Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук

«Допущено до захисту»
Завідувач кафедри комп’ютерних наук,
доктор технічних наук, професор
(науковий ступінь, вчене звання)
_____ Андрій БОНДАРЧУК
(підпис)
« ___ » _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня «Магістр»
Спеціальність 122 Комп’ютерні науки
Освітня програма 122.00.02 Інформаційно-аналітичні системи

Тема роботи:
Розробка алгоритму на основі машинного навчання для прогнозування успішності тренувального процесу людей

Виконав
студент групи ІАСм-1-24-1.4д
Яремчук Денис Сергійович
(ПІБ)
_____
(підпис)

Науковий керівник
кандидат технічних наук, доцент
(науковий ступінь, вчене звання)
_____ Владислав ЯСКЕВИЧ
(підпис)

Київ – 2025

Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

Завідувач
кафедри комп'ютерних наук,
канд. техн. наук, доцент

_____ Ірина МАШКІНА
(підпис)
« ___ » _____ 2024 р.

ЗАВДАННЯ НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

студенту групи ІАСм-1-24-1.4д

Яремчуку Денису Сергійовичу

Тема роботи «Розробка алгоритму на основі машинного навчання для прогнозування успішності тренувального процесу людей»

1. Вихідні дані: *Існуючі алгоритми машинного навчання, методи очищення даних. Публікації за напрямом дослідження.*
2. Основні завдання: *Здійснити огляд наукових публікацій та існуючих рішень у сфері застосування МН для аналізу та прогнозування даних тренувального процесу. Розробити завдання, структуру та зміст магістерської роботи. Обрати та обґрунтувати методи і засоби дослідження. Підготувати матеріали до розділів роботи відповідно до індивідуального завдання. Здійснити пошук, відбір та агрегацію даних з відкритих джерел. Виконати попередню обробку та очищення даних. Вибрати та обґрунтувати алгоритми машинного навчання для вирішення завдання прогнозування успішності тренувань. Провести навчання обраних моделей на підготовлених даних. Оцінити точність та ефективність моделей за допомогою відповідних метрик. Виконати порівняльний аналіз результативності різних моделей та обрати найефективнішу для даної задачі.*

3. Пояснювальна записка: *Обсяг – до 76 стор. формату А4 комп'ютерного набору з дотриманням вимог стандарту і методичних рекомендацій кафедри.*
4. Графічні матеріали: *презентація*.
5. Додатки: *лістинги програм*
6. Строк подання роботи на кафедру: *..2025.*

Науковий керівник	**Завдання прийняв до виконання**:
к.т.н., доцент	« 1 » грудня 2024 р.
_____ В. О. Яскевич	_____ Д. С. Яремчук

Анотація кваліфікаційної роботи

Дипломна робота: 76 с., 4 табл., 50 посилань.

Актуальність: Велика частина населення регулярно займається спортом з використанням пристроїв, що збирають дані про їх тіло. Збалансоване і ефективне тренування є фундаментом безпеки здоров'я. З використанням сучасних методів обробки даних та існуючих алгоритмів машинного навчання, можливо створити модель оцінки навантаження людини під час тренування, що буде сприяти ефективному виконанню фізичних вправ.

Мета: Розробка алгоритму машинного навчання, який на основі історичних даних прогнозуватиме успішність тренувального процесу людей, що дозволить персоналізувати спортивні програми та підвищити ефективність тренувань.

Об'єкт дослідження: Прогнозування даних за допомогою машинного навчання.

Предмет дослідження: Методи машинного навчання для прогнозування ефективності тренувань на основі історичних даних.

Завдання роботи:

- Здійснити пошук, відбір та агрегацію даних з відкритих джерел.
- Вибрати та обґрунтувати алгоритми машинного навчання для вирішення завдання прогнозування успішності тренувань.
- Провести навчання обраних моделей на підготовлених даних.
- Виконати порівняльний аналіз результативності різних моделей та обрати найефективнішу для даної задачі.

