МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки
Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження методів прогнозування результатів спортивних матчів. Дерева рішень. Градієнтний бустинг.

Виконала:
студентка 2 курсу групи ІПЗм-20-1
Смикова А.Ю.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення

Тип програми Освітньо-наукова

Керівник доц. Чуприна А. С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ З.В.Дудар

2022 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Програмної інженерії
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-наукова програма
(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____20 ____ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента *Смикової Анни Юріївни*
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів прогнозування результатів спортивних матчів. Дерева рішень. Градієнтний бустинг»

затверджена наказом університету від «24» березня 2022 р. № 412 Ст

2. Термін подання роботи до екзаменаційної комісії «__» _____ 2022 р.

3. Вихідні дані до роботи *класифікаційні методи прогнозування, критерії їх оцінки, дерево рішень, випадковий ліс, градієнтний бустинг, датасет, Python, NumPy, SciKit Learn.*

4. Перелік питань, що потрібно опрацювати в роботі *вступ, аналіз предметної області, аналіз класифікаційних методів прогнозування на основі дерев, критерії їх оцінки, постановка задачі, проведення експериментів та аналіз їх результатів, формування подальших рекомендації.*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної області	25.02.2022	виконано
2	Постановка задачі	10.03.2022	виконано
3	Проведення дослідження	01.04.2022	виконано
4	Підготовка пояснювальної записки	03.05.2022	виконано
5	Підготовка презентації та доповіді	06.05.2022	виконано
6	Попередній захист	10.05.2022	виконано
7	Перевірка на академічний плагіат	10.05.2022	виконано
8	Нормоконтроль	12.05.2022	виконано
9	Рецензування	13.05.2022	виконано
10	Знесення диплома в електронний архів	15.05.2022	виконано
11	Допуск до захисту у зав. кафедри	15.05.2022	виконано

Дата видачі завдання 17.01.2022 р.

Студент _____
(підпис)

Керівник роботи_____ доцент Чуприна А.С.
(підпис)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 70 стор., 9 рис., 1 табл., 1 лістниг, 20 джерел, 4 додатки.

МАТЧ, СПОРТ, СПОРТСМЕНИ, ТУРНІР, ДЕРЕВА РІШЕНЬ, ВИПАДКОВІ ЛІСИ, ГРАДІЄНТНИЙ БУСТИНГ, МАШИННЕ НАВЧАННЯ, НАУКА ПРО ДАНІ.

Об’єктом дослідження є методи прогнозування на основі дерев рішень.

Метою дослідження є аналіз різних за складністю та структурою класифікаційних методів прогнозування для виявлення найбільш доцільного для передбачення результатів спортивних матчів.

Методи розробки базуються на основі архітектурних підходів науки про дані, мові програмування Python та її бібліотек.

У результаті роботи було проведено ретельний аналіз існуючих методів прогнозування та виявлено найбільш влучний для поставленої задачі передбачення результату спортивного матчу, результати були продемонстровані на твореному прототипі.

MATCH, SPORTS, ATHLETES, TOURNAMENT, DECISION TREES, RANDOM FORESTS, GRADIENT BUSTING, MACHINE LEARNING, DATA SCIENCE.

The object of research is forecasting methods based on decision trees.

The aim of the study is to analyze different in complexity and structure of classification methods of forecasting to identify the most appropriate for predicting the results of sports matches.

Development methods are based on architectural approaches to data science, the Python programming language and its libraries.

As a result, a thorough analysis of existing forecasting methods was conducted and the most accurate for the task of predicting the outcome of a sports match was found, the results were demonstrated on the created prototype.

Я, Смикова Анна Юріївна, студентка групи ІПЗм-20-1, здобувач вищої освіти на другому (магістерському) рівні, кафедра Програмної інженерії, заявляю: моя кваліфікаційна робота на тему «Дослідження методів прогнозування результатів спортивних матчів. Дерева рішень. Градієнтний бустинг», що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлена з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AI	Artificial Intelligence, штучний інтелект
DT	Decision Tree, дерево рішень
GBM	Gradient Boosted Machine, градієнтний бустинг
MAE	Mean Absolute Error, середня абсолютна помилка
ML	Machine Learning, машинне навчання
MSE	Mean Squared Error, середня квадратична помилка
RF	Random Forest, випадковий ліс

ВСТУП

У наші дні існує культ здорового тіла, здорового образу життя та загалом здоров’я. Більшість людей, які мають середній рівень життя намагаються наділяти увагу підтримці свого фізичного та внутрішнього стану. Найчастіше для цих цілей люди прибігають до зайнять спортом.

На базовому рівні спорт допомагає людині підтримувати свою фізичну форму, відволікатися від повсякденних клопотів та покращувати самопочуття. В середньому людина, що, насамперед, піклується про себе, витрачає близько 2-3 днів на тиждень на часові тренування. Це може бути будь-що: зайняття фітнесом, похід до тренажерного залу, гра в настільний теніс або біг з ранку.

Але є й інша сфера спорту – професійний спорт. Це зовсім інші мотивації, вкладення часу та грошей. Професійним спортсменам потрібно прикласти немало зусиль, щоб досягти хоча б якихось успіхів. Насамперед до професійного спорту відводять ще зовсім малих дітей. З років 5-8 дитина повинна робити перші кроки на шляху до свого становлення як спортсмена. Тренування професійного спортсмена змалечку займають близько 6 годин щонайменше.

Всі ці навантаження, на які згодні йти професійні гравці, обумовлені тим, що наразі, спортсмени – це свого роду суперзірки, яких люблять та ставлять у приклад іншим. Вони гарно заробляють та становляться обличчями відомих брендів. Лише стрімкість та цілеспрямованість дозволяє досягти великих вершин у цій нелегкій сфері діяльності.

Проте спорт розвивається доволі стрімко, а з ним і програмні продукти, що полегшують життя спортсменам та дозволяють виміряти більш точні результати у різних змаганнях. Ще 50 років тому ніхто не міг подумати, що можна застосовувати 3D моделі для відображення траєкторії м’яча у матчах з футболу чи використовувати доповнену реальність для передбачення руху шайби в хокеї.

Наразі програмні продукт все більше втілюють цілі спортивних команд та турнірів, щоб полегшити збір статистичних та отримати переваги ведення менеджменту. Проте програмні продукти, інтегровані у спортивний процес,

можуть привнести набагато більше. Наприклад, існує ряд веб-додатків, які дозволяють вести статистику футбольної гри з детальним розбором кожного удару гравця та швидкістю переміщення його по полю. Я вважаю, що доволі цікавою темою для дослідження є прогнозування результатів спортивних суперечок у різних професійних видах спорту.

В прогнозуванні визначаються або прогнозуються відсутні або недоступні дані для нового спостереження на основі попередніх даних, які ми маємо, і на основі майбутніх припущень. У прогнозуванні вихід є безперервним значенням.

Метою кваліфікаційної роботи є визначення методу прогнозування, що є найбільш влучним для отримання передбачення результату матчу на основі проведеного детального аналізу та дослідження існуючих на ринку рішень.

Тому для кваліфікаційної роботи було обрано тему, що дозволить розглянути та дослідити програмні методи передбачення результатів спортивних матчів.

Об’єктом дослідження служитимуть саме методи прогнозування. Найбільш вагомим внеском у прогнозування є підходи, які імплементують функцію класифікації. Предметом дослідження виступатимуть класифікаційні методи прогнозування основані на деревах.

Класифікація — це процес пошуку хорошої моделі, яка описує класи даних або концепції, а мета класифікації — передбачити клас об’єктів, мітка класу яких невідома. Класифікація – це категоризація нових даних, що надходять, на основі поточних або минулих припущень, які були зроблені, і даних, які вже відомі.

Дерево – це тип структури даних, в якій кожен елемент приєднаний до одного або кількох елементів безпосередньо під ним. Дерева часто називають перевернутими, тому що вони зазвичай малюються з коренем у верхній частині.

Необхідно визначити найбільш влучні методи, які б вдало вирішували поставлену задачу та мали б успіх в отриманих результатах Ці методи мають бути ретельно проаналізовані на предмет точності. Методи мають бути перевірені на відкритому датасеті, який має містити необхідні поля для визначення переможця або програвшого у певному матчі. Датасет також повинен відповідати певним вимогам, зоб навчання на ньому призвело до отримання методу з великим відсотком точності.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Загальний аналіз галузі

Спорт має величезний вплив на повсякденне життя та здоров’я людини. Цей вид активної діяльності не тільки дає вам цікаву рутину, але й здорове тіло. Заняття фізичними вправами покращують роботу серця, зменшують ризики діабету, контролюють рівень цукру в крові та знижують напругу та рівень стресу. Це також привносить у життя людини позитивну енергію, дисципліну та інші похвальні якості. Заняття спортом зміцнює тіло, а також покращує м’язову пам’ять і координацію.

З проблемою ожиріння стикаються мільйони людей по всьому світу. Ожиріння збільшує ризики гіпертонії та серцево-судинних захворювань. Один з найкращих способів позбутися від ожиріння – це заняття спортом. Фізична активність допомагає контролювати свою вагу. Більшість видів спорту – це інтенсивні фізичні навантаження, які швидко та ефективно спалюють зайві калорії. Найкращий вибір у цій ситуації – це спортивні ігри, бо люди зазвичай продовжують грати навіть після втоми, тому що це подобається їм.

Гіпертонія або високий кров’яний тиск є серйозною небезпекою для здоров’я людей з усіх кінців світу. Гіпертонія може стати причиною інсульту або інших захворювань. Регулярна фізична активність і фізичні вправи допомагають підтримувати артеріальний тиск в нормі. Спорт надає усі необхідні розтяжки, біг і вправи. Таким чином, заняття спортом може стати відмінним способом боротьби з високим кров’яним тиском. Більшість експертів і лікарів рекомендують людям, які страждають на гіпертонію, регулярно займатися спортом, адже встановлено, що люди, які регулярно займаються спортом, підтримують нормальний артеріальний тиск у порівнянні з тими, хто цим не займається.

Також заняття спортом допомагають контролювати рівень холестерину. За результатами багатьох видів досліджень було доведено, що у людей з високою фізичною активністю рівень холестерину нижче, ніж у тих, хто веде малорухливий спосіб життя. У кращих спортсменів, таких як Кріштіану Рональду та інших, рівень

холестерину вражаюче низький навіть після тридцяти років.

Регулярні фізичні вправи також зміцнюють імунну систему. Тіло стає несприйнятливим до багатьох захворювань. Вправи значно збільшують швидкість потоку лейкоцитів. Коли людина потіє під час занять спортом, з її тіла виводяться токсини. Підвищення температури тіла також знижує ймовірність розвитку бактерій.

Спорт – найкращий спосіб правильно тренувати м'язи. У спортивні ігри весело грати, і це не клопіт. У той же час вони дають сильні та підтягнуті м’язи. Це можливо лише в тому випадку, якщо продовжувати регулярно займатися такими активними видами спорту, як футбол, теніс та баскетбол. Займаючись спортом, людина тонізує свої м’язи і тренує їх. Це відомо як нервово-м’язове програмування. Під час гри м’язи стають все сильнішими і сильнішими. Займаючись спортом, людина одночасно набирає м’язову масу і спалює жир. Статура найкращих спортсменів є джерелом натхнення для всіх нас.

Заняття спортом зміцнюють не тільки м’язи, а й кістки. Під час занять спортом навантажується весь скелет за допомогою великих потужних і силових рухів. Це, в свою чергу, збільшує щільність кісткової тканини, що призводить до зміцнення кісток. Наприклад, на відміну від звичайної ходьби, біг під час гри створює додаткове навантаження на кістки ніг та спини. Щоб витримати це підвищене навантаження, кістки адаптуються і стають більш щільними. Якщо продовжувати займатися спортом, кістки стають міцнішими і щільнішими через безперервний стрес. Коли люди старіють, щільність кісток продовжує зменшуватися. Заняття спортом можуть бути найпростішим способом підтримувати хорошу щільність кісток і залишатися сильними з віком.

Якщо часто займатися спортом, покращується самопочуття та підвищується самооцінка. Заняття командними видами спорту також покращують здібності до створення стратегії. Займаючись спортом, людина вчиться швидко та інстинктивно приймати рішення. Ця здатність швидкого прийняття рішень дуже корисна в повсякденному житті. Спорт також навчає зберігати спокій і мислити холодно. Кожна третя людина на землі займається спортом у тому чи іншому вигляді, саме тому ця предметна область є такою актуальною.

1.2 Класифікація професійних видів спорту

Основна відмінність між аматорськими і професійними спортсменами полягає у нагородах, які отримує кожна група за спортивні виступи. Загалом спортсменам-любителям не платять за їх спортивні виступи. Професійні спортсмени, навпаки, зазвичай отримують річну заробітну плату плюс заохочення, пов'язані з індивідуальними та командними результатами. Також професійні спортсмени отримують неабияку вигоду зі складання маркетингових кампаній для відомих брендів[1]. Наприклад, більша частка річного заробітку відомого тенісиста Роджера Федерера припадає саме на рекламні послуги, які він надає.

