Best Software Stocks to Buy According to Billionaire Paul Tudor Jones	Bill and Melinda Gates Announce End of 27-Year Marriage
Best Tech Stocks To Buy Or Watch Now: Why R…	Apples App Store Manager to Take Center Stage at Epic Trial
10 best tablets for drawing, gaming and films	How to 'be vigilant' and protect your digital assets from getting hacked
10 Best High-Yield Dividend Stocks According to Billionaire Mario Gabelli	White House Urged to Address Surge in Ransomware Attacks
What will Bill and Melinda Gates' divorce mean for their foundation?	The Tech Sector Money Machine

Рис. 4.10 Новини на початку рейтингу і в кінці

4.2 Експеримент: як впливає новина на ціну акцій

sentiment	firstment	polarity	subjectivity	FuzzySentiment	IMPORTANCE	close	close_after_news
pos	1.9345239	0.9	0.9	2.67	33.6235	248	247
pos	2.036224	0.375	1	2.67	22.6328	0	0
pos	1.9920452	0.5	0.5	2.67	13.5462	251	251
pos	1.9058243	1	0.3	2.67	13.4474	248	246
pos	1.9341958	1	0.3	2.67	13.4416	248	246
pos	1.8915328	1	0.3	2.67	13.3837	247	248
pos	1.924961	1	0.3	2.67	13.3613	0	0
pos	2.4752517	1	0.3	2.67	13.2371	0	0
pos	1.9951628	0.58	0.42	2.67	12.3983	254	252
neg	2.6238317	0.3125	0.6875	1.01	3.99299	0	0
pos	1.9378456	0.432121	0.446667	0.87	3.26471	251	254
neg	2.0804992	0.5	0.7	0.09	2.82495	0	0
neut	2.0566912	0	1	0.09	2.71713	252	251

Рис. 4.11 Дивимося ціну акцій

Close – ціна акції в момент публікації новини, close_after_news – ціна за тридцять хвилин після новини. Ціни акцій є не всюди, бо в час випуску новини торгів не було. В результаті ми бачимо, що десь 50/50 ціни корелюють із настроєм новини, бо ціни не дуже залежать від новин.

Висновок

Ми побудували модель для обчислення «важливості» новин (заголовків новин). Модель доволі непогано справляється з поставленою задачею, бо з одного боку, LSTM моделі мають високу точність, а з іншого боку, бібліотека TextBlob є гарно натренованою і добре визначає тональність речень. Звичайно, вона не претендує на стовідсоткову точність, адже це просто експериментальна модель. Також варто зазначити недоліки, які , в першу чергу, пов’язані з тією ж бібліотекою TextBlob. Вона сильно тригериться на такі слова, як “good”, “best”, “crazy” і т.д., і через це в топ новин можуть виноситися неважливі для активу, який ми вибрали, заголовки, які містять вищевказані слова. Також, як зазначалося раніше, «важливість» є абсолютно суб’єктивним значенням і не може бути визначеною однозначно. В кінці був проведений тест на те, чи впливає порахована важливість на ціну акцій. Не зовсім вона впливає, бо ціна акцій не може завжди залежить від новин.

Список літератури

1.Vashishtha, Srishti & Susan, Seba. (2019). Fuzzy Rule based Unsupervised Sentiment Analysis from Social Media Posts. Expert Systems with Applications.

2.Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 2nd Edition, Aurélien Géron, O'Reilly Media.

3. Immanuvelrajkumar, Sheeba & Vivekanandan, Kumuda. (2014). A Fuzzy Logic Based Sentiment Classification. International Journal of Data Mining & Knowledge Management Process. 4. 27-44. 10.5121/ijdkp.2014.4403.

4. Jefferson, Chris & Liu, Han & Haig, Ella. (2017). Fuzzy Approach for Sentiment Analysis. 10.1109/FUZZ-IEEE.2017.8015577.

5 Understanding LSTMs,

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

6. Himanshu Singh, Yunis Ahmad Lone 2020 253 H. Singh and Y. A. Lone, Deep Neuro-Fuzzy Systems with Python

7. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы

8 Fuzzy-Rule-based-Unsupervised-Sentiment-Analysis-from-Social-Media-Posts, https://github.com/SrishtiVashishtha/Fuzzy-Rule-based-Unsupervised-Sentiment-Analysis-from-Social-Media-Posts

9. https://www.python.org/

10. https://keras.io/

11. https://pandas.pydata.org/

12. https://matplotlib.org/

13. https://finviz.com/

14 An introduction to deep learning

https://developer.ibm.com/technologies/artificial-intelligence/articles/an-introduction-to-deep-learning/#:~:text=To%20put%20things%20in%20perspective,within%20data%20to%20make%20predictions.

15 Neural Network— Must know Model Training Tricks, https://medium.com/@abhismatrix/neural-network-model-training-tricks-61254a2a1f6b

16 Growing your own RNN cell : Simplified, https://towardsdatascience.com/growing-your-own-rnn-cell-simplified-b68ba2c0f082