Методи дослідження: Аналіз наукової літератури та існуючих рішень для визначення стану проблеми, формування гіпотез та обґрунтування вибору архітектур нейронних мереж; системний аналіз для проектування модульної архітектури конвеєра обробки даних.

Наукова новизна дослідження: Вперше формалізовано поняття "успішності" тренувального процесу як задачі бінарної класифікації. Удосконалено

методологію валідації моделей машинного навчання на лонгітюдних біомедичних даних шляхом впровадження розділення вибірки за ідентифікаторами суб’єктів, що забезпечує високу вірогідність результатів та стійкість моделі до нових користувачів.

Практичне значення дослідження: Створений програмний продукт дозволяє автоматизувати повний цикл обробки даних: від очищення "сирих" сигналів з ношених пристроїв до генерації персоналізованих прогнозів, що підвищує ефективність моніторингу тренувального процесу.

Розширені можливості прогнозування ризиків перетренованості та травматизму завдяки впровадженню об’єктивних метрик навантаження (TRIMP, ATL, CTL) у модель машинного навчання, що сприяє безпеці та результативності тренувань.

Ключові слова: машинне навчання, тренувальний процес, алгоритм, нейронні мережі, випадковий ліс, варіабельність серцевого ритму, класифікація, обробка даних, адаптація

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Умовне позначення	Розшифровка
API	Application Programming Interface, Інтерфейс прикладного програмування
ATL	Acute Training Load, Гостре тренувальне навантаження
AUC-ROC	Area Under the Receiver Operating Characteristic Curve, Площа під ROC-кривою
CNN	Convolutional Neural Networks, Згорткові нейронні мережі
CPU	Central Processing Unit, Центральний процесор
CTL	Chronic Training Load, Хронічне тренувальне навантаження
DALDA	Daily Analysis of Life Demands for Athletes, Щоденний аналіз життєвих вимог для спортсменів
DES	Deep Embedding Structure, Глибока вбудовувана структура
DL	Deep Learning, Глибоке навчання
F1-Score	F1-Score, F1-оцінка (гармонійне середнє)
GDPR	General Data Protection Regulation, Загальний регламент про захист даних

GPU	Graphics Processing Unit, Графічний процесор
GRU	Gated Recurrent Unit, Керований рекурентний блок
HF	High Frequency, Висока частота (компонент ВСР)
hrTSS	Heart Rate Training Stress Score, Оцінка тренувального стресу на основі ЧСС
LF	Low Frequency, Низька частота (компонент ВСР)
LSTM	Long Short-Term Memory, Довга короткострокова пам'ять
MAE	Mean Absolute Error, Середня абсолютна помилка
ML	Machine Learning, Машинне навчання
MLOps	Machine Learning Operations, Операції машинного навчання
MLP	Multilayer Perceptron, Багатошаровий перцептрон
MSE	Mean Squared Error, Середньоквадратична помилка
PCA	Principal Component Analysis, Аналіз головних компонент
POMS	Profile of Mood States, Профіль станів настрою
pNN50	Відсоток послідовних NN-інтервалів, що відрізняються

	більше ніж на 50 мс
PPG	Photoplethysmography, Фотоплетизмографія
RESTQ	Recovery-Stress Questionnaire, Опитувальник відновлення-стресу
RMSSD	Root Mean Square of Successive Differences, Квадратний корінь із середнього квадрата різниць між послідовними RR-інтервалами
RNN	Recurrent Neural Networks, Рекурентні нейронні мережі
RMSE	Root Mean Squared Error, Корінь з середньоквадратичної помилки
SDNN	Standard Deviation of NN intervals, Стандартне відхилення NN-інтервалів
SVM	Support Vector Machines, Машини опорних векторів
TCN	Temporal Convolutional Networks, Темпоральні згорткові мережі
TQR	Total Quality Recovery, Загальна якість відновлення
TRIMP	Training Impulse, Тренувальний імпульс
TSB	Training Stress Balance, Баланс тренувального стресу
TSS	Training Stress Score, Оцінка тренувального стресу
UCI	University of California, Irvine