Існує величезний список із сотень видів спорту з усього світу, перерахованих в алфавітному порядку. Маючи такий громіздкий список, має сенс розділити їх на менші групи. Існує безліч способів класифікувати ці види спорту – вони можуть бути групами на основі того, де, коли і як в них грають. Багато з них є варіаціями подібних видів спорту, мають спільну історію, використовують подібне обладнання або схожий ігровий процес, і ці характеристики можна використовувати для їх класифікації.

Наприклад, Глобальна асоціація міжнародних спортивних федерацій (GAISF) використовує п'ять категорій для видів спорту своїх членських федерацій. Як і будь-які такі групи, вид спорту може не вписуватися ні в одну категорію, в той час як інші потрапляють до кількох:

– переважно фізичні (наприклад, регбі, легка атлетика);

– переважно розумові (наприклад, шахи, го);

– в основному моторизовані (наприклад, перегони автомобілів, катання на моторних човнах);

– переважно координаційні (наприклад, більярд);

– переважно з залученням тварин (наприклад, кінний спорт).

Класифікувати види спорту можна також за кількістю людей, приймаючих у участь в один момент часу в одній грі, матчі, змаганні. Або просто кажучи кількість людей, що представляють одну сторону. Групи видів спорту за кількістю

учасників:

– індивідуальні,
– партнерські або парні,
– командні,
– екстремальні.

Індивідуальними називають такі види спорту, в яких учасником є одна людина, вона представляє себе та виконує усі вправи самостійно. Сюди відносять, наприклад, біатлон, бокс, гольф.

У парні видах спорту завжди є протистояння двох сторін. Тобто дві людини грають одна проти одної. Парними видами спорту також називають ігри, в яких 2 людини – пара – грають проти іншої пари. До таких видів спорту відносяться шахи, теніс, бадмінтон.

Командні види спорту представлені у вигляди гри, в якій одна команда повинна виграти іншу. В команді може бути від трьох до 10-20 людей. Найпопулярнішими командними видами спорту є футбол, баскетбол та бейсбол.

Екстремальні види спорту виокремлені від інших, тому що передбачається, що спортсмени знаходяться у небезпеці та ризикують своїм здоров'ям та життям. До екстремальних видів спорту можна віднести каякінг, дайвінг, бобслей.

Командні та парні види спорту можна розділити на чотири категорії:

– ігри про вторгнення,
– ігри з або через сітку,
– ігри з полюванням та вибиванням,
– цільові ігри.

Гра вторгнення – це будь-яка гра, в якій команда має атакувати зону іншої команди та набрати очко. Наприклад, футбол – це гра вторгнення, коли одна команда повинна утримувати м'яч, відвести м'яч у половину поля іншої команд, а потім спробувати забити гол у сітку суперника (вторгнутися в зону іншої команди).

Гра з сіткою – це коли сітка бере участь у цьому виді спорту та тримає гравців окремо. Наприклад, бадмінтон – це гра через сітку, оскільки в середині майданчика є сітка, і два гравці постійно залишаються навпроти сітки.

Гра з полюванням та вибиванням – це коли гра включає в себе групу

польових гравців і одну людину, яка одночасно вдаряє об’єкт (наприклад, м’яч) і проходить певний курс. Наприклад, у крикеті спортсмен б'є по м'ячу для крикету і повинен перебігти від однієї бази до іншої, щоб забити. Є також польові гравці з протилежної команди навколо поля для крикету.

Цільова гра – це коли гравець повинен прицілитися і спроектувати об’єкт у визначену область. Наприклад, гольф є цільовою грою, оскільки гравець повинен вдарити м’яч для гольфу в задану зону, щоб ввести м’яч у встановлену лунку.

Проаналізувавши види спорту та категорії, я дійшла висновку, що аналізувати предмет дослідження слід на парних видах спорту. Адже саме ці спортивні ігри мають чітку логіку протистояння двох сторін, які найкраще буде оцінювати у рамках класифікаційних методів передбачення. А саме існуватиме лише два класи – переможець та програвший.

1.3 Аналіз існуючих систем

Надалі слід проаналізувати існуючі на ринку реалізовані системи, які включають статистику, трекінг та прогнозування результатів спортивних ігор. Наразі, більшість систем, представлених у світі, що включають передбачення результатів спортивних матчів націлені на футбол, бо це є основною низкою гравців у ставки та букмекерських контор. Що зазвичай націлені на оцінку ставок саме на цей вид спорту. Проте одиничні системи все ж таки охоплюють більше, ніж тільки футбол.

Доволі цікавим за описом є Bespoke sports prediction software система від Gammastack[2] (рис. 2.1). Розробники цієї системи затверджують, що їх додаток має дуже вагомі можливості в передбаченні результатів матчів у більш ніж 50 видах спорту. Ця компанія використовує методи машинного навчання, штучного інтелекту, статистику та блокчейн технології в реалізації своїх програмних рішень для будь-яких задач.

Я вважаю, що їхній продукт заснований на принципах машинного, а саме на

методах нейронних мереж. Адже найбільш популярним рішенням даної задачі є саме застосування нейронних мереж, які дозволяють отримувати результати на основі вхідних даних подібні до результатів, які видає наш мозок. Адже програмується нейронна мережа подібно до нашого мозку, багатьма шарами.

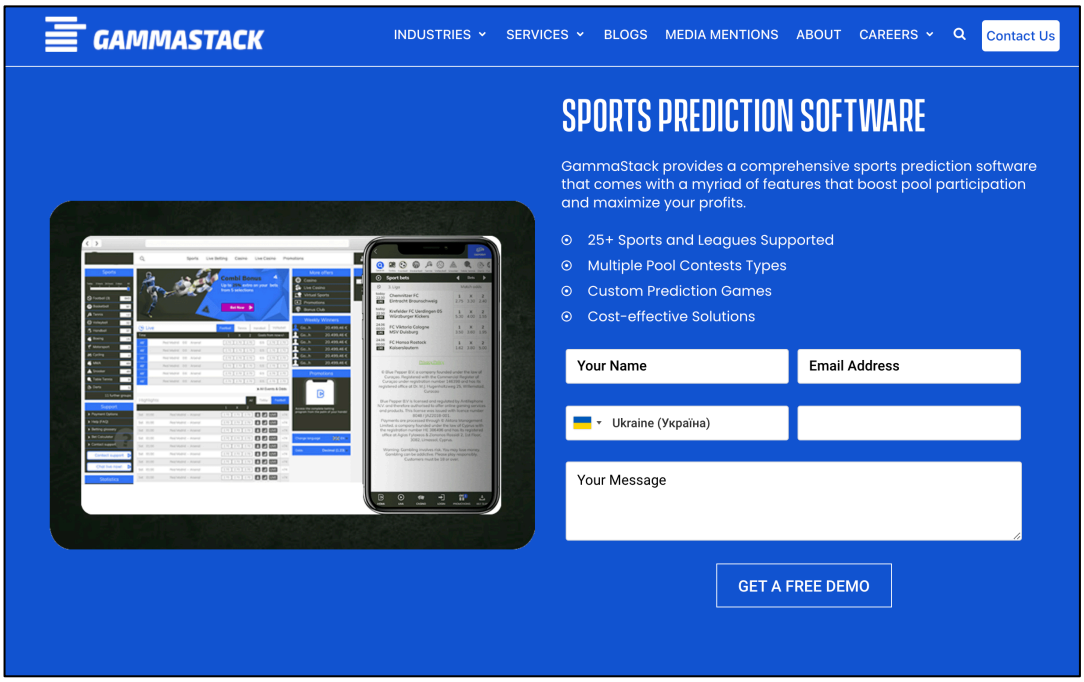

Рисунок 1.1 – Демо Sports prediction software від Gammastack

Основними перевагами цієї системи, як вони затверджують, є:

– охоплення різноманітних ліг – це програмне забезпечення для спортивних прогнозів охоплює понад 25 ліг і видів спорту, які дозволяють задовольнити потреби гравців з різними інтересами;

– результати в реальному часі – результати в прямому ефірі підживлюють хвилювання та зацікавленість;

– статистика – можливість отримати всю інформацію про матчі та результати гравців;

– дзеркальні передбачення – цей інструмент дозволяє гравцям попередніх матчів автоматично копіювати прогнози в майбутніх матчах;

– інструменти планування дозволяє краще керувати всіма своїми турнірами, лігами та змаганнями та ефективно планувати заздалегідь за допомогою інструментів для планування;

– списки матчів дозволяє клієнтам перевірити повний список матчів за улюбленими видами спорту;

– історія гравця допомагає користувачеві відстежувати свої дані, надаючи їм повну інформацію про попередні конкурси, в яких вони брали участь;

– інструменти керування матчами – адміністратори отримують повний контроль над майбутніми матчами, подіями та змаганнями.

Ще одним прикладом аналізу та трекінгу спортивних матчів є додаток Nacsport[3] (рис. 2.2), який фіксує преміщення гравців за допомогою відеозйомки. Цуй додаток представлений для десктопів, смартфонів та у вебі.

Рисунок 1.2 – Демо застосунку Nacsport

Перевагами цього застосунку заявлений наступний функціонал:

– аналіз без обмежень – можливість створювати безлімітну кількість шамблонів та використовувати необмежену кількість камер для відеозйомки;

– швидкість обробки – інструменти, розроблені для неперевершеного збору даних за рекордно короткий термін;

– трекінг кожного гравця – можливість створення теплових карт та карт зон

дозволяє точно знати, хто де знаходився у певний момент часу;

– спільний аналіз у реальному часі обіцяє командну роботу на полі та поза ним, адже надає можливість отримувати дані в реальному часі від інших аналітиків, об’єднувати їх і надсилати усім членам команди;

– покриття всіх кутів – можливість додати другу камеру для аналізу в реальному часі та до чотирьох різних ракурсів після матчу для перегляду більшої картини;

– візуалізація масових даних – інформаційні панелі, які дозволяють візуалізувати дані з кількох збігів, як глобальних, так і відфільтрованих.

Переглянувши ці та інші приклади реалізованих програмних систем з прогнозування та аналітики спортивних матчів, можна зробити висновки, що аналогічним системам не вистачає багатьох функціональних методів[4].

2 ПОСТАНОВКА ЗАДАЧІ

В даній кваліфікаційній роботі дослідження методів прогнозування результатів спортивних матчів буде проводитися на відкритому наборі даних матчів з тенісу за останні 50 років. Адже саме на прикладі матчів з тенісу можна наочно прогледіти залежність результату гри від попередніх матчів.

Для проведення експериментального дослідження знадобиться датасет з ключовими полями, на основі яких буде побудована математична модель. За допомогою цієї моделі буде проведено навчання трьома методами.

Метою роботи є виявлення найбільш влучного методу для вирішення поставленої задачі. А для цього необхідно визначити метрики та критерії оцінювання результатів, отриманих емпіричним шляхом на прикладі описаних вище методів. Саме тому, критерії оцінювання є дуже вагомою частиною дослідження.

Демонстрація отриманих результатів буде проводитися на прототипі або MVP системи аналізу спортивних матчів. Проаналізувавши системи, представлені на ринку, можна виявити основні недоліки, які потрібно зменшити в розроблюваному прототипі:

- відсутність персонального трекінгу,
- відсутність моніторингу травмувань,
- відсутність можливості прогнозування результату,
- неможливість отримання персонального плану.

Головною задачею кваліфікаційної роботи є аналіз існуючих класифікаційних методів прогнозування, що підходять для передбачення результатів матчів та визначення найбільш підходящого методу на основі проведених експериментів. А саме:

- реалізувати метод дерева рішень;
- реалізувати метод випадкового лісу;
- реалізувати метод градієнтного бустингу;
- провести навчання за обраним датасетом методом дерева рішень;

- провести навчання за обраним датасетом методом випадкового лісу;
- провести навчання за обраним датасетом методом градієнтого бустингу;
- проаналізувати отримані в експериментах результати;
- визначити кращий метод для поставленої задачі;
- зробити висновки та дати подальші рекомендації щодо використання;

Система має бути гнучкою та легко масштабованою, щоб завжди була можливість змінити вхідні параметри для навчання. Наприклад, якщо на прогнозування подальших результатів будуть впливати не тільки травмування та попередні досягнення.

Виконання поставленої задачі має виконуватись поетапно з налагодженим процесом, адже саме від налагодження процесу залежить результат програмного забезпечення. Процес реалізації буде проходити за технологією SCRUM, щоб покращити продуктивність розробки.

Обрана модель також повинна максимально чітко виявляти помилки в прогнозуванні та навчатися на основі цих помилок.

Для цього необхідно проаналізувати існуючі алгоритми машинного навчання та обрати найбільш влучний, щоб досягти поставленої мети.

3 АНАЛІЗ ІСНУЮЧИХ КЛАСИФІКАЦІЙНИХ МЕТОДІВ

3.1 Опис існуючих алгоритмів

Для заданої умови нам потрібно використати один метод машинного навчання[5], а який саме, слід встановити за аналізом існуючих алгоритмів. Для порівняння я вирішила взяти наступні методи.

