Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут телекомунікаційних систем
(повна назва інституту/факультету)

Кафедра телекомунікацій
(повна назва кафедри)

«На правах рукопису»
УДК _____

До захисту допущено
Завідувач кафедри

_____ Сергій КРАВЧУК
(підпис) (Ім'я, прізвище)

"___"_____2020_р.

Магістерська дисертація

на здобуття освітнього ступеня «магістр»

Спеціальність 172 Телекомунікації та радіотехніка,
(код і назва)

За освітньо-професійною програмою Інженерія та програмування інфокомунікацій.

на тему: «Дослідження можливості використання машинного навчання для побудови адаптивних телекомунікаційних систем»

Виконав: студент 2 курсу, групи ТЗ - 91мп
(шифр групи)

Сірант Артур Сергійович
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник к.т.н., доцент каф. телекомунікацій Міночкін Д.А. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 рік

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут телекомунікаційних систем
(повна назва)

Кафедра телекомунікацій
(повна назва)

Спеціальність 172 Телекомунікації та радіотехніка
(код і назва)

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою Інженерія та програмування інфокомунікацій.

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри

_____Явіся В.С.
(підпис) (ініціали, прізвище)

« 20 » січня 2020 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
_____Сірант Артур Сергійович_____

(прізвище, ім'я, по батькові)

1. Тема дисертації: Дослідження можливості використання машинного навчання для побудови адаптивних телекомунікаційних систем

науковий керівник дисертації к.т.н., доцент каф. телекомунікацій Міночкін Д.А.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «03» « листопада » 2020 р. № 3208-с

2. Строк подання студентом дисертації 12.12.2020 р.

3. Об'єкт дослідження: Процес побудови адаптивної системи прогнозування відтоку користувачів з використанням машинного навчання.

4. Предмет дослідження: адаптивні системи з використанням алгоритмів машинного навчання.

5. Перелік завдань, які потрібно розробити
1) Провести аналіз літератури з теми дослідження.

2) Розглянути основні парадигми і алгоритми машинного навчання а також перспективні напрями використання в телекомунікаційних системах.

3) Виконати розробку системи для виявлення користувачів схильних до відтоку.

6. Орієнтовний перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): актуальність, мета, об’єкт дослідження, предмет дослідження, загальна модель машинного навчання, теоретична модель системи відтоку, моделювання системи прогнозування, обробка результатів і висновки.

7. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

8. Дата видачі завдання 28.10.2019

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	Підбір літератури з теми дослідження	01.11.2019 09.06.2020	виконано
2	Написання розділу: Основні відомості про машинне навчання	12.01.2020 17.04.2020	виконано
3	Написання розділу: Дослідження можливостей використання машинного навчання для побудови адаптивних систем.	18.04.2020 04.08.2020	виконано
4	Написання розділу: Система передбачення відтоку користувачів на базі моделі логістичної регресії.	05.08.2020 20.09.2020	виконано
5	Розробка системи прогнозування відтоку користувачів у програмному середовищі R.	21.09.2020 27.10.2020	виконано
6	Обробка результатів моделювання	28.10.2020 06.11.2020	виконано
7	Розробка стартап-проекту	07.11.2020 01.12.2020	виконано
8	Оформлення готової роботи	02.12.2020 25.12.2020	виконано

Студент _____ Сірант А.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Міночкін Д.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Робота містить 83 сторінку, 30 рисунків, 9 таблиць. Було використано 22 джерела.

Мета роботи: огляд сучасних методів машинного навчання, визначення їх переваг та недоліків для побудови адаптивних телекомунікаційних систем. Розробка системи прогнозування відтоку.

Об'єкт дослідження: парадигми та алгоритми машинного навчання.

Предмет дослідження: інтеграція ML в телекомунікації, побудова системи прогнозування відтоку на базі моделі логістичної регресії.

В ході виконання цієї роботи проаналізовано перспективні алгоритми ML, в контексті телекомунікацій. Досліджена перспектива впровадження глибинних нейронних мереж та приведено модель прогнозування на базі LSTM архітектури. Розроблена модель прогнозування відтоку користувачів.

Для досягнення мети дослідження було поставлено та вирішено такі основні задачі:

1. Ознайомлення основними парадигмами ML.
2. Розгляд перспектив і можливостей глибинного навчання для телекомунікаційних систем.
3. Програмування моделі прогнозування відтоку на мові R.

Ключові слова: машинне навчання, глибинне навчання, логістична регресія, прогнозування відтоку користувачів, мова R.

ABSTRACT

The work contains 83 pages, 30 figures, 9 tables. 22 sources were used.

Purpose: review of modern methods of machine learning, identification of their advantages and disadvantages for the construction of adaptive telecommunications systems. Development a churn prediction system.

Object of research: paradigms and algorithms of machine learning.

Subject of research: integration of ML in telecommunications, construction the churn prediction system based on logistic regression model.

In the course of this work, promising ML algorithms in the context of telecommunications were analyzed. The perspective of introduction of deep neural networks was investigated and the model of forecasting on the basis of LSTM architecture is resulted. The model for users churn prediction has been developed.

To achieve the goal of the study, the following main tasks were set and solved:

1. Introduction to the basic paradigms of ML.

2. Consideration of prospects and opportunities for deep learning in case telecommunication systems.

3. Programming the churn prediction model in the language R.

Keywords: machine learning, deep learning, logistic regression, user churn prediction, R language.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ML – Машинне навчання

SVM – Метод опорних векторів

ANN – Штучні нейронні мережі

RBFN – мережа радіальних базових функцій

MDS – багатовимірне масштабування

PCA – аналіз головних компонентів

LDA – лінійний дискримінантний аналіз

PCR – регресія головних компонентів

HCI – Людино-комп'ютерна взаємодія

RAT – технології радіодоступу

ISP – інтернет провайдери

NLP – обробка людської мови

GPU – графічний процесор

DS – наука про дані

GP – Гаусівський процес

SGD – стохастичний градієнтний спуск

RBM – обмежена машина Больцмана

GAN – Генеративно-змагальна мережа

CNN – згорткова нейронна мережа

NN – нейронна мережа

ITS – інтелектуальні транспортні системи

ARIMA – autoregressive integrated moving average

RNN – рекурентна нейронна мережа

LSTM – Long Short-Term Memory

DBN – мережа глибокої довіри

MAE – середня абсолютна помилка

sMAPE – симетрична середня абсолютна процентна помилка

SVR – підтримуючий векторний регресор

xGBoost – екстремальне посилення градієнта

ВСТУП

Кожен день кількість даних невпинно зростає. З одного боку це ускладнює їх обробку, а з іншого відкриває можливості для їх технічного аналізу. Телекомунікаційна галузь має доступ до невичерпних об’ємів інформації. Від даних білінгу мобільних пристроїв до IoT телеметрії, що в свою чергу надає безмежні можливості для їх обробки алгоритмами машинного навчання.

Актуальність роботи

Основні види діяльності компаній, що працюють у телекомунікаційному секторі, тісно пов'язані з передачею, обміном та імпортом даних. Поєднання безпрецедентних обчислювальних потужностей та доступних даних з новими варіантами добре відпрацьованих алгоритмів машинного навчання радикально змінює стратегії оптимізації для великих телекомунікаційних компаній. Зокрема, в телеком галузі завжди доводилося мати справу зі складними системами, стохастичними контекстами, комбінаторними проблемами і важко передбачуваними користувачами.

Методи дослідження: аналіз алгоритмів машинного навчання на предмет інтеграції в адаптивні телекомунікаційні системи, порівняння альтернативних рішень з тими що уже існують на ринку, розробка моделі прогнозування в програмному середовищі R.

Практична цінність: Розроблена система прогнозування здатна заощадити значні кошти на маркетинговій політиці з утримання користувачів. Запропоновані моделі є надзвичайно гнучкими і можуть бути використані не тільки в телекомунікаційній галузі а й в суміжних областях, наприклад, електронній комерції чи в медицині.

РОЗДІЛ 1. ОСНОВНІ ВІДОМОСТІ ПРО МАШИННЕ НАВЧАННЯ

1.1 Поняття машинного навчання

Перш за все треба розглянути основні ідеї та принципи машинного навчання. На основі цього досвіду буде можливо визначити перспективні напрямки використання алгоритмів для побудови адаптивних телекомунікаційних систем.

Термін машинне навчання можна визначити як комп’ютерну програму, що вчиться на деякому досвіді відповідно до завдання з деяким показником ефективності, а її ефективність до завдання покращується із досвідом. Машинне навчання - це мультидисциплінарна сфера, що має широкий спектр наукових досліджень, що підсилюють її існування.

Моделювання моделей ML суттєво пов'язане з обчислювальною статистикою, основною метою якої є зосередження на прогнозуванні за допомогою комп'ютерів. Це також пов'язано з математичною оптимізацією, яка пов'язує моделі, програми та структуру зі сферою статистики.

Проблеми реального світу мають велику складність, що робить їх чудовими кандидатами для застосування ML. Машинне навчання може бути застосовано до різних областей обчислень, щоб розробляти та програмувати явні алгоритми з високою продуктивністю, наприклад, фільтрація електронної пошти, виявлення шахрайства в соціальних мережах, прогнозування трафіку, характер визнання та рекомендації щодо продуктів.

Машинне навчання потрібне для того, щоб комп’ютери виконували складні завдання без будь-якого втручання людей на основі навчання та постійного нарощування досвіду, необхідного для розуміння складності проблеми та відповідної адаптації.

1.2 Проблематика DS та ML

Перед тим, як перейти до вирішення проблеми, проблему потрібно класифікувати відповідним чином, щоб до нього можна було застосувати найбільш відповідний алгоритм машинного навчання. Будь-яку проблему в науці даних можна згрупувати в одну з наступних категорій, які приведено у таблиці 1.1.

Таблиця 1.1. Класифікація задач DS

Тип проблеми ML	Опис	Приклад
Класифікація	Вибір одного з n варіантів	Так/ні, кіт/людина/авто/кінь
Регресія	Прогнозування числових значень	Коефіцієнт ефективності
Кластерінг	Групування схожих прикладів	Найбільш важливий документ
Асоціативні правила навчання	Висновок щодо імовірних зв’язків у середині даних	Якщо ви купуєте вудку то ви напевно купите й підсаку
Структурований вихід	Створює складний результат	Дерева синтаксичного розбору людської мови, обмежувальні рамки розпізнавання зображень
Рейтинг	Визначає позицію за шкалою або статусом	Рейтинг результату пошуку

Таким чином, залежно від типу проблеми, може бути застосований відповідний підхід до машинного навчання. Деякі категорії пояснюються нижче:

- Проблема класифікації - проблема, в якій вихід може бути лише одним із фіксованої кількості класів результатів, відомих апріорі, таких як так/ні, істинно/хибно, називається проблемою класифікації.
- Проблема регресії - алгоритми регресії використовуються для вирішення проблем з безперервним та числовим виходом. Зазвичай вони використовуються для вирішення проблем, що стосуються таких питань, як «скільки», «який шанс».
- Проблема кластеризації підпадає під категорію алгоритмів навчання без нагляду. Ці алгоритми намагаються вивчити структури в даних і намагаються створити кластери на основі подібності в структурі даних. Потім різні класи або кластери маркуються. Під час навчання алгоритм розміщує нові дані в одному з кластерів.

ML використовується для вирішення різних проблем, які вимагають навчання з боку машини. Проблема навчання має три особливості:

- Класи завдань (Завдання, яке потрібно вивчити)
- Показник ефективності, який слід вдосконалити
- Процес набуття досвіду

Загальна модель машинного навчання складається з шести компонентів, незалежних від прийнятого алгоритму. На наступному малюнку 3 зображені ці первинні компоненти.

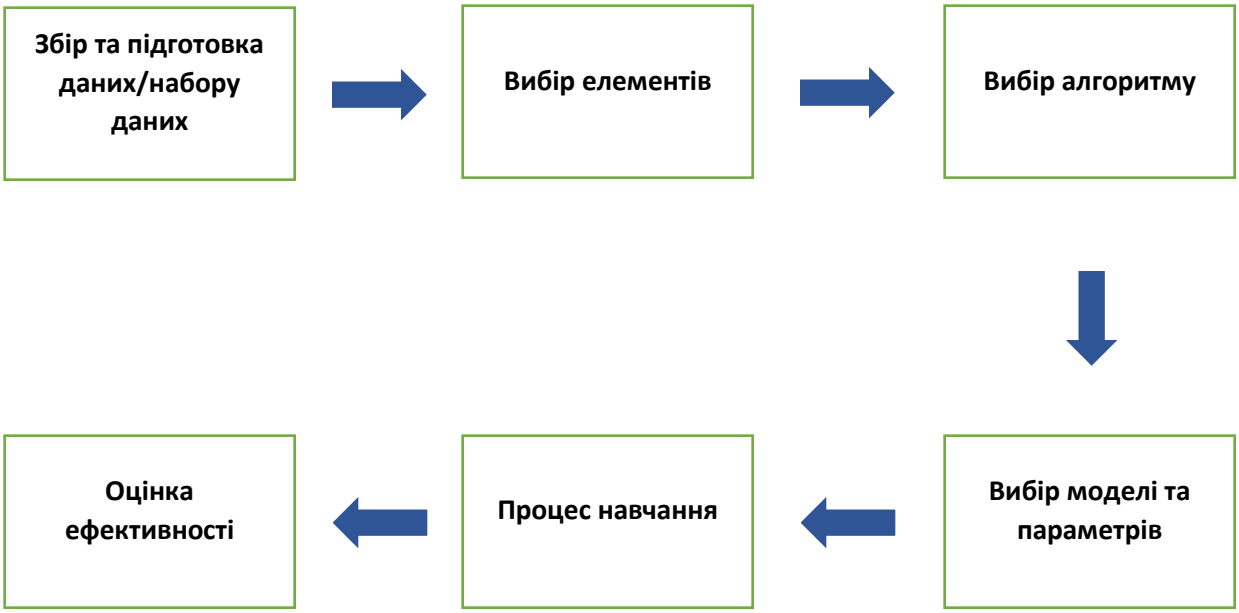

Рис.1.1. Компоненти загальної моделі ML.

Кожен компонент моделі має виконати конкретне завдання, як описано далі:

- Збір та підготовка даних: основним завданням у процесі машинного навчання є збір та підготовка даних у форматі, який може бути наданий для обробки алгоритмом. Для будь-якої проблеми може використовуватися велика кількість даних. Веб-дані, як правило, неструктуровані і містять багато шуму, тобто нерелевантні дані, а також зайві дані. Отже, їх потрібно очистити та попередньо обробити до структурованого формату.
- Вибір елементів: дані, отримані на вищевказаному етапі, можуть містити множину елементів, не всі з яких мали б відношення до навчального процесу. Ці елементи потрібно видалити та отримати підмножину найважливіших з них.
- Вибір алгоритму: один алгоритм машинного навчання не може бути використаним для вирішення всіх задач. Деякі алгоритми більше

підходять для певної проблеми класу. Вибір найкращого алгоритму машинного навчання для даної задачі є обов'язковим для отримання найкращих можливих результатів.

- Вибір моделей та параметрів: більшість алгоритмів машинного навчання вимагають певного початкового втручання вручну для встановлення найбільш відповідних значень різних параметрів.
- Процес навчання: після вибору відповідного алгоритму та відповідних значень параметрів модель потрібно навчити, використовуючи частину набору даних як навчальні дані.
- Оцінка продуктивності: перед впровадженням системи в реальному часі модель повинна бути протестована на основі тестових даних, щоб оцінити, наскільки навчання було ефективне за допомогою різних параметрів продуктивності, таких як точність та відклик.

1.3 Парадигми машинного навчання

Залежно від того, як навчається алгоритм, і на основі доступності результатів під час навчання, парадигми машинного навчання можна класифікувати на десять категорій. Сюди входять: навчання з вчителем, навчання без вчителя, часткове навчання, навчання з підкріпленням, еволюційне навчання, ансамбль методів, штучна нейронна мережа, навчання на основі екземплярів, алгоритми зменшення розмірності та гібридне навчання. Кожна з цих парадигм пояснюється нижче.

Навчання з вчителем

Під навчанням з вчителем розуміється набір прикладів або навчальних модулів з правильними вихідними даними, і на основі цих навчальних наборів алгоритм вчиться реагувати більш точно, порівнюючи свої результати з вхідними даними. Навчання з вчителем також відомо як навчання на прикладах. На наступному рисунку пояснюється концепція навчання з вчителем.

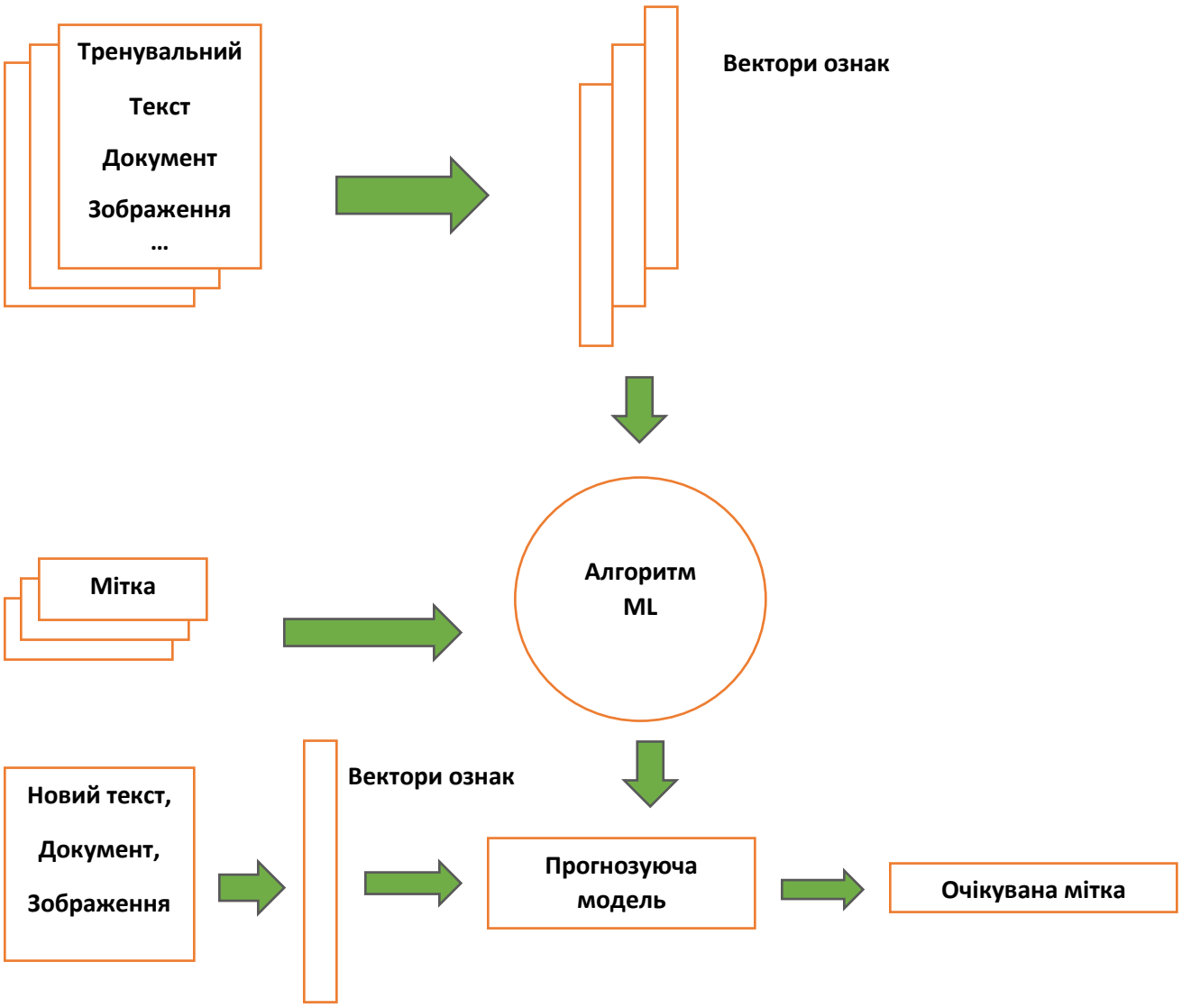

Рис.1.2. Навчання з вчителем

Навчання з вчителем знаходить застосування в прогнозуванні на основі історичних даних. Завдання навчання з учителем можна далі поділити на

завдання класифікації і завдання регресії. У разі класифікації вихідні мітки дискретні, тоді як у разі регресії вони неперервні.