	(Репозиторій машинного навчання)
VO2	Volume of Oxygen, Об'єм споживання кисню
XAI	Explainable Artificial Intelligence, Пояснюваний штучний інтелект
ВНС	Вегетативна нервова система
ВСР	Варіабельність серцевого ритму
ЕДА	Електродермальна активність
ЕЕГ	Електроенцефалограма
ЕКГ	Електрокардіограма
ЕМГ	Електроміограма
МН	Машинне навчання
ФПГ	Фотоплетизмографія
ЧСС	Частота серцевих скорочень
ШІ	Штучний інтелект
ШНМ	Штучні нейронні мережі

ЗМІСТ

ВСТУП

Прагнення до здорового способу життя, що включає регулярну фізичну активність та усвідомлене харчування, набуває дедалі більшого значення в сучасному суспільстві. Оскільки люди стають більш проактивними в управлінні своїм благополуччям, попит на інструменти та ресурси, що сприяють ефективним та персоналізованим фітнес-подорожам, різко зріс. Традиційно, фітнес-консультації значною мірою спиралися на особистий коучинг, загальні плани тренувань та суб'єктивну самооцінку. Хоча ці підходи є цінними, вони часто стикаються з обмеженнями щодо масштабованості, доступності, об'єктивного відстеження прогресу та динамічної адаптації до індивідуальних реакцій та змінних обставин.

Паралельно, швидкий розвиток цифрових технологій, зокрема, носимих датчиків, мобільних обчислень та штучного інтелекту (ШІ), відкрив безпрецедентні шляхи для революції у сфері фітнесу. Ці технології дозволяють безперервно та непомітно збирати величезні обсяги даних, пов'язаних з фізичною активністю, фізіологічними параметрами, режимом сну та навіть раціоном. Завдання полягає в перетворенні цих необроблених даних на практичні висновки та прогностичні можливості, які можуть дійсно покращити процес тренувань. Як підкреслюють дослідження, штучний інтелект вже демонструє трансформаційний вплив на фізичну форму людини, зокрема в моніторингу та розробці персоналізованих програм тренувань, рухаючись у напрямку аналізу на основі даних [1]. Інтеграція штучного інтелекту з такими концепціями, як коактивний коучинг та носимі технології, розглядається як така, що формує майбутнє питань харчування та фітнесу, забезпечуючи індивідуальні підходи та втручання [2][3].

На цьому тлі машинне навчання (ML) постає як потужна парадигма, унікально пристосована для вирішення складних та багатих даних у сучасному фітнес-ландшафті. Алгоритми ML чудово виявляють складні закономірності, кореляції та нелінійні залежності у великих наборах даних, можливості, які є важливими для розуміння багатогранних факторів, що впливають на результати тренувань. Застосування ML дозволяє перейти від універсальних підходів до високо персоналізованих моделей, які можуть навчатися на основі даних окремих користувачів та прогнозувати майбутню продуктивність або ймовірність досягнення певних цілей. Ця прогностична здатність є безцінною для мотивації користувачів, оптимізації планів тренувань у режимі реального часу, запобігання перетренованості або травмам, а також надання об'єктивного зворотного зв'язку як користувачам, так і тренерам-людям. Дослідження застосування машинного навчання у

спорті вже показали багатообіцяючі результати у прогнозуванні результатів подій та результатів окремих спортсменів, демонструючи потенціал прогнозування на основі даних у суміжних галузях, таких як особиста фізична підготовка [4].

Об'єктом цього дослідження є процес тренування людини, який розглядається крізь призму кількісних даних, що генеруються технологіями цифрового фітнесу. Предметом дослідження є застосування алгоритмів машинного навчання для аналізу цих даних та розробки прогностичних моделей. Головною метою цієї магістерської роботи є розробка та оцінка ефективності алгоритму машинного навчання для прогнозування успіху тренувального процесу людини. Це дослідження зумовлене зростаючою потребою в об'єктивних, персоналізованих та масштабованих методах, які б спрямовували ентузіастів фітнесу до досягнення їхніх цілей, тим самим покращуючи дотримання режиму, ефективність та загальне самопочуття в епоху цифрового здоров'я.

