Зміст

АНОТАЦІЯ

Дана магістерська дисертація присвячена розробці методу автоматизованої класифікації текстових даних на основі гібридних моделей.

Даний проект складається з двох основних частин: ресурсу для збору початкових даних та компоненту, що здійснює тренування та запуск моделей на отриманих даних. Частина для агрегації вхідних даних являє собою веб-додаток, що використовується для проходження опитування і формуванні результуючої таблиці на основі заповних форм. В якості проміжного етапу здійснюється підготовка та попередня обробка даних для подальшого використання. Підсистема класифікації в свою чергу поділена на два підмодуля: безпосередня імплементація розробленого методу класифікації та використання побудованої моделі для прогнозування зміни досліджуваної величини.

В рамках магістерської дисертації проведено аналіз існуючих систем для збору даних та розробка власної системи на основі адаптації готових рішень під вимоги досліджуваної області. Було здійснено дослідження існуючих алгоритмів для класифікації текстових даних та алгоритмів і бібліотек, що використовуються для побудових моделей прогнозування. Проведено оцінку предметної області, сформовано функціональні та нефункціональні вимоги до програмного забезпечення, а також проаналізовано перспективи виходу на ринок та запуску даного проекту в комерційних цілях.

У даній магістерській дисертації розроблено: архітектуру веб-ресурсу, інтерфейс для збору початкових даних, компнонент для обробки та трансформації даних, алгоритм для побудови прогностичної моделі та утиліту командного рядка для прикладного запуску моделі в якості інструменту прогнозування зміни цільової величини вхідних даних. Виконаний порівняльний аналіз з уже існуючими рішеннями та перевірка коректності роботи на основі порівняння відхилення результуючих показників з еталонни-

ми. Дана система готова розгортання, використання та інтеграції з іншими рішеннями, а також до впровадження в якості самостійного проекту на ринок, націленого на комерціалізацію продукту.

ABSTRACT

This Master's dissertation is about creating a method for automatic text data classification based on blender models.

The project consists of two main parts: data collection module and a compoment responsible for training and launching a model on the input data. Subcomponent for input data aggregation is a web-application that is used by target users to take a survey and then to form a resulting table based on an information filled. As a part of main pipeline data transformation and preprocessing takes place. Classification system by its own consists of two connected modules: implementation of a classification method and application for prediction using the model built.

In the scope of dissertation analysis of current system for data mining was made. Also new project based on requirements from domain field using solutions from existing alternatives was created. Research on existing algorithms of text data classification and comparison of libraries was conducted. The research in the domain field was performed, functional and non-functional requirements for the software were generated. Possibilities for commercial launching of the project were considered and corresponding breakdown took place.

Within the Master's dissertation following components were created: architecture of a web-resource, user interface for collecting initial data, component for data transformation and preprocessing, algorithm for predictive model creation, command line utility for launching a model built on the dataset in order to predict target value. Developed solution was compared to competitors and final measurements about system's correctness was made based on reference data. System created is ready for deployment, direct usage and integration with existing solutions as well as moving forward to the market targeting commercialization of a product.

АННОТАЦИЯ

Данная магистерская диссертация посвящена разработке метода автоматизированной классификации текстовых данных на базе гибридных моделей.

Проект состоит из двух основных частей: ресурсы для сбора начальных данных и компонента, отвечающего за тренировку и запуск моделей на существующих данных. Часть агрегации данных представляет собой веб-приложение, которое испльзуется для прохождения опроса и формирования финальной таблицы на основании заполненых форм. В качестве промежуточного этапа производится подготовка и дообработка данных для дальнейшего использования. Система классификации в свою очередь разделена на два модуля: непосредственная имплементация разработанного метода классификации и использование построенной модели для прогнозирования изменений исследуемого значения.

В рамках диссертации проведен анализ существующих систем для сбора данных, а также разработана собственная система на основании адаптации готовых решений к требованиям предметной области. Осуществлено исследование алгоритмов классификации текстовых данных, а также алгоритмов и библиотек, которые используются для построения прогностических моделей. Проведено оценку предметной области, сформировано функциональные и нефункциональные требования к программному продукту, а также проанализировано перспективы выхода на рынок и запуска проекта в коммерческих целях.

В магистерской диссертации разработано: архитектуру веб-ресурса, интерфейс для сбора начальных данных, компонент для обработки и трансформации данных, алгоритм построения прогностической модели и утилиту командной строки для запуска модели в качестве инструмента прогнозирования изменений целевого значения входящих данных. Был осуще-

ствлен анализ с уже существующими решениями и проверка корректности работы на основании сравнения отклонения результатов с еталонными значениями. Система полностью готова к разворачиванию, использованию и интеграции с другими продуктами, а также к внедрению в качестве самостоятельного решения с целью коммерциализации проекта.

1 Теоретичні основи

На відміну від штучно створених мов, наприклад мов програмування чи математичних нотацій, мови, які ми використовуємо для спілкування, розвивалися з покоління в покоління, постійно видозмінюючись, а тому досить складно відслідкувати і встановити набір чітких конкретно визначених правил. Розробка алгоритмів, що дозволяють "розуміти" людські висловлювання дає змогу покращити велику кількість аспектів взаємодії людини та комп'ютера: передбачення вводу, розпізнавання тексту, пошук інформації в неструктурованому тексті, переклад з однієї мови на іншу, аналіз емоційного забарвлення тексту та багато іншого. Створюючи інтерфейси, що дозволяють людині більш ефективно використовувати комп'ютер, ми прискорюємо розвиток багатомовного інформаційного суспільства.

1.1 Методи класифікації даних

1.1.1 Проблема класифікації даних

Задача класифікації – формалізована задача, яка містить множину об’єктів (ситуацій), поділених певним чином на класи. Задана кінцева множина об'єктів для яких відомо, до яких класів вони відносяться. Ця множина називається вибіркою. До якого класу належать інші об'єкти невідомо. Необхідно побудувати такий алгоритм, який буде здатний класифікувати довільний об'єкт з вихідної множини. Класифікувати об'єкт – означає вказати номер (чи назву) класу, до якого відноситься даний об'єкт. Класифікація об'єкта – номер або найменування класу, що видається алгоритмом класифікації в результаті його застосування до даного конкретного об'єкту. В математичній статистиці задачі класифікації називаються також задачами дискретного аналізу. В машинному навчанні завдання класифікації вирішується, як правило, за допомогою методів штучної нейронної мережі при постановці експери-

менту у вигляді навчання з учителем (supervised machine learning). Існують також інші способи постановки експерименту – навчання без вчителя (unsupervised learning), але вони використовуються для вирішення іншого завдання – кластеризації або таксономії. У цих завданнях поділ об'єктів навчальної вибірки на класи не задається, і потрібно класифікувати об'єкти тільки на основі їх подібності. У деяких прикладних областях, і навіть у самій математичній статистиці, через близькість завдань часто не відрізняють завдання кластеризації від завдання класифікації.

Деякі алгоритми для вирішення задач класифікації комбінують навчання з учителем і навчання без вчителя, наприклад, одна з версій нейронних мереж Кохонена – мережі векторного квантування, яких навчають способом навчання з учителем.

Прогностичне моделювання – використання статистичних методів для передбачення деякого цільового значення. Зазвичай, мається на увазі передбачення деякої величини в майбутньому, хоча узагальнено це не грає жодної ролі і може бути застосовано до будь-якого типу невідомої події, незалежно від того, коли вона відбулася. В багатьох випадках задача зводиться до вибору найкращої моделі, що намагається здогадатися результат на основі набору вхідних даних, наприклад визначення того, чи є деякий лист електронної пошти спамом. Моделі можуть використовувати один чи декілька класифікаторів, щоб визначати приналежність даних до деякої множини. Сам термін прогностичної моделі широко перетинається з поняттями машинного навчання в наукових статтях та в контексті розробки програмного забезпечення. В промисловому середовищі даний термін швидше відноситься до поняття прогностичного аналізу.

1.1.2 Існуючі методи класифікації даних

В залежності від вхідних даних, для задач класифікації можна виділити такі категорії:

- Характеристичний опис – найпоширеніший випадок. Кожен об'єкт описується набором своїх характеристик, які називаються ознаками. Ознаки можуть бути числовими або нечисловими.
- Матриця відстаней між об'єктами. Кожен об'єкт описується відстанями до всіх інших об'єктів навчальної вибірки. З цим типом вхідних даних працюють деякі методи, зокрема, метод найближчих сусідів, метод потенційних функцій.
- Часовий ряд або сигнал є послідовність вимірів у часі. Кожен вимір може представлятися числом, вектором, а в загальному випадку – характеристичним описом досліджуваного об'єкта в цей момент часу.
- Зображення або відеоряд.

Зустрічаються і складніші випадки, коли вхідні дані представляються у вигляді графів, текстів, результатів запитів до бази даних, і т. д. Як правило, вони приводяться до першого або другого випадку шляхом попередньої обробки даних та вилучення характеристик. Щодо класифікації сигналів та зображень, то її також називають розпізнаванням образів.

В залежності від кількості класів, на які розбиваються вхідні дані, отримуємо такий поділ:

- Двокласова класифікація (бінарна класифікація). Найпростіший в технічному відношенні випадок, який служить основою для вирішення складніших завдань.

- Багатокласова класифікація. Коли число класів досягає багатьох тисяч (наприклад, при розпізнаванні ієрогліфів або злитого мовлення), завдання класифікації стає істотно важчим.

- Непересічні класи.

- Пересічні класи. Об'єкт може належати одночасно до декількох класів.

- Нечіткі класи. Потрібно визначати ступінь належності об'єкта кожному з класів, звичайно це дійсне число від 0 до 1.

Прикладом одного з методів, що використовуэться найчастіше, є наївний баєсівський метод (байєсівський класифікатор). Наївна байєсівська модель є ймовірнісним методом навчання. Імовірність того, що документ d потрапить у клас c записується як $P(c|d)$. Оскільки мета класифікації - знайти найбільш відповідний клас для даного документа, то в наївній байєсівській класифікації задання полягає в знаходженні найбільш ймовірного класу $c_m = \underset{c \in C}{argmax} P(c|d)$.

Обчислити значення цієї ймовірност ібезпосередньо немаожливо, оскільки для цього потрібно, щоб навчальна множина містила всі (або майже всі) можливі комбінації класів і документів. Однак, використовуючи формулу Байєса, можна переписати вираз для $P(c|d)$ у вигляд $c_m = \underset{c \in C}{argmax} \frac{P(d|c)Pc)}{P(d)} = \underset{c \in C}{argmax} P(d|c)P(c)$. Використовуючи навчальну множину, ймоварність $P(c)$ можна оцінити як $\hat{P}(c|d) = \frac{N_c}{N}$, тобто відношення кількості документів у класі до загальної кількості документів у навчальній множині. Але за допомогою навчальної множини можна лише оцінити ймовірність, але не знайти її точне значеннння.

1.1.3 Машинне навчання з учителем

Машинне навчання - узагальнена назва методів штучної генерації знань з досвіду. Штучна система навчається на прикладах і після закінчення фази навчання може узагальнювати. Тобто система не просто вивчає наведені приклади, а розпізнає певні закономірності в даних для навчання.

Серед багатьох програмних продуктів машинне навчання використовують: системи автоматичного діагностування, розпізнавання шахрайства з кредитними картками, аналіз ринку цінних паперів, класифікація ланцюжків ДНК, розпізнавання мовлення та тексту, автономні системи.

Машинне навчання — розділ штучного інтелекту, має за основу побудову та дослідження систем, які можуть самостійно навчатись з даних. Наприклад, система машинного навчання може бути натренована на електронних повідомленнях для розрізняння спам і не спам-повідомлень. Після навчання вона може бути використана для класифікації нових повідомлень електронної пошти на спам та не-спам папки.

В основі машинного навчання розглядаються уявлення та узагальнення. Представлення даних і функцій оцінки цих даних є частиною всіх систем машинного навчання, наприклад, у наведеному вище прикладі повідомлення по електронній пошті, ми можемо уявити лист як набір англійських слів, просто відмовившись від порядку слів. Існує широкий спектр завдань машинного навчання та успішних застосувань. Оптичне розпізнавання символів, в яких друковані символи розпізнаються автоматично, ґрунтуючись на попередніх прикладах, є класичним прикладом техніки машинного навчання. У 1959 році Артур Самуїл визначив машинне навчання як "Поле дослідження, яке дає комп'ютерам можливість навчатися, не будучи явно запрограмованим" Samuel [1959].

Практичне використання відбувається, переважно, за допомогою алгоритмів. Різноманітні алгоритми машинного навчання можна грубо поділи-

ти за такою схемою:

- Навчання з вчителем – алгоритм вивчає функцію на основі наданих пар вхідних та вихідних даних. При цьому, в процесі навчання, «вчитель» вказує вірні вихідні дані для кожного значення вхідних даних. Одним з розділів навчання з вчителем є машинна класифікація. Такі алгоритми застосовуються для розпізнавання текстів.

- Багатокласова класифікація. Коли число класів досягає багатьох тисяч (наприклад, при розпізнаванні ієрогліфів або злитого мовлення), завдання класифікації стає істотно важчим.

- Навчання без вчителя.

- Пересічні класи. Об'єкт може належати одночасно до декількох класів.

- Навчання з закріпленням (англ. reinforcement learning): алгоритм навчається за допомогою тактики нагороди та покарання для максимізації вигоди для агентів (систем до яких належить компонента, що навчається).

Узагальнення в цьому контексті є здатність алгоритму для виконання точно на нових, невідомих прикладах після тренування на навчальному наборі даних. Основна мета учня узагальнювати свій досвід.

Також існує поняття інтелектуального аналізу даних, що за своєю природою відрізняється від машинного навчання. Два терміни часто плутають, оскільки вони не рідко використовують ті ж методи і перекриття. Вони можуть бути умовно визначені наступним чином: машинне навчання фокусується на прогноз, заснований на відомих властивостях, витягнутих з навчальних даних. Інтелектуальний аналіз даних (який є кроком виявлення знань у базах даних) фокусується на відкриття (раніше) невідомих властивостей даних.

Ці дві області перекриваються у багатьох відношеннях: інтелектуальний аналіз даних використовує безліч методів машинного навчання, але часто з дещо іншою метою. З іншого боку, машинне навчання також використовує методи інтелектуального аналізу такі як "неконтрольоване навчання" або як попередній крок оброблення для покращення точності навчальної системи. Велика частина плутанини відбувається з основних припущень: в машинному навчанні, виконання, як правило, оцінюється по відношенню до здатності відтворювати відомі знання, в той час як в інтелектуальному аналізі даних ключовим завданням є виявлення раніше невідомого знання. Необізнаний (неконтрольований) метод, який обчислюється по відношенню до відомих знань, буде легко перевершений керованими методами. В той час в типових ІАД завданнях, керовані методи не можуть бути використані через відсутність попередньої підготовки даних.

Деякі системи машинного навчання намагаються усунути необхідність в людській інтуїції під час аналізування даних, а інші обирають спільний підхід між людиною і машиною. Людська інтуїція не може бути повністю виключена, так як конструктору системи необхідно вказати, як дані повинні бути представлені і які механізми будуть використовуватися для пошуку характеристик даних.

Навчання з підкріпленням — це галузь машинного навчання натхненна біхевіористською психологією, що займається питанням про те, які дії мають виконувати програмні агенти в певному середовищі задля максимізації деякого уявлення про сукупну винагороду. Через свою загальність, дана проблема вивчається, вивчається багатьма іншими дисциплінами, такими як теорія ігор, теорія управління, дослідження операцій, теорія інформації, оптимізація на основі моделювання, багатоагентні системи, колективний інтелект, статистика та генетичні алгоритми. Галузь, що займається навчанням з підкріпленням, також називається наближеним динамічним програмуванням. Попри те, що проблема навчання з підкріпленням, вивча-

лась теорією оптимального управління, більшість досліджень стосувались саме існування оптимальних рішень та їх характеристики, а не навчання чи наближених аспектів. В економіці та теорії ігор, навчання з підкріпленням може використовуватись для пояснення того, як при обмеженій раціональності може виникати рівновага.

Навчання з учителем (англ. supervised learning) є одним із способів машинного навчання, в ході якого випробувана система примусово навчається за допомогою наявної множини прикладів «стимул-реакція» з метою визначення «реакції» для «стимулів», які не належать наявній множині прикладів.

Між входами та еталонними виходами (стимул-реакція) може існувати деяка залежність, але вона невідома. Відома лише кінцева сукупність прецедентів – пар «стимул-реакція», звана навчальною вибіркою. На основі цих даних потрібно відновити залежність (побудувати модель відносин стимул-реакція, придатних для прогнозування), тобто побудувати алгоритм, здатний для будь-якого об'єкта видати досить точну відповідь. Для вимірювання точності відповідей, так само як і в навчанні на прикладах, може вводитися функціонал якості.

Задача машинного навчання може бути представлена у вигляді експерименту. Даний експеримент являє собою окремий випадок кібернетичного експерименту зі зворотним зв'язком. Постановка даного експерименту припускає наявність експериментальної системи, методу навчання і методу випробування системи або вимірювання характеристик.

Експериментальна система у свою чергу складається з випробовуваної (використовуваної) системи, простору стимулів одержуваних із зовнішнього середовища та системи управління підкріпленням (регулятора внутрішніх параметрів). В якості системи управління підкріпленням може бути використано автоматичний пристрій, що регулюють (наприклад, термостат), або людину-оператора (вчитель), здатну реагувати на реакції випро-

бовуваної системи і стимули зовнішнього середовища шляхом застосування особливих правил підкріплення, що змінюють стан пам'яті системи.

Розрізняють два варіанти: (1) коли реакція випробовуваної системи не змінює стан зовнішнього середовища, і (2) коли реакція системи змінює стимули зовнішнього середовища. На рис. 1 зображено загальний вигляд такої експериментальної системи.