Дерева рішень (Decision Tree), Випадкові ліси (Random Forests) та Бустинг (Gradient Boosting Machine) входять до 16 найкращих інструментів науки про дані та машинного навчання, які використовуються науковцями з даних[6]. Ці три методи подібні, проте все ж таки мають значні відмінності:

- дерево рішень – це проста діаграма прийняття рішень;
- випадкові ліси – це велика кількість дерев, об’єднаних (з використанням методу середніх чи «правил більшості») в кінці процесу[7];
- градієнтний бустинг також поєднує дерева рішень, але починає процес комбінування на початку, а не в кінці.

3.2 Дерево рішень

Дерева рішень – це серія послідовних кроків, призначених для відповіді на запитання та надання ймовірностей, витрат чи інших наслідків прийняття конкретного рішення (рис. 3.1).

Вони прості для розуміння, забезпечують чітке зображення, щоб керувати процесом прийняття рішень. Однак ця простота має кілька серйозних недоліків, включаючи перенавчання (overfitting), помилку зміщення(error due to bias) та помилку через дисперсію (error due to variance).

Перенавчання відбувається з багатьох причин, включаючи наявність шуму та відсутність репрезентативних екземплярів. Також перенавчання можливе на

одному великому (глибокому) дереві.

Помилка зміщення виникає, коли встановлюється занадто багато обмежень на цільові функції. Наприклад, обмеження результату обмежувальною функцією (наприклад, лінійним рівнянням) або простим двійковим алгоритмом (наприклад, правильний/неправдивий вибір у наведеному вище дереві) часто призведе до зміщення.

Помилка дисперсії відноситься до того, наскільки результат зміниться на основі змін навчального датасету. Дерева рішень мають високу дисперсію, що означає, що крихітні зміни в даних датасету можуть спричинити великі зміни в кінцевому результаті.

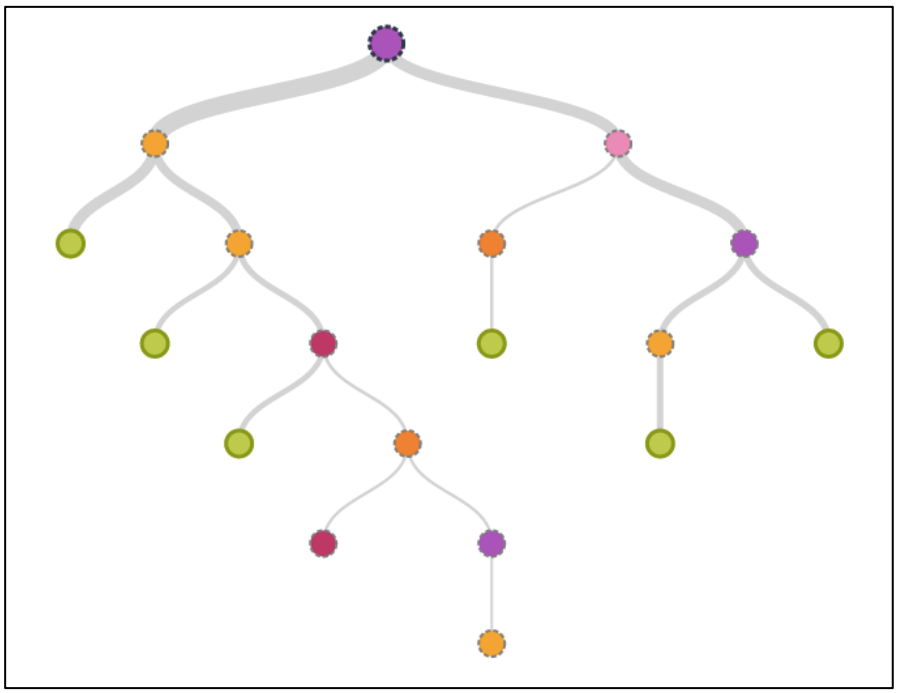

Рисунок 3.1 – Схематичний рисунок дерева рішень

Тож порівняємо детальніше випадковий ліс та дерево рішень. Як зазначалося вище, дерева рішень пов'язані з проблемами. Дерево, створене з 99 точок даних, може значно відрізнятися від дерева, створеного лише з однією іншою точкою даних. Якби існував спосіб створити дуже велику кількість дерев, усереднюючи їх рішення, то, ймовірно, отримана відповідь буде дуже близькою до істинної відповіді.

3.3 Випадковий ліс

Випадковий ліс – набір дерев рішень з єдиним агрегованим результатом. Випадкові ліси зазвичай вважаються найточнішим алгоритмом навчання (рис. 2.2).

Випадкові ліси зменшують дисперсію в деревах рішень за допомогою:

- використання різних прикладів для навчання,
- визначення випадкових підмножин ознак,
- збірка та комбінування невеликих (shallow) дерев.

Одне дерево рішень є слабким предиктором, але його побудова відносно швидка. Більше дерев дає більш надійну модель і запобігає перенавчанню. Однак, чим більше дерев, тим повільніше процес. Кожне дерево в лісі має бути створено, оброблено та проаналізовано. Крім того, чим більше є варіацій та можливостей, тим повільніший процес, який іноді може займати години або навіть дні. Зменшення набору функцій може значно прискорити процес.

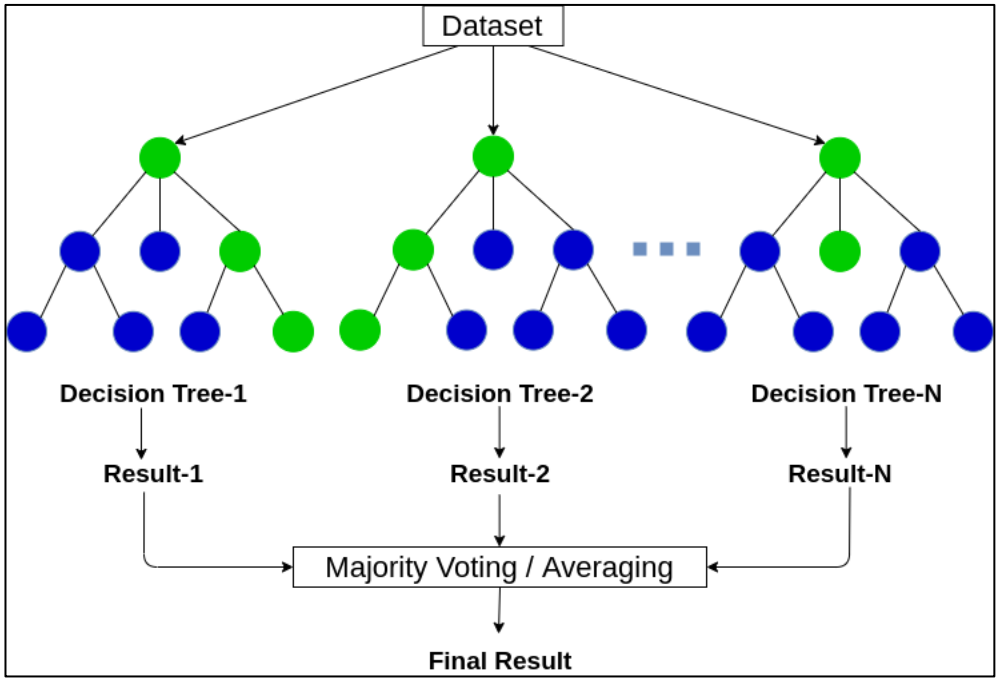

Рисунок 3.2 – Схематичний рисунок випадкового лісу

Ще одна відмінність між деревом рішень і випадковим лісом полягає в тому, що, хоча дерево рішень легко читати – потрібно просто йти по шляху і знаходити

результат – випадковий ліс дещо складніше інтерпретувати. В порівнянні з деревами рішень, крива навчання є стрімкою.

3.4 Градієнтний бустинг

А тепер слід розказати про випадковий ліс та градієнтний бустинг та порівняти ці два алгоритми. Як і випадкові ліси, градієнтний бустинг – це набір дерев рішень. Дві основні відмінності:

1. Як будуються дерева: випадкові ліси будують кожне дерево окремо, а градієнтний бустинг створює одне дерево за раз. Ця адитивна модель (ансамбль) працює поетапно, представляючи слабкого учня, щоб покращити недоліки наявних слабких учнів;
2. Об’єднання результатів: випадкові ліси об’єднують результати в кінці процесу (шляхом усереднення або «правила більшості»), тоді як градієнтний бустинг об’єднує результати на самому шляху.

Якщо ви ретельно налаштуєте параметри, бустинг може призвести до кращої продуктивності, ніж випадкові ліси. Однак градієнтний бустинг може бути не найкращим вибором, якщо у вас багато шуму, оскільки це може призвести до перенавчання. Їх також, як правило, важче налаштувати, ніж випадкові ліси.

Випадкові ліси та градієнти, що підсилюють кожен із них, є відмінними в різних областях. Випадкові ліси добре працюють для багатокласового виявлення об’єктів та біоінформатики, яка, як правило, має багато статистичного шуму. Градієнтний бустинг добре працює, коли у вас є незбалансовані дані, наприклад, при оцінці ризику в реальному часі.

З наведеного вище опису алгоритмів можна зробити висновки, що дерева рішень не підходить для нашої конкретної задачі. Тож нам слід порівняти випадковий ліс та градієнтний бустинг більш детально та обрати один з цих двох варіантів.

Градієнтний бустинг будує дерева по одному, де кожне нове дерево

допомагає виправляти помилки, допущені раніше навченим деревом.

Найкращим застосуванням бустингу є виявлення аномалій в умовах навчання під наглядом, де дані часто дуже незбалансовані, як-от послідовності ДНК, транзакції з кредитними картками або кібербезпека.

Цей метод має низку переваг. Оскільки розширені дерева отримують шляхом оптимізації цільової функції, в основному GBM (градієнтний бустинг) можна використовувати для вирішення майже всіх цільових функцій, які ми можемо виписати градієнтом. Сюди входять такі речі, як рейтингування та регресія позицій, яких за допомогою RF (випадковий ліс) досягти важче.

Проте, метод градієнтного бустингу має свої недоліки також у порівнянні з випадковим лісом:

- градієнтний бустинг є більш схильний до перенавчання, якщо дані мають «шуми»;
- навчання зазвичай займає більше часу через те, що дерева будуються послідовно;
- GBM важче налаштувати, ніж RF. Зазвичай є три параметри: кількість дерев, глибина дерев і швидкість навчання, і кожне побудоване дерево, як правило, є неглибоким.

Щодо випадкового лісу, то випадкові ліси навчають кожне дерево незалежно, використовуючи випадкову вибірку даних. Ця випадковість допомагає зробити модель більш надійною, ніж єдине дерево рішень, і з меншою ймовірністю переповнювати навчальні дані. Найкращим застосуванням випадкового лісу є багатокласове виявлення об’єктів у великомасштабних реальних проблемах комп’ютерного зору. Його методи можуть ефективно обробляти велику кількість навчальних даних і за своєю суттю підходять для багатокласових задач. Іншим гарним прикладом застосування випадкових лісів є біоінформатика, а конкретніше, медична діагностика.

Переваги даної моделі:

- RF набагато легше налаштувати, ніж GBM. Зазвичай в RF є два параметри: кількість дерев і кількість об’єктів, які потрібно вибрати в кожному вузлі.

- RF важче переобладнати, ніж GBM.

А тепер слід сказати також і про слабкі сторони моделі:

- основне обмеження алгоритму Random Forests полягає в тому, що велика кількість дерев може зробити алгоритм повільним для прогнозування в реальному часі;
- для даних, що включають категоріальні змінні з різною кількістю рівнів, випадкові ліси зміщені на користь тих атрибутів з більшою кількістю рівнів, тому оцінки змінної важливості з випадкового лісу не є надійними для цього типу даних. Для розв'язання задачі використовувалися такі методи, як часткові перестановки;
- якщо дані містять групи корельованих ознак, які мають подібну релевантність для результату, то менші групи віддають перевагу більшим групам.

Градієнтний бустинг і випадковий ліс відрізняються за способом побудови дерев, порядком і способом поєднання результатів. Було показано, що бустинг працює краще, ніж випадковий ліс, якщо параметри ретельно налаштовані, адже градієнтний бустинг навчається на основі попередніх помилок. Тобто кожне нове дерево має помилку меншу за попередню.

Наразі існують такі програмні рішення, які дозволяють оброблювати та аналізувати текст натуральною мовою[8], виконувати пошук за зображеннями[9] та розпізнавати образи і обличчя[10]. Саме тому реалізацію методу прогнозування результатів спортивних матчів виглядає актуальним та можливим до реалізації.

4 МЕТОД ПРОГНОЗУВАННЯ ГРАДІЄНТНИЙ БУСТИНГ

4.1 Опис алгоритму градієнтного бустингу

Проаналізувавши наведені вище алгоритми, я дійшла висновку, що для реалізації поставленої задачі в даній роботі я буду використовувати метод градієнтного бустингу (GBM). Адже, незважаючи на низьку швидкість навчання та складність підготовки даних, цей метод є більш влучним для конкретної ситуації. За допомогою цього методу можна біль точно спрогнозувати переможця в наступному матчі на основі зібраних про гравця даних.