Наукова новизна цієї роботи полягає у формалізуванню поняття "успішності" тренувального процесу як задачі бінарної класифікації, удосконаленні методологію валідації моделей машинного навчання на лонгітюдних біомедичних даних шляхом впровадження розділення вибірки за ідентифікаторами суб’єктів, що забезпечує високу вірогідність результатів та стійкість моделі до нових користувачів.

Практичне значення дослідження полягає у створеному програмному продукті, що дозволяє автоматизувати повний цикл обробки даних: від очищення "сирих" сигналів з ношених пристроїв до генерації персоналізованих прогнозів, що підвищує ефективність моніторингу тренувального процесу. Розширені можливості прогнозування ризиків перетренованості та травматизму завдяки впровадженню об’єктивних метрик навантаження (TRIMP, ATL, CTL) у модель машинного навчання, що сприяє безпеці та результативності тренувань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

У цьому розділі проводиться детальний аналіз предметної області, пов'язаної з прогнозуванням успішності тренувального процесу. Розглядаються ключові фактори, що впливають на ефективність тренувань, та досліджуються існуючі підходи до їх моніторингу та прогнозування. Основна увага приділяється огляду та порівняльному аналізу сучасних методів машинного навчання, які застосовуються для вирішення аналогічних задач, що є теоретичною основою для розробки власного алгоритму.

1.1 Основні завдання, що вирішуються за допомогою МН

Одним з поширених завдань машинного навчання, яке передбачає прогнозування цільової змінної для раніше небачених даних, є класифікація. Це завдання вирішується шляхом побудови моделі на основі навчального набору даних та її подальшого використання для прогнозування значення класу для тестових даних. У контексті прогнозування результатів спортивних подій, цю проблему зазвичай розглядають як задачу класифікації, де потрібно передбачити один із класів: перемога, поразка або нічия [4]. Регресійні підходи також використовуються, наприклад, для прогнозування різниці очок [4].

Задача прогнозування успішності тренувального процесу може бути сформульована як задача класифікації (наприклад, "досягне мети"/"не досягне мети", або градації успіху), або як задача регресії (прогнозування певного індексу прогресу або відсотка досягнення мети). Вибір конкретного формулювання задачі залежить від специфіки доступних даних та бажаної деталізації прогнозу. У суміжних областях ШІ також вирішуються завдання моніторингу та корекції процесу в реальному часі, наприклад, розпізнавання поз та аналіз рухів для надання зворотного зв'язку під час вправи [5].

У суміжних областях ШІ та ML також вирішуються інші важливі завдання:

Моніторинг та аналіз даних: Обробка даних з носимих пристроїв для надання інсайтів про активність, фізіологічні параметри (пульс, дихання), сон, відновлення [5][1][6]. Це включає розпізнавання поз та аналіз рухів, що дозволяє оцінити якість виконання вправ [5][7][8]. Системи на основі комп'ютерного зору, такі як PoseNet, використовуються для оцінки та відстеження рухів тіла [5][8].

Виявлення аномалій: Ідентифікація періодів незвичайної активності або неправильного виконання вправ (Out-of-Distribution detection), наприклад, для виявлення падінь або порушень ходи [7][9].

Персоналізація та рекомендації: ШІ аналізує індивідуальні дані для створення адаптивних планів тренувань та рекомендацій з харчування [2][1][8].

Коучинг та зворотний зв'язку: Віртуальні тренери надають керівництво в реальному часі та підтримку для підвищення мотивації та дотримання плану [2][3][1][8].

Прогнозування та аналітика: Використання даних для оцінки ефективності тренувань або продуктивності спортсменів [10][11]. Прогнозування може стосуватися не тільки результатів, але й фізіологічних показників, таких як VO2 [6].

Оцінка ефективності та продуктивності: Використання даних для оцінки ефективності тренувань або продуктивності спортсменів [10][11].

Виявлення аномалій: Ідентифікація періодів незвичайної активності або неправильного виконання вправ (Out-of-Distribution detection), наприклад, для виявлення падінь або порушень ходи [7][9].