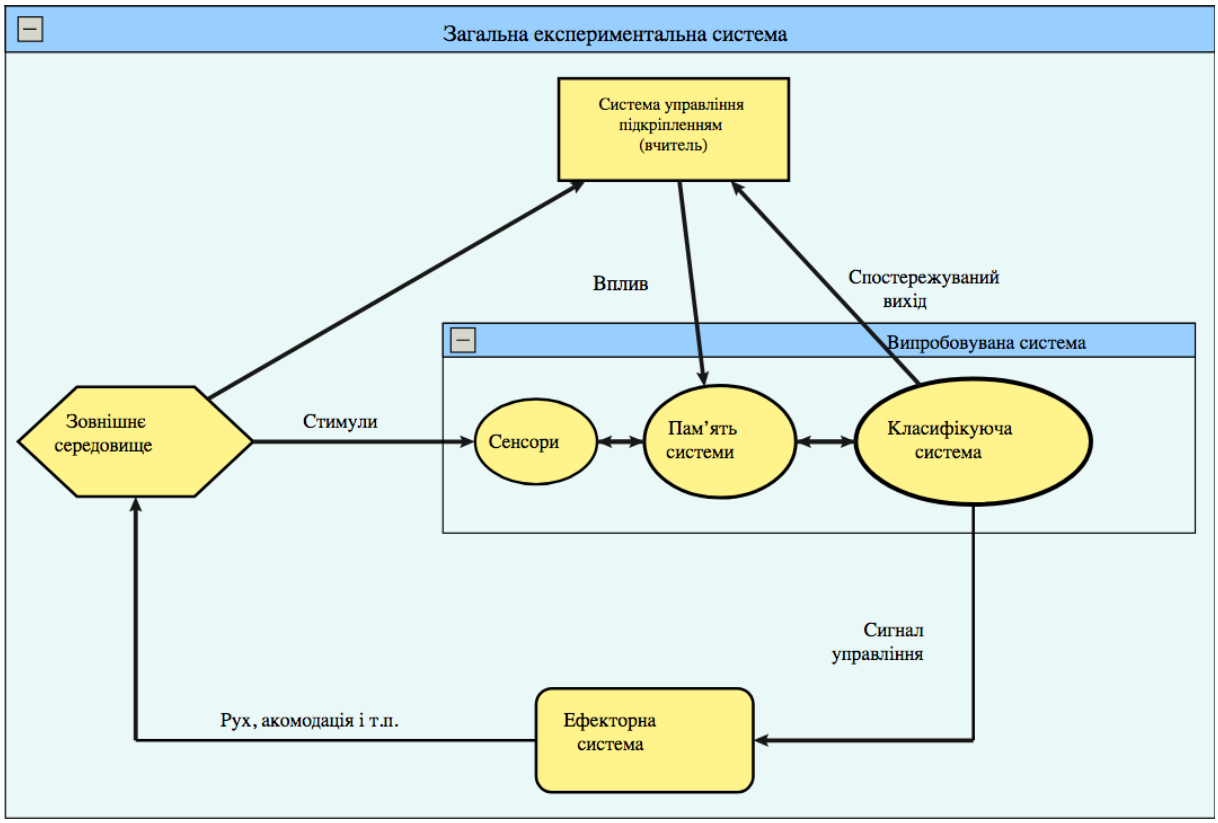

Рис. 1: Експериментальна система для навчання з учителем

В залежності від результуючих даних, отриманих від системи, можна виділити такі категорії класифікуючих систем:

- Множина можливих відповідей нескінчення (відповіді є дійсними числами або векторами). В даному випадку говорять про задачі регресії та апроксимації.
- Множина відповідей звичайна – задача класифікації та розпізнавання образів.

- Відповіді характеризують майбутню поведінку процесу або явища. В цьому випадку мова йде про задачі прогнозування (прогностичне моделювання).

Існують також вироджені системи, які характеризуються дещо зміненою поведінкою підкріплення інформації ("вчителя"):

- Система підкріплення з керуванням по реакції (R — керована система) — характеризується тим, що інформаційний канал від зовнішнього середовища до системи підкріплення не функціонує. Дана система, незважаючи на наявність системи управління, відноситься до спонтанного навчання, оскільки випробовувана система навчається автономно, під дією лише своїх вихідних сигналів незалежно від їх "правильності". При такому методі навчання для управління зміною стану пам'яті не потрібно ніякої зовнішньої інформації.

- Система підкріплення з керуванням по стимулах (S — керована система) — характеризується тим, що інформаційний канал від випробовуваної системи до системи підкріплення не функціонує. Незважаючи на не функціонування каналу від виходів випробовуваної системи, відноситься до навчання з учителем, оскільки в цьому випадку система підкріплення (вчитель) змушує випробувану систему виробляти реакції згідно певного правила, хоча й не береться до уваги наявність істинних реакцій випробовуваної системи.

Дана відмінність дозволяє глибше поглянути на відмінності між різними способами навчання, оскільки грань між навчанням з учителем і навчанням без вчителя тонша. Крім цього, таке розходження дозволило показати для штучних нейронних мереж певні обмеження для S та R — керованих систем.

1.1.4 Класифікація текстів

Класифікація текстів (документів) – одне із завдань інформаційного пошуку, яке полягає в тому, щоб віднести документ до однієї чи декількох категорій на основі вмісту документу. Класифікація може здійснюватися повністю в ручному режимі або автоматично за допомогою створеного вручну набору правил, або ж за допомогою застосування методів машинного навчання. Варто відрізняти класифікацію текстів від кластеризації, в останньому випадку тексти теж групуються за деякими критеріями, але попередньо задані категорії відсутні.

Розглянемо згадані вище три основних підходи до задачі класифікації текстів.

По-перше, класифікація не завжди здійснюється за допомогою комп'ютера. Наприклад, у звичайній бібліотеці тематичні рубрики присвоюються книгам власноруч бібліотекарем. Подібна ручна класифікація дорога і непридатна у випадках, коли необхідно класифікувати велику кількість документів з високою швидкістю.

Інший підхід полягає в написанні правил, згідно яких можна зарахувати текст до тієї чи іншої категорії. Наприклад, одне з таких правил може виглядати наступним чином: “якщо текст містить слова похідна і рівняння, то віднести його до категорії математика”. Спеціаліст, який знайомий з предметною областю і володіє навичкою написання регулярних виразів, може скласти низку правил, які потім автоматично застосовуються до класифікації нових документів. Цей підхід краще попереднього, оскільки процес класифікації автоматизується і кількість оброблюваних документів стає практично не обмеженою. Більш того, побудова правил власноруч може підвищити точність класифікації у порівнянні з машинним навчанням. Однак створення і підтримка правил в актуальному стані (наприклад, якщо для класифікації новин використовується ім'я чинного президента країни,

то відповідне правило потрібно час від часу змінювати) вимагає постійного контролю зі сторони фахівця.

Нарешті, третій підхід ґрунтується на машинному навчанні. У цьому підході набір правил або, більш загально, критерій прийняття рішення текстового класифікатора обчислюється автоматично з навчальних даних (іншими словами, проводиться навчання класифікатора).

Навчальні дані – це деяка кількість наочних зразків документів з кожного класу. У машинному навчанні зберігається необхідність ручної розмітки (термін “розмітка” означає процес надання документу певного класу), але вона є більш простим завданням, ніж написання правил. Крім того, розмітка може бути проведена в звичайному режимі використання системи. Наприклад, у програмі електронної пошти може існувати можливість позначати листи як спам, таким чином формуючи навчальну множину для класифікатора – фільтра небажаних повідомлень. Тому класифікація текстів, заснована на машинному навчанні, є прикладом навчання з учителем, де в ролі вчителя виступає людина, що задає набір класів і розмічає навчальну множину.

Класифікація за змістом є класифікацією, в якій увага приділена конкретним питанням. У документі визначається клас, до якого його зараховують. Це, наприклад, правило бібліотечної класифікації: принаймні 20% від змісту книги має бути близько класу, до якого відноситься книга. В автоматичній класифікації – це може бути кількість разів, коли дані слова з'являються в документі.

Класифікація за запитом (або індексація) є класифікацією, в якій очікуваний запит від користувачів впливає на те, як документи класифікуються. Класифікатор запитує себе: "За якими дескрипторами цей об'єкт можна знайти?" Тоді оброблюються всі можливі запити та обираються найбільш відповідні. Поняття дескриптора в даному контексті означає лексичну одиницю (слово чи словосполучення) інформаційно-пошукової мови,

яка служить для опису смислового змісту документів.

Класифікація за запитом може бути класифікацією, яка орієнтована на певну аудиторію або групу користувачів. Наприклад, бібліотека або база даних для феміністських досліджень можуть класифікувати (індексувати) документи по-різному в порівнянні з історичною бібліотекою. Це, ймовірно, краще, однак, класифікація робиться згідно деяких ідеалів і відображає мету бібліотеки або бази даних по класифікації. Таким чином, вона не обов'язково є видом класифікації або індексації на основі досліджень користувачів. Тільки якщо застосовуються емпіричні дані про використання чи користувачів, слід звернутися до орієнтованих класифікацій та розглядати в якості підходу користувача.

1.2 Опис існуючих методів кластеризації текстових документів

Інтелектуальний аналіз даних - галузь знань, яка відноситься до обробки даних, що вивчає пошук і опис прихованих, нетривіальних і практично корисних закономірностей у досліджуваних даних. До задач інтелектуального аналізу даних відноситься множина напрямків, такі як пошук документів в локальних і глобальних мережах, сортування, класифікація і кластеризація документів, автоматичне анотування та реферування, побудова тезаурусів і онтологій, системи автоматичного контролю, діалогові системи, системи, які навчаються, модифікація і поповнення баз знань, експертні системи і машинний переклад. Data Mining – дослідження і виявлення "машиною" (алгоритмами, засобами штучного інтелекту) в сирих даних прихованих знань, які раніше не були відомі, і є нетривіальними. практично корисними, доступними для інтерпретації людиною.

Під автоматичною кластеризацією текстових документів розуміють процес класифікації колекції текстових документів, який базується тільки на

аналізі та виявленні внутрішньої тематичної структури колекції без наявності апріорної інформації про неї, тобто при відсутності визначеного рубрикатора і множини документів-зразків. Класифікація документів з використанням алгоритмів кластеризації призводить до розбиття множини документів на однорідні, у відповідному розумінні, групи або кластери, шляхом автоматичного аналізу тематичної близькості між ними. Кластеризація текстів базується на гіпотезі: тісно пов'язані за змістом документи намагаються бути релевантними одним і тим же запитам, тобто документи, релевантні запиту, віддалені від тих, які не релевантні цьому запиту.

1.2.1 Мережа Кохонена (SOM)

Мережа Кохонена (англомовний термін – SOM). Призначення мережі Кохонена – розділення векторів вхідних сигналів на групи, тому можливість подання текстів у вигляді векторів дійсних чисел надасть можливість застосовувати цю мережу для їх класифікації.

Мережа складається з одного шару, що має форму прямокутних грат для g-х зв'язаних нейронів і форму соти для h-и зв'язаних.

Вектори X, що аналізуються, подаються на входи всіх нейронів. За наслідками навчання геометрично близькі нейрони виявляються чутливими до схожих вхідних сигналів, що може бути використано в завданні класифікації таким чином.

Для кожного класу визначається центральний нейрон і довірча область навколо нього. Критерієм межі довірчої області є відстань між векторами сусідніх нейронів і відстань до центрального нейрона області.

При подачі на вхід навченої мережі вектора тексту активізуються деякі нейрони (можливо з різних областей), текст належить до того класу, у довірчій області якого активізувалась найбільша кількість нейронів і якомога ближче до її центру.

Алгоритм навчання мережі полягає в наступному. Усі вектори повинні лежати на гіперсфері одиничного радіусу. Задається міра сусідства нейронів, що надає можливість визначати зони топологічного сусідства в різні моменти часу. На рис. 2 показано зміну цієї величини $NE_j(t)$ для деякого j-го нейрона.

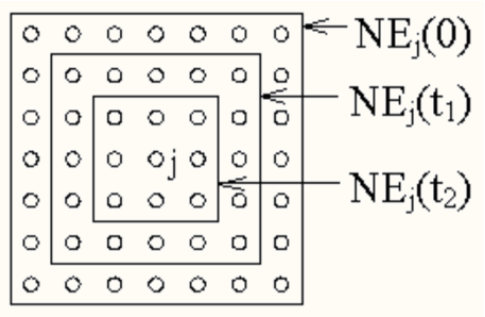

Рис. 2: Зони топологічного сусідства на мапі ознак

Крім того, задається розмір грати і розмірність вхідного вектора, а так само визначається міра подібності векторів S.

Далі виконуються такі кроки для кожного вектора навчальної вибірки:

1. Початкова ініціалізація площини може бути проведена, наприклад, довільним розподілом вагових векторів на гіперсфері одиничного радіусу.

2. Мережі подається вхдіний вектор тексту X_u, обчислюється міра подібності $S(X, W_j)$ для кожного j-го нейрона мережі. Нейрон, для якого S_j є максимальною, вважається поточним центром і для нього визначається зона сусідства $NE_j(t)$.

3. Для всіх нейронів, що потрапляють у зону $NE_j(t)$. (див. рис. 2), проводиться корекція ваг за правилом $w_{ij}(t+1) = w_{ij}(t) + \lambda(x_i(t) - w_{ij}(t))$, де λ - крок навчання, що зменшується з часом. Величина $NE_j(t)$ зменшується з часом так, що спочатку вона охоплює всю мережу, а в кінці навчання зона звужується до одного-двох нейронів, коли λ також набуває достатньо малого значення.

Як свідчать експерименти, на навчання мережі Кохонена впливають такі параметри:

1. Кількість нейронів та їх розміщення. Кількість нейронів слід вибирати не менше, ніж кількість груп, які потрібно одержати. Розташування неи?ронів на двовимірній площині залежить від завдання, що вирішується. Як правило, вибирається або квадратна матриця нейронів, або прямокутна з відношенням сторін, близьким до одиниці.

2. Початковий стан. У цьому випадку застосовується ініціалізація випадковими значеннями. Це не завжди призводить до бажаних результатів. Один із можливих варіантів покращання цього – обчислення характеристичних векторів репрезентативної вибірки текстів, що визначають межу двовимірної площини проекції. Після цього вагові вектора нейронів рівномірно розподіляються в одержаному діапазоні.

3. Характер зміни топологічної зони сусідства $NE_j(t)$. Визначає область нейронів, які підлягають навчанню. Чим швидше скорочуватиметься ця область, тим більше класів буде утворено, тим більшою є точність і меншою повнота.

4. Тип даних, що подаються на вхід. Для лексичних векторів фактично проводиться обробка по наявних в документі термах, що дає достатньо добрі результати. У цьому випадку можна виокремлювати документи за специфікою словарного набору. Проте без застосування морфологічного аналізу цеий метод неможливо застосовувати, оскільки різко збільшується обчислювальна складність.

5. Послідовність подачі на вхід векторів документів із різних груп. Оскільки коефіцієнт швидкості навчання з часом змінюється, результати подачі на вхід різних векторів текстів виявляються різними. При великому початковому значенні λ відбувається інтенсивна модифікація

всіх нейронів навколо переможця. При випадковій подачі документів із різних груп області близькості утворюються рівномірно.

1.2.2 Кластерний ієрархічний підхід

Ієрархічна кластеризація — процес організації даних в деревовидну структуру, яка побудована на основі подібності цих даних. Цей метод дуже потужний і корисний для аналізу великих наборів даних. Основна ідея — створити набір елементів у дереві. Дерево має багато гілок, якщо елементи подібні один до одного, до них приєднуються короткі гілки, і навпаки, якщо їх подібність зменшується, тоді збільшуються гілки.

Припустимо, що існує декілька текстів. Необхідно згрупувати ці тексти відповідно до подібності їх стилів. Таке групування може бути як одноріневим ("плоским", з виділенням таких кластерів, що кожен об'єкт в них є одним з текстів набору кластеризації), так і ієрархічною, коли кластери, отримані в результаті об'єднання найбільш схожих текстів самі можуть об'єднуватися в кластери, а кластери кластерів — в інші кластери і так далі. Відношення тексту до деякого кластера на певному рівні ієрархічної кластеризації може бути однозначною (кожен даний текст належить лише одному кластеру), або неоднозначною (кожен даний текст може належати декільком кластерам). Кластеризація документів була використана, аби автоматично генерувати ієрархічні кластери документів.

Текстова кластеризація автоматично виявляє групи семантично схожих документів серед заданої великої фіксованої кількості документів.

Слід зазначити, що групи формуються лише на основі попарної подібності описів документів, і жодні характеристики цих груп не задаються заздалегідь, на відміну від текстової класифікації.

На кожному робочому комп'ютері існує величезна кількість тек, у яких досить часто зберігається велика кількість документів, які, зазвичай мають

абсолютно різну тематику. Людина після певного проміжку часу ледь згадує, що у якій теці знаходиться, а якщо проміжок досягає місяців, то взагалі не може пригадати, у якій теці зберігатися необхідна йому на даний період інформація. Запропонована авторами система дозволяє вирішити цю проблему. Вона автоматично кластеризує документи у теки, які відповідають тематиці документа. Користувачеві необхідно буде скористатися запропонованою системою, і документи віднесуться до логічних за структурою документа тек. У системі на першому етапі документи проходять попередню обробку — скорочення тексту для точнішої класифікації. У нашому випадку препроцессинг (попередня обробка) складається з двох етапів. Документ, що надійшов, попередньо обробляється, перш ніж пройти останні етапи. Спочатку видаляються всі стоп-слова з документу. *Стопслова* — це набір артиклів, таких як: the, a, in, of і так далі. Потім використовується *стеммінг* — це процес виділення основи слова. Стеммінг використовується, оскільки він дозволяє максимально скоротити час обробки документа в системі, що, відповідно, веде до оптимізації системи, поліпшення якості її роботи. Для стеммінгу використовується алгоритм Портера.