Перш за все слід сказати, что градієнтний бустинг – це ансамблевий алгоритм. Ансамбль – це набір предикторів, які разом дають відповідь (наприклад, середнє значення для всіх). Причина, чому ми використовуємо ансамблі, полягає в тому, що кілька предикторів, які намагаються отримати ту саму змінну, дадуть більш точний результат, ніж один предиктор. Техніки ансамблю згодом класифікуються на бегінг (bagging) та бустинг (boosting).

Бегінг — це проста техніка, за допомогою якої ми створюємо незалежні моделі та об’єднуємо їх за допомогою якоїсь моделі усереднення (наприклад, середньозважена, більшість голосів чи звичайна середня).

Зазвичай для кожної моделі береться випадкова підвибірка даних, тому всі моделі дещо відрізняються одна від одної. Вибір базується на моделі відбору та повернення. У зв’язку з тим, що ця техніка використовує багато некорельованих моделей для побудови остаточної моделі, це зменшує дисперсію. Прикладом пакетування є модель Random Forest (RF).

Бустинг є загальним методом підвищення продуктивності будь-якого алгоритму машинного навчання. Суть його полягає у навчанні кожної наступної моделі з використанням даних про помилки попередніх моделей та подальше зниження помилок.

Цей метод теоретично можна використовувати для будь-якого слабкого алгоритму у цілях зниження помилки навчання.

Градієнтний бустинг дерев рішень дозволяє будувати адитивну функцію у

вигляді суми дерев рішень ітераційно, за аналогією з методом градієнтного спуску.

Так, на рис. 4.1 представлені ансамбль з z дерев і принцип мінімізації помилки Erz.

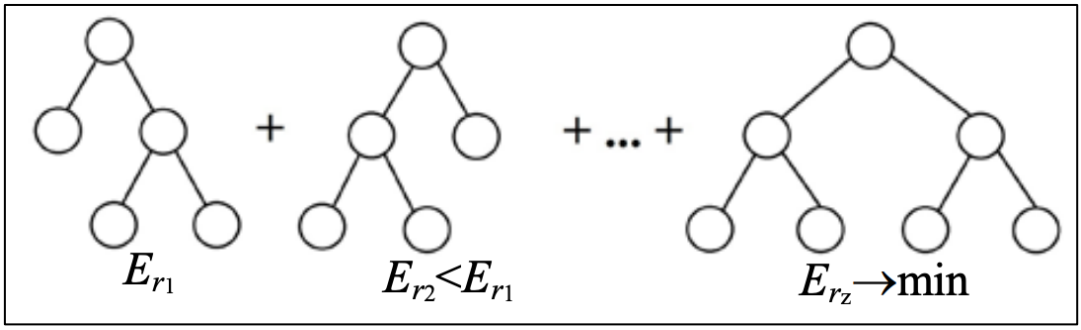

Рисунок 4.1 – Мінімізація помилки навчання

Але ми повинні вибирати критерій зупинки з обережністю, інакше це може призвести до перенавчання.

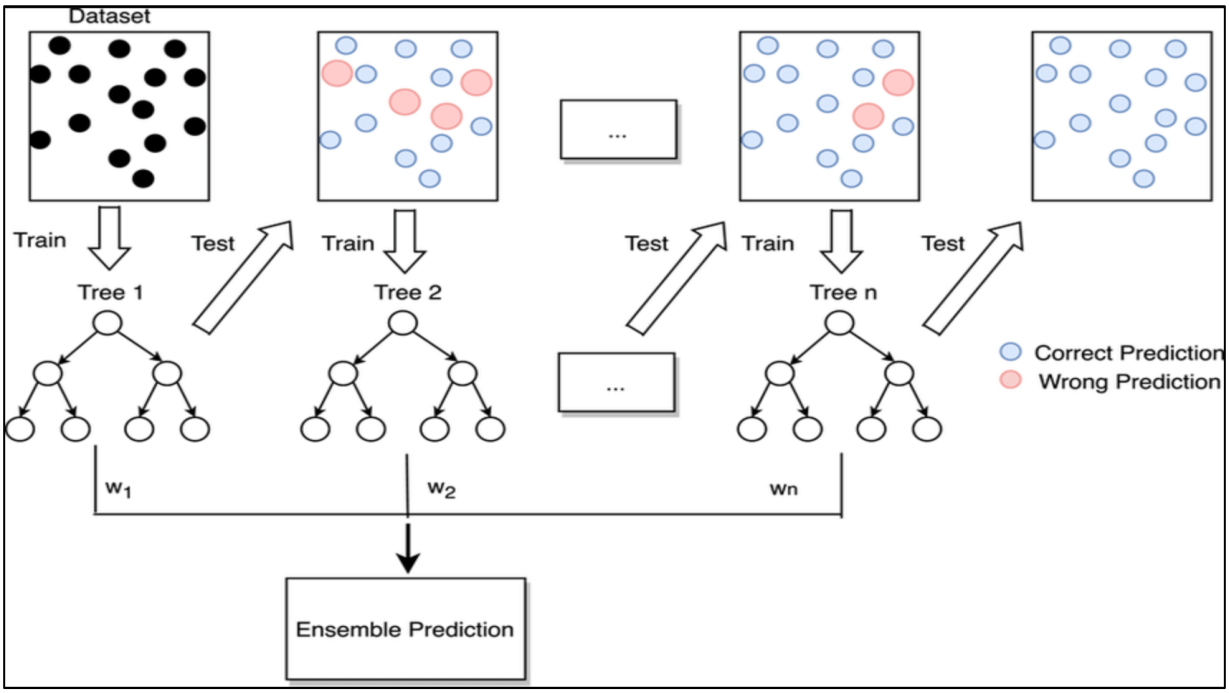

Рисунок 4.2 – Опис алгоритму Gradient Boosting Machine.

Градієнтний бустинг – це і є приклад того самого бустингу. Схема роботи класифікаційного алгоритму Gradient Boosting Machine наведена на рисунку 4.2.

Градієнтний бустинг – це метод перетворення слабких учнів у сильних[11]. При розширенні кожне нове дерево відповідає зміненій версії вихідного набору

даних. Алгоритм градієнтного бустингу найпростіше пояснити, спочатку представивши алгоритм AdaBoost. Алгоритм AdaBoost починається з навчання дерева рішень, у якому кожному спостереженню присвоюється однакова вага. Після оцінки першого дерева ми збільшуємо ваги тих спостережень, які важко класифікувати, і зменшуємо ваги тих, які легко класифікувати. Тому друге дерево вирощується на цих зважених даних. Тут ідея полягає в тому, щоб покращити передбачення першого дерева. Тому наша нова модель – Дерево 1 + Дерево 2. Потім ми обчислюємо помилку класифікації з цієї нової моделі ансамблю з 2 дерев і вирощуємо третє дерево, щоб передбачити переглянуті залишки. Ми повторюємо цей процес протягом певної кількості ітерацій. Наступні дерева допомагають нам класифікувати спостереження, які погано класифіковані попередніми деревами. Таким чином, передбачення остаточної моделі ансамблю є зваженою сумою прогнозів, зроблених попередніми моделями дерева.

Gradient Boosting тренує багато моделей поступово, адитивно та послідовно. Основна відмінність між AdaBoost і алгоритмом градієнтного бустингу полягає в тому, як ці два алгоритми визначають недоліки слабких учнів (наприклад, дерева рішень). У той час як модель AdaBoost визначає недоліки, використовуючи точки даних з великою вагою. Функція втрат — це міра, яка вказує, наскільки хороші коефіцієнти моделі для відповідності базовим даним. Нижче наведено формулу квадратичної функції втрат.

$$L\big(y_i, h(x_i)\big) = \big(y_i - h(x_i)\big)^2, \tag{1}$$

де L – функція втрат, y_i – рішення з навчальної вибірки, $h(x_i)$ – результат і-го дерева рішень. Щоб отримати помилку всього ансамблю, необхідно підсумувати її по всіх вирішальних деревах. Загальна помилка наведена у формулі нижче.

$$Q = \sum_{i=1}^{N} L\big(y_i, h(x_i)\big) = \sum_{i=1}^{N} \big(y_i - h(x_i)\big)^2, \tag{2}$$

де Q – сумарна помилка моделі. Логічне розуміння функції втрат буде залежати від

того, що ми намагаємося оптимізувати. Наприклад, якщо ми намагаємося передбачити результат матчу з тенісу за допомогою регресії, то функція втрат буде заснована на похибці між істиним та прогнозованим рахунком матчу.

Якщо порівняти алгоритм градієнтного бустингу і градієнтного спуску, який лежить в основі багатьох моделей машинного навчання, то кожне наступне дерево в ансамблі є одним кроком градієнтного спуску. Це можна наочно побачити, якщо розглянути приватну похідну помилки, яка наведена у формулі нижче.

$$\frac{\partial Q}{\partial h_i} = \sum_{i=1}^{N} \frac{\partial}{\partial h_i} (y_i - h_i)^2 = 2(y_i - h_i), \quad (3)$$

Однією з найбільших мотивів використання градієнтного бустингу є те, що воно дозволяє оптимізувати визначену користувачем функцію витрат замість функції втрат, яка зазвичай пропонує менший контроль і по суті не відповідає реальним результатам.

4.2 Аналіз та підготовка даних

Датасет – це просто набір фрагментів даних, які комп’ютер може обробляти як єдиний блок для аналітичних і прогнозних цілей. Це означає, що зібрані дані мають бути однорідними та зрозумілими для машини, яка не бачить дані так само, як люди. Для цього після збору даних важливо попередньо обробити їх, очистити та заповнити, а також анотувати дані, додавши значущі теги, які можна читати комп’ютером.

Ключом до успіху в області машинного навчання або науки про данні є практика з різними типами наборів даних. Але знайти відповідний датасет для кожного типу проекту машинного навчання є складним завданням. Крім того, хороший набір даних повинен відповідати певним стандартам якості та кількості. Для плавного та швидкого навчання необхідно переконатися, що датасет є відповідним і добре збалансованим. Слід використовувати реальні дані, коли це

можливо, і консультуватися з досвідченими фахівцями щодо обсягу даних та джерела їх збору.

Існують різні типи даних у наборах даних:

- числові дані: такі як ціна будинку, температура;
- категорійні дані: такі як так/ні, правда/неправда, синій/зелений;
- порядкові дані: ці дані подібні до категоріальних даних, але їх можна виміряти на основі порівняння.

Для роботи з проектами машинного навчання потрібна величезна кількість даних, тому що без них неможливо навчити моделі ML/AI. Збір і підготовка датасету є однією з найважливіших частин під час створення проекту ML/AI. Технологія, яка використовується в будь-яких проектах ML, не може працювати належним чином, якщо набір даних не добре підготовлений і попередньо не оброблений. Під час розробки AI інженери завжди покладаються на дані. Від навчання, налаштування, вибору моделі до тестування використовуються три різні набори даних: навчальний набір, набір для перевірки та набір для тестування. Слід зазначити, що набори перевірки використовуються для вибору та налаштування остаточної моделі ML. Можна подумати, що збору даних достатньо, але це не так. У кожному проекті штучного інтелекту класифікація та маркування наборів даних займає більшу частину часу, особливо набори даних, достатньо точні, щоб відображати реалістичне бачення ринку/світу.

Для початку слід зазначити перші два види датасетів, які нам потрібні — навчальний набір і набір тестових даних, оскільки вони використовуються для різних цілей під час створення AI проекту, і успіх проекту багато в чому залежить від них.

Набір навчальних даних використовується для навчання алгоритму, щоб зрозуміти, як застосовувати такі концепції, як нейронні мережі, для навчання та отримання результатів. Він містить як вхідні дані, так і очікуваний вихід.

Тренувальні набори становлять більшість загального обсягу даних, близько 60 %. Під час тестування моделі відповідають параметрам у процесі, який відомий як коригування ваги.

Тестовий набір даних використовується для оцінки того, наскільки добре

алгоритм був навчений за набором навчальних даних. У проектах штучного інтелекту не використовується навчальний набір даних на етапі тестування, оскільки алгоритм вже заздалегідь знатиме очікуваний результат. А це суперечить меті навчання.

Тестові набори становлять 20% даних. Передбачається, що тестовий набір є вхідними даними, згрупованими разом із перевіреними правильними виходами, як правило, шляхом перевірки людиною. Виходячи з досвіду, було б поганою ідеєю продовжувати коригування після фази тестування. Ймовірно, це призведе до перенавчання.

Перенавчання — це помилка моделювання, яка виникає, коли функція занадто точно підходить до обмеженого набору точок даних. Для результативного навчання необхідно приблизно в 10 разів більше даних, ніж кількість параметрів у моделі, що створюється. Чим складніше завдання, тим більше потрібно даних.

Після визначення з датасетом необхідно знати, що всі набори даних є неточними. У цей момент проекту нам потрібно зробити деяку підготовку даних, дуже важливий крок у процесі машинного навчання. По суті, підготовка даних полягає в тому, щоб зробити набір даних більш придатним для машинного навчання. Це набір процедур, які займають більшу частину часу, витраченого на проекти машинного навчання.