Навчання без вчителя

Підхід до навчання без вчителя полягає в розпізнаванні неідентифікованих існуючих шаблонів з даних, для виявлення правил. Цей прийом доречний в ситуації, коли категорії даних невідомі. Тут дані навчання не позначені. Навчання без вчителя розглядається як статистичний підхід до навчання і, таким чином, відноситься до проблеми пошуку прихованої структури в немаркованих даних. Наступний рисунок пояснює цю концепцію.

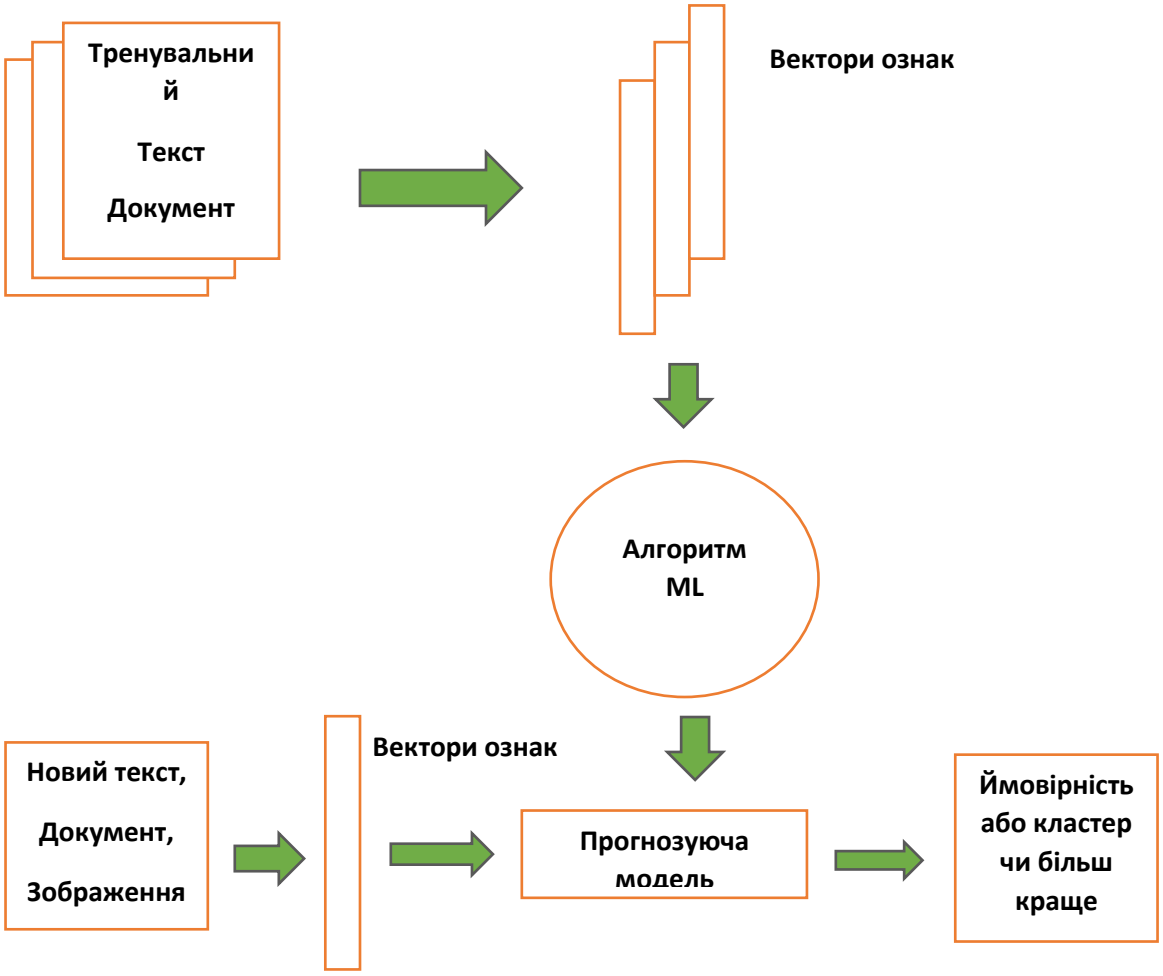

Рис.1.3. Навчання без вчителя

Навчання з підкріпленням

Навчання з підкріпленням розглядається як проміжний тип навчання, оскільки алгоритм надає тільки відповідь, яка повідомляє, чи є результат правильним чи ні. Алгоритм повинен дослідити і виключити різні можливості, щоб отримати правильний результат. Це вважається навчанням з критиком, оскільки алгоритм не пропонує ніяких пропозицій чи рішень проблеми.

Еволюційне навчання

Даний підхід натхненний біологічними організмами, які адаптуються до навколишнього середовища. Алгоритм розуміє поведінку і адаптується до вхідних даних і виключає малоймовірні рішення. Він заснований на ідеї пристосування, щоб запропонувати найкраще вирішення проблеми.

Часткове навчання

Ці алгоритми використовують можливості як навчання з вчителем, так і навчання без вчителя. У двох попередніх типах мітки вихідних даних або надаються для всіх спостережень, або не надаються. Можуть бути ситуації, коли деякі спостереження забезпечені мітками, але більшість спостережень не мають позначок через високу вартість маркування і відсутності кваліфікованого людського досвіду. У таких ситуаціях для побудови моделей найкраще підходять алгоритми часткового навчання. Часткове навчання можна використовувати з такими завданнями, як класифікація, регресія і прогнозування. Далі його можна розділити на генеративні моделі, самонавчальні і трансіндуктівні SVM.

Ансамбль методів

Це модель машинного навчання, в якій численні учні (окремі моделі) навчаються для вирішення загальної проблеми. На відміну від інших парадигм машинного навчання, які створюють одну гіпотезу з навчальних даних, ансамблеве навчання намагається вчитися, будуючи набір гіпотез з навчальних

даних та комбінуючи їх, створюючи модель прогнозування для зменшення упередженості, дисперсії або покращити прогнозування. Навчання ансамблю можна розділити на дві групи:

- Послідовний підхід - це методи, в яких базові учні будуються послідовно (AdaBoost). Цей метод використовує залежність між базовими учнями.
- Паралельні підходи - у них базові учні не залежать один від одного, тому ці взаємозв'язки використовуються шляхом паралельної побудови базових учнів (наприклад, випадковий ліс)
- Пакування: Він використовую однорідних учнів у семплах та бере середнє значення серед усіх прогнозів. Наприклад, M різних дерев можна навчити різними підмножинами даних і обчислити ансамбль як:

$$f(x) = 1/M \sum_{m=1}^{M} f_m(x)$$

- Підсилення: це ітераційний прийом, який регулює вагу спостереження на основі останньої класифікації. Він намагається підібрати послідовність слабких моделей учнів, які працюють трохи краще, ніж просто випадкове передбачення, наприклад невеликі дерева рішень. AdaBoost означає адаптивне підсилення і є найбільш широко застосовуваним алгоритмом підсилення.

Штучна нейронна мережа

Штучні нейронні мережі (ANN) працюють подібно біологічній нейронній мережі. Нейронна мережа - це взаємозв’язок нейронів, які допомагають електричним імпульсам поширюватися через мозок.

Основною одиницею навчання в нейронної мережі є нейрон, який являє собою нервову клітину. Нейрон складається з чотирьох частин: дендритів

(рецептор), соми (обробника електричного сигналу), ядра (ядра нейрона) і аксона (передавального кінця нейрона). Аналогічно біологічній нейронній мережі, ANN працює на трьох рівнях: вхідному, прихованому та вихідному. Цей тип мережі має зважені взаємозв'язки і навчається, регулюючи вагу взаємозв'язків, щоб виконувати паралельну розподілену обробку. Алгоритм навчання Perceptron, алгоритм зворотного розповсюдження, мережі Hopfield, мережа радіальних базових функцій (RBFN) - деякі популярні алгоритми. Виходячи з навчальної поведінки, ANN можна далі класифікувати як:

- Контрольована нейронна мережа - входи та виходи надаються мережі як навчальні дані. Мережа навчається на цих даним, регулюючи вагу, щоб отримати точні результати. Коли він повністю навчений, йому передаються тестові дані для прогнозування результату.
- Непідконтрольна нейронна мережа. У непідконтрольній нейронній мережі мережа не має вихідних даних. Вона намагається знайти якусь структуру або кореляцію між вхідними даними та об’єднати ці дані разом у групу чи клас. Коли вона повністю навчена, їй подаються тестові дані для прогнозування результату.
- Підкріплення нейронної мережі - оскільки люди взаємодіють із навколишнім середовищем і навчаються на помилках, нейронна мережа підкріплення також вчиться на своїх минулих рішеннях шляхом покарання за неправильні рішення та винагороди за правильні рішення. Вага з'єднань, що дають позитивний результат, посилюється, а та, що дає негативний результат, послабляється.

На відміну від інших методів машинного навчання, де чітке визначення цільової функції забезпечується на основі даних навчання, цей метод навчання на початку не описує жодної цільової функції. Швидше він просто зберігає навчальний екземпляр, а узагальнення відкладається до класифікації нового

екземпляра. Звідси він також відомий як ледачий учень. Такі методи створюють базу даних навчальних екземплярів, і щоразу, коли нові дані подаються на вхід, вони порівнюють ці дані з іншими екземплярами бази даних, використовуючи міру подібності, щоб знайти найближчу відповідність та зробити прогноз. Ледачий учень оцінює цільову функцію по-різному та локально для кожного нового екземпляра, який потрібно класифікувати, замість того, щоб оцінювати її глобально для всього простору екземпляра, отже, швидше тренуватися, але для прогнозування потрібен час.

Алгоритми зменшення розмірності

Протягом останніх декількох десятиліть інтелектуальні моделі машинного навчання були застосовані в багатьох складних програмах з великим обсягом даних. Однак існуючі системи, засновані на ML, недостатньо ефективні та перепоною для обробки об'ємних даних. Висока розмірність даних виявилася прокляттям для обробки даних. Інший виклик - розрідженість даних. Найбільш точний, з технічної точки зору, підхід є дорогим для даного типу даних. Алгоритм зменшення розмірності допомагає зменшити обчислювальні витрати за рахунок зменшення кількості розмірів даних. Це досягається за рахунок зменшення кількості надлишкових і нерелевантних даних і очищення даних для підвищення точності результатів. Зниження розмірності працює неконтрольованим чином для пошуку і використання неявної структури в даних. Існує безліч алгоритмів зменшення розмірності, які можна адаптувати за допомогою алгоритмів класифікації і регресії, таких як багатовимірне масштабування (MDS), аналіз головних компонентів (PCA), лінійний дискримінантний аналіз (LDA), регресія головних компонентів (PCR) і лінійний дискримінантний аналіз (LDA).

Гібридне навчання

Хоча ансамблеве навчання значно спростило задачу дослідників, що мають справу з загальними проблемами обчислювальної складності, надмірної підгонки

і дотримання локальних мінімумів в алгоритмах класифікації, дослідники виявили проблеми з ансамблевим навчанням. Складний набір декількох класифікаторів ускладнює реалізацію і ускладнює аналіз результатів. Замість підвищення точності моделі ансамблі можуть збільшувати помилку на рівні окремого базового учня. Ансамблі можуть призвести до низької точності в результаті вибору комбінації поганих класифікаторів. Останнім підходом до вирішення таких проблем є гібридизація, тобто створення ансамблю різнорідних моделей. При цьому комбінується більше одного методу, наприклад, об'єднання кластеризації і дерева рішень або кластеризації і асоціативного аналізу і т. д..

З усіх вищезазначених парадигм навчання з учителем є найбільш популярним в науковому середовищі.

1.4 Алгоритми машинного навчання

У цьому підрозділі розглянуті популярні алгоритми машинного навчання з різних парадигм, пояснені у попередньому підрозділі. Хоча кількість алгоритмів, що потрапляють у кожну парадигму величезна, далі буде розглянуті лише деякі з них. Наступна таблиця коротко пояснює деякі з цих алгоритмів.

Таблиця 1.2. Класифікація задач DS

Парадигма	**Алгоритм**	**Опис**
Навчання з вчителем	Дерево рішень	Дерево рішень - це один із популярних і потужних алгоритмів машинного навчання. Це непараметричний контрольований метод навчання, який можна використовувати як для класифікації, так і для регресії. Мета полягає в тому, щоб створити модель, яка передбачає значення цільової змінної шляхом вивчення простих правил прийняття рішень, виведених із

		особливостей даних. Для класифікаційної моделі цільові значення мають дискретний характер, тоді як для регресійної моделі цільові значення представлені безперервними. Найбільш популярні алгоритми: Дерево класифікації і регресії (CART), Ітеративний діхотомайзер 3 (ID3), Автоматичне виявлення взаємодії (CHAID), Хі-квадрат C4.5 і C5.0 і M5.
	Баєсовий класифікатор	Наївний Байес класифікує за допомогою теореми ймовірності Байеса. Теорема Байєса обчислює апостеріорну ймовірність події (A) з урахуванням деякої апріорної ймовірності події B, представленої P (A/B), в такий спосіб: $$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$ Де, • A і B - події. • P (A) і P (B) – ймовірності спостереження A і B незалежно один від одного. • P(A/B) - умовна ймовірність, тобто ймовірність спостереження A, якщо B істинно. • P(B/A) - ймовірність спостереження B, якщо A істинно. Наївний Баєсовський класифікатор підпадає під категорію простих імовірнісних

		класифікаторів, заснованих на концепції теореми Байєса, що має суворі припущення щодо незалежності серед ознак. Це особливо підходить, коли розмірність входів висока.
	Метод опорних векторів	SVM можна використовувати як для класифікації, так і для вирішення завдань регресії. Це контрольований алгоритм навчання. Він працює за принципом розрахунку маржі. У цьому алгоритмі кожен елемент даних відображається як точка в n-вимірному просторі (де n - кількість функцій, які є в нашому наборі даних). Значення кожного об'єкта - це значення відповідної координати. Він класифікує дані по різних класах, знаходячи лінію (гіперплощину), яка розділяє набори навчальних даних на класи. Він працює, максимізуючи відстані між найближчою точкою даних (в обох класах) і гіперплощиною.
	Регресійний аналіз	Регресійний аналіз - це метод прогнозного моделювання, який досліджує відношення між залежними і незалежними змінними. Це важливий інструмент для аналізу і моделювання даних. У цьому методі підганяється пряма/крива до точок даних, щоб мінімізувати різницю між відстанями точок даних від кривої або прямої. Існують різні види регресійного аналізу, такі як лінійний, логістичний та поліноміальний.

Навчання без вчителя	Кластеризація K- mean	K-mean - популярний некерований алгоритм машинного навчання для кластерного аналізу. Його мета полягає в розподілі спостережень «n» та «k» кластери, у яких кожне спостереження належить кластеру, що має найближче середнє значення, слугуючи прототипом кластера. Середнє значення спостережень у певному кластері визначає центр цього кластера.
Навчання на основі екземплярів	K-nearest Neighbours	Це непараметричний метод, що використовується для класифікації та регресії. Враховуючи N тренувальних векторів, алгоритм KNN визначає k-найближчих сусідів невідомого вектору ознак, клас якого слід ідентифікувати
Ансамбль методів	Random Forest	Це метод ансамблевого навчання, який використовується при класифікації і регресії. Він використовує метод пакування для створення групи дерев рішень з випадковим підмножиною даних. Вихідні дані всіх дерев рішень у випадковому лісі об'єднуються для створення остаточних дерев рішень. В алгоритмі випадкового лісу є два етапи: один - створити випадковий ліс, а інший - зробити прогноз на основі класифікатора випадкового лісу, створеного на першому етапі.
Зменшення розмірності	Алгоритм основних компонентів (PCA)	Він в основному використовується для зменшення розмірності набору даних. Це допомагає зменшити кількість ознак набору

		даних або кількість незалежних змінних у наборі даних. Він використовує ортогональне перетворення для перетворення корельованих змінних у набір лінійно некорельованих, які називаються головними компонентами.

Висновки до розділу 1

1. Широкий спектр застосувань ML зумовлений тим, що існує більше десятку різноманітних парадигм навчання, які в свою чергу, включають в себе низку алгоритмів і методів. Таке різноманіття дозволяє вирішувати майже будь-які специфічні задачі, що робить ML привабливою технологією для побудови адаптивних систем.
2. Алгоритми машинного навчання вимагають великих обсягів даних, для точного та ефективного функціонування, що не завжди є загальнодоступним. Такі технічні гіганти, як Facebook та Google, мають доступ до таких величезних даних, саме тому вони лідирують у галузі штучного інтелекту.
3. Алгоритми машинного навчання постійно розробляються і стають більш поширеними з кожним роком. Вони корисні в наступних областях телекомунікаційної галузі: глибоке навчання для прогнозування тенденцій на ринку телекомунікаційних послуг, видобуток та аналіз білінгових даних, для створення самокерованих систем, побудова та вдосконалення HCI інтерфейсів.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ ВИКОРИСТАННЯ ГЛИБИННОГО НАВЧАННЯ ДЛЯ ПОБУДОВИ АДАПТИВНИХ СИСТЕМ

2.1. Область застосування, можливості та переваги глибинного навчання

Мобільні пристрої, підключені до інтернету, проникають у всі аспекти життя, роботи та розваг людей. Зростаюча кількість смартфонів і поява все більш різноманітних додатків викликають сплеск мобільного трафіку даних. Дійсно, останні дані показують, що річне споживання IP-трафіку з смартфонів уже кілька років як перевищує ПК-трафік та продовжує невпинно зростати. Враховуючи зміну переваг користувачів у напрямку бездротового підключення, поточна мобільна інфраструктура стикається з великими вимогами до пропускної здатності.