Загальна структура цієї моделі даних починається з відображення будь-якого документа як вектора слів, які з'являються в документах набору даних. Вага (зазвичай частоти термів) слів також міститься в кожному векторі. Після попередньої обробки ми використовуємо векторну модель. На сьогоднішній день векторна модель, широко використовується для зображення даних в класифікації і кластеризації документів. Подібність між двома документами визначається на підставі двох відповідних за властивостями векторів, наприклад Jaccard measure, Euclidean distance та інші. Найчастіше використовується cosine measure. Часто використовується попереднє оброблення — скорочення тексту для точної класифікації. З обробкою методів, різні документи можуть бути створені як структуровані відображення документів. Зазвичай, завдання попереднього оброблення дій включає

стандартизацію документа, токенізацію, лематизацію і стеммінг.

На сьогоднішній день існує багато різноманітних методів і реалізацій кластерного аналізу документів. Але попри це існує невеликий ряд методів, які є основою для більшості інших – головна частина з них була описана вище. Всі алгоритми кластеризації ґрунтуються на вимірюваннях схожості по різних критеріях. Деякі використовують слова, які часто з'являються разом (лексичну близькість), інші використовують вилучені особливості (такі як імена людей і т. п.). Різниця полягає також і в кластерах, що створюються.

Виділяють три основні типи методів кластеризації документів:

- ієрархічний – створює дерево зі всіма документами в кореневому вузлі і одним документом у вузлі-листі. Проміжні вузли містять різні документи, які стають все більш і більш спеціалізованими у міру наближення до листя дерева;

- бінарний – забезпечує угрупування і перегляд документальних кластерів по посиланнях подібності. У один кластер поміщаються найближчі по своїх властивостях документи;

- нечіткий – включає кожен документ у всі кластери, але при цьому зв'язує з ним вагову функцію, що визначає ступінь приналежності даного документа до певного кластеру.

Найбільш передовими є алгоритми, що базуються на самоорганізаційних картах Кохонена. Структура кластерів при використанні алгоритму самоорганізуючих карт Кохонена може бути відображена шляхом візуалізації відстані між опорними векторами (ваговими коефіцієнтами нейронів). При використанні цього методу найчастіше використовується уніфікована матриця відстаней (u-matrix), тобто обчислюється відстань між вектором ваги нейрона в сітці і його найближчими сусідами. Потім ці значення вико-

ристовуються для визначення кольору, яким цей вузол буде відображатися. Зазвичай використовують градації сірого, причому, чим більше відстань, тим темніше відображається вузол. При такому використанні, вузлам з найбільшою відстанню між ними та сусідами відповідає чорний колір, а навколишнім вузлам – білий. Таким чином, розташовані поблизу кластери зі схожими кольорами утворюють більш глобальні кластери. Зазвичай в них розташовані близькі за ознаками документи.

Можна зробити висновок, що базовими для реалізацій інших методів, що використовуються у сучасному програмному забезпеченні є методи k-means, метод n-грам, самоорганізаційні карти Кохонена та деякі інші.

Метод k-means показує гарні результати і має високу швидкість знаходження кластерів. Одним з недоліків даного методу є великий об'єм словника, який використовується у методі. Залежність розміру словника є більшою за лінійну. Але незважаючи на це даний метод є популярним на сьогоднішній день. Він активно використовується в різноманітних мовах програмування, операційних системах, пошукових системах й іншому програмному забезпеченні.

Метод n-грам має високу швидкість знаходження результатів, високу точність і легкість в реалізації. Одним з недоліків даного методу є не 100% ймовірність виявлення кластеру. Але в порівнянні з багатьма іншими методами, даний метод має не такий високий словник, що значно пришвидшує пошук необхідного елемента в ньому. Даний метод з'явився вже давно, але незважаючи на це і всі його недоліки він є популярним і в наш час. Його часто використовують для визначення помилки в слові. Тобто він виступає підготовчим для наступних методів.

Вагомою перевагою карт самоорганізації і нейронних мереж зустрічного розповсюдження є велика кількість обмежень і передумов для використання інструментарію дискримінантного аналізу, зокрема, відносно стаціонарності досліджуваних процесів, незмінності зовнішніх умов і т. п. Ця

модель здатна швидко адаптуватися до нових даних, не потребує залучення експертів і дозволяє виявляти приховані нелінійні закономірності.

1.3 Препроцесинг даних

Попередня обробка (препроцесинг) - розділ аналізу даних що займається отриманням характеристик для подальшого використання у наступних розділах аналізу даних. Загалом препроцесинг можна поділити на декілька основних етапів.

1. Обчислення базових характеристик (центральні моменти).
2. Перевірка основних гіпотез (симетричності, однорідності).
3. Перевірка стохастичності вибірки.
4. Видалення аномальних спостережень.
5. Розвідувальний аналіз.

Препроцесинг даних – один із найважливіших кроків в дата майнінгу. Методи для збору даних зазвичай не є жорстко контрольованими, за рахунок цього можемо отримати комбінації несумісних даних або втрачені значення. Аналіз таких даних може призвести до похибок та невірних результатів. Тому представлення та якість вхідних даних є першою вимогою перед безпосереднім аналізом.

Результатом препроцесингу даних є фінальний тренувальний набір даних.

1.3.1 Очищення даних

Очищення даних – це процес виявлення та коригування (зміни чи видалення) хибних, невірних чи не досить точних записів з таблиці, бази даних

чи вхідного файлу. Також до цього процесу відноситься і ідентифікація неповної, некоректної чи неточної частини цих даних. Очищення може бути здійснене в інтерактивному режимі за допомогою спеціально створених інструментів або пакетної обробки з допомогою певних скриптів. Після очищення набір даних буде консистентним з іншими подібними наборами даних. Процес очищення відрізняється від процесу валідації тим, що вхідні дані відкидаються після додавання до системи, а не під час створення запису. Також до чистки даних може відноситися і доповнення існуючої інформації. Наприклад, доповнення адреси її поштовим кодом або заміна скорочень на їх повні відповідники.

1.3.2 Якість даних

Визначання якості даних залежить від набору певних критеріїв, що характерні для даних. Дані критерії включають в себе:

- Придатність – відповідність даних до вимог та поставлених обмежень. Обмеження бувають декількох типів:
  - обмеження на типи – значення можуть бути лише визначених типів, наприклад числа, булеві значення чи дати;
  - обмеження на діапазон значень – наприклад, числа можуть бути в межах мінімального та максимального значення;
  - обмеження на вміст – значення не можуть бути порожні і мають містити певну інформацію;
  - обмеження на унікальність – значення не можуть повторюватися в рамках одного датасету, наприклад двоє людей не можуть мати один ідентифікаційний номер;
  - обмеження на набір значень – поле може бути лише одним значень з наперед визначеної множини, наприклад стать чоловіча,

жіноча або невідома.

  - обмеження на набір значень – поле може бути лише одним значень з наперед визначеної множини, наприклад стать чоловіча, жіноча або невідома.
  - обмеження регулярним виразом – текстові поля повинні підпадати під регулярний вираз, наприклад номер телефону може вимагатися в певному форматі, описаному регулярним виразом;
  - обмеження-зв'язок – обмеження, яке визначається зв'зками між даними у вхідному наборі. Наприклад, це може бути сума всіх полів датасету, що не може перевищувати задане значення. Також, значення може бути присутнім лише за умови наявності відповідного значення в іншій таблиці.

- Точність – значення повинні бути в межах допустимої похибки. Загалом, отримати необхідну точність дуже важко, оскільки мова йде не про обчислення, а про роботу з уже існуючими даними, покращити точність яких може бути дуже ресурсно затратною операцією, а інколи і взагалі неможливою процедурою.

- Повнота – гарантія існування всіх полів, що вимагаються. Прикладом може бути дата: якщо є лише час, то потрібно доповнити його поточною датою або іншим значенням за замовчуванням, щоб поля дня, місяця і року були заповнені.

- Консистентність – ступінь відповідності частини даних до інших аналогічних входжень в інших місцях. Наприклад, якщо користувач в одній таблиці має вказаний номер телефону, а в іншій номер має відмінне значення – виникає внутрішня неконсистентність. Проблемою є те, що таку неконсистентність інколи неможливо вирішити: який з телефонів правильний вирішити неможливо без додаткових даних.

- Однорідність – дані мають відповідати єдиній системі вимірів. Наприклад, якщо деяка колонка містить відстань, то вона буде представлятися певним числом, але якщо це число в одних записах виражене в метрах, а в інших – в кілометрах, дані будуть хибними, навіть задовольняючи всі вимоги вище.

Також варто згадати поняття *інтегрованості* даних – воно охоплює в собі всі вище перелічені дані і означає можливість запису бути доданим до системи, не порушуючи її цілісності.

1.3.3 Процес попередньої обробки даних

Аудит даних – перевірка для виявлення аномалій та невідповідностей, отримання характеристик аномалій та їх входжень. Визначення робочого процесу – виявлення та видалення аномалій здійснюється послідовністю операцій над даними, також відомою як робочий процес (*workflow*). Для ефективного робочого процесу необхідно знати причини виникнення аномалій та помилок в даних.

Виконання робочого процесу (*workflow launch*) – запуск процесу, визначеного на попередньому етапі з урахуванням наявних обчислювальних потужностей та ефективносі обраного алгоритму, реалізація якого має враховувати обсяг оброблюваних даних.

Контроль та післяобробка – перевірка отриманих результатів на коректність. Контроль може включати в себе ще один етап аудиту та циклічний запуск процесу, якщо дані не пройшли відбір вимогами. Щоб дані всередині певної структури були високої якості необхідно дотримуватися таких вимог:

- відповідальність за внесення нових даних;
- інтеграція між різними середовищами та додатками;

- постійне вимірювання та покращення якості даних.

Також до процесу попередньої обробки можуть додаватися такі кроки:

- Парсинг – виявлення синтаксичних помилок. Контроль вхідних даних на відповідність деякій граматиці.
- Трансформація даних – перетворення вхідних даних в попередньо визначений формат, з яким уміє працювати існуючий додаток або який представляє дані у більш зручній формі.
- Видалення дублікатів – виявлення дублікатів та видалення однакових входжень. Зазвичай слідує за сортуванням вхідних даних за певним ключем, оскільки дублікати в даному випадку будуть знаходитися поруч, що спрощує їх пошук.
- Статистичні методи – отримуючи інформацію про середнє значення, стандартну похибку чи діапазон значень, можна встановити, що деякі значення не відповідають очікуванням, а отже є хибними.

Для процесу очищення даних характерні і недоліки, а саме: вартість проекту може сягати великих розмірів, оскільки вимагає роботи з величезними об'ємами даних; час - обробка може зайняти багато часу навіть на потужних кластерних системах; захист та безпека - надання доступу до інформації, обмін даних між додатками всередині системи є потенційно небезпечним і може спричинити витік чи перехоплення даних.

1.4 Отримані результати

В ході попереднього дослідження було виділені дві основних групи-представники машинного навчання, а саме: навчання з учителем та без нього. До методів навчання з учителем відноситься проблема класифікації на відміну від проблем кластеризації, що відносяться до методів навчання без учителя. Нада-

лі нас будуть цікавити саме алгоритми класифікації та їх робота з вхідними текстовими даними. Метою класифікації текстів є розподіл документів на групи наперед визначених категорій.

Оскільки робота з текстовими даними вимагає їх попереднього збору, було також розглянуто підходи до накопичення та препроцесингу даних. Це один з важливих кроків, який забезпечить розроблюваний алгоритм якісними вхідними даними для того, щоб зосередитися більше на технічній імплементації і математичному апараті, замість тонкого налаштування і обробки крайніх умов, враховуючи особливості датасету. Саме тому методам для обробки даних розглядаються окрема, а також було розроблемо незалежний компонент, що відповідає за нормалізацію даних та конвертацію їх формат, повністю придатний для роботи з іншими компонентами системи.

Також було враховано особливості алгоритмів, що здійснюють класифікацію: більшість з них працює з деяким числовим набором даних і очікує представлення будь-якої інформації саме у вигляді числових значень чи наборів значень (n-вимірних векторів для довільного багатовимірного простору). Тому надалі були розглянуті і методи перетворення текстових даних у числовий формат та можливі технічні проблеми, що виникають на таких етапах.

2 Розробка алгоритму

Для побудови прогностичних моделей розроблено та використовується величезна кількість алгоритмів. В загальному вигляді процес використання прогностичних моделей для вхідних даних можна представити у вигляді діаграми на рис. 3.

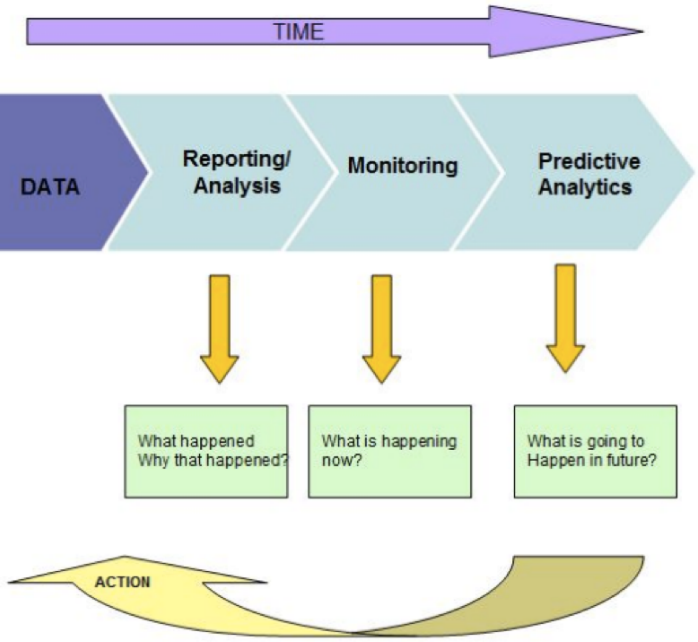

Рис. 3: Процес зміни даних з часом

2.1 Існуючі алгоритми

Враховуючи те, що результуюча гібридна модель буде використовувати деяку модель в якості базової, доцільно розглянути найбільш поширені алгоритми, що використовуються для класифікації даних. Це дасть змогу зрозуміти головні припципи роботи алгоритмів класифікації, а також за рахунок варіативності даних алгоритмів підкреслити універсальність даного підходу та можливість роботи з будь-якого типу базовими моделями.

2.1.1 Logistic regression

В найпростішому випадку опис деякої взаємодії можна звести до двох-трьох категорій змінних за допомогою простих операцій статистики. Проте, в більшості реальних ситуацій потрібно враховувати вплив куди більшої кількості таких змінних, які в загальному випадку можуть бути представлені як багатовимірні таблиці I × J × K. Саме тому доцільно розглянути узагальнений клас моделей, що носять назву *узагальнених лінійних моделей (**generalized linear models, GLMs**)*. Структурно дані моделі описують шаблони взаємодій та асоціацій. Параметри моделі показують силу таких асоціацій, а основний фокус під час створення таких моделей лягає в оцінку необхідних величин даних параметрів. Узагальнена формула для таких моделей:

$$y_i \sim N(x_i^T \beta, \sigma^2) \tag{1}$$

Значення вектору β містить коефіцієнти, які і потрібно визначити в ході тренування моделі. В таких моделях значення результуючої змінної підкоряється деякому закону експоненційного розподілу з середнім μ_iзначенням, який припускається є деякою нелінійною функцією від $x_i^T \beta$. Будь-яка модель даного класу містить три основних компоненти:

- Випадковий компонент (*random component*) - відноситься до розподілу ймовірностей результуючої величини Y, для лінійної регресії це звичайний нормальний розподіл Y. Також носить назву *шуму* або *помилки моделі*.

- Систематичний компонент (*systematic component*) - визначає набір додаткових величин $(X_1, X_2, \ldots X_k)$ для моделі, а саме їх лінійну комбінацію для створення так званого лінійного провісника (*linear predictor*), наприклад $\beta_0 + \beta_1 x_1 + \beta_2 x_2$ для лінійної регресії.

- Зв'язуюча функція (*link function*, $\eta, g(\mu)$) - виражає зв'язок між випад-

ковим та систематичним компонентом. Вона показує, яким чином вихідне значення зв'язане з прогнозованим значенням додаткових величин (наприклад, $\eta = g(E(Y_i)) = E(Y_i)$ для лінійної регресії).

Загалом серед переваг *GLM*-моделей можна виділити такі:

- не потрібно видозмінювати вихідну величину Y, щоб отримати нормальний розподіл;
- вибір зв'язуючої функції не залежить від вибору випадкового компоненту, таким чином ми отримуємо більшу гнучкість на етапі моделювання;
- якщо зв'язуюча функція адитивна, то нам не потрібно стала дисперсія;
- підходи для перевірки моделей є загальними для всіх підвидів класу, тому немає необхідності здійснювати використання окремих інструментів для кожного типу моделі.

2.1.2 Linear Discriminant Analysis

Linear Discriminant Analysis та Quadratic Discriminant Analysis - два класифікатори, що пропонують лінійну та квадратичну поверхню виборів відповідно. Даного роду алгоритми є досить привабливими, оскільки можуть бути обраховані без додаткових витрат, перевірені на практиці та не мають додаткових метапараметрів для тонкого налаштування.