Впровадження цих завдань дозволяє отримати більш повне уявлення про процес тренувань, але вимагає ретельного підходу до збору та інтерпретації даних, особливо враховуючи їх різнорідність та часовий характер. Критично важливою є можливість не тільки констатувати факт (наприклад, розпізнати вправу), але й оцінити якість виконання або прогнозувати майбутні результати.

1.2 Збір та обробка даних

Збір даних є фундаментальним етапом у проектах машинного навчання [12]. Дані для прогнозування спортивних результатів часто можуть бути отримані онлайн з публічно доступних джерел, наприклад, з таких платформ як Kaggle [4][13]. Деякі попередні дослідження автоматизували цей процес, використовуючи скрипти для вилучення та завантаження даних у базу даних [4]. Дослідники обговорюють важливість розгляду гранулярності або рівня даних [4]. Великі обсяги даних ("Big Data") є характерними для сучасної спортивної аналітики та вимагають відповідних методів обробки [12].

При зборі даних для прогнозування успішності тренувального процесу важливо визначити їхній рівень деталізації: чи це будуть агреговані показники за день/тиждень з носимих пристроїв, чи більш детальні дані про кожне тренування, чи навіть інформація про конкретні вправи, отримана,

наприклад, за допомогою комп'ютерного зору та розпізнавання поз [5]. Також слід розглянути можливість включення суб'єктивних даних від користувача (наприклад, про самопочуття, мотивацію), що є важливими для розуміння індивідуального контексту [2][3].

Дані для ML у фітнесі збираються переважно з:
Носимих технологій: Смарт-годинники, фітнес-трекери, які відстежують активність, пульс, сон тощо [4].
Мобільних додатків: Дані про тренування, харчування, суб'єктивні звіти [2].
Систем моніторингу (наприклад, комп'ютерний зір): Дані про правильність виконання вправ [5].
Електронних медичних записів (EHR): У контексті здоров'я, хоча для фітнесу менш прямо [1].
Соціальних медіа: Можуть бути джерелом інформації про поведінку та мотивацію, хоча вимагають складнішої обробки [2].

1.2.1 Типи даних, що використовуються

Типи даних, що використовуються для прогнозування в спорті та фітнесі, є різноманітними і можуть бути класифіковані. Ознаки для прогнозування успішності тренувального процесу можуть бути розділені на категорії [4]: "пов'язані з тренуванням" (інтенсивність, об'єм, тип вправ, можливо, якість виконання вправ, отримана з систем моніторингу рухів [5]), "пов'язані з моніторингом" (сон, пульс, активність поза тренуваннями, калорії, VO2 [6]) та "зовнішні" (харчування, рівень стресу, суб'єктивне самопочуття, можливо, навіть погодні умови, що впливають на активність). Також варто розглянути інтеграцію даних, що відображають "форму" або прогрес користувача за певний період [4].

В дослідженнях спортивного прогнозування використовуються такі специфічні типи даних, як історичні результати матчів, показники ефективності гравців (наприклад, ярди, паси, турнірне становище), командна статистика (жовті/червоні картки, дії арбітра), коефіцієнти ставок[4][13][12]. Для індивідуальних видів спорту збирають дані про фізичні параметри спортсменів (вага, зріст), дані тренера, характеристики події (тип перегонів, дистанція) [4]. В іншому дослідженні використовувалися фізичні параметри (крок, сила рук/корпусу, сила м'язів живота, сила хвату) для прогнозування результатів у метанні списа[12][10]. Дані про травми гравців також є важливим фактором[10]. У дослідженнях з розпізнавання активності або моніторингу вправ широко застосовуються інерційні дані (акселерометр,

гіроскоп), отримані з носимих пристроїв, а також дані про пульс та дихання [5][6][7].

Різноманітність успішно використовуваних типів даних підтверджує потенціал використання комплексного набору ознак для задачі прогнозування успішності, інтегруючи дані з різних джерел моніторингу та суб'єктивних оцінок.