У відповідь на цей зростаючий попит на ранніх етапах роботи пропонується гнучке виділення ресурсів і розподілене управління мобільністю. Однак в довгостроковій перспективі інтернет провайдери (ISP) повинні розробити інтелектуальні гетерогенні архітектури та інструменти, які можуть відповідати більш суворим вимогам кінцевих користувачів до додатків. Зростаюче різноманіття та складність архітектур мобільних мереж зробило моніторинг та управління безліччю елементів мережі важкодоступними. Тому впровадження універсального машинного інтелекту в майбутні мобільні мережі викликає сильний науковий інтерес. Ця тенденція знаходить своє відображення в рішеннях на основі машинного навчання , від вибору технології радіодоступу (RAT) до виявлення шкідливих програм, а також в розробці мережевих систем, що підтримують методи машинного навчання. Машинне навчання дозволяє систематично витягувати цінну інформацію з даних трафіку і автоматично виявляти кореляції, які в іншому випадку були б занадто складними для вилучення людьми.

Будучи флагманом машинного навчання, глибинне навчання досягло видатних результатів в таких областях, як комп'ютерний зір і обробка людської мови (NLP).

Дані, що генеруються за допомогою мобільних середовищ, стають дедалі неоднорідніші, оскільки вони, як правило, збираються з різних джерел, мають різний формат і мають складні кореляційні зв’язки. Як наслідок, цілий ряд конкретних проблем стає занадто складним або непрактичним для традиційних засобів машинного навчання (наприклад, неглибокі нейронні мережі). Це пояснюється тим, що:

- Їх продуктивність не покращується, якщо надходить більше даних.
- Вони не можуть обробляти багатовимірні простори станів/дій в задачах управління. Навпаки, великі дані підживлюють продуктивність глибокого навчання, оскільки виключають досвід предметної області і замість цього використовують ієрархічне витяг ознак.

По суті, це означає, що інформацію можна ефективно переганяти, і з даних можна отримувати дедалі абстрактніші кореляції, зменшуючи зусилля попередньої обробки.

Паралельні обчислення на базі графічного процесора (GPU) додатково дозволяють глибинному навчанню робити висновки за мілісекунди. Це полегшує аналіз та управління мережею, долаючи обмеження виконання традиційних математичних методів (наприклад, опуклу оптимізацію, теорію ігор, метаевристику).

У таблиці 2.1 приведено переваги використання глибинного навчання для вирішення проблем мережевої інженерії. Зокрема:

1. Відомо, що розробка функцій, надзвичайно важлива для продуктивності традиційних алгоритмів машинного навчання та вимагає великих витрат. Ключовою перевагою глибокого навчання є те, що воно може автоматично отримувати високорівневі функції з даних, що мають складну структуру і внутрішні кореляції. Процес навчання не обов'язково повинна супроводжувати людина, що значно спрощує попередню людську роботу. Важливість цього посилюється в контексті мобільних мереж, оскільки мобільні дані зазвичай генеруються з різнорідних джерел, часто зашумлені і демонструють нетривіальні просторово-тимчасові патерни, маркування яких в іншому випадку потребувало б величезних людських зусиль.

Таблиця 2.1. Переваги використання глибинного навчання.

Ключовий аспект	**Опис**	**Переваги**
Вилучення функцій	Глибокі нейронні мережі можуть автоматично витягувати високорівневі функції через шари різної глибини	Скоротить обсяг дорогої ручної розробки функцій при обробці різнорідних і зашумлених мобільних даних
Експлуатація Big Data	На відміну від традиційних інструментів ML, ефективність глибокого навчання зазвичай значно зростає із збільшенням обсягу навчальних даних	Ефективно використовує величезні обсяги мобільних даних, що генеруються з високою швидкістю

Продовження таблиці 2.1.

Навчання без вчителя	Глибинне навчання ефективне при обробці немаркованих або напівмаркованих даних, що уможливлює навчання без учителя.	Обробка великих обсягів немаркованих даних, які часто зустрічаються в мобільних системах
Багатозадачне навчання	Функції, навчені нейронними мережами через приховані шари, можуть бути застосовані до різних завдань шляхом передачі навчання	Знижає вимоги до обчислень та пам’яті під час виконання багатозадачного навчання в мобільних системах

2. По-друге, глибоке навчання здатне обробляти великі обсяги даних. Мобільні мережі генерують великі обсяги різних типів даних з великою швидкістю. Навчання традиційними алгоритмам машинного навчання (наприклад, Support Vector Machine (SVM) і Gaussian Process (GP)) іноді вимагає зберігання всіх даних в пам'яті, що неможливо з обчислювальної точки зору при великих обсягах даних. Більше того, продуктивність ML не зростає суттєво через великі обсяги даних, а відносно швидко досягає свого плато. Навпаки, стохастичний градієнтний спуск (SGD), який використовується для навчання NN, вимагає тільки підмножини даних на кожному етапі навчання, що гарантує масштабованість глибинного навчання з великим об’ємом даних. Глибинні нейронні мережі отримують додаткові переваги, оскільки навчання з використанням великих даних запобігає переналаштуванню моделі.
3. Традиційне навчання з учителем ефективно тільки при наявності достатньої кількості розмічених даних. Однак більшість сучасних

мобільних систем генерують немарковані або частково марковані дані. Глибинне навчання надає безліч методів, які дозволяють використовувати немарковані дані для вивчення корисних шаблонів без учителя, наприклад, обмежена машина Больцмана (RBM), Генеративно-змагальна мережа (GAN). Програми включають в себе кластеризацію, апроксимацію розподілів даних, неконтрольоване/частково контрольоване навчання і один/нуль навчання.

4. Компресивні уявлення, засвоєні глибокими нейронними мережами, можна розподілити між різними завданнями, хоча це обмежено або важко досягти в інших парадигмах ML (наприклад, лінійна регресія, випадковий ліс тощо). Тому одну модель можна навчити виконувати кілька задач, не вимагаючи повної перекваліфікації моделі для різних завдань. Це важливо для інженерії мобільних мереж, оскільки знижує вимоги до обчислювальних ресурсів і пам'яті мобільних систем при виконанні багатозадачного навчання.
5. Глибинне навчання ефективне при передачі геометричних мобільних даних, в той час як для інших підходів машинного навчання складна задача. Геометричне уявлення відноситься до багатовимірних даних, представлене координатами, топологією, метриками і порядком. Мобільні дані, такі як місце розташування мобільного користувача і точка підключення до мережі, можуть бути природним чином представлені у вигляді хмар точок і графіків, які мають важливі геометричні властивості. Ці дані можна ефективно моделювати за допомогою спеціалізованої архітектури глибокого навчання, такої як PointNet ++ і Graph CNN. Використання цих архітектур має великий потенціал в геометричному аналізі мобільних даних.

2.2. Принципи обчислювальних методів в глибинному навчанні

Глибоке навчання - це, по суті, відгалуження машинного навчання, яке дозволяє алгоритму робити прогнози, класифікації або приймати рішення на основі даних без явного програмування. Класичні приклади включають лінійну регресію, класифікатор k-найближчих сусідів і Q-навчання. На відміну від традиційних інструментів машинного навчання, які в значній мірі покладаються на функції, визначені експертами в предметній області, алгоритми глибокого навчання ієрархічно витягують знання з необроблених даних через кілька рівнів нелінійних модулів обробки, щоб робити прогнози або вживати дії відповідно до деякої цільової мети. Найбільш відомими моделями глибокого навчання є нейронні мережі (NN), але тільки NN, які мають достатню кількість прихованих шарів (зазвичай більше одного), можуть розглядатися як «глибинні» моделі. Крім глибинних NN, є інші архітектури які мають кілька рівнів, такі як глибинні гаусові процеси, нейронні процеси і глибинний випадковий ліс, їх також можна розглядати як структури глибинного навчання. Таким чином, основною перевагою глибинного навчання перед традиційним машинним навчанням є автоматичне вилучення функцій, за допомогою якого можна обійти дорогий етап ручної розробки функцій. Далі проілюстровано взаємозв'язок між глибинним навчанням, машинним навчанням і штучним інтелектом (ШІ) на рис. 2. В цілому ШІ - це обчислювальна парадигма, яка наділяє машини інтелектом, прагнучі навчити їх працювати, реагувати і вчитися як люди. Багато методів підпадають під це визначення, включаючи машинне навчання, експертні системи і еволюційні алгоритми. Серед них машинне навчання дозволяє штучним процесам поглинати знання з даних і приймати рішення без явного програмування. Глибинне навчання - це сімейство методів машинного навчання, які імітують біологічні нервові системи і виконують репрезентативне навчання

за допомогою багаторівневих перетворень, що поширюються на всі парадигми навчання.

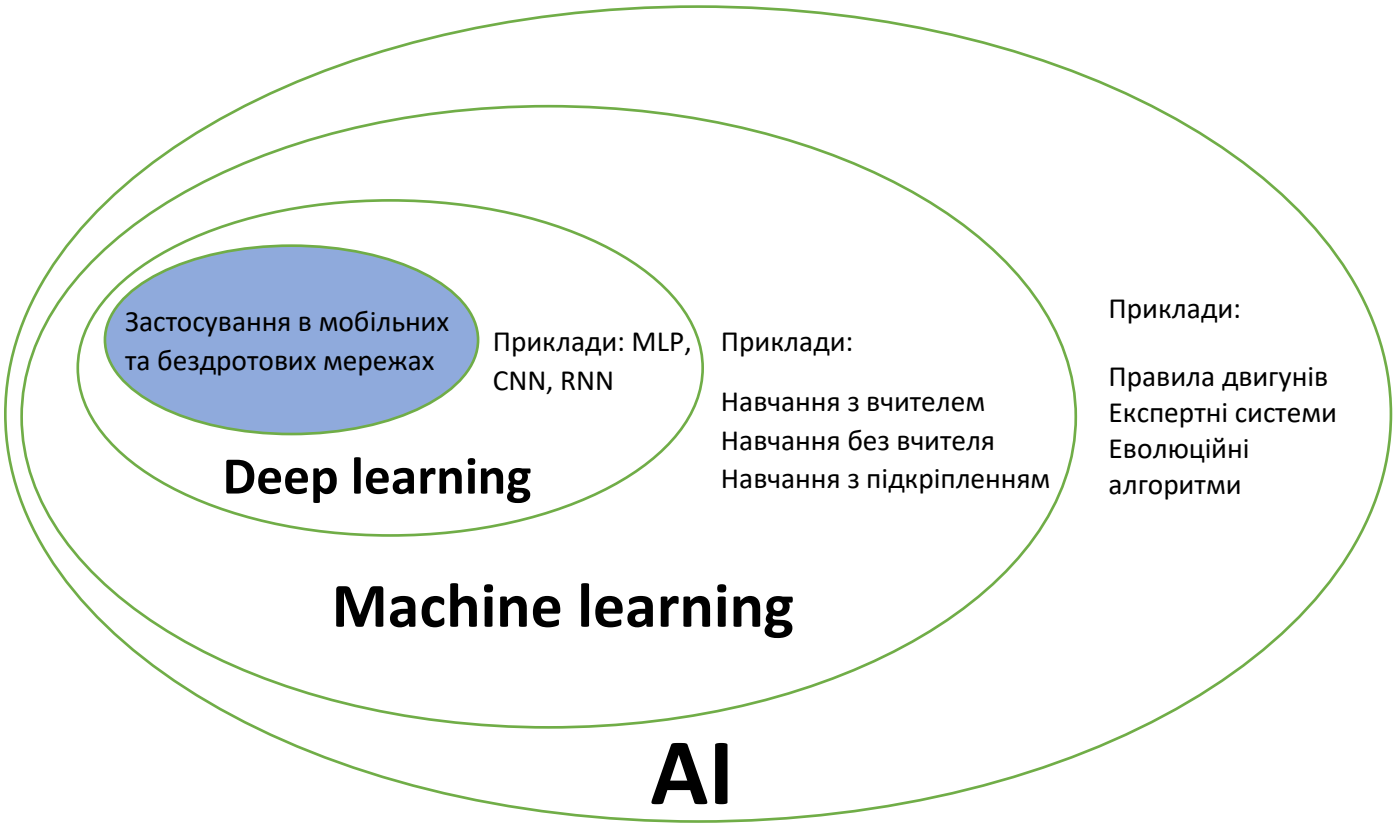

Рис. 2.1. Діаграма Венна взаємозв'язку між глибинним навчанням, машинним навчанням та ШІ.

Ключовою метою глибоких нейронних мереж є апроксимація складних функцій за допомогою композиції простих і заздалегідь визначених операційних одиниць (або нейронів). Така цільова функція може бути майже будь-якого типу, наприклад, зіставлення зображень і міток їх класів (класифікація), обчислення майбутніх цін на акції на основі історичних значень (регресія) або навіть визначення наступного оптимального шахового ходу з урахуванням поточного статусу на дошці (управління).

Виконувані операції зазвичай визначаються комбінацією вагомості певної групи прихованих модулів з нелінійною функцією активації, в залежності від структури моделі. Такі операції разом з модулями виводу називаються «шарами». Архітектура нейронної мережі нагадує процес сприйняття в мозку, коли певний набір одиниць активується в поточному середовищі, впливаючи на вихід моделі нейронної мережі.

2.2.1. Пряме і зворотне поширення

У математичному відношенні архітектура глибинних нейронних мереж зазвичай диференційована, тому ваги (або параметри) моделі можна вивчити, мінімізуючи функцію втрат, використовуючи методи градієнтного спуску шляхом зворотного поширення, дотримуючись основного правила ланцюга. Далі проілюстрований принципи процесів навчання і логічного висновку глибокої нейронної мережі на рис. 3, де була використана двовимірна (2D) згорткова нейронна мережу (NN).

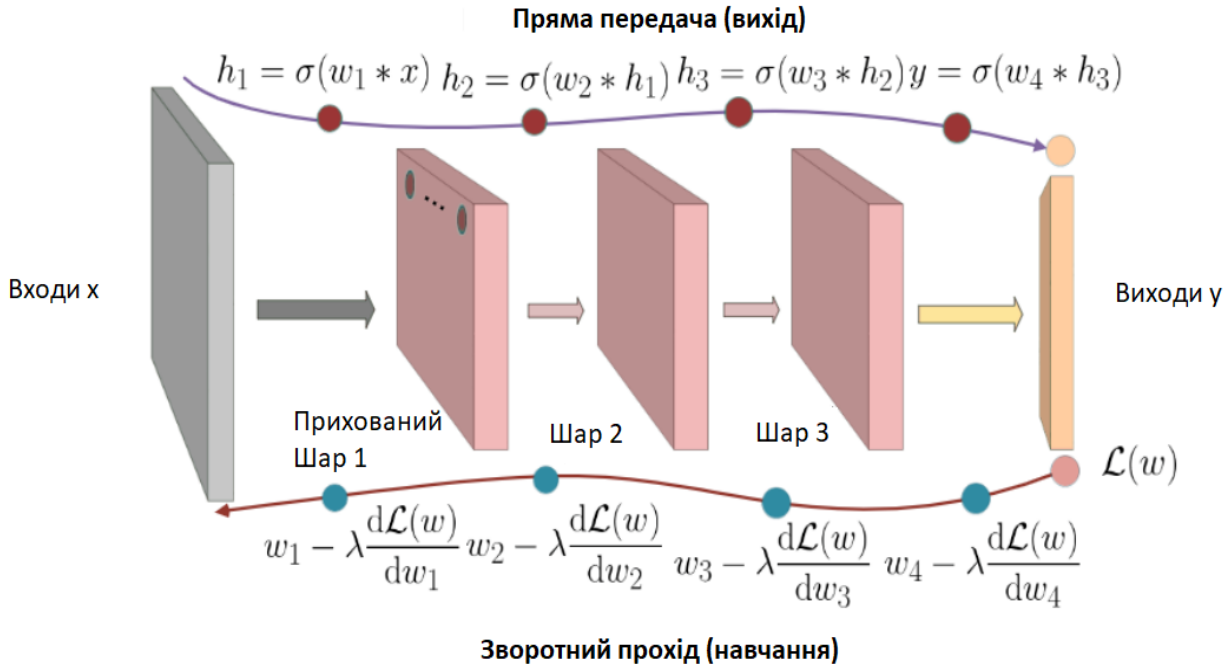

Рис. 2.2. Ілюстрація процесів навчання та виводу 4-рівневої CNN

де w - позначає ваги кожного прихованого рівня (шару),

σ - це функція активації,

λ - швидкість навчання,

$*$ - позначає операцію згортки,

$\mathcal{L}(w)$ - функція втрат, яку слід оптимізувати.

Пряме поширення: на рисунку показана CNN з 5 шарами, вхідним шаром (сірий), 3 прихованими шарами (червоний) і вихідним шаром (помаранчевий). При прямому поширенні двовимірний вхід x (наприклад, зображення) спочатку обробляється згортковим шаром (рівнем), який виконує наступну згорткову операцію:

$$h_1 = \sigma(w_{1^*}x) \tag{2.1}$$

де h_1 - вихід першого прихованого рівня (шару),

w_{1^*} – згортковий фільтр,

σ (·) - функція активації, спрямована на поліпшення нелінійності та репрезентативності моделі.

Вихідний сигнал h_1 згодом надається в якості вхідних даних і обробляється двома наступними згортковими шарами, що в підсумку дає остаточний вихідний сигнал y. Це може бути, наприклад, вектор ймовірностей для різних можливих патернів (форм), виявлених на вході (зображення). Щоб правильно навчити CNN, використовується функція втрат $\mathcal{L}(w)$ для вимірювання відстані між виходом **y** і контрольним значенням **y***. Мета навчання - знайти найкращі коефіцієнти ваги w, щоб мінімізувати функцію втрат $\mathcal{L}(w)$. Цього можна досягти зворотним поширенням через градієнтний спуск.

Зворотне поширення: під час зворотного поширення обчислюється градієнт функції втрат L (w) за вагою останнього прихованого шару і оновлюється вага шляхом обчислень:

$$w_4 = w_4 - \lambda \frac{d\mathcal{L}(w)}{dw_4} \tag{2.2}$$

Тут λ позначає швидкість навчання, яка контролює значення кроку руху в напрямку, зазначеному градієнтом. Така ж операція виконується для кожного вагового коефіцієнту по ланцюгу. Процес повторюється, і в кінцевому підсумку градієнтний спуск призведе до набору, який мінімізує $\mathcal{L}(w)$. Для інших структур NN процеси навчання і виведення аналогічні.