На рис. 4 зображено встановлені алгоритмами межі для двох класів. Нижній рядок показує, що лінійний аналіз може здійснити лише лінійний поділ, в той час як квадратичний є більш гнучким. Даний алгоритм часто використовується для зменшення розмірностей простору деякої величини. Він проектує вхідні дані на лінійний підпростір, що максимізує межу поділу між класами. Розмірність вихідних даних завжди менше, ніж кількість

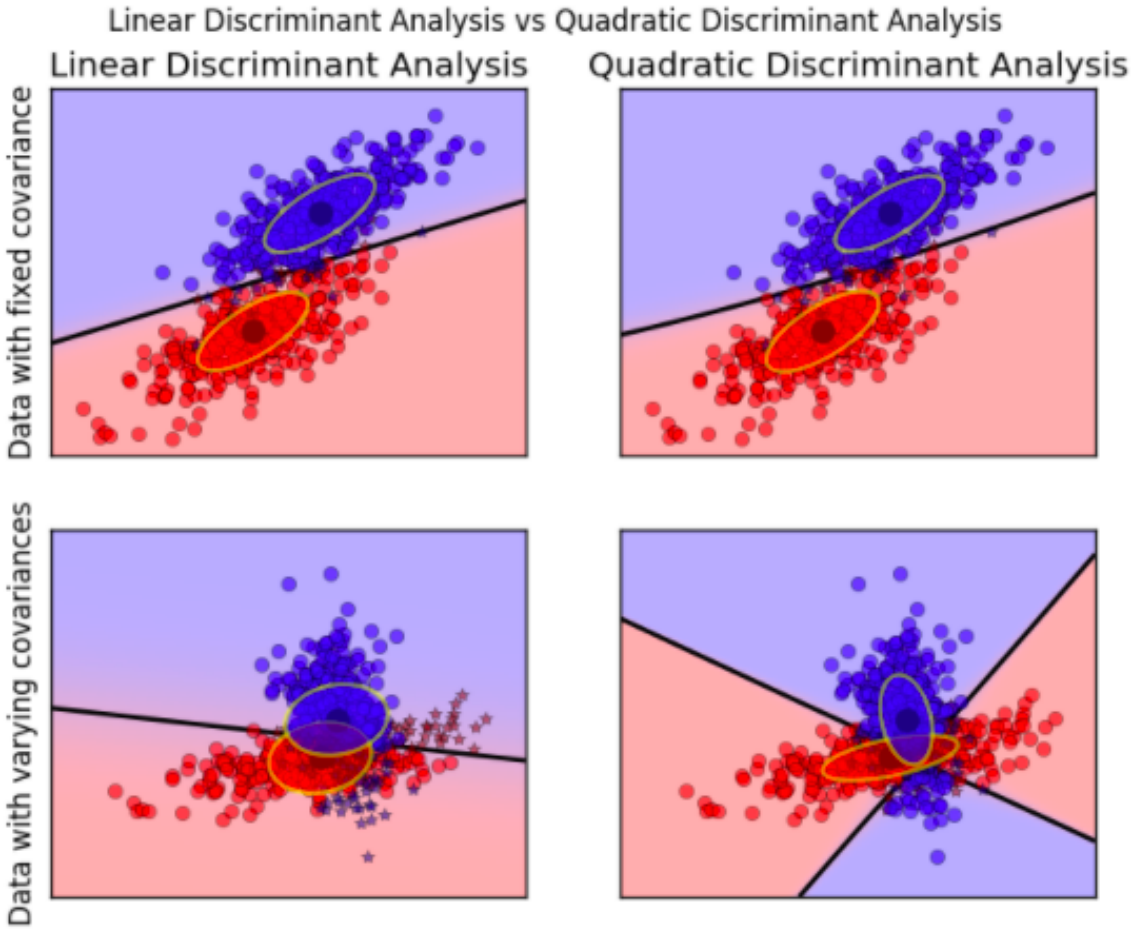

Рис. 4: Візуалізація лінійного та квадратичного дискримінантного аналізу

класів, тому такий прийом має сенс лише під час виконання мультикласифікації. Обидві моделі унаслідувані від простих ймовірносних моделей на основі умовного розподілу даних $P(X|yk)$ для кожного класу k. Прогнозована величина може бути отримана, використовуючи формулу Байєса:

$$P(y = k|X) = \frac{P(X|y = k)P(y = k)}{P(X)} = \frac{P(X|y = k)P(y = k)}{\sum_l P(X|y = l) \cdot P(y = l)} \quad (2)$$

Потім ми обираємо клас k, який максимізує умовну ймовірність. Щоб використовувати дану дану модель в якості класифікатора необхідно визначити з вхідних тренувальних даних такі величини: $P(y = k)$ - відношення екземплярів для класу k, середні значення для класу μ_i та коваріативну матрицю. У випадку, якщо кількість вхідних даних для тренування досить мала в порівнянні з кількість незалежних змінних датасету (*features*), мо-

жна скористатися параметром *усадки* (*shrinkage*). Це значення в межах від 0 до 1, де 1 означає, що діагональна матриця дисперсії буде використана для обрахунку коваріативної матриці. Використання усадки на вхідних даних дає змогу збільшити точність класифікації для варіанту, згаданого вище (рис. 5).

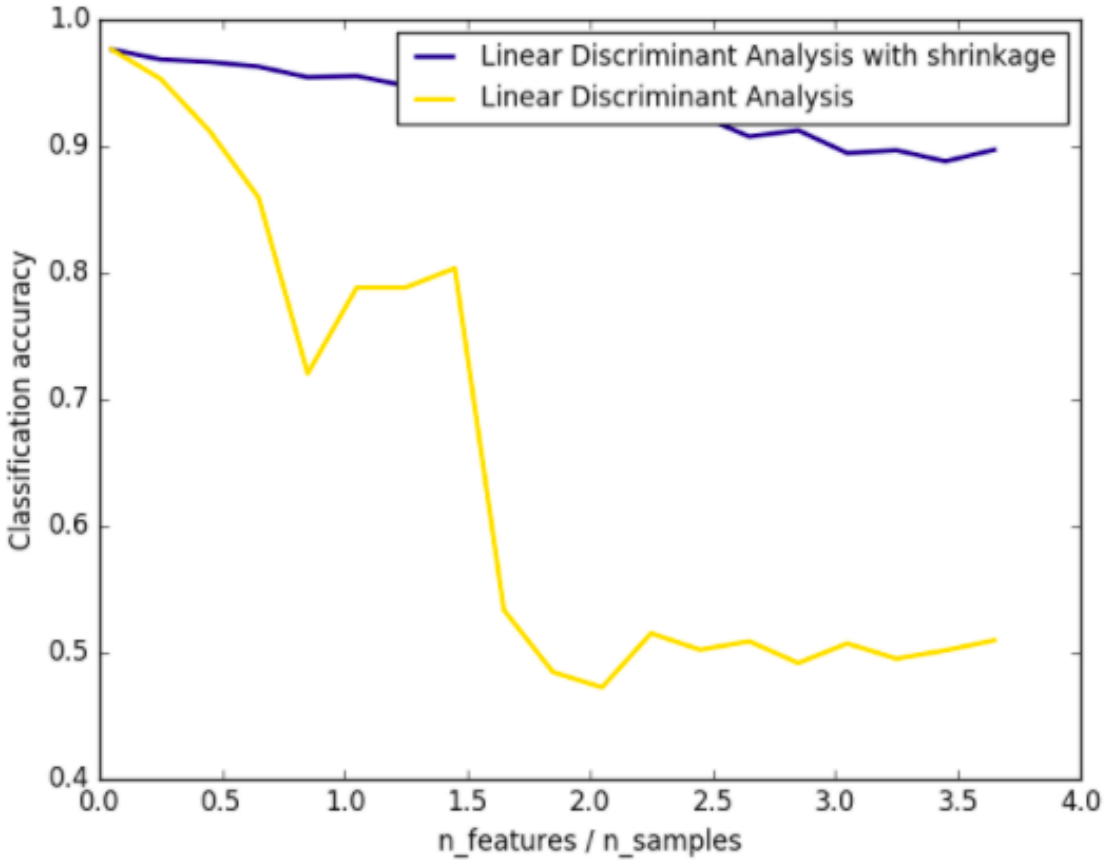

Рис. 5: Показники точності з використання усадки та без неї

2.1.3 Gaussian Naive Bayes

Наївні Байєсівські методи - це набір методів, що використовуються для навчання з учителем та ґрунтуются на застосуванні теореми Байєса з "наївним" припущенням, що кожна пара колонок вхідного датасету (*features*) є незалежною між собою. Маючи змінну класу та залежний вектор характеристичних величин, теорема Байєса відображає таку залежність

$$P(y|x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n|y)}{P(x_1, \ldots, x_n)} \tag{3}$$

Припускаючи незалежність величин

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y) \tag{4}$$

для всіх i-тих елементів, дане відношення можна спростити до вигляду

$$P(y|x_1, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_i, \ldots, x_n)} \tag{5}$$

З даної формули тепер можна вивести таке правило для класифікації:

$$\hat{y} = \underset{y}{argmax} P(y)\prod_{i=1}^{n} P(x_i|y) \tag{6}$$

Одним з часто використовуваних підвидів Байєсівського класифікатора є класифікатор з використанням розподілу Гауса:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}) \tag{7}$$

Різні підвиди наївного Байєсівського класифікатора відрізняються в основному припущеннями, які вони здійснюють стосовно розподілу $P(x_i|y)$. Незважаючи на використання таких занадто спрощених припущень, наївні Байєсівські класифікатори дуже добре працюють в багатьох реальних прикладних ситуаціях. Велика кількість класифікаторів документів, а також спам-фільтрів побудована саме на такого роду класифікаторах. Вони вимагають невеликої кількості вхідних даних для тренування та швидко встановлюють необхідні параметри. Ще однією з переваг є те, що такі класифікатори показують високу швидкодію в порівнянні з більш складними методами.

2.1.4 Support Vector Machines

Support Vector Machines - це також множина методів, що використовуються для класифікації, а також для визначення аномалій вхідних даних. Основними перевагами для використання цих методів є такі:

- ефективність в багатовимірних просторах;
- непогані показники навіть за умов, коли кількість розмірностей більша за кількість екземплярів;
- використання підмножини точок тренувального набору в якості функції рішення, що робить алгоритм ефективним з точки зору використання пам'яті;
- різносторонність: в якості функції рішення можна вибрати велику кількість уже існючих або використати будь-яку свою.

Щодо недоліків цих методів включають:

- низька швидкодія за умови, коли кількість характеристичних ознак набагато більша за кількість зразків;
- сам метод не надає жодних функцій для оцінки ймовірностей та вимагає додаткового кроку крос-валідації

В якості функції рішення (*kernel function*) можна обрати будь-яку з наведених:

- linear: $\langle x, x' \rangle$;
- polynomial: $(\gamma \langle x, x' \rangle + r)^d$;
- rbf: $exp(-\gamma |x - x'|^2)$;
- sigmoid: $tanh(\gamma \langle x, x' \rangle + r)$.

З математичної точки зору даний метод будує гіперплощину чи множину гіперплощин в багаторозмірному просторі, які потім використовуються для класифікації.

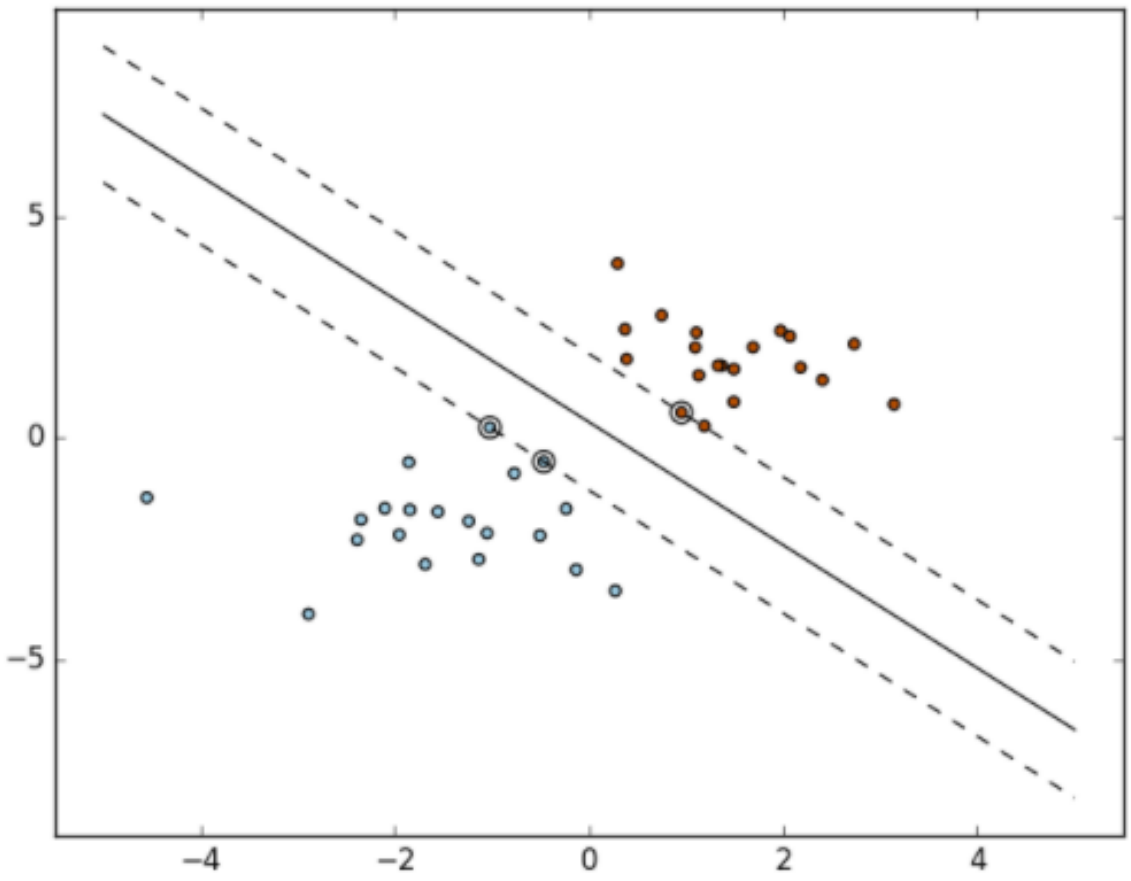

Рис. 6: Знаходження межі та поділ на класи

Поділ на класи виконується за рахунок знаходщення такої гіперплощини (рис. 6), що дає найбільшу відстань між найближчими точками з тренувального набору даних будь-якого класу, оскільки чим більша межа, тим мена помилка узагальнення для класифікатора. Для класифікації використовується така функція вибору:

$$sgn(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho) \tag{8}$$

2.1.5 Decision Tree Classifier

Дерева вибору (*decision trees*) - набір непараметризованих методів для машинного навчання з учителем, що використовуються для класифікації. Метою є створення такої моделі, що буде передбачати значення досліджуваної величини на основі простого набору правил, виведених з даних характере-

стичних величин. Приклад на рис. показує як за допомогою дерев вибору здійснити апроксимацію синусоїди за допомогою набору правил *"якщо-тоді-інакше"*. Чим більше рівнів у дерева, тим більша кількість правил і тим точнішою є модель.

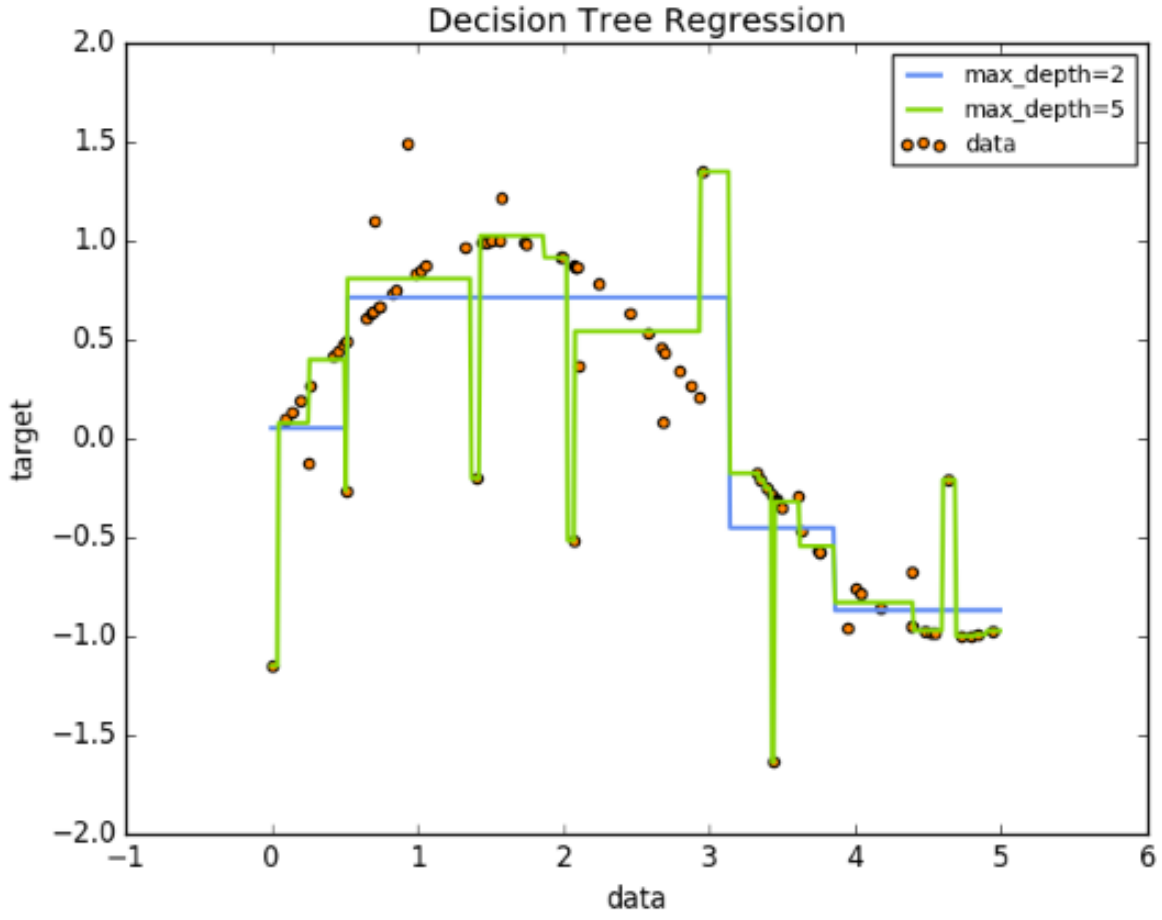

Рис. 7: Апроксимація синусоїди деревом вибору

Серед переваг використання дерев вибору виділимо такі:

- простота розуміння та інтерпретації;
- можливість простої візуалізації;
- майже не вимагає попередньої підготовки даних; нормалізація чи видалення порожніх значень можуть бути пропущені;
- логарифмічна складність алгоритму;

- можливість роботи як з числовими, так і з нечисловими даними;
- використання у алгоритмах, що потребують множинних відповідей;
- використання моделі *white box* (відкритої коробки): обґрунтування вибору моделі може бути виведене звичайною булевою алгеброю, в той час як рішення інших моделей (наприклад, нейронних мереж) може бути складним для пояснення і розуміння людиною;
- можливість валідації моделі, використовуючи лише статистичні перевірки;
- показує гарні результати, навіть якщо припущення у деяких умовах є хибними.