Навіть якщо дані готові, все одно можна зіткнутися з проблемами, з їх якістю, а також зі зміщеннями, прихованими у навчальних датасетах. Простіше кажучи, якість навчальних даних визначає продуктивність систем машинного навчання. З роками вчені з’ясували, що деякі популярні набори даних, які використовуються для навчання розпізнавання зображень, містять гендерні зміщення.

Як наслідок, створення додатків AI займає більше часу, оскільки потрібно переконатися, що дані правильні та інтегровані належним чином.

Може статися, що не вистачає даних, необхідних для інтеграції рішення AI. Якщо покладатися на паперові документи або файли .csv, щоб створити набір даних, готовий до штучного інтелекту, потрібен час. Найкращим рішенням є перекласти цю відповідальність на іншу команду.

Створення культури, керованої даними, в організації є, мабуть, найскладнішою частиною роботи спеціаліста з штучного інтелекту. Дійсно, збір даних може бути дратівливим завданням, яке обтяжує співробітників. Проте можна автоматизувати більшість процесу збору даних.

Іншою проблемою може бути доступність даних і право власності. В такому разі необхідно створити зв'язки між блоками даних в організації. Щоб отримати особливу інформацію, слід зібрати дані з кількох джерел.

Що стосується власності, то те, що доступ до інформації наявний, не означає, що право її використовувати також присутнє. В таких випадках слід запитати про це юридичну команду (одним із прикладів є GDPR в Європі).

Хорошою ідеєю було б почати з моделі, яка була попередньо навчена на великому наявному наборі даних, і використовувати навчання передачі, щоб налаштувати її на менший набір даних, який був зібраний.

Наступним кроком є попередня обробка даних. На цьому кроці вже зібрані дані, які є важливими, різноманітними та репрезентативними для даного проекту AI. Попередня обробка включає вибір правильних даних із повного набору даних і створення навчального набору. Процес об'єднання даних у цьому оптимальному форматі відомий як перетворення ознак:

1. Формат: дані можуть бути поширені в різних файлах. Наприклад, результати продажів із різних країн з різною валютою, мовами тощо, які потрібно зібрати разом, щоб сформувати набір даних;

2. Очищення даних: на цьому кроці мета — розібратися з відсутніми значеннями та видалити небажані символи з даних;

3. Вилучення функцій: на цьому кроці ми зосередимося на аналізі та оптимізації кількості функцій. Зазвичай необхідно з'ясувати, які функції важливі для передбачення, і вибрати їх для швидших обчислень і низького споживання пам'яті.

Попередня обробка даних — це процес перетворення необроблених даних у зрозумілий формат. Це також важливий крок у аналізі даних, оскільки ми не можемо працювати з необробленими даними. Якість даних слід перевірити перед застосуванням алгоритмів машинного навчання або інтелекту.

Попередня обробка даних полягає в основному в перевірці якості даних. Якість можна перевірити наступним методами:

– точність: для перевірки правильності введених даних чи ні;

– повнота: щоб перевірити, чи доступні дані чи ні;

– узгодженість: щоб перевірити, чи зберігаються однакові дані в усіх місцях, які збігаються чи не збігаються;

– своєчасність: дані повинні оновлюватися правильно;

– достовірність: дані повинні бути надійними;

– інтерпретація: зрозумілість даних.

Основні завдання попередньої обробки даних:

– очищення даних,

– інтеграція даних,

– скорочення даних,

– перетворення даних.

Очищення даних – це процес видалення неправильних даних, неповних і неточних даних із наборів даних, а також заміна відсутніх значень. Існують деякі методи очищення даних.

Обробка відсутніх значень: для заміни відсутніх значень можна використовувати стандартні значення, як-от «Недоступно» або «Немає». Відсутні значення також можна заповнити вручну, але це не рекомендується, якщо цей набір даних великий. Середнє значення атрибута можна використовувати для заміни відсутнього значення, коли дані розподіляються нормально, при цьому у випадку ненормального розподілу може використовуватися медіанне значення атрибута. При використанні алгоритмів регресії або дерева рішень пропущене значення можна замінити на найбільш ймовірне значення.

Шумні дані: зазвичай означає випадкову помилку або містить непотрібні точки даних. Ось деякі з методів обробки даних із шумом:

– бінінг: цей метод призначений для згладжування або обробки даних із шумом. Спочатку дані сортуються, а потім відсортовані значення відокремлюються і зберігаються у вигляді ящиків. Існує три способи згладжування

даних у кошику. Згладжування за методом середнього значення bin: у цьому методі значення в bin замінюються середнім значенням bin; Згладжування за медіаною бенза: у цьому методі значення в корзині замінюються на середнє значення; Згладжування за межею розділу: у цьому методі беруться мінімальні та максимальні значення значень бігу, а значення замінюються найближчим граничним значенням;

– регресія: використовується для згладжування даних і допоможе обробляти дані, коли є непотрібні дані. Для аналізу регресія мети допомагає визначити змінну, яка підходить для нашого аналізу;

– кластеризація: це використовується для пошуку викидів, а також для групування даних. Кластеризація зазвичай використовується в навчанні без нагляду.

Інтеграція даних: процес об'єднання кількох джерел в один набір даних. Процес інтеграції даних є одним з основних компонентів в управлінні даними. Під час інтеграції даних необхідно враховувати деякі проблеми:

– інтеграція схеми: інтегрує метадані (набір даних, що описують інші дані) з різних джерел;

– проблема ідентифікації сутностей: ідентифікація сутностей з кількох баз даних. Наприклад, система або використання повинні знати, що студент _id однієї бази даних і student_name іншої бази даних належить до тієї ж сутності;

– виявлення та вирішення концепцій значень даних: дані, взяті з різних баз даних під час об'єднання, можуть відрізнятися. Подібно значення атрибутів однієї бази даних можуть відрізнятися від іншої бази даних. Наприклад, формат дати може відрізнятися як «ММ/ДД/РРРР» або «ДД/ММ/РРРР».

Скорочення даних: це процес, який допомагає зменшити обсяг даних, що полегшує аналіз, але дає той самий або майже той самий результат. Це зменшення також допомагає зменшити простір для зберігання. Існують деякі методи скорочення даних: зменшення розмірності, зменшення кількості, стиснення даних.

Зменшення розмірності: цей процес необхідний для реальних додатків, оскільки розмір даних великий. У цьому процесі відбувається зменшення випадкових величин або атрибутів, щоб можна було зменшити розмірність набору

даних. Об’єднання та об’єднання атрибутів даних без втрати вихідних характеристик. Це також допомагає зменшити простір для зберігання та зменшити час обчислень. Коли дані мають велику розмірність, виникає проблема, яка називається «Прокляття розмірності».

Зменшення кількості: у цьому методі представлення даних зменшується за рахунок зменшення обсягу. При цьому скороченні втрати даних не буде.

Стиснення даних – стиснена форма даних називається стисненням даних. Це стиснення може бути без втрат або без втрат. Коли під час стиснення немає втрати інформації, це називається стисненням без втрат. Тоді як стиснення з втратами зменшує інформацію, але видаляє лише непотрібну інформацію.

Ще одним способом підготовки датасету є трансформація даних. Зміна, внесена у формат або структуру даних, називається перетворенням даних. Цей крок може бути простим або складним залежно від вимог. Існують деякі методи перетворення даних:

– згладжування: за допомогою алгоритмів ми можемо видалити шум із набору даних і допоможемо дізнатися важливі характеристики набору даних. Згладжуючи, ми можемо знайти навіть просту зміну, яка допомагає прогнозувати;

– агрегація: у цьому методі дані зберігаються та представлені у вигляді підсумку. Набір даних із кількох джерел інтегрується в опис аналізу даних. Це важливий крок, оскільки точність даних залежить від кількості та якості даних. Коли якість і кількість даних хороші, результати є більш релевантними;

– дискретизація: безперервні дані тут розбиваються на інтервали. Дискретизація зменшує розмір даних. Наприклад, замість того, щоб вказувати час занять, ми можемо встановити інтервал, наприклад (15:00-17:00, 18:00-20:00);

– нормалізація: це метод масштабування даних, щоб вони могли бути представлені в меншому діапазоні. Приклад у діапазоні від -1,0 до 1,0.

Найуспішніші проекти штучного інтелекту – це ті, які інтегрують стратегію збору даних протягом життєвого циклу послуги/продукту. Дійсно, збір даних не може бути серією одноразових вправ. Він повинен бути вбудований в сам основний продукт. По суті, щоразу, коли користувач взаємодіє з продуктом/послугою, слід збирати дані про взаємодію. Мета полягає в тому, щоб використовувати цей

постійний новий потік даних для покращення свого продукту/послуги.

При досягненні цього рівня використання даних, кожен новий клієнт збільшує набір даних і продукт стає кращим, що залучає більше клієнтів, покращує набір даних тощо. Це таке собі позитивне коло.

Найкращі та довгострокові проекти ML – це проекти, які використовують динамічні, постійно оновлювані набори даних. Перевага створення такої стратегії збору даних полягає в тому, що конкурентам стає дуже важко відтворити набір даних. З даними ШІ стає кращим, а в деяких випадках, як-от спільна фільтрація, дуже цінний. Спільна фільтрація дає пропозиції на основі подібності між користувачами. Це покращиться з доступом до більшої кількості даних; чим більше даних користувача, тим більша ймовірність того, що алгоритм зможе знайти схожого користувача.

Це означає, що потрібна стратегія для постійного вдосконалення набору даних до тих пір, поки є якісь переваги користувача для підвищення точності моделі.

Інший підхід полягає в тому, щоб підвищити ефективність конвеєра маркування, наприклад, раніше багато покладалися на систему, яка могла б запропонувати мітки, передбачені початковою версією моделі, щоб етикетки могли швидше приймати рішення. Деякі компанії просто наймають більше людей, щоб позначити нові навчальні матеріали. Це вимагає часу і грошей, але це працює, хоча це може бути складно в організаціях, які традиційно не мають резерву в своєму бюджеті для такого роду витрат. Незважаючи на те, що кажуть більшість компаній SaaS, машинне навчання вимагає часу та підготовки.

4.2 Опис обраного датасету

Обраний набір даних містить наступні типи даних матчів: файли гравців ATP, історичні рейтинги, результати та статистику матів.

Стовпці файлів гравця є наступними: player_id, name_name, place_name, hand,

birth_date, country_code, height (cm).

Стовпцями для файлів рейтингу є rank_date, ranking, player_id, ranking_points (якщо доступно).

Історичні рейтинги ATP в основному повні з 1985 року по теперішній час. 1982 р. відсутній, а рейтинги за 1973-1984 рр. лише переривчасті.

Результати та статистика представлена іншим чином. За сезон можна отримати до трьох файлів: один для матчів основної сітки на рівні туру (наприклад, 'atp_matches_2014.csv'), один для кваліфікаційних матчів на рівні туру та матчів основної сітки суперників і один для майбутніх матчів.

Більшість стовпців у файлах результатів є очевидними. Щоб спростити використання файлів результатів для більшої кількості людей було додано невелику надлишковість до біографічних файлів і файлів рейтингу: кожен рядок містить кілька стовпців біографічної інформації, а також рейтингові бали для обох гравців. Дані про рейтинг, а також вік вказано на tourney_date, яка майже завжди є понеділком на початку події або близько неї.

Статистика матчів включена там, де її можливо отримати. Загалом, це означає 1991-до теперішнього часу для матчів на рівні туру, 2008-до теперішнього часу для претендентів і 2011-до теперішнього часу для кваліфікаційного рівня туру. Стовпці MatchStats мають бути зрозумілими, але вони можуть бути не тими, що звикли бачити. Це все цілі підсумки з яких можна обчислити традиційні відсотки.

Є кілька матчів на рівні туру, у яких відсутня статистика. Деякі відсутні, тому що у ATP їх немає. Інші були видалені, тому що вони не пройшли деяку перевірку на цілісність (переможений виграв 60% балів, або час матчу був менше 20 хвилин тощо). Тобто, датасет вже без викидів.