2.3. Застосування глибинного навчання для мобільних мереж

Глибоке навчання має широкий спектр додатків в мобільних і бездротових мережах. Нижче представляємо найбільш важливі результати досліджень застосування глибинного навчання в різних областях мобільних мереж і порівняно їх дизайн і принципи.

1. **Аналіз мобільних даних на рівні мережі на основі глибинного навчання** фокусується на додатках глибинного навчання, побудованих на великій кількості мобільних даних, зібраних в мережі, включаючи прогнозування мережі, класифікацію трафіку і інтелектуальний аналіз записів відомостей про дзвінки (CDR).
2. **Аналіз мобільних даних на рівні додатків на основі глибинного навчання** зосереджує увагу на аналітиці мобільних даних на периферійних пристроях.
3. **Аналіз мобільності користувачів на основі глибокого навчання** пропонує використання глибоких нейронних мереж для розуміння моделей пересування користувачів на груповому або індивідуальному рівні.

4. **Локалізація користувачів, заснована на глибокому навчанні**, в основу покладено використання глибинних нейронних мереж для локалізації користувачів в приміщенні або на відкритому повітрі на основі різних сигналів, отриманих від мобільних пристроїв або бездротових каналів.
5. **Управління мережею на основі глибокого навчання** досліджує використання глибинного навчання з підкріпленням і глибинного імітаційного навчання для оптимізації мережі, маршрутизації, планування, розподілу ресурсів і радіоуправління.
6. **Мережева безпека, заснована на глибокому навчанні**, використовує глибинне навчання для поліпшення мережевої безпеки, яку можливо згрупувати в залежності від інфраструктури, програмного забезпечення і конфіденційності.
7. **Обробка сигналів, заснована на глибокому навчанні**, вивчає аспекти фізичного рівня, які можливо покращити за допомогою глибинного навчання.

2.4. Прогнозування міського руху за допомогою глибинного навчання

Інформація про дорожній рух має велике значення для великих міст, і протягом багатьох років ведеться робота по точному прогнозуванню транспортних потоків. Прогнозування міського трафіку спрямоване на використання складних моделей для захоплення прихованих характеристик трафіку з історичних даних про мобільність, а потім використання навчених моделей для прогнозування умов дорожнього руху в майбутньому. Завдяки потужним можливостям вивчення уявлень і вилучення ознак глибинне навчання стає потужною альтернативою такому моделювання трафіку.

Вичерпна інформація про міський рух приносить користь повсякденному життю громадян та покращує ефективність міського транспорту. Точні прогнози

такої інформації про дорожній рух необхідні для планування маршруту, навігації та інших служб мобільності. Також ця інформація може бути використана для планування розгортування нових мобільних мереж, або оптимізації існуючих. У прогнозуванні міського трафіку зазвичай використовуються моделі для аналізу різних історичних даних і даних про трафік в реальному часі для прогнозування майбутніх умов руху.

Завдяки популярності в останні роки широкому застосуванню датчиків та інтелектуальних транспортних систем (ITS), зараз можливо збирати величезний масив даних про мобільність за допомогою різних мобільних пристроїв (таких як смартфони і бортові пристрої GPS) і пристроїв автоматичного збору плати за проїзд (AFC), що широко використовуються міськими транспортними системами (наприклад, метро, автобуси і таксі).

Щоб використовувати ці переваги, традиційні методи використовують статистичні моделі або моделі машинного навчання для прогнозування потоків трафіку. Вони покладаються на створені людиною функції для виявлення і захоплення основних характеристик трафіку і подальшого отримання миттєвих вимірювань умов трафіку в якості вхідних даних разом з моделями, побудованими на отриманих характеристиках, для прогнозування майбутніх умов руху. Однак на практиці на транспортні потоки можуть впливати різні чинники, наприклад, правила перевезення, погодні умови і так далі. Через складність можливих кореляцій, функції створені людиною не здатні враховувати усі можливі впливи, отже, не можуть забезпечити точні прогнози.

Доступність великих даних і можливість швидкої обробки цих даних разом роблять можливим і перспективним для впровадження алгоритми глибинного навчання. У порівнянні з класичними моделями машинного навчання, наприклад, SVM і ANN, які мають тільки поверхневу архітектуру для захоплення функцій, моделі глибинного навчання використовують багаторівневу (тобто

«глибинну») архітектуру для виявлення складних структур і складних шаблонів, де різні шари захоплюють об'єкти з різних точок зору і, нарешті, разом утворюють багаторівневу абстракцію.

2.4.1. Концепції прогнозування міського руху

Прогнозування міського руху стосується прогнозування умов дорожнього руху на період від кількох секунд до кількох годин на майбутнє на основі поточної та історичної інформації дорожнього руху. Системи спрямовані на точне моделювання транспортних індикаторів, таких як швидкість руху, транспортний потік, передбачення аварій, а також передбачення умов руху. На рисунку 2.3 показана високорівнева процедура прогнозування міського трафіку, включаючи збір даних з мобільності, розширене моделювання трафіку і цілі прогнозування трафіку.

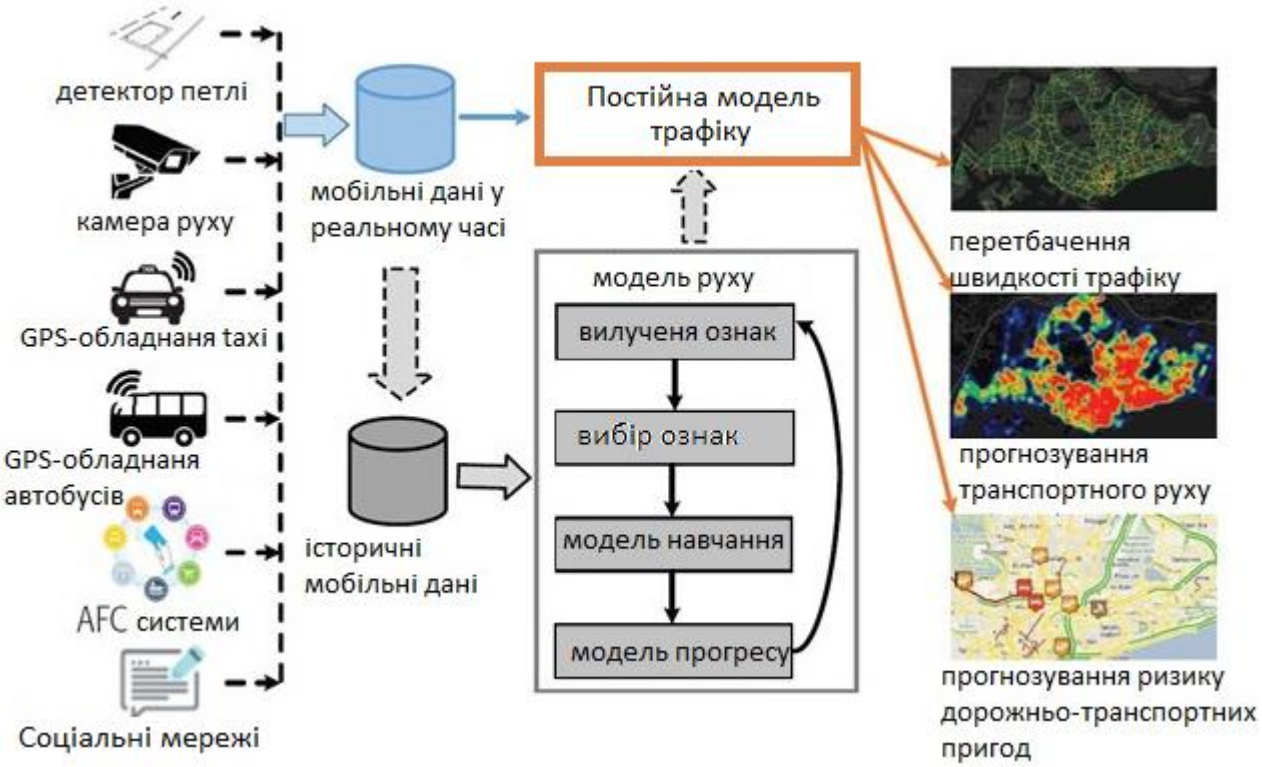

Рис. 2.3. Основні компоненти прогнозування міського руху

Дані про мобільність, що беруть участь в прогнозуванні трафіку, можна розділити на наступні категорії.

Дані про трафік з міської інфраструктури: багато пристроїв інфраструктури, наприклад, петльові детектори і камери трафіку, були розгорнуті в містах для постійного збору інформації про дорожній рух. Детектори петель приховані під смугами руху деяких важливих доріг і можуть виявляти проїжджаючі автомобілі. Такі вимірювання використовуються для розрахунку швидкості руху кожного окремого транспортного засобу, а також підрахунку загальної кількості транспортних засобів, що проїжджають повз (тобто транспортного потоку). Так само й камери розміщуються над перехрестями доріг і використовуються для зйомки проїжджаючих автомобілів. На основі методів комп'ютерного зору також можна визначити швидкість руху транспортних засобів і транспортні потоки.

Маршрутні дані від транспортних засобів: в урбаністичних містах велика кількість громадських транспортних засобів (наприклад, таксі та автобуси) було обладнано пристроями GPS і, таким чином, може періодично повідомляти про свій статус, включаючи місце свого перебування, швидкість руху, напрямок і т. д ., звіти показують маршрути транспортних засобів, які містять вимірювання стану дорожнього руху на дорогах.

Записи AFC від транзитних систем: сучасні мережі громадського транспорту в значній мірі покладаються на пристрої AFC для автоматичного збору плати за проїзд з пасажирів автобусів і метро, яким необхідно підключати свої смарт-карти, що зчитує пристрій AFC, коли вони входять і виходять з автобусів або метро. Таким чином, системи AFC реєструють станції посадки/висадки (автобус, метро або трамвай, такі записи можна використовувати для побудови матриці відправлення-призначення (OD) поїздки, яка показує потоки мобільності.

Інші джерела даних: існують і інші джерела даних, які можна використовувати для прогнозування трафіку. Наприклад, звіти про аварії, які містять місце розташування, тяжкість і події кожної аварії, надають корисну інформацію для оцінки потенційного ризику аварій в кожній точці в межах міста. Сервіси соціальних мереж можуть розглядати людей як датчики для дослідження динаміки в місті, таким чином, дані соціальних мереж також можуть допомогти зробити висновки про аномалії руху (наприклад, аваріях). Дані стільникового зв'язку з мобільних пристроїв вказують на пересування користувачів в межах міста на рівні веж стільникового зв'язку і дають підказки для припущень про умови дорожнього руху. Крім того, дані зондування з систем краудсорсингу також є важливим джерелом даних для прогнозування трафіку.

Міський рух складний і, як правило, нелінійний, отже, деякі передові моделі дорожнього руху є кращими, наприклад, статистичні моделі або моделі машинного навчання, щоб фіксувати приховані характеристики трафіку з даних мобільності, а потім полегшувати прогнозування на основі введення даних реального часу.

Як показано на рис. 2.3, вдосконалене моделювання трафіку - це ітераційний процес, який складається з декількох фаз. Щоб побудувати модель трафіку, спочатку потрібно витягнути деякі бажані значення (тобто особливості) з необроблених даних мобільності. Такий набір ознак корелює із цільовими умовами руху. Беручи як приклад стан дорожнього руху C_i сегменту дороги S_i, на C_i впливають не тільки умови руху сусідніх сегментів дороги S_i' в просторовому вимірі, але також впливає час доби (наприклад, час пік та час простою), день тижня (наприклад, робочий день та вихідні) у часовому вимірі. Ці просторово-часові фактори разом визначають еволюцію C_i та відіграють важливу роль у точному прогнозуванні його майбутнього статусу. Після етапу вилучення ознак, на основі деяких критеріїв, наприклад, інформаційна ентропія, для

спрощення моделювання та розширення можливостей узагальнення моделі обирається невеликий набір найбільш відповідних ознак. Побудувавши модель трафіку лише з використанням найбільш інформативних і ненадлишкових функцій, можливо налаштувати параметри за допомогою масиву навчальних даних та оцінити отриману модель за допомогою даних тестування. Весь процес моделювання трафіку можна повторити до досягнення цільових показників (наприклад, точність). Модель постійного трафіку - це модель, яка кодує характеристики трафіку і може бути використана для прогнозування трафіку, враховуючи дані вхідної мобільності в реальному часі.

Існуючі роботи в основному покладаються на моделі, такі як ARIMA, ANN та SVM, щоб зафіксувати складний трафік. При побудові таких моделей трафіку вилучення та вибір особливостей є суттєво важливими, оскільки вони визначатимуть остаточну ефективність моделі трафіку. Однак ці процедури в значній мірі залежать від створеної людиною функцій, які вимагають багатого досвіду та експертних знань.

2.4.2 Прогнозування швидкості руху, транспортного потоку та ризику дорожньо-транспортних пригод

Відповідно до цілей прогнозування, прогнози міського руху можуть бути додатково поділені на прогнозування швидкості руху, прогнозування транспортного потоку і прогнозування ризику дорожньо-транспортних пригод. У таблиці 2.2 наведені типи прогнозів трафіку, а також відповідні джерела даних і бажаний результат.

Таблиця 2.2. Зведення різних прогнозів міського трафіку

Категорія	**Залучені джерела даних**	**Бажаний результат**
Прогнозування швидкості руху	Міська інфраструктура, вбудовані GPS	Середня швидкість руху (або ступінь заторів)
Прогнозування транспортного потоку	Міська інфраструктура, AFC система	Загальна кількість об’єктів, що проходять дорогою/регіоном
Прогнозування ризику дорожньо-транспортних пригод	Міська інфраструктура, AFC система, дані в соціальних мережах, історичні дані	Ймовірність аварії для кожної дороги/регіону

Швидкість руху - це широко використовуваний індикатор для вимірювання умов руху на одній ділянці дороги, який зазвичай розраховується як середня швидкість руху всіх транспортних засобів, які відбирають проби, на даній ділянці дороги. Системи отримують такі вимірювання швидкості транспортних засобів або побічно з даних, зібраних петльовими детекторами і камерами, або безпосередньо з транспортних засобів, оснащених GPS. Вони створюють модель швидкості руху на основі історичних даних, застосовуючи класичні моделі машинного навчання, і використовують швидкість вибірки в реальному часі в якості вхідних даних для прогнозування швидкості руху в майбутньому. Прогнозовані швидкості трафіку можуть бути переведені на певні рівні заторів (наприклад, повільний, нормальний і швидкий) відповідно до деяких правил картографування.

Загалом, транспортний потік визначається як загальна кількість цільових об'єктів (наприклад, транспортних засобів або людей), які проходять через

територію за період часу. Район може бути відрізком дороги або районом міста. На відміну від традиційних робіт, в яких міститься багато припущень про мобільність людей, більш сучасні підходи моделюють і прогнозують потоки руху на основі реалістичних даних про мобільність людей, зібраних з міських інфраструктур і систем AFC. Транспортні потоки виявляють рух натовпу і потенційно визначають розподіл трафіку.

Дорожньо-транспортні пригоди, хоча і відбуваються рідко, але серйозно впливають на міське рух. Отже, необхідно оцінювати ризики дорожньо-транспортних пригод для кожної конкретної дороги і регіону, які можна виміряти як ймовірність, тобто наскільки ймовірно, що дорожньо-транспортні пригоди можуть статися на дорозі/регіоні. Недавні практики в основному пов'язують ризики нещасних випадків з поточними умовами руху і мобільністю людей, і тому вони розробляють моделі для виявлення взаємозв'язків між даними про мобільність і історичними звітами про події для прогнозування ризику дорожньо-транспортних пригод.

2.4.3 Рішення на базі глибинного навчання

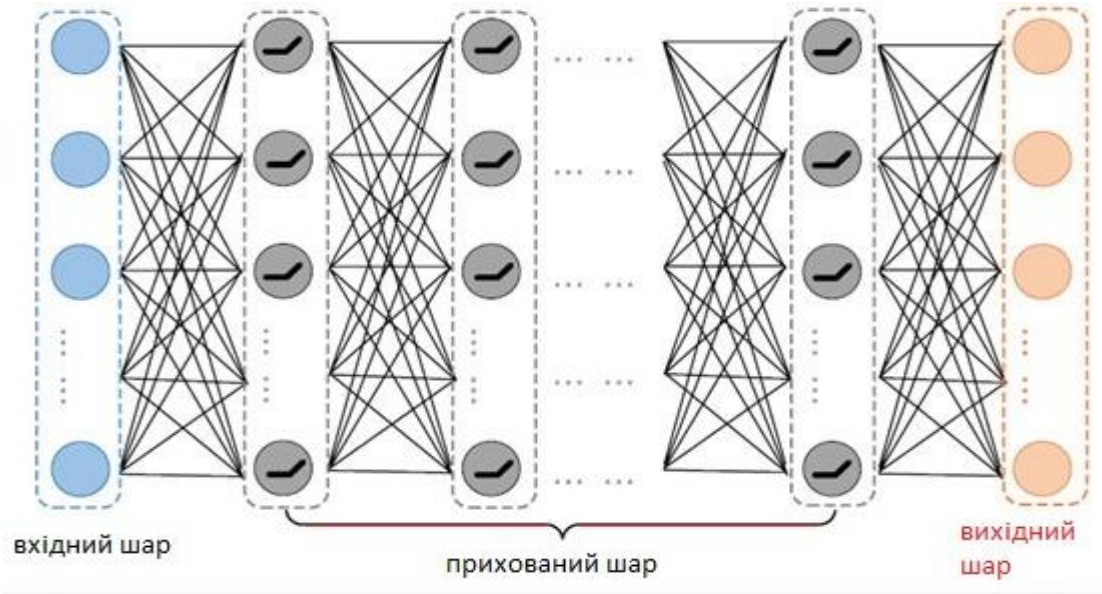

Рис. 2.4. Глибинна нейронна мережа з повнозв'язними шарами

Хоча існують різні форми моделей глибокого навчання, вони мають загальну архітектуру, як показано на рис. 2.4, яка містить вхідний рівень, вихідний рівень і від кількох до більш ніж тисячі прихованих шарів між ними. Необроблені дані ініціалізують значення вхідного шару, в той час як вихідний шар генерує бажані висновки. Всі приховані шари відповідають за перетворення станів вхідного рівня в очікувані висновки вихідного шару шляхом захоплення абстракцій високого рівня. Кожен рівень в мережі містить кілька одиниць, і розміри можуть відрізнятися для різних рівнів. Зв'язки існують між блоками будь-яких двох сусідніх шарів, і кожне посилання пов'язане з вагою. У кожної одиниці є функція активації, яка визначає, як обчислити свій власний стан на основі одиниць попереднього рівня, в свою чергу, передає свій стан наступному шару. Однією з найпопулярніших функцій активації є випрямлений лінійний блок (ReLU), який представляє собою напівхвильовий випрямляч $f(x) = \max(x,0)$. Далі наведено декілька перспективних моделей.