Хоча дерева вибору і мають велику кількість переваг, на противагу їм є і достатня кількість недоліків:

- навчаючись, модель може створити занадто громіздке дерево, яке насправді не зможе гарно узагальнити вхідні дані. Також можливий ефект надмірної точності (*overfitting*), коли модель занадто добре працює на вхідних даних, але показує незадовільні результати для будь-яких інших даних;
- дерева можуть бути нестабільними, оскільки навіть невелика зміна чи відхилення у вхідних даних призводить до побудови зовсім іншого дерева. Даний проблема вирішується використанням іншого класу алгоритмів на основі множини дерев.
- для того, щоб створити оптимальне дерево вибору потрібно вирішити *NP*-повну задачу, а це означає, що ми не завжди отримаємо найкращий результат, оскільки потрібно використовувати жадібні алгоритми чи евристики;

- дерева вибору схильні давати упереджені результати, якщо деякий клас є домінуючим. Результати передбачення будуть зміщені в сторону домінуючого класу. Щоб вирішити таку проблему використовують попередню обробку чи балансування вхідних даних.

Критерій класифікації, що застосовується для дерев вибору, виражається так:

$$p_{mk} = 1/N_m \sum_{x_i \in R_m} I(y_i = k) \tag{9}$$

Це буде виражати пропорцію входжень деякого класу k у вузлі m. Для визначення похибки користуються формулою:

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk}) \tag{10}$$

2.1.6 K Nearest Neighbours

Алгоритми групи "найближчих сусідів" можуть бути використані для машинного навчання як з учителем, так і без нього. Методи даного класу дуже часто застосовуються в кластеризація та є базою для багатьох інших методів. Основний принцип даних алгоритмів дуже простий - знайти визначену кількість входжень серед заданого вхідного набору даних (сусідів), що мають найменшу відстань до вибраної нової точки і на основі цього віднести її до того ж самого класу. Кількість таких входжень регулюється деякою константою k, звідси і назва алгоритму. Відстань може бути як і звичайною Евклідовою відстанню (що використовується найчастіше), так і будь-якою іншою метрикою для довільного простору. Незважаючи на простоту алгоритму, він використовується під час вирішення досить складних задач, таких як розпізнавання рукописного тексту чи знімків супутника. Метод є непараметризованим, а тому використовується в ситуаціях, коли межа поділу між класами досить нечітка і непостійна.

Для класифікації на основі цього алгоритму достатньо більшості голосів для кожної точки, тобто знаходження найбільшої кількості екземплярів деякого класу серед вибраної кількості сусідів. Значення буде віднесено до того класу, який містить найбільшу кількість представників серед вибраних найближчих сусідів. Якщо вхідні дані розподілені нерівномірно, доцільно обрати не наперед визначену кількість сусідніх точок, а деякий радіус r, та розглянути всі точки, що потрапляють в отриману множину.

Також інколи доцільно застосовувати "зважений" метод. Це модифікація яка надає деяке значення ваги кожному з сусідів. У найпростішому випадку вагою може бути коефіцієнт від відстані: чим ближче точка-сусід знаходиться до нашої цільової точки, тим більше значення ваги вона матиме. Для ваги можна визначити будь-яку користувацьку функцію, що робить алгоритм досить гнучким і дозволяє більш точно визначити межі класів.

Щоб зробити знаходження найближчих сусідів швидким процесом продиться велика кількість досліджень, оскільки найпростіший спосіб зводиться до повного перебору та обчислення відстаней до всіх точок. Для N точок в D вимірному просторі це вимагає порядку $O[DN^2]$ операцій, що прийнятно лише для невеликої кількості точок. Наразі існують підходи на основі дерев, що дозволяють зробити пошук дещо швидшим: *K-D Tree* та *Ball Tree*.

2.2 Оптимізація алгоритмів для використання у предметній галузі

Оскільки велика кількість алгоритмів вимагають специфічного середовища чи платформи, а для побудови інших необхідні значні обчислювальні потужності, постає проблема платформозалежності та неможливості використання кращих підходів за умови існування додаткових обмежень. Щоб уникнути даних проблем, достатньо розробити рішення, що буде відповід-

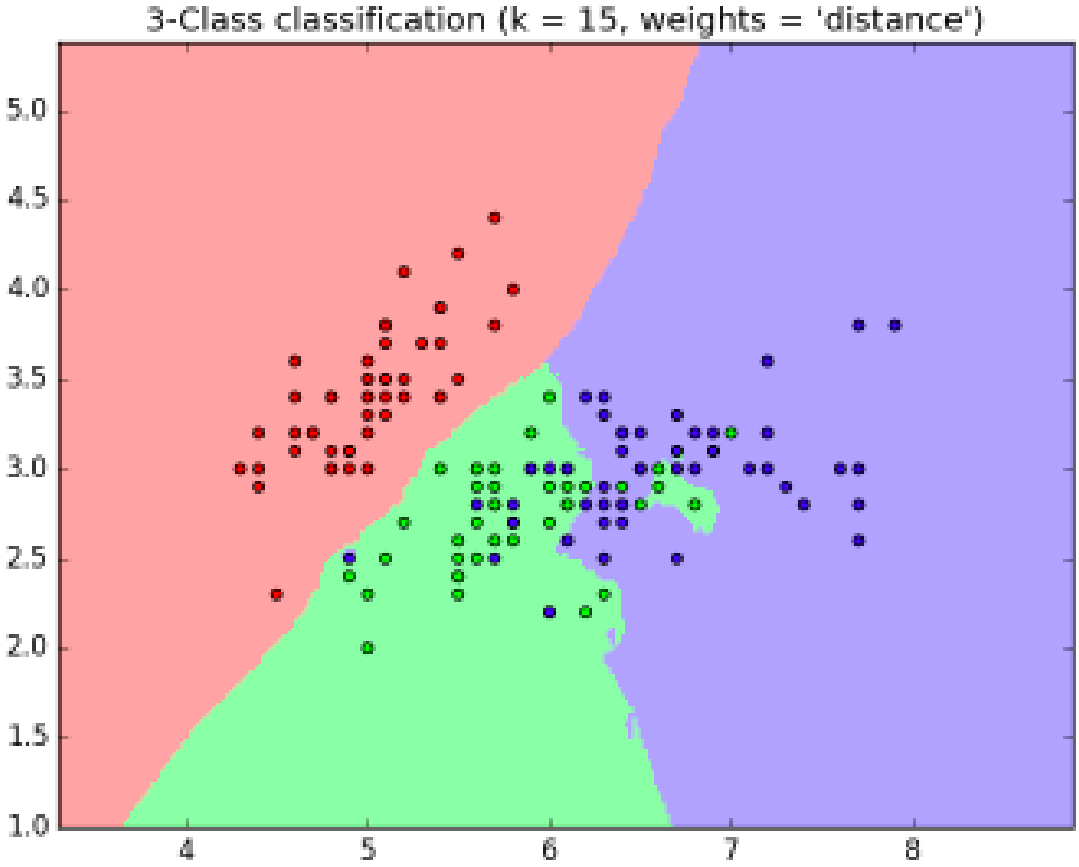

Рис. 8: Класифікація методом найближчих сусідів для трьох класів

ати поставленим нижче вимогам:

- доступ користувача до коду моделі на будь-якій платформонезалежній мові, що дозволить запускати її в довільному середовищі та не спиратися на використання сторонніх бібліотек;
- будова моделі та деталі її внутрішньої реалізації повинні бути відкритими, тобто користувач повинен мати змогу переглянути вихідний код і в разі необхідності самостійно відтворити довільний крок та отримати аналогічний результат передбачення для однакового набору вхідних даних;
- модель повинна мати точність максимально наближену до точності моделей, що показують найкращі результати для вибраних вхідних

даних. Модель повинна мати аналогічні показники щонайменше для 95% всіх вхідних наборів даних;

- виконання коду програми повинно бути швидким (близько 1 мс на рядок вхідних даних).

Єдиного рішення, що дозволило б відмовитися від наявних алгоритмів на користь одного, визначеного вимогами вище, поки що немає, але існує підхід, що дозволяє покрити перелік всіх умов. Даний підхід носить назву апроксимаційної моделі. Основна ідея полягає в припущенні, що деяка відносно проста модель, побудована на основі передбачень більш складної моделі може показати схожі результати в межах допустимого відхилення. Саме такою моделлю є RuleFit-модель, або модель на основі класу визначених правил. Принцип роботи полягає в серії тренувань дерев вибору на вхідних даних з наступною конвертацією гілок дерев в класи правил. Наприклад, одне правило може виражатися такою формулою: $20 < age <= 30 and income > 10000$. Новий набір даних створюється на основі оригінальних вхідних даних, генеруючи набір індикаторів $0/1$ таким чином, що кожен рядок позначає негативне чи позитивне значення в залежності від результату застосування правила до цього рядка. Дані індикатори потім використовуються в якості значень передбачення для узагальненої лінійної моделі.

$$y(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p \tag{11}$$

Формула 11 описує звичайну модель, що є лінійною комбінацією вхідних значень та вектора коефіцієнтів w, а y – це значення, для якого здійснюється передбачення.

RuleFit є простою моделлю в тому плані, що це лише список правил з відповідними коефіцієнтами для кожного з них. Однак існують і недоліки, пов'язані з тим, що класи правил можуть містити подібні правила, що

ускладнює інтуїтивне розуміння впливу коефіцієнтів, тобто вносить складність для розуміння людиною.

Існує дві основних частини в реалізації алгоритму: безпосереднє тренування і передбачення та rulefit задача. Дана задача містить багато параметрів, найголовнішим з яких є альфа-параметр, що визначає розмір регуляризації, що застосовується до лінійної моделі RuleFit.

Кінцевим етапом створення такої моделі повинна бути валідація згенерованого коду. Оскільки немає жорсткої прив'язки до використовуваної мови, потрібно переконатися, що код компілюється та виконується коректно. Далі потрібно запустити код та зберегти файл з прогнозованими значеннями для подальшого порівняння зі значеннями, що отримуються від оригінальної моделі. Якщо похибка лежить в межах допустимого відхилення валідація вважається успішно пройденою.

3 Програмна реалізація

В якості мови програмування було обрано Python, оскільки це високорівнева інтерпретована мова програмування, що дозволяє швидко здійснювати побудову прототипу та скорочує час на розробку продукту вцілому. Значна частина системи являє собою веб-додаток, що стало ще одним фактором під час вибору даної мови, оскільки вона є веб-орієнтованою та містить величезну кількість бібліотек та сторонніх модулів для розробки саме веб-ресурсів. Python також дуже часто використовують в сфері збору та аналізу даних, тому що за рахунок можливості роботи в інтерактивному режимі інтерпретатора можливо значно зекономити час під час роботи з будь-якого роду даними. Гомогенність мов розроблюваних додатків дозволяє зберігати контекст і не перемикатися на синтаксичні відмінності чи особливості мови під час розробки програми. Це, в свою чергу, зменшує час на написання та дозволяє уникнути необхідності працювати з декількома різними мовами одночасно. *aiohttp* був обраний в якості веб-фреймворку за рахунок своєї асинхронної природи: оскільки основна мета веб-ресурсу коректно обробити вхідні дані від усіх користувачів, потрібно мати змогу асинхронно опрацьовувати велику кількість з'єднань. *Redis* було обрано в якості локальної бази даних за рахунок його швидкодії. Дані зберігаються не на диску, а в пам'яті, що доволяє значно прискорити швидкодію в ситуаціях постійної роботи з записом та читанням. *Redis* також дозволяє здійснювати операцію зберігання поточного стану на диск, тому було також розроблено компонент, що виконує дану діяльність періодично протягом всього часу роботи системи.

Архітектура мікросервісів була обрана значною мірою тому, що в даному програмному продукті ми маємо окремі повністю незалежні компоненти, які до того ж відповідають за зовсім різні функції системи. Виокремлення системи збору даних дозволяє як і розробляти її окремо та незалежно від

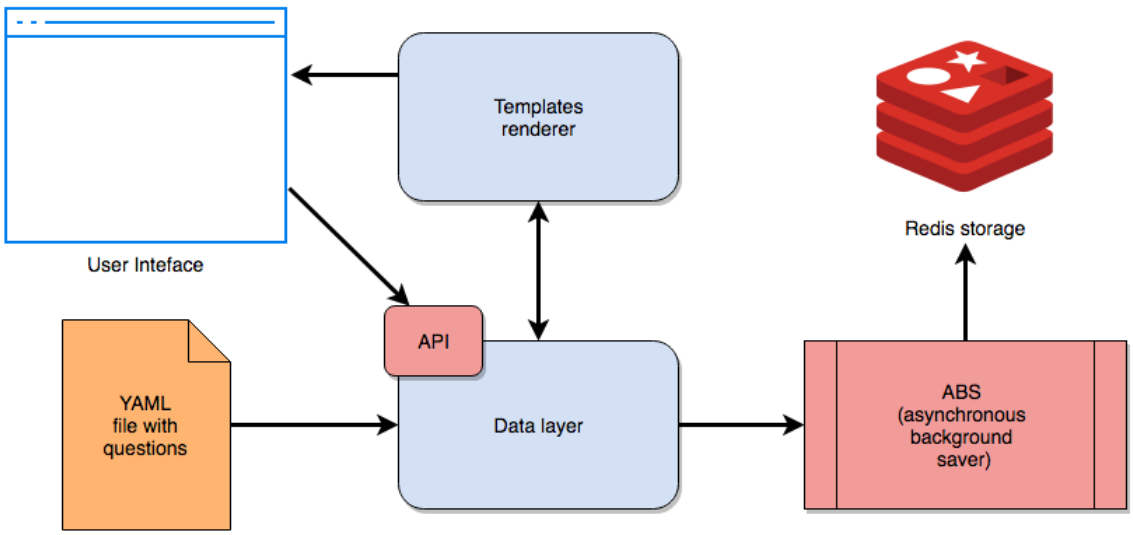

Рис. 9: Архітектура фреймоврку для проведення опитувань

інших складових, так і використовувати її в рамках інших систем, а зібрані в результаті її роботи дані в рамках будь-яких інших додатків. Платформа збору даних може бути використана взагалі незалежно в якості платформи для опитування, оскільки була розроблена як фреймворк і відповідно налаштовується під конкретні вимоги користувача. Архітектура фреймворку для опитування зображена на рис. 9.

3.1 Збір та попередня обробка даних

Напрямок збору даних розвивався разом з розвитком комунікаційних технологій, а особливо як складова будь-якого бізнес процесу компаній. Проблеми оптимізації систем роботи з клієнтами спричинили появу підходів, які зараз вважаються класичними. В переважній більшості ці підходи покривають 99,9% поточних бізнес задач. Враховуючи те, що за мету було поставлено не тільки розробку самого алгоритму для побудови моделі, але й збір даних для перевірки роботи цієї моделі, постала потреба загрегувати дані, представити їх у необхідному форматі та обробити їх належним чином. Для роботи з даними була обрана бібліотека *Pandas*, а для роботи

безпосередньо з числовими даними, векторами та статистичними функціями - бібліотека *NumPy*. Серед науковців та спеціалістів з обробки даних також широко використовується мова програмування *R*, яка уже містить значну частину вбудованих методів та навіть засобів візуалізації в стандарнтій бібліотеці самої мови, та *Python* був обраний з урахуванням необхідності розробки всього програмного комплексу в рамках однієї мови програмування.

Веб-додаток, що відповідав за збір даних представляє собою веб-сайт з декількома основними формами. Перша форма - це форма вибору типу респондента. На вибір були доступні три варіанти: студент, викладач та випускник. Далі, в залежності від вибору, відображалися форми з запитаннями для кожної групи користувачів. Основна увага приділена користувачам групи студентів, оскільки саме дані їх анкет використовувалися для подальшої обробки та слугували в якості вхідних даних для алгоритму. Загальний вигляд сторінки з запитанням зображено на рис. 10. Деякі питання мали наперед визначені відповіді, що можуть бути представлені у числовому форматі, інша група запитань зводилася до категорій відповідей (що також за допомогою таблиці відповідностей зводяться до множини визначених числових значень) та остання група - питання з розгорнутою відповіддю, - саме вони вимагають додаткової обробки та класифікації тексту відповіді для розуміння приналежності цих даних до деякого класу.