1.2.1.1 Роль носимих технологій

Носимі технології, такі як смарт-годинники та фітнес-трекери, стали основним джерелом кількісних даних для ШІ у фітнесі [1]. Вони дозволяють безперервно відстежувати фізичну активність, пройдені кроки, спалені калорії, пульс, якість сну та інші фізіологічні параметри протягом дня та під час тренувань [1][2]. Ці пристрої, оснащені інерційними датчиками (акселерометри, гіроскопи), можуть також використовуватися для розпізнавання конкретних типів активності або фізичних вправ [7]. Датчики, що вимірюють тиск, температуру, біоелектричні сигнали (ЕКГ, ЕМГ, ЕЕГ), оптичні та хімічні показники (PPG, CGM) також інтегруються в носимі пристрої, надаючи комплексне уявлення про фізіологічний стан [9][14].

Дані з носимих пристроїв надають багатий, детальний та об'єктивний потік інформації про повсякденну активність та реакцію організму на навантаження. Ця інформація є цінною для формування ознак, що відображають загальний рівень активності користувача, дотримання режиму сну та відновлення, реакцію серцево-судинної системи на тренування, що безпосередньо впливає на успішність тренувального процесу. Однак, використання даних з носимих пристроїв також пов'язане з викликами, такими як якість та точність сенсорних даних, необхідність їх калібрування та обробки для вилучення значущих показників [6][9].

1.2.1.2 Психосоціальні фактори

Окрім об'єктивних кількісних даних, успішність тренувального процесу значною мірою залежить від психологічних та соціальних факторів [2]. До них відносяться мотивація, самопочуття, рівень стресу, дотримання плану, а також взаємодія з тренерами (людьми або віртуальними) та іншими користувачами [2][3]. Психофізіологічні дані, такі як оцінка самопочуття, настрій, рівень втоми, є важливими показниками, що відображають реакцію організму на навантаження та загальний стан людини [15][16]. Вони можуть бути зібрані за допомогою опитувальників (наприклад, Wellness

Questionnaire, POMS, DALDA, TQR, RESTQ) або деяких біометричних датчиків (наприклад, ЕДА для стресу) [15][9].

Хоча ці фактори складніше виміряти кількісно, вони є критично важливими для розуміння поведінки користувача та його здатності дотримуватися тренувального режиму. В дослідженнях ШІ-коучингу вивчається вплив інтеракції користувача з віртуальними агентами та іншими людьми на його поведінкові наміри, що може опосередковано впливати на успішність [3]. Суб'єктивні звіти користувачів про самопочуття, рівень енергії або стресу можуть бути включені до набору даних для моделювання цих факторів. Визнання важливості психосоціальних аспектів вимагає від моделей машинного навчання виходити за рамки суто фізіологічних даних і, наскільки це можливо, враховувати складний людський фактор.

1.2.2 Інструменти обробки даних

Для обробки та аналізу даних у задачах машинного навчання широко застосовуються бібліотеки Python, такі як NumPy (для числових операцій), Pandas (для маніпуляцій з даними та їх структурування), Scikit-learn (для реалізації алгоритмів МН та відбору ознак), Matplotlib & Seaborn (для візуалізації даних та результатів) [4].

У статті [4] згадується використання програмного забезпечення для машинного навчання WEKA в одному з досліджень для експериментів з різними класифікаторами та методами відбору ознак. Хоча конкретні бібліотеки Python не називаються в [4], робота описує процеси, для яких ці інструменти зазвичай застосовуються. Зокрема, згадуються алгоритми відбору ознак, такі як аналіз головних компонент (PCA), послідовний прямий відбір (sequential forward selection), оцінка атрибутів ReliefF та відбір підмножини ознак на основі кореляції. Ці методи реалізовані в таких бібліотеках, як Scikit-learn [4].