Модель CNN в основному призначена для обробки двовимірних даних, наприклад, зображень. Як показано на рис. 2.5, модель CNN складається з вхідного та вихідного шарів, а також безлічі прихованих шарів, якими можуть бути згорткові, об'єднані або повністю пов'язані шари. Згорткові шари використовують згорткові фільтри, які застосовують певні перетворення до вхідних даних для захоплення їх властивостей. Далі йдуть рівні об'єднання, які об'єднують вихідні дані поодиноких кластерів на попередньому рівні в єдиний блок на наступному рівні, використовуючи фільтр максимуму або мінімуму. Рівень об’єднання вивчає більш абстрактні подання даних, а тим часом діє як форма зменшення розмірності, щоб спростити всю модель. Повністю зв'язаний шар використовується для завершення виведення.

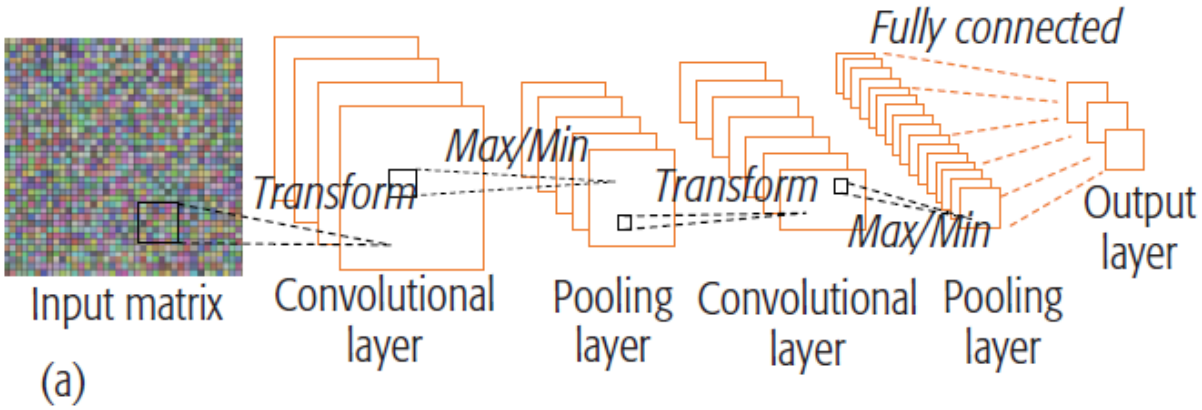

Рис. 2.5. Типова архітектура моделі CNN

Рекурентна нейронна мережа (RNN): RNN нейронні мережі - це нейронні мережі, які зберігають тимчасову розмірність даних часових рядів, використовуючи повторюваний прихований стан. Основне функціонування мереж RNN включає в себе цикли або "рекурентні" компоненти для підключення нейрона до себе і багаторазового розгортання, що дозволяє зберегти часову розмірність даних, знайдену з послідовних даних. Мережі RNN мають приховані стани, які оновлюються послідовною інформацією, отриманою з вхідних часових рядів даних, з виходами, що залежать від цих прихованих станів. Простий механізм розгортання RNN в мережі показаний на рисунку 2.6. U і V представляють ваги прихованого і вихідного рівнів відповідно, а W - ваги переходу в прихований стан. Приховане стан мережі на момент часу t обчислюється поелементним добутком вхідного вектору і попереднього мережевого прихованого стану h_{t-1}. Це обчислюється математично за допомогою:

$$h_t = \sigma(W_{hx}x_t + W_{hh}h_{t-1} + b) \quad (2.3)$$

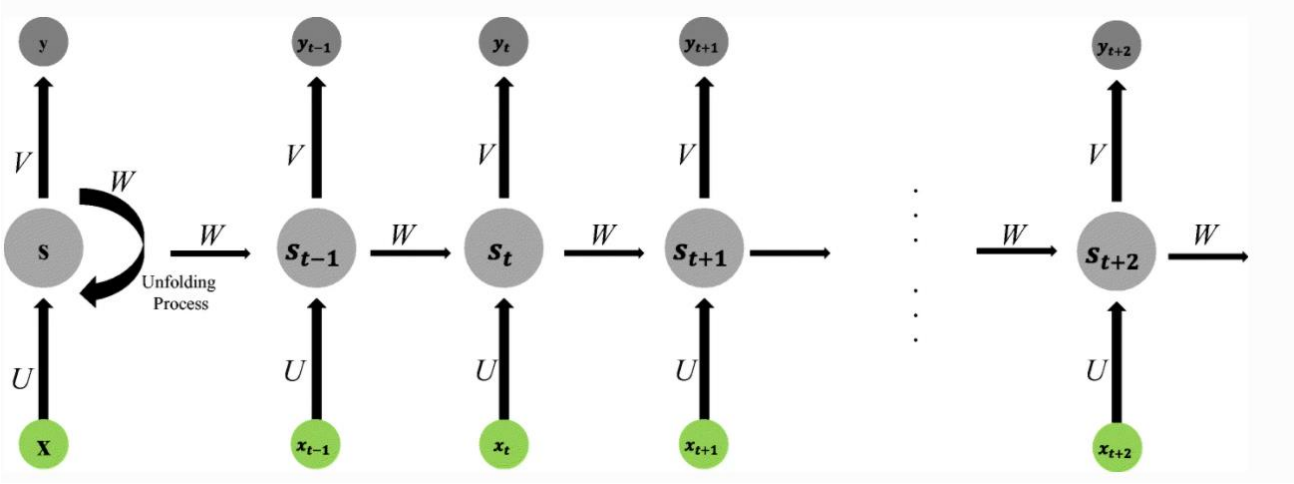

Рис. 2.6. Типова архітектура моделі RNN

Де W_{hx} представляє вага між вхідним і повторюваним прихованим вузлом, W_{hh} є вага між повторюваним вузлом і попереднім часовим кроком самого прихованого вузла, b і σ представляють зміщення і нелінійну (сигмоїдальну) активацію відповідно. Хоча RNN краще справляються з завданнями прогнозування часових рядів, у них все ще є проблеми, які ще треба вирішити. Наприклад, з рівняння 2.3 вище, повторюваний прихований стан h_t наближається до нуля відповідно зі збільшенням часового інтервалу, що призводить до проблеми зменшення градієнта. З цієї причини RNN не можуть вчитися на часових рядах, що мають великі часові затримки. Ця проблема вирішується допомогою роботи Long Short-Term Memory RNN - основною метою якої є моделювання довгострокових часових залежностей у часових рядах. Модель LSTM замінила повторюваний прихований блок на «комірку пам'яті».

Архітектура LSTM-NN, що має один блок пам'яті, зображена на рисунку 2.7. Блок пам'яті містить вхідні, вихідні та ворота збросу, які відповідно виконують функції запису, читання та скидання в кожній комірці. Мультиплікативні ворота, $\oplus$ та $\otimes$, відносяться до операторам складання матриць і скалярного добутку відповідно і дозволяють моделі зберігати інформацію

протягом тривалих періодів, тим самим усуваючи проблему зникаючого градієнта, яка зазвичай спостерігається у традиційних моделях нейронної мережі.

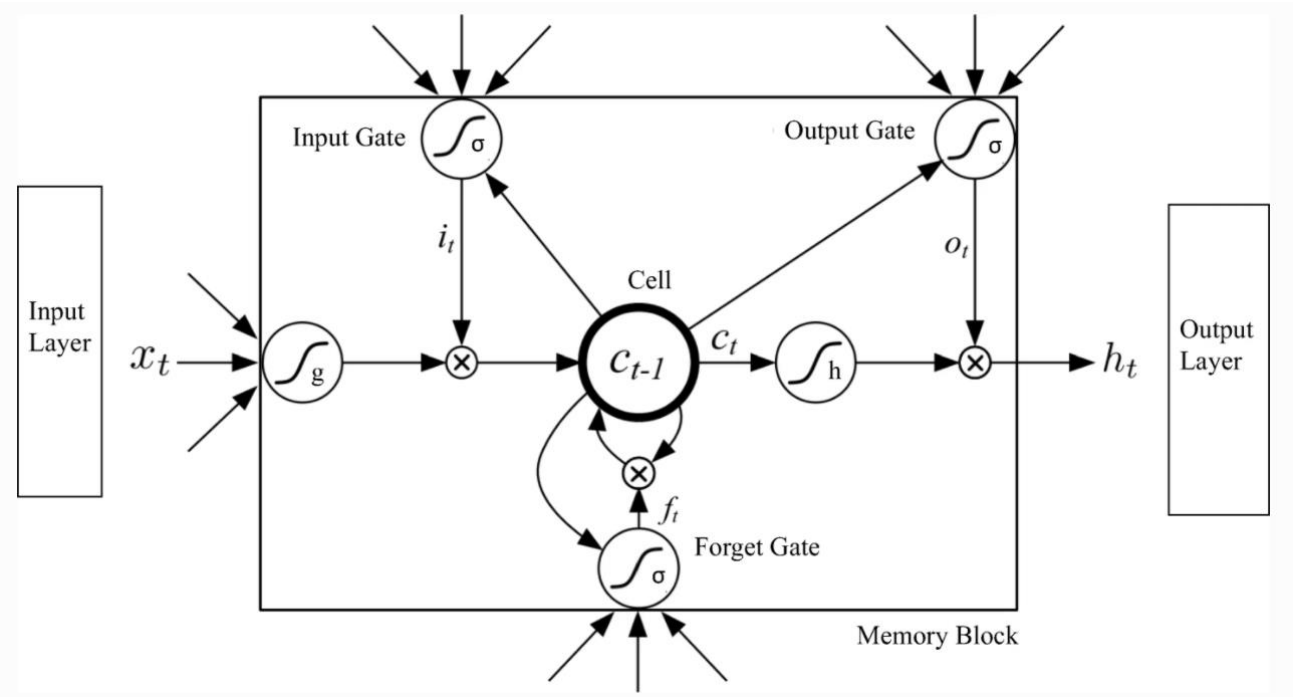

Рис. 2.7. Модель LSTM з одним блоком пам’яті

Глибинні нейронні мережі LSTM широко використовуються в дослідженнях прогнозування трафіку, модель LSTM-NN використовувалася для прогнозування швидкості трафіку, а результати порівнювалися з іншими непараметричними алгоритмами (Support Vector Machines (SVMs), Kalman Filter, та ARIMA). Результати показали перевагу моделі LSTM в точності прогнозування. Крім того, модель глибокого навчання LSTM та мережа глибоких переконань (DBN), для прогнозування короткострокової швидкості руху, з використанням даних про трафік і опади показали, що об'єднання джерел даних

Запропонована модель є вдосконаленою версією моделі, представленої вище, де було використано стекові автокодери для навчання моделі. Основною перевагою автокодерів є навчання стислому поданням (кодуванням) набору

вхідних векторів даних. Іншими словами, автокодер, який одночасно є методом зменшення розмірності часових рядів та інструментом стиснення даних при аналізі зображень, може бути навчений за допомогою дуже великих наборів даних за короткий час. З цієї причини, великі дані можна обробляти на значно коротших часових інтервалах за допомогою автокодерів LSTM, порівняно з звичайними глибокими нейронними мережами LSTM.

Автоенкодер - це нейронна мережа з прямим зв'язком, яка приймає вхідний вектор x і перетворює його в приховане уявлення або прихований простір h. Іншими словами, автокодери стискають вхідний вектор в «код» меншою розмірності і намагаються відновити вихідні дані з цього заданого уявлення. Автоенкодер складається з трьох основних компонентів: кодувальника, коду і декодера.

Глибинні двонаправлені LSTM

Глибинна мережа LSTM - це повторювана мережа, яка має багато шарів. Вона замінює звичайні повторювані блоки комірками пам'яті LSTM. Особливою перевагою застосування глибинних мереж LSTM є той факт, що вони можуть вивчати довгострокові залежності в складних структурах даних ієрархічно. У порівнянні з одно- або дворівневими (тобто поверхневим навчанням/мережами) LSTM-мережами, глибокі LSTM можуть ієрархічно витягувати часові залежності в складних часових рядах або послідовних наборах даних .

Структура двонаправленого LSTM складається з двох односпрямованих LSTM, покладених в протилежних напрямках. Отже, як попередні, так і майбутні вектори часового ряду застосовуються в двунаправленном циклі навчання LSTM. Таким чином, дані обробляються в обох напрямках з використанням двох окремих прихованих шарів, які потім передаються на один вихідний рівень. Рисунок 2.8 показана структура двонаправленого LSTM. Як видно, мережа

обчислює як пряму приховану послідовність $\vec{h}$, так і зворотну приховану послідовність $\overleftarrow{h}$. Вихідні дані потім обчислюються шляхом ітерації зворотного шару в зворотному хронологічному порядку (тобто від $t = T$ до 1), в той час як прямий шар повторюється від $t = 1$ до T. Таким чином, глибинний двонаправлений LSTM є глибинною двобічною мережею LSTM, яка є критичним компонентом успіху, записаного в архітектурі глибинного навчання. Як зазначалося раніше, мережі глибинного навчання можуть ієрархічно створювати уявлення шарів в складних наборах даних. Глибокі двонаправлені LSTM створюються шляхом вертикального накладення декількох рівнів двонаправленої LSTM. Таким чином, вихідна послідовність для одного шару слугує вхідною послідовністю для наступного шару.

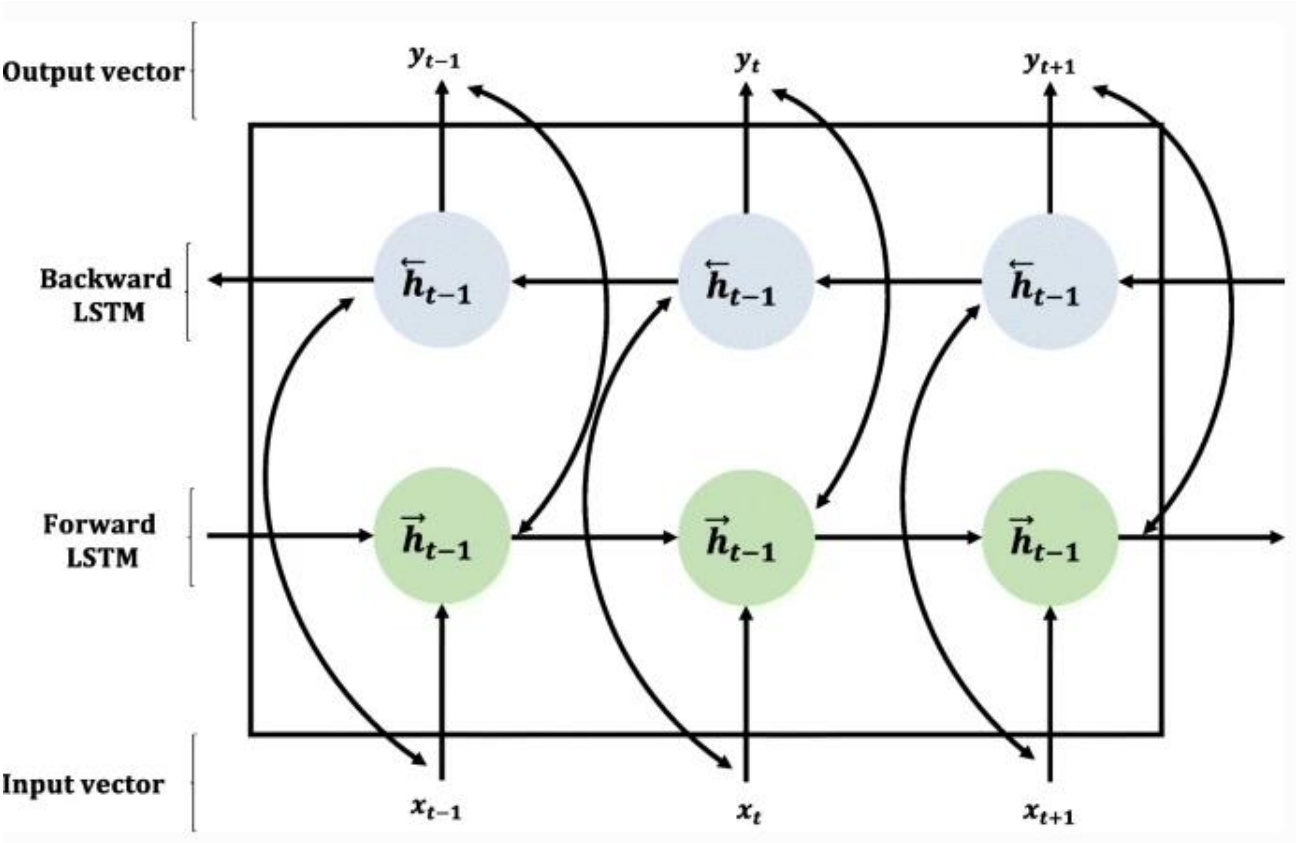

Рис. 2.8. Архітектура двонаправленої LSTM

2.4.4 Оцінка продуктивності моделі

Для цього дослідження використано обернений метод оцінки прогнозів. Звичайні методи оцінки, такі як k-кратна перехресна перевірка, не підходять для використання в даних часових рядів, тому що вони не враховують часовий або послідовний порядок, або розмірність вхідного набору даних. Було застосовано два показника оцінки точності статистичного прогнозування - середня абсолютна помилка (MAE) і симетрична середня абсолютна процентна помилка (sMAPE), які визначаються наведеними нижче рівняннями.

$$MAE = \frac{1}{n}\sum_{e=1}^{n}|e_i|, \tag{2.4}$$

$$sMAPE = \frac{200}{n}\sum_{i=1}^{n}\left|\frac{x_i - yi}{x_i + y_i}\right| \tag{2.5}$$

де e_i, $i = 1, 2, \ldots n$ являє собою n зразків модальних помилок,

x_i та y_i відповідно представляють вхідні та вихідні значення.

Далі проводиться порівняння продуктивності запропонованої моделі з обраними сучасними базовими моделями машинного навчання. Використовуючи показники оцінки продуктивності, описані в формулами (2.4) і (2.5), було порівняно запропоновану модель з такими базовими моделями: підтримуючий векторний регресор (SVR), екстремальне посилення градієнта (xGBoost) і регресор RandomForest. Для кожної з базових моделей використовувався ідентичний навчальний набір даних, щоб гарантувати справедливість і об'єктивність в процесі оцінки моделі.