Процес обробки даних, трансформації значень та представлення даних в узагальненому форматі можна зобразити у вигляді діаграми на рис. 11

В кінцевому результаті дані зберігаються в уніфікованому *json*-форматі, що дає змогу: використовувати їх безпосередньо в консолі браузера для простішого перегляду і відлагодження; проводити легку конвертацію в *csv*-формат, який очікує більшість інструментів з обробки даних; перетворювати їх в *Python*-об'єкти для роботи з ними в коді програми; передавати на обробку в інші методи та функції, що вже готові працювати з даним форматом без

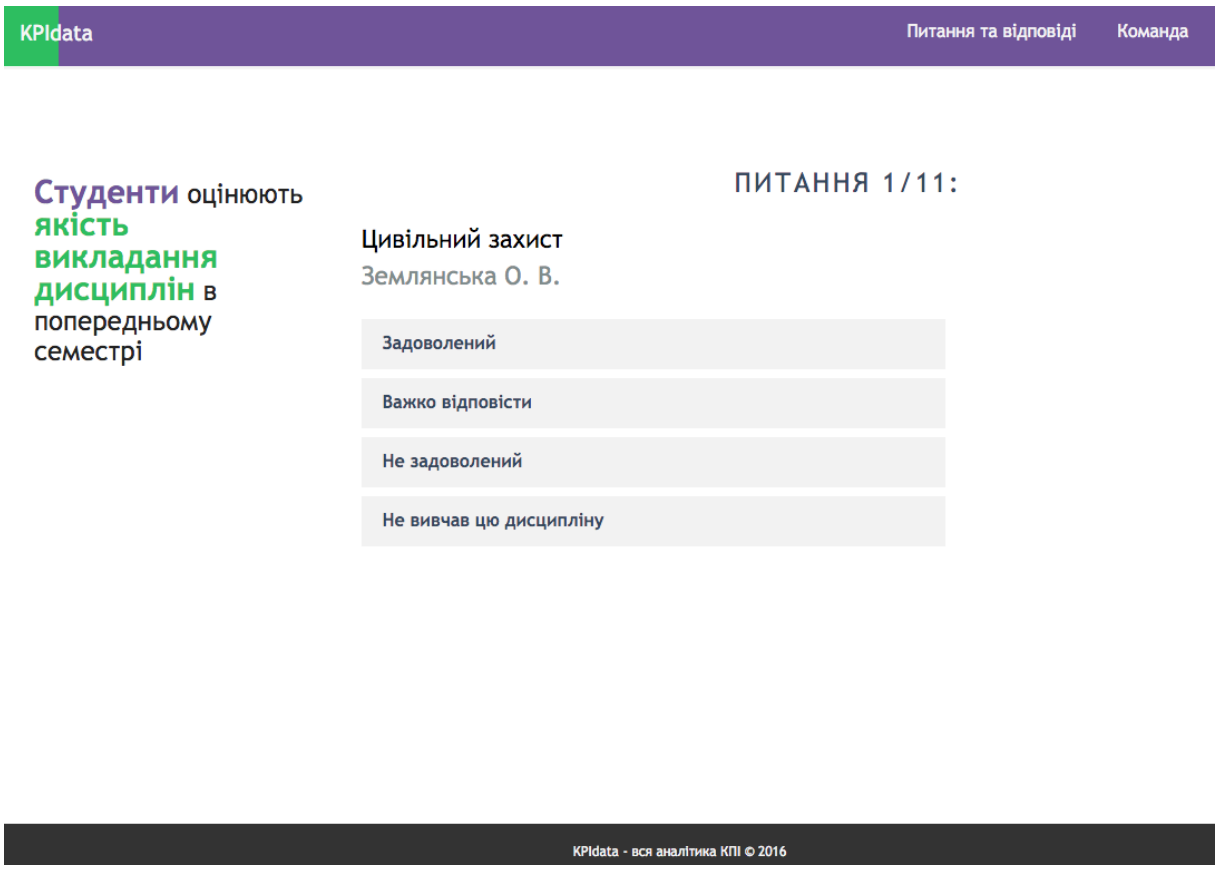

Рис. 10: Етапи попередньої обробки даних

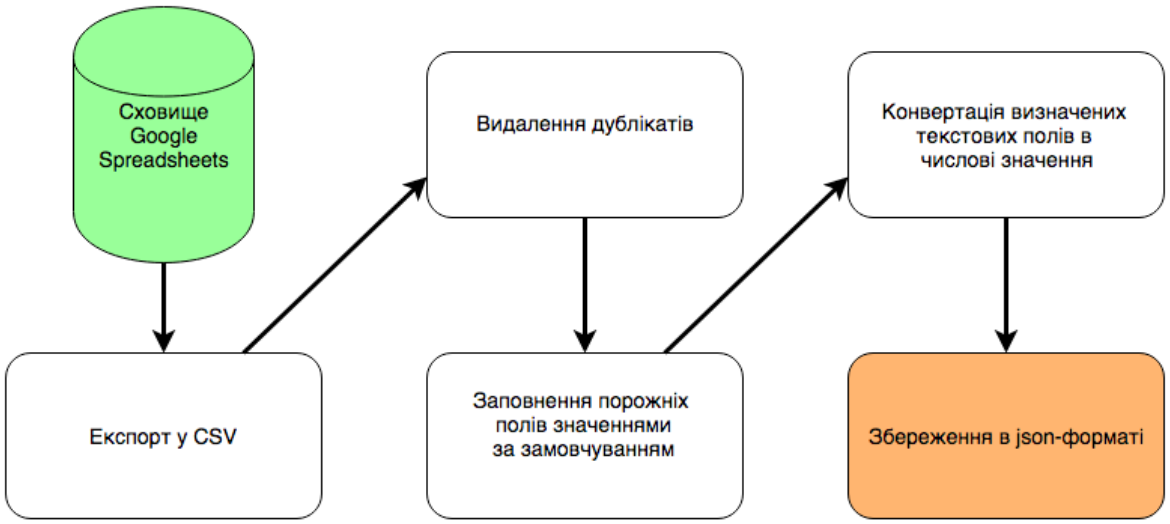

Рис. 11: Етапи попередньої обробки даних

необхідності додаткових серіалізацій/десеріалізацій; здійснювати конвертацію цих даних з додаванням відношень і зв'язків між ними (наприклад, для зберігання в реляційній *SQL*-базі даних); зберігати на диск з можливі-

стю перегляду в будь-якому текстовому редакторі.

Перед етапом обробки даних та жорсткого втручання в них (видалення записів, модифікація існуючих значень, заміна чи конвертація полів) необхідно мати загальне уявлення про вміст та загальні характеристики даних. На даному етапі на допомогу приходить статистика та її методи. *Статистика* - за своїм строгим визначенням не є технологією збору даних, проте саме вона використовується задля того, що знайти закономірності в даних та для наступної побудови прогностичних моделей. Також, з точки зору користувача, ви завжди будете стикатися з тими чи іншими інструментами статистики в будь-яких інших методах збору та аналізу даних. Статистика загалом являє собою розділ математики, пов'язаний зі збором та описом даних. Статистика займається підрахунком ключових значень та ймовірностей. Використання її в процесі збору даних допомагає відповісти на ряд важливих запитань, що відносяться до ваших даних: які закономірності прослідковуються з базі даних; яка ймовірність того, що обрана подія настане; які закономірності найбільш важливі; яку загальну інформацію можна отримати про дані, щоб зрозуміти з якого роду значеннями ми маємо справу.

Дану інформацію надалі можна використовувати для роботи з даними, оскільки це свого роду надає додаткову інформацію про доменну область (наприклад, на рис. зображена гістограма, що дає змогу швидко встановити факт про вік переважної кількості цільової аудиторії: більше 50). Інші з найчастіше вживаних метрик статистики:

- *max* - максимальне значення цільової величини;
- *min* - мінімальне значення цільової величини;
- *mean* - середнє значення обраної величини;
- *median* - значення, що розділяє вибірку на дві підмножини з макси-

мально наближеної кількістю елементів у кожній;

- *mode* - значення, що зустрічається найчастіше;
- *variance* - показник відхилення цільового значення від середнього у вибірці.

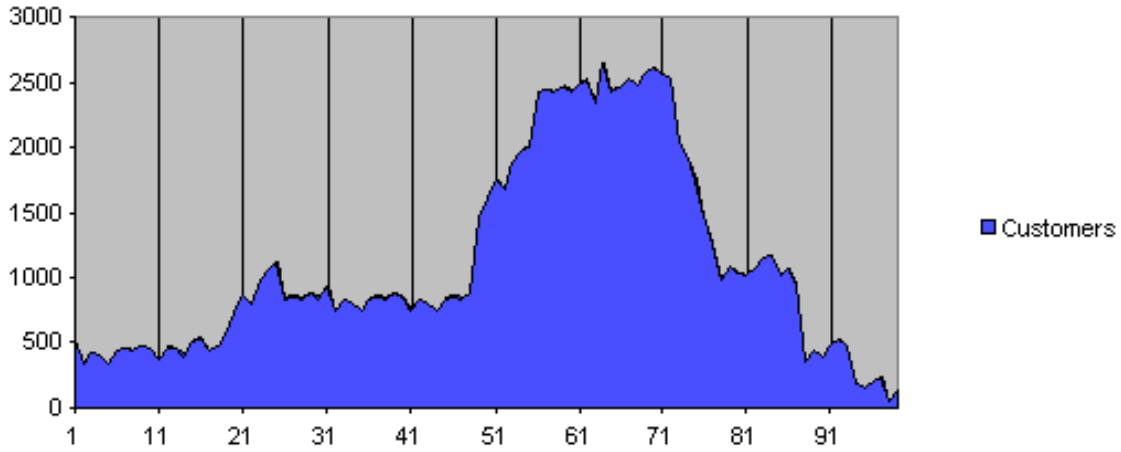

Рис. 12: Гістограма розподілу кількості клієнтів за віком

Останній показник, дисперсія, дещо складніший для розуміння: чим вище значення, тим більш дані різнорідні і різняться між собою. Якщо ж значення менше - дані загалом схожі і мало відрізняються від середнього по вибірці. Базуючись на статистичних даних, користувач має змогу налаштувати модель таким чином, щоб передбачуване значення максимально відоражало реальну зміну величини.

Для збору даних та формування початкового датасету було створено допоміжний додаток у вигляді веб-ресурсу. Він являє собою веб-сайт, на якому здійснюються опитування серед різних класів респондентів: студентів, випускників та викладачів. Кожен учасник опитування заповнює невелику тематичну анкету, на основі якої формується таблиця вхідних даних. Загальну архітектуру додатку можна побачити на рис. 13

Система побудована на основі архітектури мікросервісів і дотримання принципу делегування частини роботи зовнішнім ресурсам. За рахунок та-

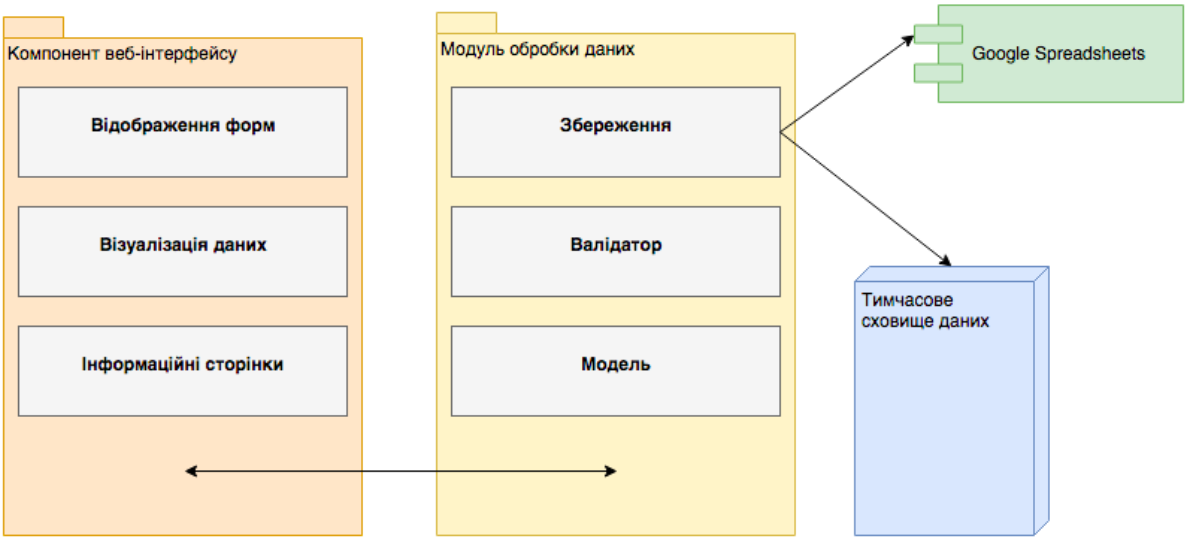

Рис. 13: Загальна архітектура додатку опитування

кого підходу можна отримати більшу ефективність, стабільність та пропускну здатність роботи системи, але знижується надійність, оскільки вихід хоча б одного компоненту з ладу призведе до повної недієздатності системи. Враховуючи важливий фактор роботи з даними, потрібно розуміти, що втрати на етапі збору значно впливають на подальші результати. Саме тому було додамо внутрішнє тимчасове сховище даних, яке слугує для збереження резервних копій. Система складається з таких компонентів:

- Веб-інтерфейс - являє собою веб-додаток, що побудований на основі aiohttp веб-фреймоврку та здійснює основні функції обробки користувацької логіки. Він відповідає за рендеринг сторінок, видачу статичних ресурсів, обробку та роутинг запитів та спілкується з іншими компонентами для подальшоої передачі даних.
  - Відображення форм - підкомпонент, що відповідає за відображення користувацького інтерфейсу та безпосередню взаємодію з користувачем за допомогою веб-браузера.
  - Візуалізація даних - представлення вхідних, існуючих, а також відображення статусу обробки поточних даних для розуміння

стану даних та прогресу під час заповнення форми опитування.

  - Інформаційні сторінки - відображення статичних сторінок вебресурсу для надання додаткової інформації та для отримання зворотного зв'язку.

- Модуль обробки даних - здійснює фільтрацію, нормалізацію та перетворення даних таким чином, щоб вони були уніфікованого формату та могли бути використанні надалі іншими компонентами. Обробка здійснюється з використанням можливостей самої мови, а також з допомогою сторонніх бібліотек *NumPy* та *Pandas*.

  - Підкомпонент збереження даних - створення об'єктів для кінцевих даних форм на основі моделей; створення асинхронних задач збереження даних; переріодична інкрементальна відправка проміжного стану для формування єдиної бази, використовуючи сторонній сервіс *Google Spreadsheets*. Збереження виконується як локально, використовуючи базу даних в оперативній пам'яті, так і віддалено, використовуючи основну базу даних на основі таблиць.

  - Валідатор - здійснення перевірки даних на коректність та відповідність вхідним обмеженням. Видача повідомлень про помилку в разі невідповідності.

  - Модель - *data access object*, представлення вхідних даних у вигляді об'єктів мови програмування для надання зручного доступу до даних з коду програми. Серіалізація моделі дозволяє зберегти дані для подальшого використання та можливої додаткової обробки. За допомогою формального представлення даних у вигляді моделі можливо з легкістю проводити маніпуляції з даними в рамках інструментів мови програмування.

Особливості даних можна встановити ще на етапі їх збору. Навіть не здійснюючи жодного аналізу чи попередньої обробки можливо отримати деякі важливі характеристики даних або просто візуалізувати їх для зручнішого сприйняття чи для кращого усвідомлення того, з якого роду даними доведеться працювати. Саме для таких цілей і використовується *exploratory data analysis* - аналіз даних з метою попереднього дослідження вхідних даних.

Візуалізація даних загалом здійснюється для таких цілей:

- Комунікативна складова:
  - представити дані та ідеї;
  - проінформувати;
  - підтримати і аргументувати;
  - вплинути і переконати;
- Дослідницька:
  - вивчити (дослідити) дані;
  - проаналізувати ситуацію;
  - визначити наступні кроки;
  - прийняти рішення стосовно деякого питання;

Оскільки одним із компонентів збору даних був Google Spreadsheets, то початкова візуалізація даних не становила жодної складності, тому що даний ресурс містить вбудовані засоби для цього рис. 14

Для збору даних потрібно враховувати декілька важливих нюансів. По-перше, це *робота з неповними даними* - ситуація, коли зібрані дані не відповідають поставленим критеріям чи не проходять валідацію, але не можуть бути відкинуті, оскільки є досить важливими. Схожа ситуація може

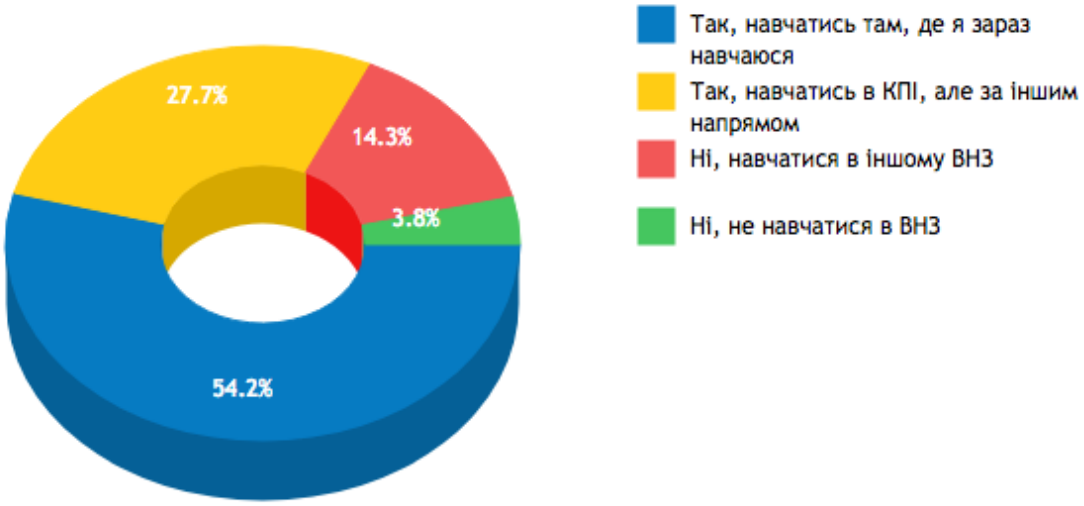

Рис. 14: Візуалізація відповідей однієї з форм анкети опитування

відбутися, коли дані взагалі зберігаються в різної кількості форматах та структур, тому виділити один загальний критерій для валідації може бути досить важко. Тому потрібно завжди враховувати випадок обробки та роботи з неповними даними.

Врахування ефективності алгоритмів, що використовуютьс з роботи з даними - під час обробки невеликої кількості даних ефективність алгоритму не відіграє значної ролі, оскільки обчислювальних ресурсів комп'ютера зазвичай більш ніж досить, тому користувач майже не помічає затримок у роботі. Але з ростом об'єму даних (які можуть сягати декількох сот гігабайт) час на їх обробку може зростати нелінійно, а отже операції можуть перевищувати встановлені обмеження на тривалість роботи. Тому необхідно враховувати кількість даних, з якими повинна буде працювати системи і обирати максимально ефективні алгоритми для роботи.

Отримання великого обсягу початкових даних - під час скрапінгу вебресурсів або під час потокового отримання вхідних даних (*streaming*) можуть виникнути проблеми, пов'язані з неможливістю процесу обробити всі дані одночасно чи зберігати їх в рамках однієї сесії в оперативній пам'яті.