Багато стовпців у файлах «матчів» є очевидними або дуже схожі на попередні стовпці:

– tourney_id – унікальний ідентифікатор для кожного турніру, наприклад 2020-888. Точні формати запозичені з кількох різних джерел, тому, хоча перші чотири символи завжди є роком, решта ідентифікатора не має передбачуваної структури;

– tourney_name – назва турніру;

– surface – поверхня;

– draw_size – кількість гравців у жеребкуванні, часто округляється до найближчого ступеня 2. (Наприклад, турнір з 28 гравцями може відображатися як 32);

– tourney_level – рівень турніру, наприклад: "G" – турніри Великого шолома, "M" – Masters 1000s, "A" – інші події на рівні туру, "C" – претенденти, "S" – сателіти/ITF, "F" – фінали туру та інші події на завершення сезону, а "D" – Кубок Девіса;

– tourney_date – вісім цифр, РРРРММДД, зазвичай понеділок турнірного тижня;

– match_num – відповідний ідентифікатор матчу. Часто починаючи з 1, іноді з відліком від 300, а іноді довільно;

– winner_id - player_id, використаний у цьому репозиторії для переможця матчу;

– winner_entry – вхід в турнір: 'WC' = wild card, 'Q' = кваліфікаційний, 'LL' щасливий невдах, 'PR' = захищений рейтинг, 'ITF' = запис ITF, і є кілька інших, які іноді використовуються;

– winner_name – ім'я переможця;

– winner_hand – основна рука переможця: R = праворуч, L = ліворуч, U невідомо, для гравців-амбідекстрів це їхня подача;

– winner_ht – зріст в сантиметрах, якщо є;

– winner_ioc – тризначний код країни;

– winner_age – вік, у роках на дату турніру;

– score – рахунок;

– best_of – кількість виграних сетів, «3» або «5», що вказує на кількість сетів для цього матчу;

– round, minutes – тривалість, якщо є;

– w_ace – кількість найкращих ударів переможця;

– w_df – кількість парних помилок переможця;

– w_svpt – кількість очок подачі переможця;

– w_1stIn – кількість зроблених перших подач переможця;

– w_1stWon – кількість виграних очок за першу подачу;

– w_2ndWon – кількість виграних очок за другу подачу;

– w_SvGms – кількість переможців подачі геймів;

– w_bpSaved – кількість збережених брейк-пойнтів переможця;

– w_bpFaced – кількість отриманих пойнтів переможця;

– winner_rank – рейтинг ATP або WTA переможця на дату tourney_date або остання дата рейтингу перед датою tourney_date;

– winner_rank_points – кількість балів рейтингу, якщо є.

Найкраще відобразити обраний датасет за допомогою heatmap діаграми. Теплові карти кореляції – це тип графіка, який візуалізує міцність зв’язків між числовими змінними.

Кореляційні графіки використовуються, щоб зрозуміти, які змінні пов’язані одна з одною, а також силу цього зв’язку. Графік кореляції зазвичай містить ряд числових змінних, кожна з яких представлена стовпцем.

Рядки представляють зв’язок між кожною парою змінних. Значення в клітинках вказують на міцність зв’язку, причому позитивні значення вказують на позитивний зв’язок, а від’ємні значення вказують на негативний зв’язок. Теплові карти кореляції можна використовувати, щоб знайти потенційні зв’язки між змінними та зрозуміти силу цих зв’язків. Крім того, кореляційні графіки можна використовувати для ідентифікації викидів і виявлення лінійних і нелінійних зв’язків. Кольорове кодування клітинок дозволяє легко визначити зв’язки між змінними з першого погляду. Теплові карти кореляції можна використовувати для пошуку як лінійних, так і нелінійних зв’язків між змінними.

Теплова карта кореляції — це графічне представлення кореляційної матриці, що представляє кореляцію між різними змінними.

Значення кореляції може приймати будь-яке значення від -1 до 1. Якщо значення дорівнює 1, говорять, що це позитивна кореляція між двома змінними. Це означає, що коли одна змінна збільшується, інша змінна також збільшується. Якщо значення дорівнює -1, говорять про негативну кореляцію між двома змінними. Це означає, що коли одна змінна збільшується, інша змінна зменшується. Якщо

значення дорівнює 0, немає кореляції між двома змінними. Це означає, що змінні змінюються випадковим чином один щодо одного.

Кореляція між двома випадковими величинами або двовимірними даними не обов'язково передбачає причинно-наслідковий зв'язок. Кореляцію між двома змінними також можна визначити за допомогою діаграми розсіювання між цими двома змінними. Обраний датасет можна також відобразити на прикладі heatmap, на якому зображені вагомі для дослідження поля (див. рис. 4.3).

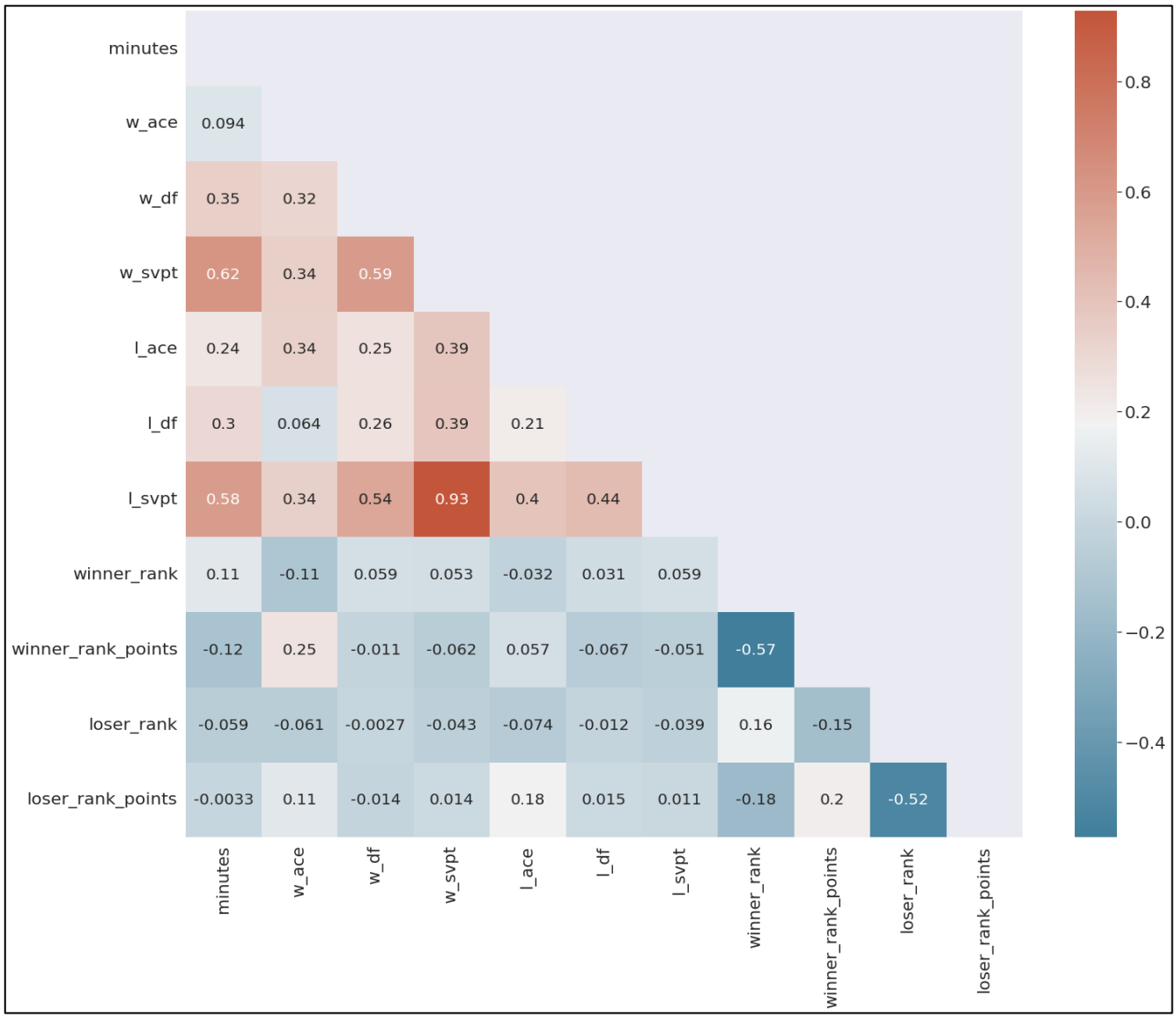

Рисунок 4.3 – Кореляційний heatmap обраного датасету

Щоб намалювати теплову карту кореляції для набору даних спортивних матчів, що представляє кореляцію між різними змінними, включаючи змінні предиктора та відповіді, слід звернути увагу на деякі з наступних:

– пакет Pandas використовується для читання табличних даних за допомогою методу read_table;

– метод corr() викликається у Pandas DataFrame, щоб визначити кореляцію між різними змінними, включаючи змінні предиктора та відповіді;

– метод Seaborn heatmap() використовується для створення теплової карти, що представляє кореляційну матрицю.

На даній тепловій карті можна побачити, що поля l_svtp та w_svtp корелюються прямо пропорційною Це властивість, яка відповідає за набрані під час подачі пойнти. У той самий час поля winner_rank_points та winner_rank мають зворотню кореляцію. Саме тому, не має сенсу використовувати обидві ці властивості.

А такі поля як w_svtp та winner_rank у свою чергу, навпаки, ніяк не корелюють між собою.

Подивившись на дану мапу було визначенно, які поля є взаємнозамінними, а які мають великий вплив на подальше навчання.

5 МЕТРИКИ ТА КРИТЕРІЇ ОЦІНЮВАННЯ

Оцінка алгоритму машинного навчання є важливою частиною будь-якого проекту. Математична модель може дати вам задовільні результати при оцінці за допомогою метрики, скажімо, оцінка точності, але може дати погані результати при оцінці за логарифмічною втратою або будь-яким іншим показником. У більшості випадків використовується точність класифікації для вимірювання продуктивності моделі, однак цього недостатньо, щоб по-справжньому судити про модель.

5.1 Точність класифікації

Точність класифікації – це те, що зазвичай мають на увазі, коли використовують термін точність. Це відношення кількості правильних прогнозів до загальної кількості вхідних вибірок (див. формулу нижче).

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}, \quad (4)$$

Ця метрика добре працює лише за умови однакової кількості зразків, що належать до кожного класу.

Наприклад, якщо в навчальному наборі є 98% зразків класу A і 2% зразків класу B, тоді модель може легко отримати 98% точності навчання, просто передбачивши кожну навчальну вибірку, що належить до класу A. Коли ту саму модель тестують на тестовому наборі з 60% зразків класу A і 40% зразків класу B, точність тесту впаде до 60%. Точність класифікації чудова, але дає хибне відчуття досягнення високої точності.

Справжня проблема виникає, коли вартість помилкової класифікації зразків другорядного класу дуже висока. Якщо ми маємо справу з рідкісною, але

смертельною хворобою, то ціна того, що не вдається діагностувати хворобу хворої людини, набагато вища, ніж вартість відправлення здорової людини на додаткові дослідження.

5.2 Логарифмічна втрата

Логарифмічна втрата або Log Loss, працює шляхом покарання за помилкові класифікації. Він добре працює для багатокласової класифікації. При роботі з Log Loss класифікатор повинен призначити ймовірність кожному класу для всіх вибірок. Припустимо, є N зразків, що належать до M класів, тоді втрата журналу обчислюється, як показано нижче:

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log\left(p_{ij}\right), \quad (5),$$

де y_{ij} вказує, чи належить зразок i до класу j чи ні, а p_{ij} вказує ймовірність належності вибірки i до класу j.

Log Loss не має верхньої межі і існує в діапазоні [0, ∞). Log Loss ближче до 0 вказує на більш високу точність, тоді як якщо Log Loss знаходиться далеко від 0, то це вказує на нижчу точність. Загалом, мінімізація втрат журналу дає більшу точність для класифікатора.

5.3 Матриця плутанини

Матриця плутанини, як випливає з назви, дає нам матрицю як вихідну інформацію та описує повну продуктивність моделі.

Припустимо, що у нас проблема бінарної класифікації. У нас є деякі зразки,

що належать до двох класів: ТАК чи НІ. Крім того, у нас є власний класифікатор, який прогнозує клас для даної вхідної вибірки. При тестуванні нашої моделі на 187 зразках ми отримуємо наступний результат (див. табл. 5.1).

Таблиця 5.1 – Матриця плутанини

N=187	Фактичні позитивні	Фактичні негативні
Очікувані позитивні	55	19
Очікувані негативні	8	105

Є 4 важливі терміни для матриці плутанини:

– справжні позитивні моменти: випадки, коли ми передбачили ТАК, фактичний результат також був ТАК;

– справжні негативні: випадки, коли ми передбачили НІ, а фактичний результат був НІ;

– помилково позитивні: випадки, коли ми передбачили ТАК, а фактичний результат був НІ;

– помилково негативні: випадки, коли ми передбачили НІ, а фактичний результат був ТАК.

Точність для матриці можна обчислити, узявши середнє значення, що лежать поперек «головної діагоналі», тобто:

$$Accuracy = \frac{TruePositive+TrueNegative}{TotalSample} = \frac{55+105}{187} = 0{,}86, \qquad (6),$$

де $TruePositive$ – справжні позитивні випадки, $TrueNegative$ – справжні негативні, $TotalSample$ – сума всіх випадків.

5.4 Середня абсолютна похибка

Середня абсолютна похибка (MAE) — це середнє значення різниці між вихідними значеннями та прогнозованими значеннями. Це дає нам міру того, наскільки далекі були прогнози від фактичного результату. Однак вони не дають нам жодного уявлення про напрямок помилки, тобто про те, чи ми передбачили недостатньо чи перевищили дані. Математично він представляється у вигляді:

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}|y_j - \widehat{y_J}|, \quad (7),$$

де N – сума експериментів, y_j – отриманий результат, а $\widehat{y_J}$ – зпрогнозований.