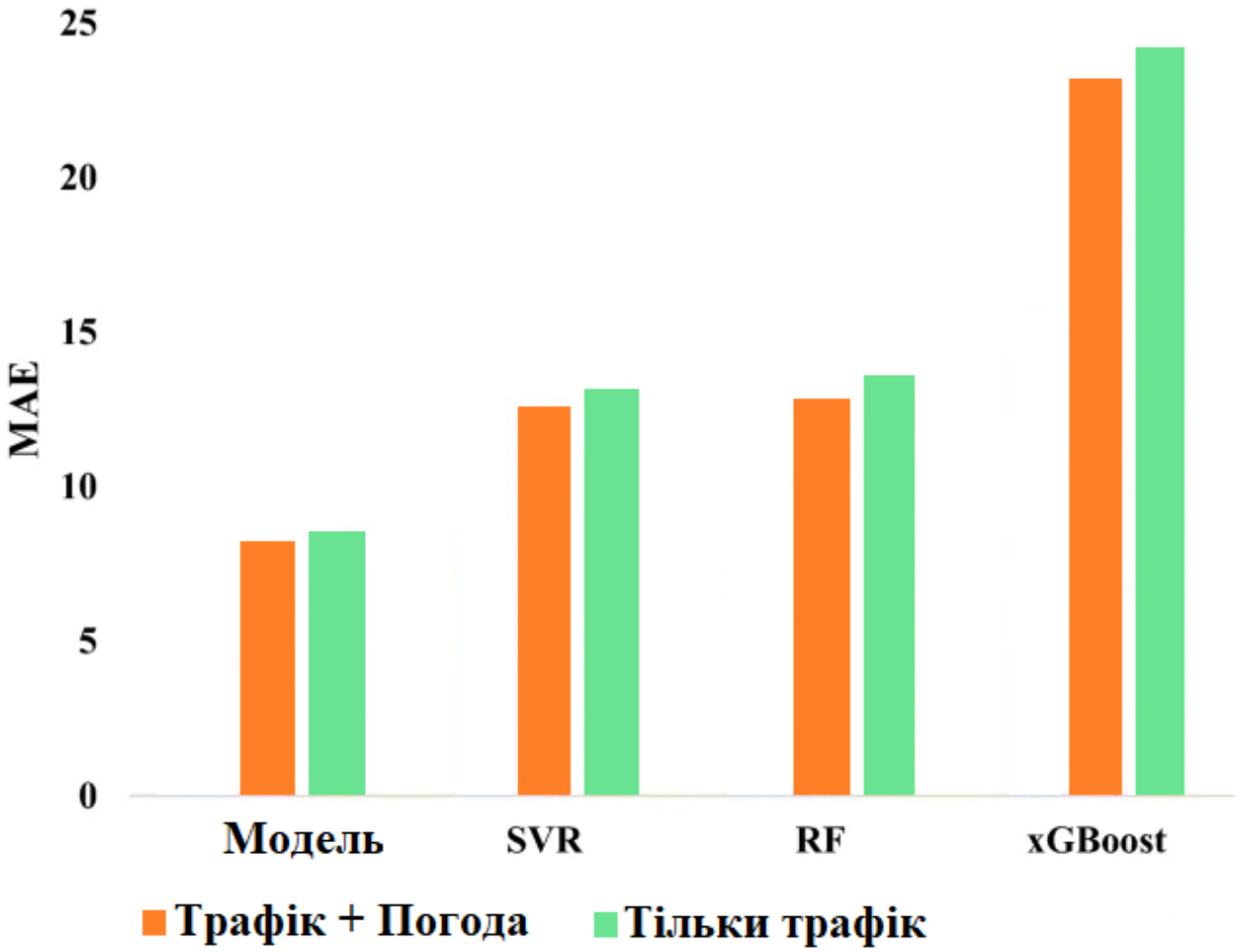

Рис. 2.9. Показник оцінки точності MAE для різних систем глибинного навчання

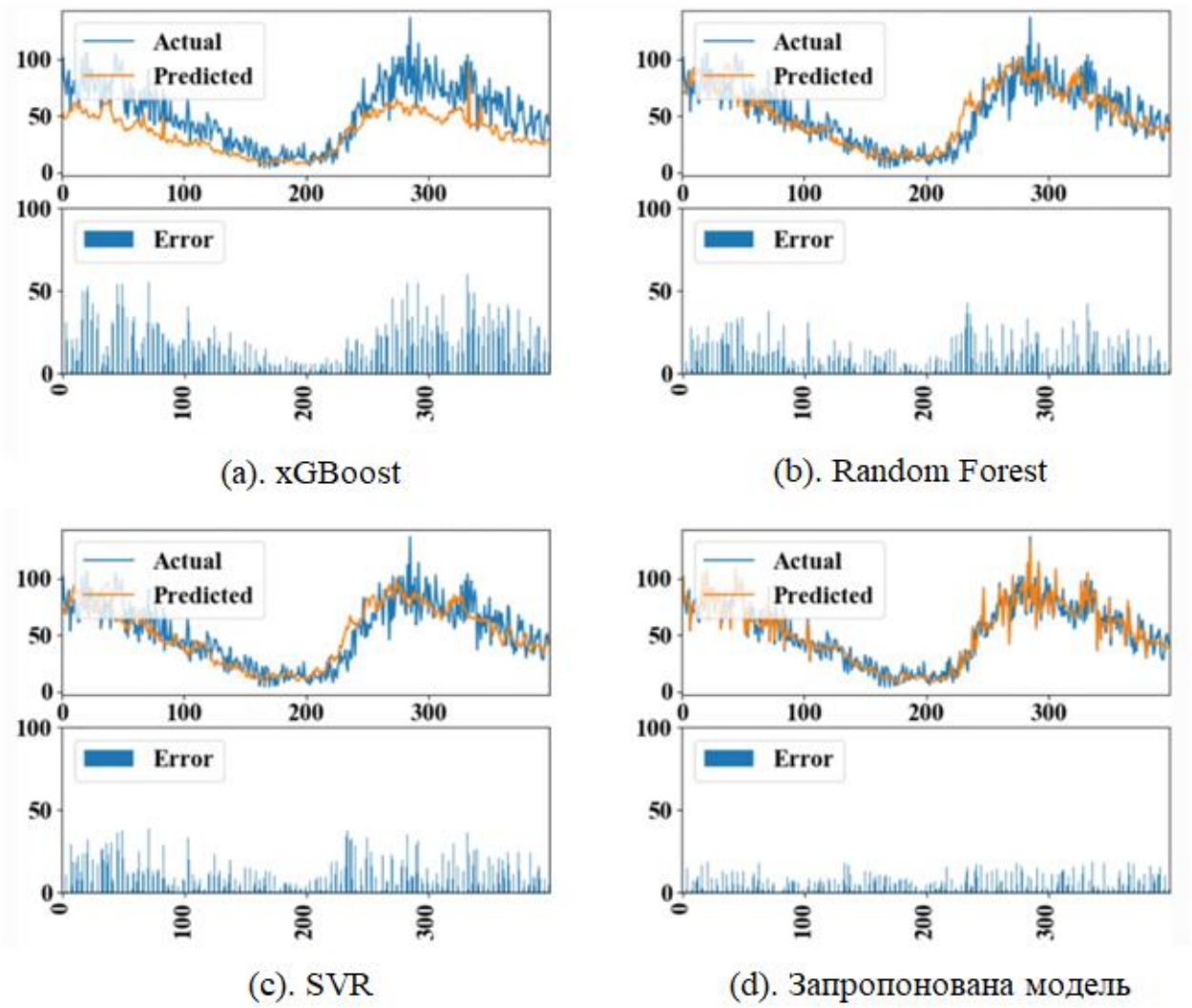

Рис. 2.10. Показник оцінки точності sMAPE для різних систем глибинного навчання

Як видно з рисунків 2.9, 2.10, підвищилася точність прогнозування при використанні наборів даних погоди і трафіку разом. Можна також зауважити, що запропонована модель, перевершила традиційні базові показники машинного навчання.

Висновки до розділу 2

1. Глибинне навчання є найбільш ефективним методом машинного навчання для інтеграції в системи обробки даних з мобільних пристроїв. Основними перевагами перед іншими методами є здатність працювати з великими масивами необроблених даних.
2. Моделі глибинного навчання є досить гнучкими і здатні виконувати різні задачі без переналаштування моделі.
3. Глибинне навчання ефективне при обробці геометричних мобільних даних. Як наслідок, знаходить застосування в аналізі даних розташування і переміщення користувача, точок підключення до мережі та геометричному аналізі.
4. У цьому розділі було розглянуто двонаправлену LSTM систему глибинного навчання для прогнозування міського трафіку. В результаті проведеного дослідження точність прогнозування запропонованої моделі перевершила показники існуючих систем глибинного навчання.

РОЗДІЛ 3 СИСТЕМА ПРОГНОЗУВАННЯ ВІДТОКУ КОРИСТУВАЧІВ НА БАЗІ МОДЕЛІ ЛОГІСТИЧНОЇ РЕГРЕСІЇ.

3.1 Відтік користувачів

Відтік клієнтів став надзвичайно важливим для компаній через посилення конкуренції в галузі, зростаючу важливість маркетингових стратегій та свідому поведінку споживачів протягом останніх років. Клієнти можуть легко схилятися до альтернативних послуг, можуть переключатися між операторами без будь-яких труднощів у будь-який час.

постійно шукає кращих послуг за нижчою вартістю Компанії повинні розробляти різні стратегії запобігання таким тенденціям, залежно від послуг, які вони надають. Під час оцінки можливого відтоку можуть використовуватися білінгові дані попередніх років.

Щорічний відтік користувачів у телекомунікаційній галузі становить близько 30 відсотків. Існує три основні стратегії отримання більших доходів: придбання нових клієнтів, збільшення продажів існуючих клієнтів та збільшення періоду утримання клієнтів. Однак порівняння цих стратегій з урахуванням величини рентабельності інвестицій кожної з них показало, що третя стратегія є найбільш вигідною.

Витрати через відтік клієнтів для європейських та американських телекомунікаційних компаній, за оцінками, становлять 4 мільярди доларів США щорічно. А коефіцієнт (витрати на залучення клієнтів / витрати на утримання або задоволення споживачів) для постачальників бездротового зв’язку становить близько 8. Здається розумним інвестувати більше в управління відтоком. замість залученням нових користувачів, особливо, якщо вартість придбання нового клієнта у вісім разів більша, ніж утримання існуючого.

3.2 Методологія побудови системи передбачення відтоку

Оскільки ми маємо доступ до існуючих даних клієнтів, ми можемо спробувати створити модель машинного навчання, щоб передбачити відтік клієнтів. Побудову системи передбачення відтоку можна поділити на такі етапи:

- Модель на основі логістичної регресії
- Підготовка даних
- Підгонка моделі
- Прогнозування

Далі використана низка метрик оцінки машинного навчання (ROC, AUC, чутливість, специфічність), а також метрики, орієнтовані на бізнес (економія коштів). Створена модель є найпростішою для розуміння, для уникнення зайвої абстракції, водночас демонструє високі показники ефективності. Такий підхід дозволить показати економічний вплив на існуючу бізнес модель та дозволить зробити висновки, щодо перспектив використання алгоритмів машинного навчання в телекомунікаціях.

3.3 Теоретична модель на базі логістичної регресії

Логістична регресія є лінійним класифікатором. Оскільки ми намагаємося передбачити наявність відтоку або його відсутність, то модель класифікації - саме те, що нам потрібно.

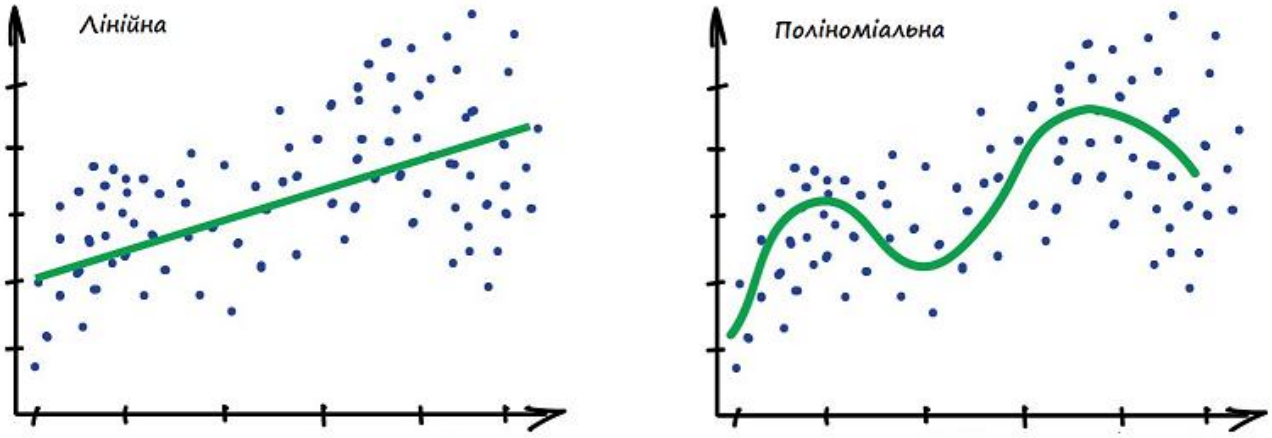

Рис. 3.1

Це чудова модель, оскільки її легше інтерпретувати, ніж багато інших моделей, таких як Random Forest. Що мається на увазі під поняттям "інтерпретувати", це те, що ми можемо легше побачити взаємозв'язок між ознаками та результатом.

Недоліком логістичної регресії є те, що вона має ухил до лінійних підходів. Якщо межа прийняття рішення не є лінійною, вона може не працювати так добре, як модель Random Forest. В основному системи покладено її простота та гнучкість. Це завжди враховується при впровадженні моделей машинного навчання.

У нашому випадку це бінарна логістична регресія. Логістична регресія приймає реальне значення і робить передбачення подібно до вхідного класу, що має значення 0. Якщо передбачення > 0,5, тоді воно приймає вихідне значення як клас 0, інакше висновок приймається як клас 1 (тут клас 0 має значення що клієнт не схильний до відтоку, а 1 – що користувач буде втрачений). Логістична регресія досягається шляхом прийняття логарифмічного коефіцієнта

$$Z_i = \ln\left(\frac{P_i}{1 - P_i}\right) = \beta_0 + \beta_1 X_1 + .. + \beta_n X_n$$

Де ***P*** ймовірність відтоку чи не відтоку. ***P*** завжди буде в діапазоні від 0 до 1.

β – коефіцієнти, розрахунок яких є задачею бінарної логістичної регресії, а

$X_1 - X_n$ незалежні змінні. Тут відтік користувачів є залежною змінною, решта - незалежні.

В алгоритмі машинного навчання значення коефіцієнта оцінюється за допомогою стохастичного градієнта. Він просто обчислює значення прогнозу для кожного екземпляра в навчальному наборі та обчислює помилку для кожного прогнозу. Крім того, цей процес триває, поки модель не стане досить точною. Крім того, коефіцієнт постійно оновлюється в процесі.

3.4 Процес побудови моделі

3.4.1 Вибір середовища розробки

Для вирішення поставлених задач була обрана мова програмування **R**. R - мова програмування, використовується для статистичної обробки даних і роботи з графікою, а також вільне програмне середовище обчислень з відкритим вихідним кодом в рамках проекту GNU.

```
R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
```

Рис. 3.2 версія ПЗ

Для обробки даних і роботи з графікою були встановлені наступні бібліотеки :

- ROCR
- Caret

- Tidyverse

```
> library(tidyverse)
-- Attaching packages --------------------------------------- tidyverse 1.3.0 --
v ggplot2 3.3.2     v purrr   0.3.4
v tibble  3.0.4     v dplyr   1.0.2
v tidyr   1.1.2     v stringr 1.4.0
v readr   1.4.0     v forcats 0.5.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
> library(caret)
Загрузка требуемого пакета: lattice

Присоединяю пакет: 'caret'

Следующий объект скрыт от 'package:purrr':

    lift

> library(ROCR)
>
```

Рис. 3.3 Встановлення додаткових бібліотек

3.4.2 Імпортування бази даних

Збір даних є дуже важливим і складним завданням для моделі прогнозування відтоку в телекомунікаціях. Будь-яка телекомунікаційна компанія не надаватиме базу даних громадськості, задля запобігання зловживання приватною інформацією клієнтів. Однак деякі бази даних телекомунікаційних компанії є загальнодоступними на веб-сайтах сховищ даних. База даних, використана в цьому дослідженні взята з IBM Sample Data Sets.

Зміст бази

Кожен рядок представляє клієнта, кожен стовпець містить атрибути клієнта, описані в стовпці метадані. Вихідні дані містять 7043 рядки (клієнти) та 21 стовпець (функції).

Набір даних включає інформацію про:

- Клієнтів, які відмовились від послуг протягом останнього місяця - графа називається Churn.
- Послуги, на які підписаний кожен клієнт - телефон, кілька ліній, Інтернет, безпека в Інтернеті, резервне копіювання в Інтернеті, захист пристрою, технічна підтримка та потокове ТБ та фільми.
- Інформація про рахунок клієнта - як довго вони є клієнтами, контракт, спосіб оплати, безготівкова оплата, щомісячні платежі та загальна вартість.
- Демографічна інформація про клієнтів - стать, віковий діапазон та наявність у них партнерів.

Далі імпортуємо дані у датафрейм datafr за допомогою функції read_csv

```
>  datafr <- read_csv("TelecoCC.csv")

-- Column specification --------------------------------
cols(
  .default = col_character(),
  SeniorCitizen = col_double(),
  tenure = col_double(),
  MonthlyCharges = col_double(),
  TotalCharges = col_double()
)
i Use `spec()` for the full column specifications.

>
```

Рис. 3.4 Підключення бази даних

```
> glimpse(datafr)
Rows: 7,043
Columns: 21
$ customerID       <chr> "7590-VHVEG", "5575-GNVDE", "3668-QPYBK", "7795-CFOCW...
$ gender           <chr> "Female", "Male", "Male", "Male", "Female", "Female",...
$ SeniorCitizen    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ Partner          <chr> "Yes", "No", "No", "No", "No", "No", "No", "No", "Yes...
$ Dependents       <chr> "No", "No", "No", "No", "No", "No", "Yes", "No", "No"...
$ tenure           <dbl> 1, 34, 2, 45, 2, 8, 22, 10, 28, 62, 13, 16, 58, 49, 2...
$ PhoneService     <chr> "No", "Yes", "Yes", "No", "Yes", "Yes", "Yes", "No", ...
$ MultipleLines    <chr> "No phone service", "No", "No", "No phone service", "...
$ InternetService  <chr> "DSL", "DSL", "DSL", "DSL", "Fiber optic", "Fiber opt...
$ OnlineSecurity   <chr> "No", "Yes", "Yes", "Yes", "No", "No", "No", "Yes", "...
$ OnlineBackup     <chr> "Yes", "No", "Yes", "No", "No", "No", "Yes", "No", "N...
$ DeviceProtection <chr> "No", "Yes", "No", "Yes", "No", "Yes", "No", "No", "Y...
$ TechSupport      <chr> "No", "No", "No", "Yes", "No", "No", "No", "No", "Yes...
$ StreamingTV      <chr> "No", "No", "No", "No", "No", "Yes", "Yes", "No", "Ye...
$ StreamingMovies  <chr> "No", "No", "No", "No", "No", "Yes", "No", "No", "Yes...
$ Contract         <chr> "Month-to-month", "One year", "Month-to-month", "One ...
$ PaperlessBilling <chr> "Yes", "No", "Yes", "No", "Yes", "Yes", "Yes", "No", ...
$ PaymentMethod    <chr> "Electronic check", "Mailed check", "Mailed check", "...
$ MonthlyCharges   <dbl> 29.85, 56.95, 53.85, 42.30, 70.70, 99.65, 89.10, 29.7...
$ TotalCharges     <dbl> 29.85, 1889.50, 108.15, 1840.75, 151.65, 820.50, 1949...
$ Churn            <chr> "No", "No", "Yes", "No", "Yes", "Yes", "No", "No", "Y...
>
```

Рис. 3.5 Структура і розмірність масиву вхідних даних

3.4.3 Підготовка вхідних даних

Перш ніж використовувати модель логістичної регресії, нам доведеться провести невелику очистку масиву.