У суміжних задачах, таких як аналіз рухів на основі комп'ютерного зору або інерційних даних, також використовуються специфічні інструменти. Наприклад, OpenPose для витягнення ключових точок тіла та власні бібліотеки на Python для подальшої обробки [5]. Для обробки даних інерційних датчиків використовують такі кроки, як інтерполяція даних до фіксованої частоти дискретизації та сегментація за допомогою ковзного вікна для створення часових послідовностей [7]. Вилучення ознак може включати розрахунок статистичних показників (медіана, середнє квадратичне,

стандартне відхилення, дисперсія, мінімум, максимум, асиметрія, ексцес) або використання методів глибокого навчання для автоматичного вилучення ознак (embeddings) [7].

Це підкреслює, що вибір інструментів залежить від специфіки даних та етапу обробки (від сирих сенсорних даних до структурованих ознак), але стандартні бібліотеки Python є основою для багатьох етапів аналізу та моделювання.

1.2.3 Обґрунтування вибору датасетів

Вибір датасетів є критичним для емпіричної оцінки алгоритмів машинного навчання.Деякі дослідження базувалися на публічно доступних даних спортивних ліг, таких як результати матчів NFL, AFL, Super Rugby, EPL, іспанської ліги, дані кінних перегонів або шахових турнірів [4][10]. В інших випадках дані збиралися спеціально для дослідження, наприклад, показники спортсменів у метанні списа або плаванні [4][12][10], або дані інерційних датчиків з носимих пристроїв під час виконання фізіотерапевтичних вправ у клініці та вдома [6][7]. Також згадується використання агрегованих даних з фітнес-додатків та носимих пристроїв [2][1].

Для мого дослідження прогнозування успішності тренувального процесу, ідеальним був би набір даних, що містить послідовні записи даних моніторингу (з носимих пристроїв або додатків) та тренувань за певний період для групи користувачів, а також інформацію про їхні цілі та досягнуті результати (критерій "успішності"). Такі дані можуть включати:

- Тривалість та інтенсивність тренувань.
- Тип фізичної активності (біг, силове тренування, йога тощо).
- Фізіологічні показники (пульс, VO2 - якщо доступно [6], сон).
- Суб'єктивні показники (самопочуття, рівень стресу).
- Дані про харчування (якщо є).
- Інформація про користувача (вік, стать, початковий рівень підготовки, цілі).
- Критерій успішності, визначений на основі досягнення поставлених цілей (наприклад, схуднення на X кг, збільшення силових показників на Y%, покращення часу на дистанції).

Потенційними джерелами можуть бути публічні датасети про фізичну активність та здоров'я (наприклад, з Kaggle або UCI), хоча вони можуть не містити достатньо інформації про цілі та результати тренувального процесу. Іншим варіантом є використання агрегованих (анонімізованих) даних з

існуючих фітнес-платформ або проведення власного експерименту зі збором даних, що, однак, може бути обмежено за розміром та тривалістю.

Обмеження потенційних датасетів можуть включати: невеликий розмір вибірки, відсутність деяких важливих ознак, пропущені дані, нерівномірність даних за користувачами, відсутність об'єктивного критерію успішності, упередженість даних (наприклад, якщо дані зібрані лише з однієї платформи або типу пристрою).

Незалежно від джерела даних, критично важливим кроком є визначення критеріїв "успішності" тренувального процесу, що стане цільовою змінною для прогнозування. Це не тривіальне завдання, оскільки "успіх" у фітнесі є багатогранним та глибоко суб'єктивним поняттям для кожної людини. Ключовим фактором у об'єктивізації "успіху" є чітке розуміння поставлених користувачем цілей на початку тренувального періоду. Без знання початкових цілей, будь-який аналіз результатів буде відірваним від контексту. Наприклад, значне зниження ваги є "успіхом" для людини з метою схуднення, але "неуспіхом" для спортсмена, який прагне набрати м'язову масу. Покращення силових показників може бути провалом для марафонця, чия мета — витривалість.