5.5 Середня квадратична помилка

Середня квадратична помилка (MSE) дуже схожа на середню абсолютну помилку, єдина відмінність полягає в тому, що MSE приймає середнє значення квадрата різниці між вихідними значеннями та прогнозованими значеннями. Перевага MSE полягає в тому, що легше обчислити градієнт, тоді як середня абсолютна помилка вимагає складних інструментів лінійного програмування для обчислення градієнта. Оскільки береться квадрат помилки, ефект більших помилок стає більш вираженим, ніж менша похибка, отже, модель тепер може більше зосередитися на більших помилках.

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}\left(y_j - \widehat{y_J}\right)^2, \quad (8),$$

де N – сума експериментів, y_j – отриманий результат, а $\widehat{y_J}$ – зпрогнозований результат.

6 ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ

Тепер слід обрати програмні методи та інструменти для реалізації заданого рішення. Мовою програмування була обрана мова Python[12] через її властивості швидкого написання та легкого читання. А також через те, що це найбільш розповсюджена мова програмування серед програмних науковців. А це означає, що вона має велику кількість бібліотек для будь-яких програмних задач.

Python — це високорівнева, динамічно типізована багатопарадигмальна мова програмування. Часто кажуть, що код Python майже схожий на псевдокод, оскільки він дозволяє висловлювати дуже потужні ідеї в кількох рядках коду, будучи дуже читабельним.

У якості IDE був обраний PyCharm, адже компанія JetBrains створює, на мою думку, найкращі продукти для розробників та програмних інженерів. У кожному продукті можна знайти щось нове, чого немає в аналогів, та найбільш зручний інтерфейс завжди надихає на нові досягнення.

Як вже було зазначено вище, Python дуже розповсюджена мова програмування та має великий набір готових рішень та бібліотек. Саме тому я скористаюся декількома з них, які опишу нижче.

Numpy є основною бібліотекою для наукових обчислень на Python. Він надає високопродуктивний багатовимірний об’єкт масиву та інструменти для роботи з цими масивами. Масив numpy – це таблиця значень, усі одного типу, і індексується кортежем цілих невід’ємних чисел. Кількість вимірів – це ранг масиву; форма масиву являє собою кортеж цілих чисел, що надають розмір масиву вздовж кожного виміру.

Pandas — це бібліотека Python для аналізу даних. Започаткована Весом МакКінні у 2008 році через потребу в потужному та гнучкому інструменті кількісного аналізу, pandas перетворилася на одну з найпопулярніших бібліотек Python. Він має надзвичайно активну спільноту дописувачів. Pandas побудовано на основі двох основних бібліотек Python — matplotlib для візуалізації даних і NumPy для математичних операцій. Pandas діє як обгортка над цими бібліотеками,

дозволяючи отримати доступ до багатьох методів matplotlib і NumPy з меншою кількістю коду. Наприклад, .plot() pandas поєднує кілька методів matplotlib в один метод, що дозволяє побудувати діаграму в кілька рядків[13].

Scikit-learn – це бібліотека, яка містить кілька алгоритмів навчання класифікаторів або регресорів (дерево рішень, лінійна регресія, персетрон тощо) та кілька наборів даних для дидактичного навчання. За допомогою кількох рядків коду можна навчити машину виконувати певне завдання..

Тож реалізуємо основний алгоритм Gradient Boosted Tree на мові програмування Python з використанням вище наведених бібліотек.

```
import numpy as np
from sklearn.metrics import zero_one_loss
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split

def gradient_boosted_tree(featues, targets):
    train_features, test_features, train_targets, test_targets =
train_test_split(features, targets, test_size=0.2, random_state=111)
    ensemble = GradientBoostingClassifier(max_depth=1,
n_estimators=210, random_state=111).fit(train_features,
train_targets)

    predictions = ensemble.predict(test_features)
    print("Bias loss = " + str(np.round(zero_one_loss(predictions,
test_targets), 2)))
    return predictions

gradient_boosted_tree(features, targets)
```

Лістінг 6.1 – Імплементація алгоритму Gradient Boosted Tree

У результаті було реалізовано алгоритм класифікації Gradient Boosting Machine. Це дуже ефективний алгоритм для прогнозування відношення до певного класу, а саме це і необхідно нам у контексті поставленої задачі.

7 АНАЛІЗ РЕЗУЛЬТАТІВ ПРОВЕДЕНИХ ЕКСПЕРИМЕНТІВ

У ході проведення експериментів порівняно методів машинного навчання таких як: дерево рішень, випадковий ліс, градієнтний бустинг, до обраного для навчання та тестування результатів набору даних було застосовано метод зменшення розмірності PCA. Це було зроблено для спрощення візуалізації результатів роботи моделей та можливості їх графічного порівняння[14].

Аналіз головних компонентів (PCA) — це метод лінійного зменшення розмірності, який можна використовувати для вилучення інформації з простору високої вимірності шляхом проектування її в підпростір нижчого виміру. Він намагається зберегти суттєві частини, які мають більше варіацій даних, і видалити несуттєві частини з меншою кількістю варіацій. Розміри — це не що інше, як ознаки, які представляють дані.

Одна важлива річ, яку слід зазначити щодо PCA, полягає в тому, що це метод безконтрольного зменшення розмірності, ви можете групувати подібні точки даних на основі кореляції між ними без будь-якого контролю[15]. PCA — це статистична процедура, яка використовує ортогональне перетворення для перетворення набору спостережень можливо корельованих змінних у набір значень лінійно некорельованих змінних, які називаються головними компонентами.

На рисунку А.1 в додатку А зображено графічну ілюстрацію результатів експерименту. Перший графік відображає розподіл даних у наведеному до двомірної розмірності наборі даних.

На наступних графіках відповідно зображено розбиття простору ознак на класи за результатами роботи конкретного методу машинного навчання.

Дерева рішень схильні до перенавчання, особливо коли дерево є глибоким. Це пов’язано з кількістю специфічності, яку ми розглядаємо, що призводить до меншої вибірки подій, які відповідають попереднім припущенням. Ця невелика вибірка може призвести до необґрунтованих висновків. Прикладом цього може бути прогноз, чи переможуть "Бостон Селтікс" у сьогоднішньому баскетбольному матчі "Маямі Хіт". Перший рівень дерева може запитати, чи грають «Селтікс»

вдома чи на виїзді. Другий рівень може запитати, чи мають «Селтікс» вищий відсоток перемог, ніж їхній суперник. Третій рівень запитує, чи грає найкращий бомбардир «Селтика»? Четвертий рівень запитує, чи грає другий найкращий бомбардир «Селтика». П'ятий рівень запитує, чи повертаються «Селтікс» на східне узбережжя після 3 або більше послідовних ігор на західному узбережжі. Хоча всі ці запитання можуть бути актуальними, можливо, лише дві попередні ігри були виконані умовами сьогоднішньої гри. Використання лише двох ігор як основи для нашої класифікації не було б достатнім для прийняття зваженого рішення. Одним із способів боротьби з цією проблемою є встановлення максимальної глибини. Це обмежить наш ризик перенавчання, але, як завжди, це буде за рахунок помилок через упередженість. Таким чином, якщо ми встановимо максимальну глибину 3, ми запитаємо лише, домашня гра чи на виїзді, чи мають «Селтікс» вищий відсоток перемог, ніж їхній суперник, і чи є їхнім найкращим бомбардиром. Це простіша модель з меншою дисперсією від вибірки до вибірки, але в кінцевому підсумку вона не буде сильною прогнозною моделлю.

В ідеалі ми хотіли б мінімізувати як помилки через перенавчання, так і помилки через дисперсію. Випадкові ліси добре пом'якшують цю проблему. Їхня здатність обмежувати перенавчання без істотного збільшення похибок є причиною таких потужних моделей.

Якщо ми використовуємо багато дерев у нашому лісі, зрештою буде включено багато або всі наші ознаки. Таке включення багатьох ознаки допоможе обмежити нашу помилку через перенавчання та помилку через дисперсію. Якби об'єкти не вибиралися випадково, базові дерева в нашому лісі могли б стати дуже корельованими. Це пов'язано з тим, що деякі ознаки можуть бути особливо передбачуваними, і, таким чином, ті самі характеристики будуть вибрані в багатьох базових деревах. Якби багато з цих дерев містили ті самі характеристики, ми б не боролися з помилками через дисперсію.

З огляду на це, випадкові ліси є потужною технікою моделювання і набагато надійнішою, ніж одне дерево рішень[16]. Вони об'єднують багато дерев рішень, щоб обмежити переобладнання, а також помилки через упередження і, отже, дають корисні результати.

Одним з недоліків бустінгу є те, що він чутливий до викидів, оскільки кожен класифікатор зобов'язаний виправляти помилки попередників. Таким чином, метод занадто залежить від викидів. Ще одним недоліком є те, що метод практично неможливо розширити. Це пов'язано з тим, що кожен оцінювач базує свою правильність на попередніх, що ускладнює впорядкування процедури.

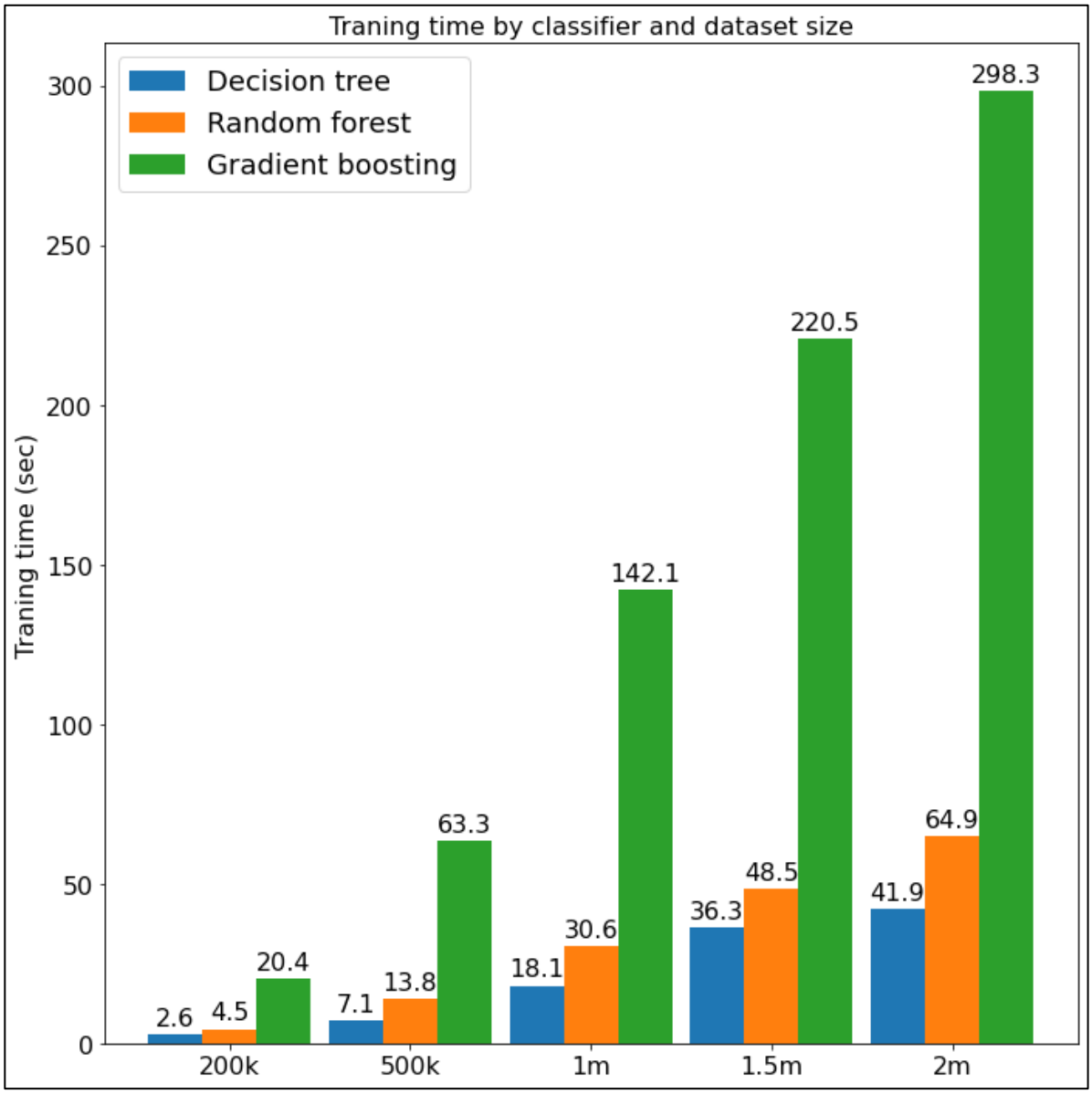

Рисунок 7.1 – Діаграма часу навчання в залежності від розміру вибірки

Якщо розглядати час навчання моделі як важливий параметр, то виходячи з отриманих у результатах експериментів даних (рис. 7.1) найбільш підходящим є алгоритм випадкового лісу, так як він має не дуже відрізняються показники в

порівнянні з деревом рішень при цьому забезпечує більш високі показники точності і менш схильний до перенавчання.

Моделі машинного навчання недостатньо розумні, щоб знати, які гіпер параметри призведуть до максимально можливої точності для даного набору даних[17].