Почнемо з перетворення символів змінних до коефіцієнтів, для цього в коді використовуємо функцію mutate_if, а також використаємо конвеєрний оператор %>%. Основне призначення цього оператора полягає в тому, що він робить код більш читабельним. Також необхідно змінити змінну SeniorCitizen із цілого числа на коефіцієнт. Далі перевіримо масив на відсутність значень. Ми можемо зробити це за допомогою функції map.

```
> datafr <- datafr %>% mutate_if(is.character, as.factor)
> datafr$SeniorCitizen <- as.factor(datafr$SeniorCitizen)
> datafr %>% map(~ sum(is.na(.)))
```

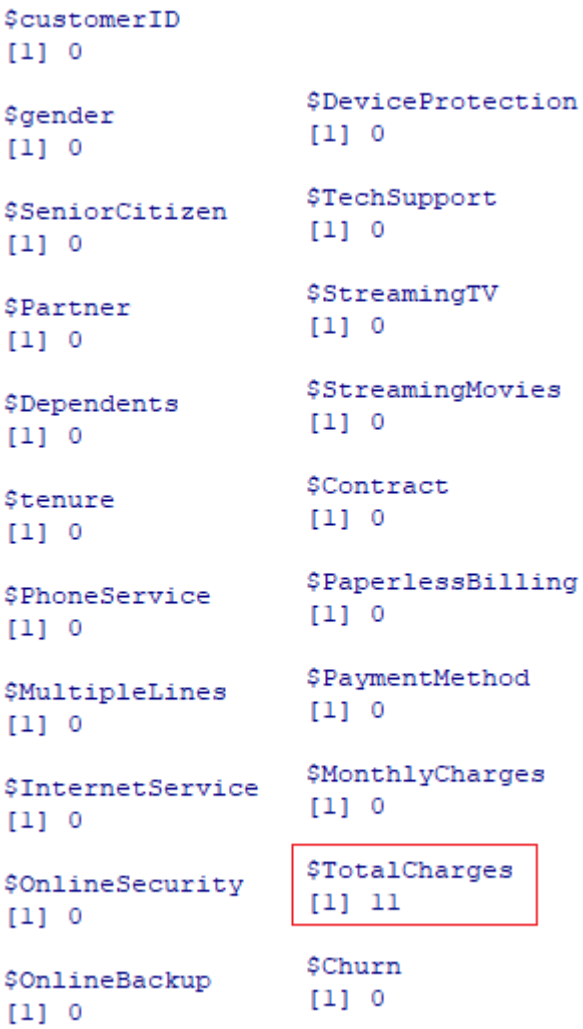

Рис. 3.6 Проміжний результат обробки вхідних даних

Ми бачимо, що TotalCharges має 11 відсутніх значень. Щоб замінити ці відсутні значення просто зробимо просту заміну.

Останнє, що треба зробити, - це вилучити функцію CustomerID. Це унікальний ідентифікатор для кожного клієнта, тому, ймовірно, він не додасть корисної інформації до нашої моделі.

```
datafr <- datafr %>%
  mutate(TotalCharges = replace(TotalCharges,
                                is.na(TotalCharges),
                                median(TotalCharges, na.rm = T)))
```

3.4.4 Поділ даних на групи тестування і навчання

Щоб переконатись, що ми не переобладнали нашу модель, ми розділимо дані на навчальний і тестовий набори. Це відоме як перехресна перевірка.

Ми навчатимемо модель на тренувальному наборі, а потім перевірятиме її ефективність на тестовому наборі даних. Ми випадковим чином відберемо 75% даних для нашого навчального набору та 25% для нашого тестового набору. Для більш точних результатів рекомендовано спробувати різні спліти, щоб побачити, як співвідносяться результати (можливі співвідношення тренувальних і тестових наборів 80% / 20% та 60% / 40%). Далі створимо тестові фрейми.

```
> set.seed(5)
> trainData <- createDataPartition(y = datafr$Churn, p=0.75, list=FALSE)
> train <- df[trainData,]
>
> test <- df[-trainData,]
>
```

Рис. 3.7 Створення тестового і навчального фрейму

3.4.5 Налаштування навчальної моделі

Тепер, коли розділили дані на навчальні та тестові набори, настав час підібрати модель. Для реалізації моделі логістичної регресії було використано функцію узагальнених лінійних моделей (GLM), glm.

Існують різні типи GLM, що включає логістичну регресію. Щоб вказати, що ми хочемо виконати двійкову логістичну регресію, ми будемо використовувати аргумент family = binomial. Ось повний код для встановлення моделі логістичної регресії:

```
fit <- glm(Churn~., data=train, family=binomial)
```

3.4.6 Прогнозування

Тепер, підігнавши модель, настав час подивитися, як вона працює. Для цього було зроблено прогнози, використовуючи набір тестових даних. Передано дані в модель підгонки з попереднього розділу. Для прогнозування ймовірностей вказуємо type = ”response”.

Треба перетворити ці ймовірності у двійкову відповідь, оскільки, змінна Churn “Так” або “Ні”. Переглядаючи результат, можна відмітити, що позитивний результат кодується як 1. Тепер знаючи кодування відповіді можливо перетворити передбачувані результати на відповіді Так і Ні. Встановимо поріг відповіді 0,5, тому, якщо передбачувана ймовірність перевищує 0,5 перетворимо цю відповідь на Так. Останній крок - перетворення відповідей символів коефіцієнти. Це робиться для того, щоб кодування було правильним для моделі логістичної регресії.

```
  > churn.probs <- predict(fit, test, type="response")

> head(churn.probs)
         1          2          3          4          5          6
0.04124488 0.02885267 0.68876825 0.79465353 0.60547789 0.03380915
> contrasts(datafr$Churn)
    Yes
No    0
Yes   1
> glm.pred = rep("No", length(churn.probs))
> glm.pred[churn.probs > 0.5] = "Yes"
> glm.pred <- as.factor(glm.pred)
>
```

Рис. 3.8 Перетворення ймовірності

Важливою частиною прогнозування є оцінка та перевірка моделі. Далі детально розглянемо деякі метрики оцінки та закінчимо більш суворим підходом до перевірки моделі, k-кратним перехресним підтвердженням.

3.4.7 Оцінка моделі

Після того як були зроблені прогнози, настав час оцінити модель. Корисним інструментом для швидкого здійснення цього є використання функції confusionMatrix від Caret. Ми передамо наш масив прогнозів glm.pred, а також фактичні результати тесту $Churn. Нарешті, ми визначимо позитивний клас як "Yes", використовуючи positive = " Yes ".

```
> confusionMatrix(glm.pred, test$Churn, positive = "Yes")
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1167  213
       Yes  126  254

               Accuracy : 0.8074
                 95% CI : (0.7882, 0.8256)
    No Information Rate : 0.7347
    P-Value [Acc > NIR] : 5.638e-13

                  Kappa : 0.4747

 Mcnemar's Test P-Value : 2.999e-06

            Sensitivity : 0.5439
            Specificity : 0.9026
         Pos Pred Value : 0.6684
         Neg Pred Value : 0.8457
             Prevalence : 0.2653
         Detection Rate : 0.1443
   Detection Prevalence : 0.2159
      Balanced Accuracy : 0.7232

       'Positive' Class : Yes
```

Рис. 3.9 Матриця помилок

Ця функція створює як матрицю помилок, так і іншу цікаву статистику. Матриця помилок показує, скільки правильних і неправильних прогнозів було зроблено для кожного класу. Ось швидкий погляд на матрицю помилок з нашої моделі:

Рис. 3.10 Матриця помилок

Видно, що модель правильно передбачила "No" 1167 разів, і неправильно передбачила "No", коли фактична відповідь була "Yes" 213 разів. Так само модель правильно передбачила "Yes", коли фактична відповідь була "Yes" 254 рази. У той же час вона неправильно передбачила "Yes", коли фактична відповідь була "No" 126 разів. Наша загальна точність становить 81%. Простою базовою моделлю є передбачення класу більшості, який у нашому випадку є "No". Якби ми просто передбачили мажоритарний клас, наша точність склала б 73%. У тестовому наборі 1760 спостережень, а 1293 - "No".

Деякі інші корисні показники - це чутливість та специфічність. Оскільки наші класи дещо незбалансовані (близько 73% - "No" та відповідно 27% - "Yes"), ці показники можуть бути більш корисними.

Чутливість вказує TPR. Це міра того, наскільки точно ми передбачали позитивний клас, який у нашому випадку є "Yes". Функція confusionMatrix повідомляє про це безпосередньо, але також можливо обчислити це самостійно

за формулою $TPR = \frac{TP}{TP+FN}$, де TNR – чутливість, а TP – істино позитивні випадки і FN – хибно негативні. Специфічність рахується за наступною формулою $TNR = \frac{TN}{TN+FP}$, де TN істино негативні випадки, а FP – хибно позитивні.

Ще однією корисною метрикою є площа під кривою робочої характеристики приймача (ROC), також відома як AUC. Спочатку впроваджений для аналізу радіолокаційних сигналів, ROC є графіком справжньої позитивної ставки проти хибної позитивної ставки. ROC є чудовим інструментом, оскільки він будує TPR проти FPR, оскільки порогове значення варіюється. Далі складемо графік кривої ROC, використовуючи бібліотеку ROCR.

```
> pred <- prediction(churn.probs, test$Churn)
> predf <- performance(pred, measure = "tpr", x.measure = "fpr")
> plot(predf)
>
```

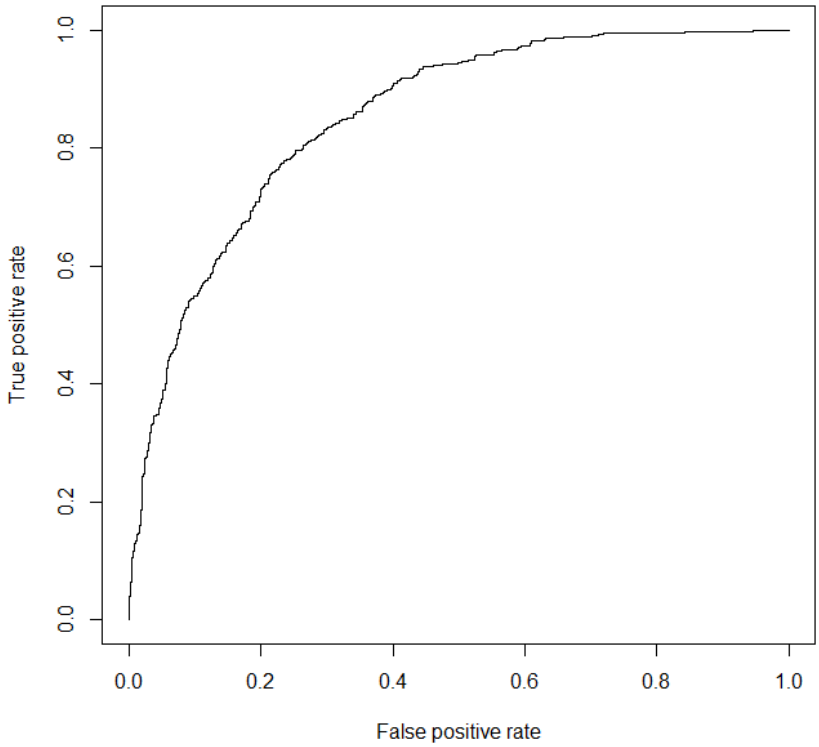

Рис. 3.11 Крива ROC

Як згадувалося раніше, ще одним корисним показником є площа під кривою ROC, відома як AUC. AUC може приймати будь-яке значення від 0 до 1, причому 1 є найкращим. Це зручний спосіб звести ROC до одного числа для оцінки моделі.

```
> auc <- performance(pred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8504366
> 
```

Рис. 3.12 Значення AUC

Наша модель має AUC 0,85, що досить добре. Якщо просто робити випадкові передбачення, крива ROC була би лінією під 45 градусів. Це відповідало б AUC 0,5. Можна зробити висновки, що запропонована модель дійсно демонструє гарні показники ефективності.

Тепер, коли модель навчена, протестована та оцінена, можливо провести поглиблену, більш ретельну оцінку.

Раніше вхідний масив даних був розділений на навчальні та тестові набори даних, що є непоганим способом запобігання переналаштування моделі. Ще кращий підхід – використання К-кратної перехресної перевірки.

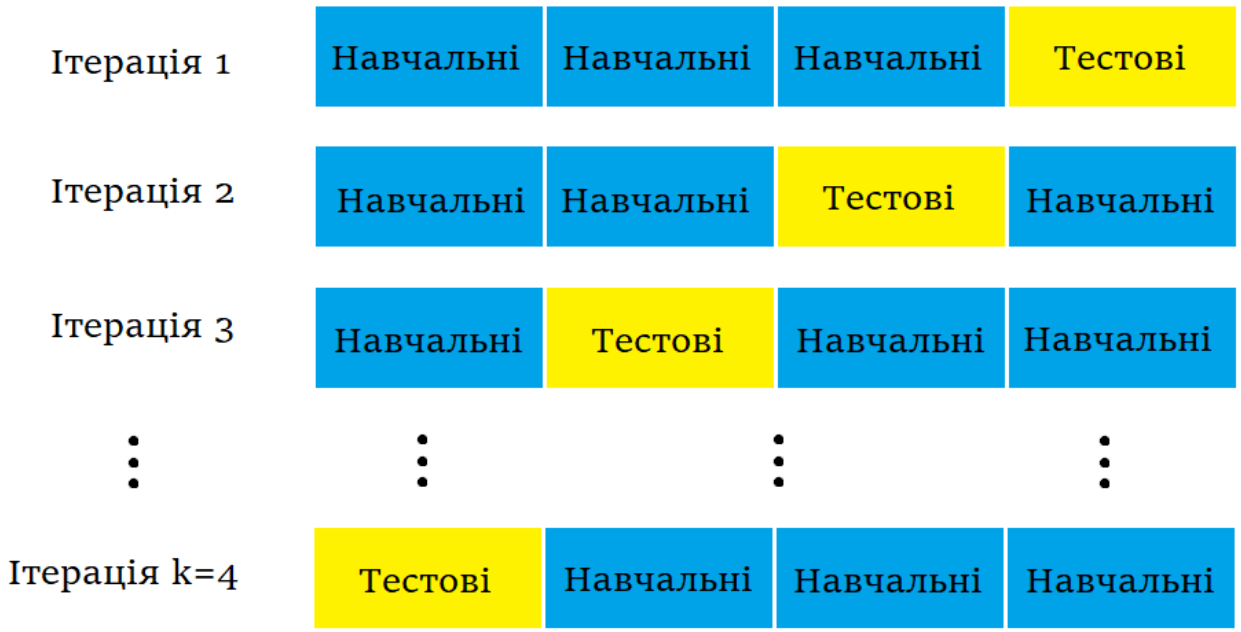

Рис. 3.13 k-кратна перехресна валідація вхідних даних, k=4

Для k-кратної перехресної перевірки моделі ми випадковим чином розподіляємо дані на тестові та навчальні набори, вказуючи певну кількість ітерацій. У наведеному вище прикладі кількість ітерацій становить k = 4.

Після того, як запуститься модель на кожній ітерації, проводиться оцінка метрики. Отже, якщо запустили модель чотири рази, використовуючи ROC, отримаємо усереднене кожне значення з чотирьох ROC. Це достатньо ефективний спосіб запобігти переналаштуванню моделі.

Загальна кількість ітерацій, які слід використовувати, - 10, тому саме така кількість ітерацій була використана для поділу вхідного масиву. Також процес був повторений 3 рази, задля того, щоб додати трохи більше технічної строгості підходу.

```
> set.seed(10)
> datafr$Churn <- as.character(datafr$Churn)
> datafr$Churn[datafr$Churn == "No"] <- "Y"
> datafr$Churn[datafr$Churn == "Yes"] <- "N"
> datafr$Churn[datafr$Churn == "Y"] <- "Yes"
> datafr$Churn[datafr$Churn == "N"] <- "No"
> fitControl <- trainControl(
+ method = "repeatedcv",
+   number = 10,repeats = 3,
+   classProbs = TRUE,
+   summaryFunction = twoClassSummary)
> logicRegres <- train(Churn ~., datafr,
+  method = "glm",
+                 family = "binomial",
+                 trControl = fitControl,
+                 metric = "ROC")

> logicRegres
Generalized Linear Model

7043 samples
  19 predictor
   2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 6338, 6339, 6339, 6339, 6338, 6338, ...
Resampling results:

  ROC        Sens      Spec
  0.8459719  0.550206  0.8975048

>
```

Рис. 3.14 Результат застосування 10-кратної перехресної валідації

Все починається з налаштування k-кратної перехресної валідації за допомогою функції trainControl. Усі вхідні дані досить прості. Як вже зазначалось раніше у коді використовується 10 ітерацій, повторених 3 рази. Далі відбувається тренування моделі. Знову використовуємо сімейство “binomial” із методу “glm”. Для оцінки моделі будемо використовувати “ROC”. Модель насправді повідомляє AUC, але спосіб, яким ми це вказуємо функція train з параметром metric = ”ROC”.

Результати схожі на ті, що ми отримали раніше. Як і раніше AUC дорівнює 0,85. Це повідомляється у виведенні як ROC, але насправді це AUC.

TPR (чутливість) становить 0,55, а TNR (специфічність) - 0,90.

3.5 Бізнес аналітика запропонованої моделі

Наразі було використано перехресну валідацію k-fold та логістичну регресію для прогнозування відтоку клієнтів. Було розглянуто корисні показники ефективності моделі, такі як AUC, чутливість та специфічність.

Реальна мета при розробці цієї моделі - показати вплив на бізнес. У цьому конкретному випадку це буде економія коштів. Далі буде поетапно продемонстрована економічна користь даної моделі.