Для вирішення такої проблеми можна скористатися такими підходами: кешування інформації на проміжних етапах, використання вбудованих функцій замість написання власних послідовностей обробки на стороні бізнес-логіки (прикладом може бути використання stored procedures під час роботи з *SQL* базами даних), побудова алгоритмів, що використовують принцип *pipeline* (вихід одніє функції відразу слугує вхідними даними для іншої, таким чином відпадає необхідність зберігати дані в проміжному стані та витрачати додатковий простір на жорсткому диску), пакетна обробка даних.

Обробка даних, що містять зв'язки між собою - досить поширена ситуація, коли одні дані містять посилання на інші, або потрібно відслідковувати зв'язки між певними компонентами вхідної інформації. В такому випадку немає простих рішень чи рекомендації, оскільки в будь-якому випадку потрібно буде підтримувати структуру даних, що буде відповідати за збереження даних зв'язків. Очевидним рішенням є використання реляційних баз даних, які будуть підтримувати структуру і відношення чи альтернативне використання графових баз даних, принцип яких саме в побудові бази таким чином, щоб вона представляла собою граф відносин між даними.

Обробка даних, що містять зв'язки між собою - досить поширена ситуація, коли одні дані містять посилання на інші, або потрібно відслідковувати зв'язки між певними компонентами вхідної інформації. В такому випадку немає простих рішень чи рекомендації, оскільки в будь-якому випадку потрібно буде підтримувати структуру даних, що буде відповідати за збереження даних зв'язків. Очевидним рішенням є використання реляційних баз даних, які будуть підтримувати структуру і відношення чи альтернативне використання графових баз даних, принцип яких саме в побудові бази таким чином, щоб вона представляла собою граф відносин між даними.

Підтримка гетерогенності джерел інформації - системи, що використовуються для отримання даних можуть не мати уніфікованого інтерфейсу та не надавати однакові можливості зі свого боку для здійснення запитів.

Потрібно враховувати відмінності між кожним окремим джерелом, а також розуміти встановлені обмеження (наприклад, багато веб-ресурсів встановлюють обмеження на кількість запитів, що здійснюються за певний проміжок часу). Тому якщо додаток буде розроблено без урахування таких відмінностей - він може показати добрі результати під час роботи з одним провайдером інформації, але для інших він не буде функціонувати належним чином.

3.2 Побудова моделі

Для створення ефективної кінцевої моделі розроблюваний алгоритм повинен підповідати таким вимогам:

- відкритий доступ користувача до коду моделі на будь-якій платформонезалежній мові, що дозволить запускати її в довільному середовищі та не опиратися на використання сторонніх бібліотек;
- будова моделі та деталі її внутрішньої реалізації повинні бути відкритими, тобто користувач повинен мати змогу переглянути вихідний код і в разі необхідності самостійно відтворити довільний крок та отримати аналогічний результат передбачення для однакового набору вхідних даних;
- модель повинна мати точність максимально наближену до точності моделей, що показують найкращі результати для вибраних вхідних даних. Модель повинна мати аналогічні показники щонайменше для 95% всіх вхідних наборів даних;
- виконання коду програми повинно бути швидким (близько 1 мс на рядок вхідних даних).

Використовуючи отримані на попередніх етапах дані та проаналізувавши висунуті вимоги, опишемо розроблений алгоритм для побудови кінце-

вої моделі (припускаємо, що етап збору даних уже пройдено і вони зберігаються в централізованому сховищі):

1. Виконання алгоритмів попередньої обробки та трансформації даних для початкового "сирого" набору даних. Здійснюються всі необхідні кроки для очищення та нормалізації, а також побудова додаткової статистики для опису властивостей вхідних даних.

2. Конвертація текстових даних для характеристичних змінних, що представлені не у числовому вигляді. На даному етапі здійснюється безпосереднє перетворення текстових даних в числовий чи векторний формат, з яким можуть працювати всі алгоритми класифікації.

3. Визначення множини алгоритмів класифікації, для яких на основі вхідних даних буде відбуватися створення і тренування моделі. Наразі будуються всі моделі з множини наперед визначених алгоритмів не залежно від набору даних. Перед етапом побудови виконується поділ датасету на дані для тренування та валідації, а також допоміжні розділи, якщо цього вимагає алгоритм.

4. Валідація всіх побудованих моделей та здійснення прогнозування досліджуваної величини. Також перевірка точності всіх моделей за обраною метрикою задля сортування та вибору найкращої з усіх моделей.

5. Для обраної найкращої моделі виконати побудову "гібридної моделі" за допомогою методу *RuleFit*, таким чином здійснити її апроксимацію для забезпечення мінімальної похибки передбачення, порівнюючи з базовою моделлю.

6. Для всіх наступних даних, для яких ми хочемо прогнозувати значення досліджуваної величини, використовувати лише одну побудовану на попередньому етапі модель. Саме це забезпечить поєднання то-

чності передбачення найкращої моделі та швидкодію реалізації кінцевої моделі.

7. Здійснити збереження моделі для подальшого використання в разі надходження нових даних, якими ми хочемо скористатися для передбачення.

Даний алгоритм можна спрощено зобразити у вигляді блок-схеми (рис. 15). Як бачимо на схемі, спочатку здійснюється побудова N моделей, для кожної з яких проводиться тренування і методом оцінки за вибраним критерієм визначається найкраща з них. Потім на основі моделі, що дала найкращі показники, здійснюється побудова кінцевої "гібридної" моделі, що усюди надалі використовується для прогнозування передбачуваної величини.

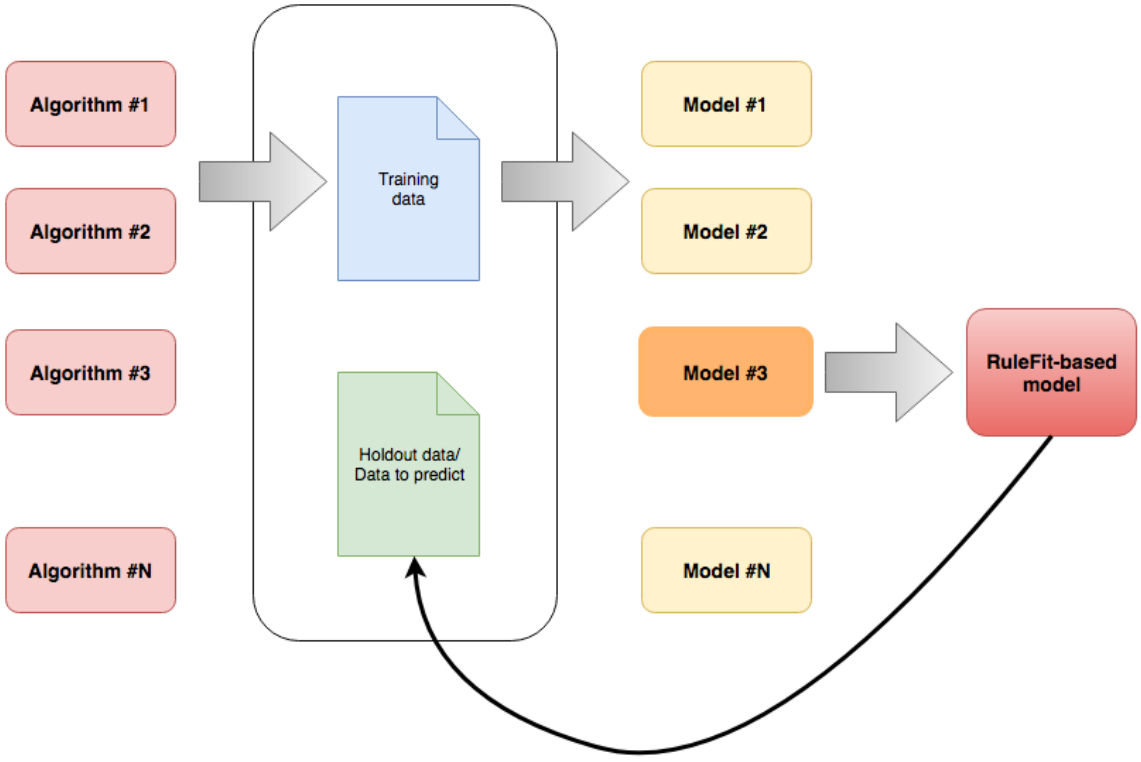

Рис. 15: Алгоритм побудови гібридної моделі

Можливий частковий випадок, коли декілька моделей будуть мати однакові показники для точності передбачення. В такому разі можна скориста-

тися кількома метриками і оцінювати моделі за множиною параметрів, а не лише за одним критерієм. Також можливим рішенням є використання будь-якої моделі чи використання їх обох за умови, що витрати на час є доцільними. Одним із цікавих варіантів розвитку даного дослідження у такому випадку є створення деякої композиції k найкращих моделей (використання їх паралельно та подальше усереднення результатів чи, в загальному випадку, створення деякої лінійної комбінації вихідних даних всіх моделей для подальшого визначення найбільш точного результату), що з точки зору інкапсуляцію будуть поводити себе як одна модель, проте давати більш точні передбачення, що безсумнівно є досить привабливою перевагою для науковців.

3.3 Результати реалізації

Отже, було розроблено три основних компоненти системи: програма-агрегатор у вигляді веб-додатку для накопичення та збереження сирих вхідних даних; програма для попередньої обробки, очищення та трансформації даних і, відповідно, збереження у форматі, готовому для роботи з алгоритмами класифікації; компонент, що здійснює побудову множини моделей, визначає найкращу з них та будує гібридну модель на її основі, а також безпосередньо використовується для подальшого прогнозування.

Реалізація стабільно показує чудові результати валідації відносно базової моделі в межах допустимої похибки, а також аналогічні показники для даних, що прогнозуються. Задачі класифікації текстів дають дещо гірші показники за рахунок втрати даних на момент перетворення їх у числове представлення (для текстових категорій схожі втрати відстуні), але загалом не відрізнюються від методів-аналогів. Головною підтвердженою перевагою залишається швидкодія робити в поєднанні зі збереженням допустимої точності моделі.

4 Аналіз рішення

З практичної точки зору недоцільно розглядати лише математичний апарат запропонованого методу, оскільки корисність та необхідність використання алгоритму на реальних вхідних даних вимагає емпіричного підтвердження чи порівняння роботи з уже готовими рішеннями. Оскільки розроблюваний метод є апроксимацією довільно обраної моделі, було проведено порівняння з найкращою моделлю (що надалі була взята за базову) на основі алгоритму Support Vector Machines, а також був проведений аналіз для вибірки датасетів. Всі результати були зіставлені з визначеними заздалегідь нефункціональними вимогами для підтвердження відповідності програмного продукту поставленим критеріям.

4.1 Порівняльний аналіз

Було проведено порівняльний аналіз рішення з існуючими реалізаціями для підтердження ефективності використання даного алгоритму. Обрані такі критерії для побудови порівняльної таблиці:

- відхилення від еталонної величини;
- середнє квадратичне відхилення;
- час роботи на рядок вхідних даних датасету.

Також для найкращої моделі та гібридної моделей були побудовані таблиці помилок (*confusion matrix*) - матриці, що допомагають візуалізувати ефективність алгоритмів. Кожна колонка містить кількість результатів в передбачуваному класі, в той час як кожен рядок містить дійсну кількість елементів у класі (рис. 16).

Результати для моделі *Support Vector Machines* та побудованої гідбридної моделі (табл. 1) дають змогу зрозуміти, обидві моделі відносять еле-

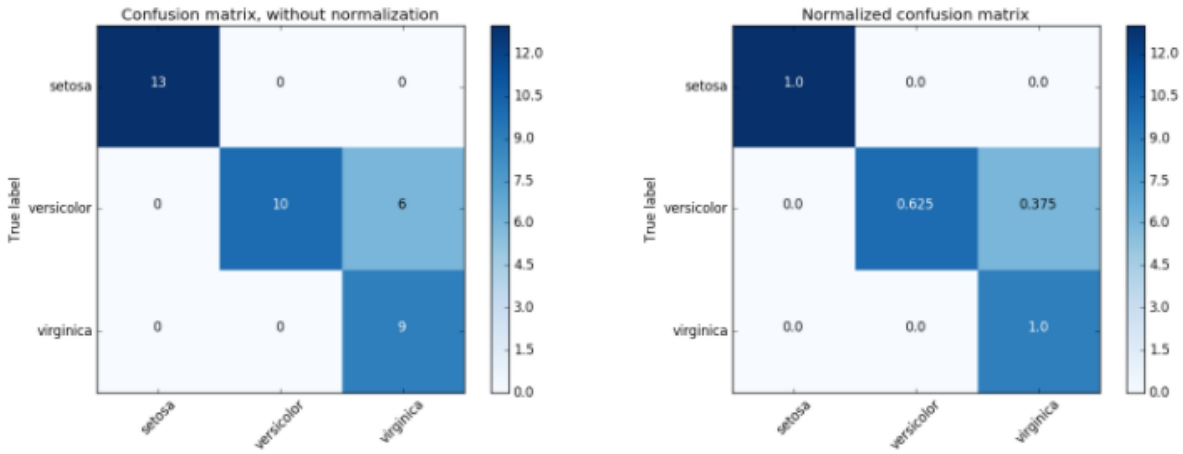

Рис. 16: Приклад матриці помилок для вхідного набору даних Iris

менти до однакових класів. Це означає, що хоч і точність передбачення не є максимально можливою, проте дозволяє показати стабільні консистенті результати для обох моделей, а це, у свою чергу, дозволяє підтверджувати надійність розроблюваного підходу.

	setosa	versicolor	virginica
setosa	7	0	0
versicolor	0	10	2
virginica	0	0	11

Табл. 1: Матриця помилок

4.2 Відповідність поставленим критеріям

Для перевірки відповідності висунутим вимогам було проведено аналіз для кожного окремо висунотого пункту нефункціональних вимог. Для перевірки швидкодії було обрано 5 різних різнорідних вхідних наборів даних (2).

Було обрано різні датасети розміром від декількох кілобайт до десятків мегабайт для перевірки поведінки моделі під час роботи з різними об'ємами даних. Результати показали, що зі збільшенням об'єму даних, час роботи алгоритму росте лінійно, тобто залишається сталим час на обробку одного

№ п/п	Назва	Розмір	Час (мс на рядок)	Точність (validation)
1	creditcard.csv	68.08 MB	0.02	0.9833
2	movie_metadata.csv	579.78 KB	0.01	0.9754
3	mushrooms.csv	365.24 KB	0.01	0.9666
4	77_cancer_proteomes_CPTAC_itraq.csv	5.41 MB	0.01	0.9342
5	menu.csv	29.29 KB	0.007	0.9303

Табл. 2: Швидкодія роботи на різних вхідних даних

рядку вхідних даних. Це означає, що модель теоретично готова приймати необмеженого розміру дані на вхід (зрозуміло, спираючись на обмеження оперативної пам'яті та дискового простору).

Для перевірки точності було обраховано середнє квадратичне відхилення, щоб дізнатися, на скільки в середньому відхиляються індивідуальні значення передбачуваної величини від її середнього значення (3).

№ п/п	Назва алгоритму	Значення відхилення
1	Logistic regression	0.04082
2	Linear Discriminant Analysis	0.03818
3	Gaussian Naive Bayes	0.05335
4	Support Vector Machines	0.02511
5	Decision Tree Classifier	0.04064
6	K-nearest neighbors	0.03333

Табл. 3: Середнє квадратичне віхилення для досліджуваних алгоритмів

Порівняння дає змогу зрозуміти, який з алгоритмів менше за все схильний до видачі "несподіваних" результатів, тобто таких значень, що будуть давати помітні стрибки досліджуваної величини. Навіть за умови досить точного передбачення існують такі варіанти використання, де відхилення буде відігравати набагато важливішу роль. Тобто, інколи важливіше не ви-

ходити за межі допустимих обмежень по всіх даних загалом, ніж отримати більш точні результати для індивідуальних записів. Для даного порівняння алгоритм класу *Support Vector Machines* показав найкращі результати, саме тому гібридна модель для таких початкових даних буде будуватися на основі цієї моделі.

4.3 Доцільність вибору інструментарію

4.4 Отримані результати та шляхи покращення

Отже, в результаті отримана реалізація показала відповідність усім висунутим вимогам та продемонструвала стабільні показники незалежно від виду та об'єму вхідних даних. На поточному етапі такі результати повністю задоволняють як користувачів алгоритму, так і розробників, які бажають покращити алгоритм чи модифікувати його будь-яким чином для забезпечення кращих показників точності чи швидкодії.

Серед напрямків для покращення алгоритму можна виділити такі основні:

- Визначення оптимального алгоритму не шляхом повного перебору існуючих моделей, а за допомогою деякої евристики. Зараз для вхідних даних необхідно побудувати всі моделі, користуючись медотом повного перебору (*brute force*) і виконати їхнє порівняння за деякою обраною метрикою. Лише після цього на основі кращої моделі буде побудована наша гібридна модель. Якщо ж скористатися деяким набором правил, евристикою чи іншими додатковими знаннями в доменній залузі - можна обмежити кількість моделей, що будуть побудовані, таким чином в декілька разів зменшити витрати на час на етапі побудови моделей. Виграш буде значно відчутний на великій кількості моделей за умови, що лише кілька з них дійсно показують

стабільно найкращі результати для схожих вхідних даних. Однією із можливих реалізацій такого прийому може бути додаткова модель-класифікатор, що буде на основі вхідних даних обирати клас алгоритмів (деяке значення T), для яких варто проводити побудову моделей. Іншим варіантом може бути підтримка структури у вигляді словника, що буде зберігати наперед визначені користувачем набори типу "критерій"-"клас алгоритмів" і значно швидше (за лінійний час) буде обирати потрібну множину алгоритмів. Останній підхід є значно швидшим, але вимагає додаткової початкової ініціалізації та втручання людини для підготовки такого словника.