Після налаштування гіпер параметрів всіх моделей, що використовуються в дослідженні, ми сформували підсумкову діаграму порівняння (рис. 7.2).

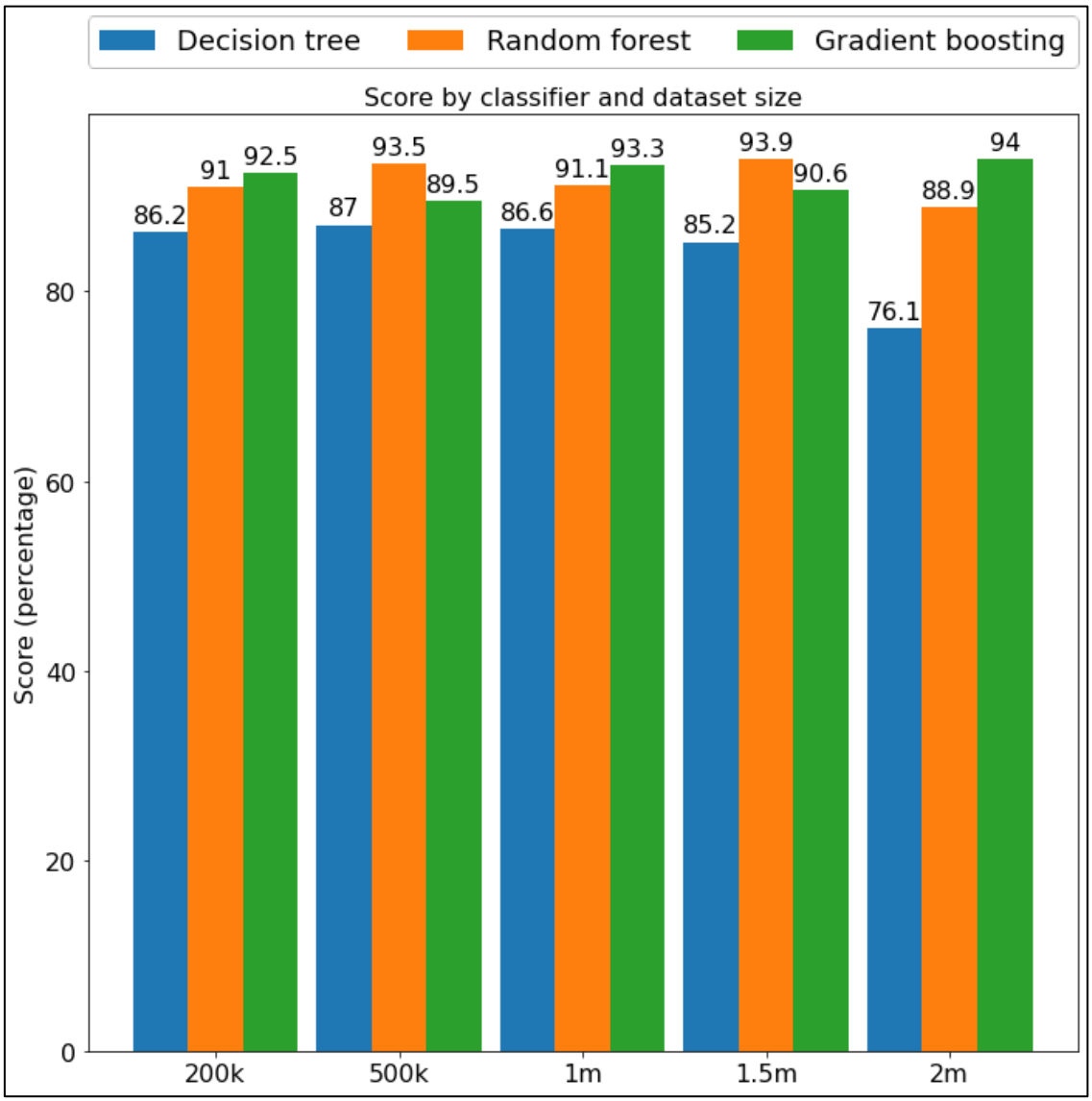

Рисунок 7.2 – Діаграма порівняння точності моделей в залежності від розміру вибірки

Значення гіпер параметрів, якщо вони встановлені правильно, можуть створювати високоточні моделі, і, таким чином, ми дозволяємо нашим моделям

пробувати різні комбінації гіпер параметрів під час процесу навчання та робити прогнози з найкращою комбінацією значень гіпер параметрів. Деякі з гіпер параметрів у класифікаторі випадковий ліс — це загальна кількість дерев у лісі, глибина кожного дерева в лісі та критерій[18].

З отриманих результатів можна сказати, що дерева рішень, як теоретично і передбачалося, виявилися більш схильними до перенавчання, а метод градієнтного бустингу показав найкращі результати зі збільшенням розміру навчальної вибірки.

8 РЕКОМЕНДАЦІЇ

Як і для всіх аналітичних методів, у методу дерева рішень існують обмеження, про які потрібно знати. Основним недоліком є те, що він може бути перенавчений, особливо при використанні невеликого набору даних. Ця проблема може обмежити узагальненість та надійність результуючих моделей. Інша потенційна проблема полягає в тому, що сильна кореляція між різними потенційними вхідними змінними може призвести до вибору змінних, які покращують статистику моделі, але не мають причинно-наслідкового зв'язку з результатом. Таким чином, слід бути обережними при інтерпретації моделей дерева рішень і при використанні результатів цих моделей для розробки причинно-наслідкових гіпотез.

Перехресна перевірка для машинного навчання важлива, оскільки вона допомагає оцінити, чи точна модель у реальному середовищі з новими динамічними даними. Кількість доступних навчальних даних часто обмежена[19]. Для належної підготовки високоякісних маркованих наборів даних потрібні час і ресурси. Можна очікувати, що модель, навчена навчальним даним в автономному середовищі, може знизитися в точності при розгортанні в новому живому середовищі. Перехресна перевірка — це процес тестування моделі з новими даними, щоб оцінити точність прогнозування з невидимими даними.

Тому перехресна перевірка є важливим кроком у процесі розробки моделі машинного навчання. Техніка є корисним методом для зміщення відбору в даних навчання. Процес оптимізації машинного навчання означає, що алгоритм буде навчений, щоб бути максимально ефективним із навчальними даними. Однак модель може бути надмірно оптимізована або переповнена навчальним даним. Це означає, що модель буде менш точною під час впливу живого середовища, яке не контролюється так ретельно, як середовище для навчання. Перехресна перевірка – це спосіб висвітлити та виміряти цю проблему за допомогою узагальнення. Методика допоможе організаціям оцінити рівень помилки між моделлю, яка обробляє навчальні дані, і невидимими даними.

Оцінка точності моделі машинного навчання є важливим кроком перед розгортанням[20]. Модель починає приносити користь після розгортання, тому забезпечення ефективного функціонування алгоритму є життєво важливим. Хоча модель може досягти високої точності на даних навчання, це не гарантується з реальними даними в динамічній обстановці. Після розгортання модель може бути неефективною з новими даними, якщо її точність не була перехресно перевірена. Результати процесу перехресної перевірки можна використовувати для сигналізації про подальшу оптимізацію моделі машинного навчання перед розгортанням.

ВИСНОВКИ

В даній кваліфікаційній роботі було проведено дослідження методів прогнозування результатів спортивних матчів на відкритому наборі даних матчів з тенісу за останні 50 років.

Для проведення експериментального дослідження використовувався датасет з ключовими полями, на основі яких була побудована математична модель. За допомогою цієї моделі було проведено навчання трьома методами.

В роботі було виявлено найбільш влучний метод для вирішення поставленої задачі. А для цього було визначено метрики та критерії оцінювання результатів, отриманих емпіричним шляхом на прикладі описаних методів.

Для демонстрація отриманих результатів був створений прототип системи аналізу спортивних матчів.

Так як головною задачею кваліфікаційної роботи є аналіз існуючих класифікаційних методів прогнозування, що підходять для передбачення результатів матчів та визначення найбільш підходящого методу на основі проведених експериментів, то в роботі було виконано наступне:

- реалізація методу дерева рішень;
- реалізація методу випадкового лісу;
- реалізація методу градієнтного бустингу;
- проведення навчання за обраним датасетом методом дерева рішень;
- проведення навчання за обраним датасетом методом випадкового лісу;
- проведення навчання за обраним датасетом методом градієнтого бустингу;
- аналіз отриманих в експериментах результати;
- визначення кращого методу для поставленої задачі;
- зроблено висновки та дано подальші рекомендації щодо використання;

В результаті проведеної роботи було отримано результати на прикладі проведених експериментів. Результати було порівняно за описаними в роботі метриками. Було виявлено кращий метод для прогнозування результатів матчів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. У. Тімоті Голви. Теніс. Психологія успішної гри. 2016 − 208 с.

2. Gammastack [Електронний ресурс] – Режим доступу до ресурсу: https://www.gammastack.com (дата звернення: 19.04.2022).

3. Nacsport [Електронний ресурс] – Режим доступу до ресурсу: https://www.nacsport.com (дата звернення: 19.04.2022).

4. Л. Белл, М. Брантон-Сполл, Р. Смит, Дж. Бэрд. Безпека розробки в Agile-проектах. Liters. 4 вересня 2019 – 439 с.

5. Елбон Кріс. Машинне навчання з використанням Python. БЧВ-Петербург. 2020 – 384 с.

6. Аміт Курмар Тьягі. Data Science та Data Analytics. 22 вересня 2021 – 482 с.

7. Павлов Ю.Л. Random Forests. Walter de Gruyter GmbH & Co KG – 14 січня 2019 – 122 с.

8. K. Smelyakov, D. Karachevtsev, D. Kulemza, Y. Samoilenko, O. Patlan. Effectiveness of preprocessing algorithms for natural language processing applications. IEEE - 2020. p. 187-191.

9. K. Smelyakov, A. Chupryna, D. Sandrkin, M. Kolisnyk. Search by Image Engine for Big Data Warehouse. IEEE – 2020. P. 1-4.

10. K. Smelyakov, A. Chupryna, O. Bohomolov, N. Hunko. The Neural Network Models Effectiveness for Face Detection and Face Recognition. IEEE – 2021. p. 1-7

11. Corey Wade. Hands-On Gradient Boosting with XGBoost and scikit-learn. Packt Publishing Ltd – Oct 16, 2020 – 310 p.

12. David M. Beazley. Python Essential Reference. Addison-Wesley Professional – 2009 – 717 p.

13. А. Богачов. Графіки, котрі переконують всіх. Litres – Квітень 21, 2020 – 370 с.

14. D. J. Hand and N. M. Adams, “Data Mining,” in Wiley StatsRef: Statistics Reference Online, Chichester, UK: John Wiley & Sons, Ltd, 2015, pp. 1–7.

15. I. Karakatsanis et al., “Data mining approach to monitoring the requirements of the job market: A case study,” Inf. Syst., vol. 65, pp. 1–6, Apr. 2017.

16. A. Mazidi, F. Roshanfar, and V. Parvin Darabad, “A Review of Outliers: Towards a Novel Fuzzy Method for Outlier Detection ,” J. Appl. Dyn. Syst. Control, vol. 2, no. 1, pp. 7–17, Jun. 2019.

17. A. Mazidi and F. Roshanfar, “PSPGA: A New Method for Protein Structure Prediction based on Genetic Algorithm,” J. Appl. Dyn. Syst. Control, vol. 3, no. 1, pp. 9–16, Jun. 2020.

18. A. Mazidi, M. H. Saddredini, and H. Tahayori, “Proposing A New Algorithmto Detect Local Outliersin Data Stream,” vol. 4, no. 4. Journal of A. Mazidi and F. Roshanfar, “PSPGA: A New Method for Protein Structure Prediction based on Genetic Algorithm,” J. Appl. Dyn. Syst. Control, vol. 3, no. 1, pp. 9–16, Jun. 2020.

19. A. Mazidi, M. H. Saddredini, and H. Tahayori, “ProposingA New Algorithm to Detect Local Outliersin Data Stream,” vol. 4, no. 4. Journal of Soft Computingand Information Technology (JSCIT), pp. 31–42, 01-Jan-2016.Soft Computingand Information Technology (JSCIT), pp. 31–42. 01, Jan, 2016.

20. N. Etminan, E. Parvinnia, and A. Sharifi-Zarchi, “FAME: Fast And Memory Efficient multiple sequences alignment tool through compatible chain of roots,” Bioinformatics, 2020.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

8. K. Smelyakov, D. Karachevtsev, D. Kulemza, Y. Samoilenko, O. Patlan. Effectiveness of preprocessing algorithms for natural language processing applications. IEEE - 2020. p. 187-191.

9. K. Smelyakov, A. Chupryna, D. Sandrkin, M. Kolisnyk. Search by Image Engine for Big Data Warehouse. IEEE – 2020. P. 1-4.

10. K. Smelyakov, A. Chupryna, O. Bohomolov, N. Hunko. The Neural Network Models Effectiveness for Face Detection and Face Recognition. IEEE – 2021. p. 1-7