Для початку було зроблено деякі припущення щодо різних витрат. Припустимо, що вартість залучення клієнтів у телекомунікаційній галузі становить приблизно 200 доларів. Якщо ми зробимо прогноз, що клієнт не схильний до відтоку, але він насправді належить до групи відтоку (помилково негативний, FN), то компанії доведеться витратити 200 доларів, щоб залучити нового клієнта.

Відомо, що залучити нового клієнта в 8 разів дорожче, ніж утримати існуючого. Таким чином для утримання клієнта компанії доведеться витратити 25 доларів.

Іноді модель правильно передбачує, що клієнт відмовляється від послуг (істинно позитивний, TP), а іноді ми будемо неправильно прогнозувати, що клієнт належить до групи відтоку (помилково позитивний, FP). В обох випадках ми витратимо 25 доларів, щоб утримати клієнта.

Нарешті, є сценарій, коли модель правильно прогнозує, що клієнт не перестане користуватись послугами компанії (істино негативний, TN). У цьому

випадку компанія не буде витрачати гроші. Це клієнти, які були коректно визначені системою.

Ось короткий підсумок витрат:

- **FN** (передбачають, що клієнт не припинить взаємодію з компанією, але насправді це робить): 200 доларів
- **TP** (істино передбачає, що клієнт схильний до відтоку): 25 доларів
- **FP** (прогнозують, що клієнт збиватиметься, коли насправді цього не робить): 25 доларів
- **TN** (передбачає, що клієнт припинить взаємодію з компанією, коли він насправді цього не робить): 0 доларів

Якщо помножити кількість кожного типу прогнозування (FN, TP, FP та TN) на вартість, пов’язану з кожним, і підсумувати їх, то ми можемо розрахувати загальну вартість, пов’язану з нашою моделлю. Ось як виглядає це рівняння:

Вартість = FN ($200) + TP ($25) + FP ($25) + TN ($0). Тепер застосуємо цю оцінку витрат до запропонованої моделі.

Перш за все створимо пороговий вектор та вектор вартості клієнта. Пороговий вектор буде містити поріг для кожної моделі. Раніше було використано порогове значення в 0.5, але далі розглянемо порогові значення з кроком 0.1 від 0 до 1. За допомогою цикла for, робимо прогнози, використовуючи різні порогові значення, та оцінюємо вартість кожного із них.

У тестовому наборі є 1760 спостережень, Таким чином можливо обчислити вартість одного користувача. Тепер припустимо, що в даний час компанія використовує так звану "контрольну" модель, яка за замовчуванням має значення

0.5. Тож далі можливо підігнати цю модель, зробити прогнози і розрахувати вартість.

```
> tr <- glm(Churn~., data=train, family=binomial)
> churn.probs <- predict(tr, test, type="response")
> thresh <- seq(0.1,1.0, length = 10)
> cost = rep(0,length(thresh))
> for (i in 1:length(thresh)){
+
+   glm.pred = rep("No", length(churn.probs))
+   glm.pred[churn.probs > thresh[i]] = "Yes"
+   glm.pred <- as.factor(glm.pred)
+   x <- confusionMatrix(glm.pred, test$Churn, positive = "Yes")
+   TN <- x$table[1]/1760
+   FP <- x$table[2]/1760
+   FN <- x$table[3]/1760
+   TP <- x$table[4]/1760
+   cost[i] = FN*200 + TP*25 + FP*25 + TN*0
+ }

  > glm.pred = rep("No", length(churn.probs))
  > glm.pred[churn.probs > 0.5] = "Yes"
  > glm.pred <- as.factor(glm.pred)
  >
  > x <- confusionMatrix(glm.pred, test$Churn, positive = "Yes")
  > TN <- x$table[1]/1760
  > FP <- x$table[2]/1760
  > FN <- x$table[3]/1760
  > TP <- x$table[4]/1760
  > cost simple = FN*200 + TP*25 + FP*25 + TN*0
  >  dat <- data.frame(
  +    Модель = c(rep("налаштована",10),"контрольна"),
  + Вартість_на_користувача = c(cost,cost_simple),
  + Поріг = c(thresh,0.5)
  + )
```

Рис. 3.15 Налаштована модель в залежності від порогового значення

Замість того, щоб використовувати загальну кількість кожного результату для TN, FP, FN та TP, було використано відсоткове значення. Кількість кожного результату була розділена на загальну, тобто 1760.

Тепер можливо помістити всі результати на графік і побудувати його (залежність вартості одного користувача від порогового коефіцієнта).

```
>  ggplot(dat, aes(x = Поріг, y = Вартість_на_користувача, group = Модель, colour = Модель)) +
+ geom_line() +
+ geom_point()
>
```

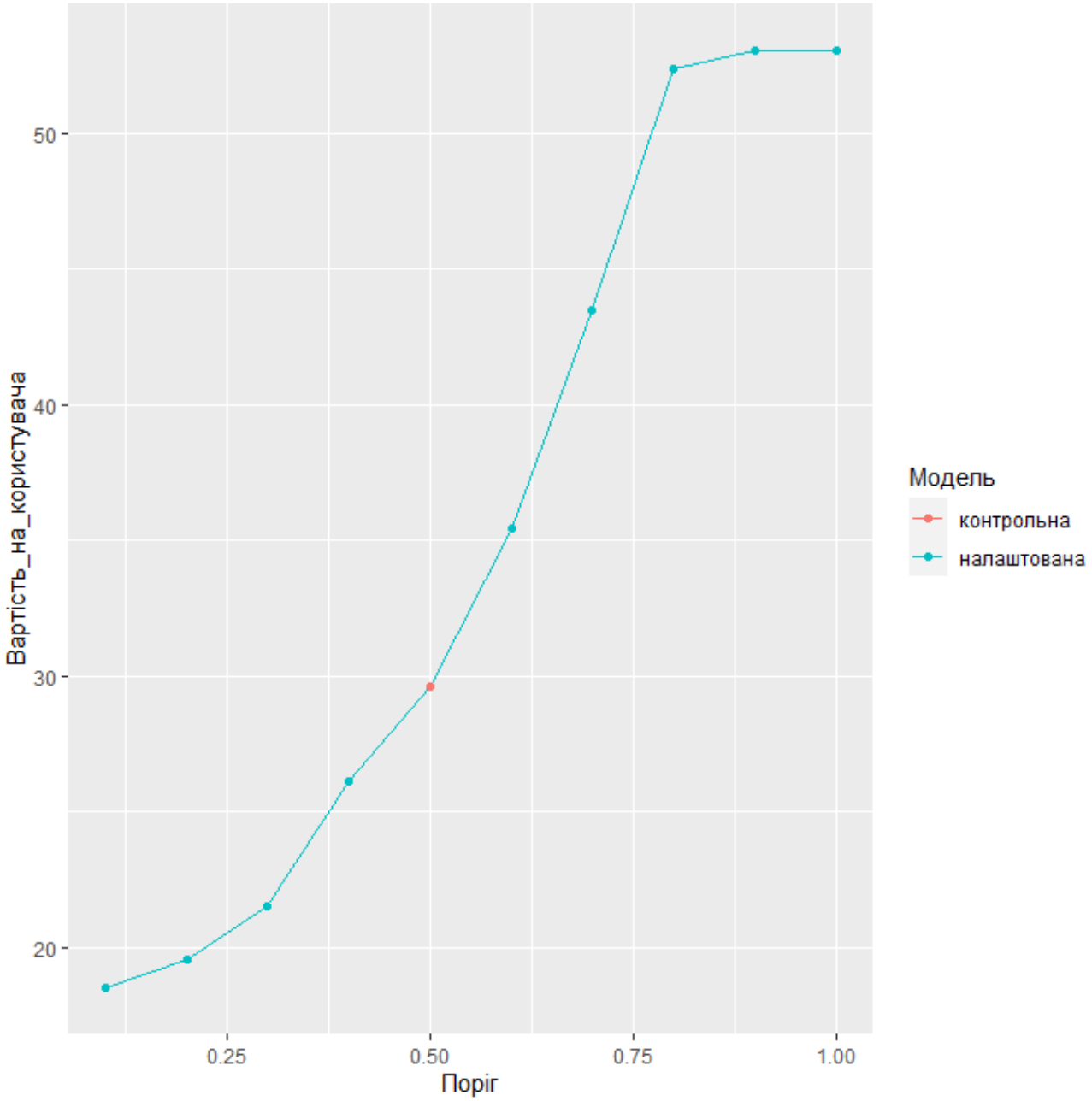

Рис. 3.16 Залежність вартості користувача від порогового значення

Можна зробити висновок, що мінімальна вартість одного клієнта становить близько 18 доларів при порозі 0,2.

Контрольна модель, яку в даний час впроваджує компанія, коштує близько 29 доларів на користувача при граничному значенні 0,50. Якщо припустити, що у компанії є близько 1 000 000 клієнтів, перехід від простої моделі до оптимізованої забезпечує економію 11 мільйонів доларів.

```
> savings_per_customer = cost_simple - min(cost)
>
> total_savings = 1000000*savings_per_customer
> total_savings
[1] 11093750
>
```

Рис. 3.17 Загальна економія від оптимізованої моделі

Висновки до розділу 3

1. У цьому розділі була розроблена модель машинного навчання для прогнозування відтоку клієнтів. Зокрема, детально розглянули такі моменти як: логістична регресія, підготовка даних, підгонка моделі, складання прогнозів та вплив на бізнес. В результаті була розроблена оптимізована модель логістичної регресії для проблеми відтоку клієнтів.
2. Припускаючи, що компанія використовує модель логістичної регресії з порогом за замовчуванням 0.5, було визначено, що оптимальним порогом насправді є 0.2. Це зменшило витрати на клієнта з 29 до 18 доларів. При клієнтській базі приблизно 1 000 000 це призведе до щорічної економії витрат у розмірі 11 мільйони доларів. Що в свою чергу має значний вплив на бізнес.

РОЗДІЛ 4 РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

В даному розділі приведено економічний аналіз перспектив інтеграції системи передбачення відтоку користувачів від постачальників мобільного зв’язку.

4.1 Опис ідеї стартап проекту

Проект спрямований на мінімізацію фінансових збитків телекомунікаційних компаній на маркетинг, шляхом адаптивного заохочення існуючих клієнтів для запобігання відтоку.

Таблиця 4.1

Опис ідеї проекту

Зміст ідеї	Напрямки застосування	Вигода для користувача
Адаптивна система для виявлення користувачів схильних до відтоку, що надає змогу вибірковому заохоченню .	1. Виявлення кореляцій, тенденцій та параметрів, що впливають на відтік. 2. Прогнозування відтоку. 3. Зменшення витрат на утримання абонентів.	Постачальники стільникового зв’язку отримають можливість гнучко виділяти кошти саме до групи клієнтів схильних до відтоку, тим самим заощаджуючи на маркетинговій політиці

Таблиця 4.2

Технологічна здійсненність проекту

No п/п	Ідея проекту	Технології реалізації	Можливості реалізації	Доступність реалізації
1	Адаптивна система прогнозування відтоку користувачів	Мова R для статистичної обробки даних	Бібліотеки для роботи з графікою: ROCR, Caret, Tidyverse.	Є доступним та безкоштовним для використання
2		База даних IBM Sample Data Sets	Попередня обробка вхідних даних з бази	Є доступним та безкоштовним для використання

4.2 Аналіз ринкових можливостей

При дослідженні можливостей ринку, в першу чергу аналізовано попит: наявність попиту, обсяг, динаміка розвитку ринку. Дані приведені у таблиці нижче.

Таблиця 4.3

Попередня характеристика потенційного ринку стартап-проекту

No п/п	Показники стану ринку	Характер стану
1	Кількість головних гравців, од	< 3
2	Динаміка ринку (якісна оцінка)	Зростає
3	Наявність обмежень для входу	Немає
4	Специфічні вимоги для стандартизації, специфікації	Немає

Враховуючи сьогоднішню необхідність ринку рішень стосовно зменшення відтоку клієнтів у галузі, за попереднім оцінюванням ринок є привабливим для входження.

Таблиця 4.4

Характеристика потенційних клієнтів стартап проекту

No п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Зменшення кількості клієнтів, що припиняють користування послугами мобільного	Оператори зв’язку	Оператори зв’язку в свою чергу можуть визначати самостійно формат взаємодії	• Висока точність системи прогнозування відтоку. • Можливість визначати фактори та

Продовження Таблиці 4.4

	зв’язку та підвищення їх лояльності		та ввести свої коригування.	тенденції, що впливають на відтік. • Можливість виокремлювати абонентів схильних до відтоку.

Таблиця 4.5

Фактори загроз

No п/п	Фактор	Зміст загрози	Можлива реакція компанії
	Відсутність зацікавленості у продукті	Успіх адаптивної системи виявлення користувачів, які мають намір припинити користуватися послугами оператора зв’язку, залежить від попиту з боку провайдерів комунікаційних послуг середнього та малого сегмента. Крупні гравці зазвичай схильні до рішень, що вже мають позитивний досвід і перевірені часом.	Інтеграція продукту для малого і середнього бізнесу для створення позитивного досвіду та подальшого просування системи на новий ринок збуту.

Висновки до розділу 4

В даному розділі приведено маркетинговий аналіз перспектив впровадження системи виявлення відтоку користувачів. В ході дослідження було встановлено, що запропонована система є надзвичайно гнучкою та здатна ефективно працювати майже з будь-якою базою клієнтів, така універсальність значно розширює ринок збуту. проекту. Конкурентна ситуація не є досить щільною, що в свою чергу робить перспективним впровадження продукту, так як аналогічна продукція від компаній-конкурентів реалізує лише частковий функціонал запропонованої системи.

В результаті існуючі системи-аналоги не створюють прямої конкуренції на ринку України. Основною проблемою є вихід на ринок великих телекомунікаційних компаній через недовіру до нового продукту.

Проведений аналіз підтверджує, що подальша імплементація проекту є доцільною.

ВИСНОВКИ

1. Проаналізовано різноманітні парадигми та відповідні їм алгоритми машинного навчання. Серед яких найперспективнішим є глибоке навчання для прогнозування тенденцій на ринку телекомунікаційних послуг, видобутку та аналізу білінгових даних, для створення самокерованих систем

2. Глибинне навчання надає необхідну гнучкість і є перспективною архітектурою для побудови адаптивних телекомунікаційних систем, завдяки здатності працювати з великими об’ємами необроблених даних та легкій масштабованості для різних задач.

Для задачі прогнозування міського трафіку система глибинного навчання на базі двонаправленної архітектури LSTM продемонструвала точність близько 92%, що помітно більше показників точності існуючих архітектур (xGBoost, SVR, регресія RandomForest).

3. Запропонована і розроблена система прогнозування відтоку користувачів на базі моделі логістичної регресії: засобами мови програмування R. Моделі прогнозування на базі логістичної регресії є перевірені часом та демонструють досить високі показники ефективності прогнозування (85% в даному випадку). Було проведено гіпотетичний бізнес аналіз результатів використання такої системи, який показав, що компанія з одним мільйоном користувачів за рік може заощадити на утримані існуючих клієнтів близько 11 мільйонів доларів.

4. Розроблено стартап проект по реалізації системи прогнозування відтоку користувачів. Визначено сильні та слабкі сторони такого проекту, завади до його впровадження. До переваг розробленої моделі можна віднести простоту у використанні, гнучкість та масштабованість, можливість кастомізації та ефективність. До основних вад системи належить необхідність попередньої обробки даних людиною, часткова автоматизація процесу обробки. Можливі такі фактори загроз для впровадження проекту як недовіра ринку та конкуренція.

ПЕРЕЛІК ПОСИЛАНЬ

1. Ільченко М.Ю., Кравчук С.О. Телекомунікаційні системи. – Київ: Наукова думка, 2017. – 730 с.
2. Лисенко О.І., Тачиніна О.М., Алєксєєва І. В. «Математичні методи моделювання та оптимізації. Частина 1. Математичне програмування та дослідження операцій: підручник» – К.: НАУ, 2017. – 212 с. ISBN 978-966-932-063-6.
3. Досягнення в телекомунікаціях 2019 / за наук. ред. М.Ю.Ільченка, С.О.Кравчука: монографія. - Київ: Інститут обдарованої дитини НАПН України, 2019.- 336 с. Рекомендовано до друку ВР КПІ ім.І.Сікорського (прот.№10 від 04.11.2019 р.) ISBN 978-617-7734-12-2.
4. Abidin, A.F., Kolberg, M., Hussain, A.: Integrating twitter traffic information with Kalman filter models for public transportation vehicle arrival time prediction. In: big-data analytics and cloud computing. Pp. 67–82. Springer international publishing, Cham (2015).
5. Castanedo, F.: A review of data fusion techniques. Sci. World J. (2013).
6. Cisco. Cisco Visual Networking Index: Forecast and Methodology, 2016-2021, June 2017.
7. N. Bui and J. Widmer, “Owl: a reliable online watcher for lte control chann measurements,” ACM All Things Cellular, 2016.
8. Jia, Y., Wu, J., Xu, M.: Traffic flow prediction with rainfall impact using a deep learning method. J. Adv. Transp., (2017). https://doi.org/10.1155/2017/6575947.
9. Elman, J.L.: Finding structure in time. Cogn. Sci. **14**, 179–211 (1990). https://doi.org/10.1016/0364-0213(90)90002-E.
10. Hoang Duy Trinh, Lorenza Giupponi, Paolo Dini Mobile Traffic Prediction from Raw Data Using LSTM Networks 2018.
11. Archived | AI models for the telecommunications and media industries https://developer.ibm.com/articles/cc-cognitive-media-telco-5/.

12. Cisco. 2017. Cisco Visual Networking Index: Forecast and Methodology, 2016-2021.White Paper. Cisco Systems.
13. Customer Churn Prediction https://www.kaggle.com/c/customer-churn-prediction-2020.
14. Customer churn prediction in telecommunication DOI: 10.1109/SIU.2015.7129808 (2015).
15. https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d
16. https://towardsdatascience.com/ml-notes-why-the-least-square-error-bf27fdd9a721Proceedings of the IEEE 101, 1410-1423 (2013).
17. Panagiotis Kasnesis, Charalampos Patrikakis, and Iakovos Venieris.Changing the game of mobile data analysis with deep learning.ITProfessional, 2017.
18. C. L. Philip Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data,"Inf. Sci.,vol. 275, pp. 314–347, Aug. 2014.Pendry, J. B. Negative refraction makes a perfect lens. Physical Review Letters 85, 3966-3969 (2000).
19. Understanding Machine Learning From Theory to Algorithms https://doi.org/10.1017/CBO9781107298019
20. Qian Mao, Fei Hu, and Qi Hao. Deep learning for intelligent wireless networks: A comprehensive survey.IEEE Communications Surveys & Tutorials, 2018.