- Запуск побудови кожної моделі в окремому потоці. Процес побудови моделі є процесом, що в першу чергу вимагає процесорний час для виконання (*cpu-bound*), тому з апаратної точки зору прискорити його роботу можливо за рахунок розпаралелювання на декількох ядрах процесора. Найпростішим варіантом є використання багатоядерних процесорів і виконнання алгоритму на окремому ядрі. Сучасні відеокарти з підтримкою технологій *Nvidia CUDA* та *AMD OpenCL* теж можуть бути використані для запуску даних алгоритмів. Порівняння показують, що використання відеокарт для схожих обрахунків може надати приріст у розмірі 90-95х кратного прискорення роботи. Аналогічним чином можна скористатися розподіленими та багатопроцесрними системами, коли алгоритми будуть виконуватися окреми на різних машинах в межах одного кластеру. Головним недоліком таких систем є підвищення порогу входження для розробки, адже це вимагає додаткових знань як для написання коду (*C++, MPI, OpenMPI*), так і розуміння архітектури розподілених систем вцілому. Наприклад, для написання алгоритму для кластеру потрібно розуміти, що кожна модель на окремій ноді кластеру повинна мати доступ до вхідного датасету, а

отже потрібно забезпечити централізований неблокуючий доступ до даних або реалізувати можливість спільної пам'яті (*shared memory*), що теж вимагає додаткових зусиль з точки зору програміста. З переваг варто відмітити однократність даної операції та практично необмежений лінійний ріст ефективності пропорційно до кількості початкових алгоритмів.

- Під час сумісної роботи над проектом виникає необхідність обміну даними між розробниками. Науковці хочуть мати змогу надсилати моделі іншим, а також мати змогу їх зберегти для подальшого використання. Тому ще одним можливим шляхом для вдосконалення може бути можливість серіалізації моделей. Таким чином модель можна буде побудувати і зберегти в бінарному форматі на одному комп'ютері, а потім використати в майбутньому без необхідності повторної її перебудови. Звісно такий підхід працює лише для одного набору вхідних даних, тому область його застосування досить обмежена. Проте, якщо над деякими даними працює команда науковців, саме це дасть змогу швидко обмінюватися результатами чи використовувати напрацювання інших. Найпростішою реалізаціює тут може бути знімок об'єкта у пам'яті, але таким чином втрачається кросплатформенність та машинонезалежність. Вбудовані підходи до серіалізації (наприклад, *pickle*) також будуть страждати від подібних нюансів. Саме тому необхідно буде розробити додатковий алгоритм для серіалізації моделей, що і є головним фактором, який стримує додавання даної можливості до існуючого коду.

Розглянуті напрямки покращення дозволять збільшити швидкодію програми вцілому, а також спростити використання її в межах команди науковців, а тому доцільно розглянути подальшу роботу над проектом саме в одному із запропонованих напрямків.

5 Стартап

Опис ідеї стартап-проекту

numeric literals	integers	in decimal	\FontspecSetCheckBoolFalse\Fontsp
		in octal	\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
		in hexadecimal	\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
	fractionals	in decimal	\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
char literals			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp
string literals			\FontspecSetCheckBoolFalse\Fontsp
			\FontspecSetCheckBoolFalse\Fontsp

Ринок є доволі привабливим для входження: пристойна середня норма рентабельності, що трохи вищ аза середній банківський відсоток на вклади у гривні, а спадання ринку потенційно відкриває його для нестандартних інноваційних рішень, оскільки існує дуже висока необхідність в розробці універсального методу для відновлення зображень.

Обрано альтернативу 2 як таку, що має на увазі довше життя проекту.

В якості цільових груп обрано: 1 та 2.

5.0.1 Тут субсубсекція з висновками

Проведений детальний аналіз ринку та перспектив розвитку проекту дав змогу отримати такі результати:

Зміст ідеї	Напрямки застовування	Вигоди для користувача
Система побудови універсальних прогностичних моделей та метод їх додавання	1. Використання науковими та інженерними спеціалістами з аналізу даних для підвищення їх ефективності та продуктивності роботи загалом	Зручний та зрозумілий інтерфейс дозволить працювати значно ефективніше, тим самим зосереджуючись на прикладних задачах, замість деталей реалізації
2. Узагальнення алгоритмів для роботи з різними типами даних	Універсальність моделі дозволить не перемикати контексти під час роботи з різними типами даних, використовуючи однаковий підхід для вхідної інформації	
3. Отримання кращих результатів передбачень для даних, що змінюються з часом	Допомога під час роботи з величинами, що залежать від часу: курси валют, показники біржі, зміни клімату	

Табл. 4: Опис ідеї стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	1
2	Загальний обсяг продаж, грн/ум. од	914 218 млн грн
3	Динаміка ринку (якісна оцінка)	Спадає
4	Наявність обмежень для входу (вказати характер обмежень)	Висока доля невизначеності, відсутність попереднього досвіду та необхідних статистичних даних
5	Специфічні вимоги до стандартизації та сертифікації	-
6	Середня норма рентабельності в галузі (або по ринку), %	18-20%

Табл. 5: Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Необхідність для інвесторів знайти перспективний метод для вкладень	Люди, які мають фінансову можливість та зацікавленість робоити інвестиції у інноваційні проекти	Люди, які мають фінансову можливість та зацікавленість робити інвестиції у інноваційні проекти мають на меті збільшення свого капіталу, підвищення свого іміджу, а також долучитися до новітніх технологій, щоб бути у тренді	Необхідно розробити методику оцінювання та рекомендації, які б з високою ймовірністю розраховували потенційні необхідні інвестиції та шляхи попередження ключиових ризиків
2	Необхідність команди для побудови цього	Активні люди, які бажають втілити у життя свій проект	Необхідність проаналізувати всі ключові фактори, щоб визначити,	Високоточний метод оцінки відновлення зображень, щоб визначи-

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Попит	Не вдасться розробити унікальний метод, який би можна було застосовувати для будь-яких алгоритмів та адаптувати для роботи з різними типами даних	Розробка максимально універсального методу
2	Конкуренція	Можливість появи конкурентів з дуже схожими функціями, їх вихід на ринок раніше за нас	Доопрацювання якості розроблюваного методу з фокусом на зручність та простоту використання, розробка нових властивостей, яких немає у конкурента. Розгляд можливості об'єднання компаній для подальної спільної роботи.
3	Економічні	Зменшення доходу інвесторів, що призведе до зменшення кількості інвестицій	Моніторинг економічної ситуації у країні, пошук закордонних користувачів та адаптація для світового ринку

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Попит	Унікальність пропонованого функціоналу та додаткових можливостей при умові невисокої конкуренції дозволить захопити велику частку ринку, особливо зацікавивши додатком невеликих інвесторів (бізнес-ангелів) та команди проектів, які не потребують значних інвестицій	Адаптація до ринку, що розширяється, моніторинг новітніх розробок та ризиків, які тільки нещодавно з'явилися
2	Науково-технічні	Поява нових технологій, виникнення нових ринкових умов та факторів, які виявлять значний вплив на розвиток алгоритмів класифікації	Активне використання використання рішення; у випадку, якщо наше рішення буде адним з перших та матиме суттєві відмінності від аналогів, захист інтелектуальної власності розробників, патентування цієї технології та додання її до

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції - чиста конкуренція	Велика кількість методів відновлення зображень, частина з яких є запатентованою інтелектуальною	Звертати увагу на якість та універсальність методу відновлення зображень

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Прямих конкурентів немає, непрямі - різноманітні методи побудови прогностичних моделей	Нові розробки у галузі	Інвестори диктують умови розвитку ринку: ключова умова - проект повинен бути потрібним користувачам та приносити користь	Кількість зацікавлених клієнтів, рівень зацікавленості в такому типі послуг	Поява схожих дешевших або якісніших продуктів-конкурентів
Висновки	Прямих конкурентів немає	- можливості входу в ринок присутні, необхідно вирішити проблему пошуку та адаптації стати-	Успіх нашого проекту залежить від рівня довіри інвесторів та команд проекту до новітнього	Клієнти формують попит на таку послугу	Універсальних методів, які могли б замінити запропонований проект немає

№ п/п	Фактор конкурентноспромо-жності	Обгрунтування (наведення чинників, що роблять фактор для порівнянння конкурентних проектів значущим)
1	Фактор часу	Ідея є частково новою, для перейняття ідеї та втілення її у життя потенційним конкурентам знадобиться час
2	Фактор новизни товару	Початковий успіх продукту очікується через його новизну та інтерес цільової аудиторії до нових інноваційних рішень
3	Фактор якості послуг та надання інформації	Науковці та експерти з обробки даних потребують універсальний метод побудови прогностичних моделей

Табл. 11: Обґрунтування факторів конкурентноспроможності

№ п/п	Фактор конку-рен-тно-спро-мо-жності	Бали 1-20 Рей-тинг товарів-конкурентів у по-рів-нянні з ін-шими мето-дами оціню-вання	2	3	4	5	6
1	Фактор часу	15			+		
2	Фактор нови-зни товару	20		+			
3	Фактор якості послуг та на-дання інфор-мації	17		+			

Табл. 12: Порівняльний аналіз сильних та слабких сторін методу

Сильні сторони: Якість послуг, що надаються Новизна послуг Можливість використання як інвесторами, і командою з розробки	Слабкі сторони: Відсутність статистичних даних та попереднього досвіду в реалізації подібних рішень
Можливості: Створення нової ринкової ніші Потреба у ефективному та компактному методі створення прогностичних моделей Необхідність закладати у бюджет можливі ризики та зміни ринкових умов	Загрози: Різка зміна ринку, поява нових стартапів, економічна криза

Табл. 13: SWOT-аналіз стартап-проекту

- Існує можливість ринкової комерціалізації проекту, на ринку наявний попит на пропонований продукт.
- Ринок відкритий для інновацій, прослідковується позитивна динаміка ринку.
- Рентабельність роботи на ринку вища за прибутковість банківських вкладів, а отже приваблює як інвесторів, так і розробників для роботи над перспективним проектом.
- З огляду на потенційні групи клієнтів існує потенціал та перспектива входу на ринок.
- Істотні бар'єри для входження відсутні.
- В якості варіанту для впровадження для ринкової реалізації проекту доцільно обрати довгострокову роботу та утримання клієнтів, роботу над покращенням розробленого методу з використанням багатовимірного статистичного аналізу.

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	- Ціль: отримання прибутку в короткостроковій перспективі - Конкуренція: цінова та партнерська (пропонуємо свої нові послуги розповсюдження інформації про партнерів - рекламні послуги) - Взаємодія з фірмами: активна боротьба за долю ринку, що належить конкурентам	В короткостроковому плані - велика В довгостроковому плані - значний ризик втратити долю ринку, якщо займатися лише ціновою конкуренцією	8-12 місяців після запуску проекту
2	- Ціль: захоплення частини ринку, підтримання її розміру та поступове нарощення об'ємів - Конкуренція: нецінова (акцент на тому, що пропонуємо інноваційні послуги) - Взаємодія з конкурентами: спів-	Висока ймовірність отримання ресурсів та утримання їх протягом довгого проміжку часу. Більш ймовірний розвиток компанії та постійне покращення продукту	8-12 місяців після запуску проекту - для отримання перших фінансових надходжень від розповсюдження інформації про акції магазинів-партнерів, та їх реклама. Далі фінансові надходже-

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Високозабезпечені люди, які зацікавлені у пошуку перспективних проектів для інвестування	Споживачі слідкують за найновітнішими технологіями, бажають бути в тренді та готові сприйняти новий продукт	Потенційно високий, інвестори хочуть бути впевненими у доцільності своїх інвестицій та подальшому отриманні прибутку	Практично відсутня	При наявності достойної та доручної реклами - досить просто
2	Ініціативні люди та науковці, які мають хорошу ідею в схожій сфері та хочуть втілити її у життя	Споживачі готові сприйняти продукт, так як зацікавлені у глибинному аналізі ситуації	Високий попит	Практично відсутня	При наявності достойної та доречної реклами - досить просто

Табл. 15: Вибір цільових груп потенційних споживачів

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Захоплення, підтримання та захист частки ринку	Стратегія концентрованого маркетингу	- Новизна послуг - Доступність продукту - Простота в користуванні продуктом - Додаткові зручні аспекти, які враховуються під час розрахунку ефективності та інвестиційної привабливості побудови прогностичних моделей, що вигідно виділяють наш продукт серед конкурентів	Стратегія диференціації

Табл. 16: Визначення базової стратегії розвитку

№ п/п	Чи є проект "першопрохідцем" на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів	Чи буде компанія копіювати основні характеристики товару конкурента і які?	Стратегія конкурентної поведінки
1	Частково	Нові споживачі, частково забиратиме споживачів конкурентів	Частково. Новий метод оцінювання ефективності побудови прогностичних моделей буде агрегувати декілька методик аналізу, що дозволить оцінювати проекти більш точно з використанням більшої кількості факторів, що впливають на проект	Стратегія лідера

Табл. 17: Визначення базової стратегії конкурентної поведінки

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Універсальність методу оцінювання з точки зору інвестора	Стратегія диференціації	Врахування всіх аспектів оцінювання проекту з точки зору інвестиційної привабливості	- Ваші гроші ефективно працюють у інноваційному прогресивному проекті
2	Універсальність методу оцінювання з точки зору команди проекту	Стратегія диференціації	Врахування всіх аспектів оцінювання проекту з точки зору інвестиційної привабливості та життєздатност іпроекту, доцільність реалізовувати інноваційний проект	- Реальна можливість втілити у життя ідею завдяки глибинному аналізу ключових аспектів та пошуку інвесторів
3	Необхідність враховувати	Стратегія диференціації	Врахування ключових	- Детальний облік ризиків

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Універсальний метод оцінювання ефективності та інвестиційної привабливості проекту, який буде корисний як для інвесторів, так і для команди проекту	Методика оцінювання дозволить уникнути передчасного закриття проекту та перевитрат люджету завдяки високоточній оцінці на ранніх етапах проекту	Оцінювання проекту як з точки зори витрат та ефективності їх використання командою стартапу, так і з урахуванням потенційних ризиків, прихованих стратегічних переваг на недоліків.

Табл. 19: Визначення ключових переваг концепції потенційного товару

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	Безкоштовно	Безкоштовно	Більше 10000 грн/місяць	-

Табл. 20: Визначення меж встановлення ціни

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Купують право на використання методики	Зберігання, сортування, встановлення контакту інформування	Одноріневий	Залучена

Табл. 21: Формування системи збуту

- Конкуренція практично відсутня, а конкурентноспроможність самого продукту достатньо висока. …

Враховуючи описані вище ключові моменти, можна зробити висновок, що подальша імплементація даного проекту є доцільною та обґрунтованою.

№ п/п	Специфіка поведінки цільових клієнтів	Канали комуніка-цій, якими користу-ються цільові клієнти	Ключові позиції, обрані для позиціону-вання	Завдання реклам-ного повідомле-ння	Концепція реклам-ного звернення
1	Позитивне відно-шення до інно-вацій та швидкий розвиток технологій призводять до появи великої кількості нових методів, росту кількості даних і побудова прогно-стичних моделей стає більш актуаль-ною	Соціальні мережі (facebook, twitter), тематичні ресурси	Універ-сальний метод побудови прогно-стичних моделей	Впевнити клієнта у тому, що метод є унікаль-ним та універ-сальним	Повідом-лення у соціальних мережах, статті на веб-ресурсах, короткі демон-страційні ролики

6 Висновки

Тут багато незрозумілих слів, ще більше води, ніж у всіх інших частинах диплому

Результати показують, що стабільно показують чудові результати для завдань класифікації текстів, суттєво перевищуючи показники інших методів.

Описаний підхід до створення моделі дозволяє отримати високі показники та обмежитися мінімально необхідними ресурсами для запуску на вхідних даних теоретично необмежено розміру з мінімальними втратами точності. Створення такої моделі вимагає додаткового кроку побудови з уже існуючої моделі, але такі затрати є цілком виправданими. Простота реалізації даного підходу дає змогу покращити та уніфікувати процес побудови прогностичних моделей і їх наступне використання як звичайним науковцям з обробки даних, так і комплексним системам, що містять архітектуру різного рівня складності.

Whatever

Text mining. Классификация текста. Пример классификации документов с использованием программных алгоритмов statistica. URL \FontspecSetCheckBoolFalsehttp://statosphere.ru/blog/135-text-mining1.html.

Hinrich Schütze Christopher D. Manning, Prabhakar Raghavan. *An Introduction to Information Retrieval.* Cambridge University Press., 2009.

John Doe. *The Book without Title*. Dummy Publisher, 2100.

Robert W. Fairlie. *Kauffman Index of Entrepreneurial Activity*. Kansas City: Ewing Marion Kauffman Foundation, 2014.

Brad Feld. *Startup communities: Building an entrepreneurial ecosystem in your city*. Hoboken, NJ: John Wiley & Sons, 2012.

Steven Finlay. *Predictive Analytics, Data Mining and Big Data. Myths, Misconceptions and Methods (1st ed.).* Basingstoke: Palgrave Macmillan., 2014.

Micheline Kamber Han Jiawei and Jian Pei. *Data mining: concepts and techniques.* Morgan Kaufmann., 2006.

Helland I.S. *Steps Towards a Unified Basis for Scientific Models and Methods*. World Scientific., 2010.

Stapleton J.H. *Models for Probability and Statistical Inference.* Wiley-Interscience., 2007.

Young-Hoon Kwak. *A brief history of Project Management*. Greenwood Publishing Group, 2005.

Arthur Samuel. *Some Studies in Machine Learning Using the Game of Checkers.* IBM Journal of Research and Development, Volume 44, 1959.

F. Sebastiani. *Machine Learning in Automated Text Categorization*.

Dane Stangler. *The Economic Future just Happened*. Kansas City: Ewing Marion Kauffman Foundation., 2009.

Martin Stevens. *Project Management Pathways*. Association for Project Management. APM Publishing Limited, 2002.

Sholom M. Weiss and Nitin Indurkhya. *Predictive Data Mining.* Morgan Kaufmann., 1998.

Rand R. Wilcox. *Fundamentals of Modern Statistical Methods.* New York: Springer, 2010.

Шмидт С. Бирман Г. *Капиталовложения. Экономический анализ инвестиционных проектов.* М.: ЮНИТИ-ДАНА, 2003.

Лапыгин Ю. Н. *Управление проектами: от планирования до оценки эффективности*. М.: Омега-Л, 2008.