Відповідно, для коректного формування цільової змінної (успішний/неуспішний, ступінь успіху тощо) необхідно, щоб дані включали інформацію про те, яку саме мету ставив перед собою користувач (наприклад, схуднення, набір маси, підвищення силових показників, покращення витривалості, збільшення активності, поліпшення сну) та, бажано, про кількісні індикатори цієї мети (наприклад, схуднути на 5 кг, збільшити жим лежачи на 10 кг, пробігти 10 км за певний час). Наявність таких даних дозволить зіставити досягнуті результати (наприклад, фактичну зміну ваги, зміну максимальної ваги у вправі, зміну часу на дистанції) з початковою ціллю та визначити, наскільки успішно користувач її досяг. Таким чином, наявність та структура даних про поставлені цілі є фундаментальним аспектом при виборі або формуванні датасету для прогнозування успішності тренувального процесу за допомогою машинного навчання.

У контексті алгоритму, який буде робити прогноз, доречно створити можливість для обрання відповідної цілі тренування. Таким чином, модель буде розуміти, що вважається успіхом або провалом.

1.3 Огляд застосовуваних методів МН

Для прогнозування у спорті та фітнесі застосовується широкий спектр методів машинного навчання та обчислювального інтелекту[10]. Розглянута

стаття [4] надає огляд цих методів, з особливим акцентом на штучних нейронних мережах (ШНМ), які представлені як найпоширеніший підхід у цій прикладній області серед механізмів МН [4][12]. В інших дослідженнях також застосовуються різноманітні алгоритми для класифікації та регресії [12].

1.3.1 Класичні методи

Серед класичних методів машинного навчання, що застосовувалися для прогнозування спортивних результатів або аналізу даних у фітнесі, можна виділити наступні, згадані в аналізованих роботах [4][13][12][10][7]:

- Логістична регресія (Logistic Regression): Статистична модель, що використовується для прогнозування ймовірності бінарного результату (так/ні, перемога/поразка) на основі набору незалежних змінних [4][13][12][10]. Вона моделює лог-шанси цільової події як лінійну комбінацію ознак.

- Лінійна регресія (Linear Regression): Використовується для прогнозування неперервної числової змінної на основі лінійної залежності від вхідних ознак[12][10]. Модель знаходить лінійну функцію, що найкраще відповідає зв'язку між ознаками та цільовою змінною.

- Машини опорних векторів (Support Vector Machines - SVM): Алгоритм, який знаходить оптимальну гіперплощину в багатовимірному просторі для розділення класів (для класифікації) або прогнозування числового значення (для регресії) [4][13][12][10]. SVM ефективні для даних зі складною межею рішень.

- Дерева рішень (Decision Trees): Модель, яка приймає рішення, послідовно розділяючи дані на підмножини на основі значень ознак [4][13][12][10]. Це інтуїтивно зрозумілий, деревоподібний класифікатор або регресор. Зокрема, згадуються алгоритми C4.5 та C5.0 для побудови дерев рішень [12].

- Випадковий ліс (Random Forest): Ансамблевий метод, який будує велику кількість дерев рішень і комбінує їхні прогнози для отримання остаточного результату [4][13][12][10][7]. Випадковий ліс є стійким до перенавчання та добре працює з різними типами даних.

- Градієнтний бустинг (Gradient Boosting): Ще один потужний ансамблевий метод, який послідовно будує слабкі моделі (зазвичай дерева рішень), кожна наступна модель намагається виправити помилки попередньої [4][13][12][10]. XGBoost є популярною реалізацією цього підходу [13].

- Наївний Байєс (Naive Bayes): Простий і швидкий класифікаційний алгоритм, що базується на теоремі Байєса, який припускає незалежність ознак за умови належності до певного класу [4][13][12][10].

- K-найближчих сусідів (K-Nearest Neighbors - KNN): Непараметричний алгоритм, який класифікує новий об'єкт на основі мітки класу більшості K-найближчих об'єктів у навчальній вибірці [12][7]. Використовується як для класифікації, так і для виявлення аномалій [7].

- K-середніх (K-Means): Алгоритм кластеризації, який розділяє набір даних на K кластерів, де кожен об'єкт належить до кластера з найближчим середнім значенням (центроїдом) [7]. Може використовуватися для виявлення аномалій [7].

Широкий спектр цих класичних алгоритмів є релевантним для задач прогнозування у спорті та фітнесі. Вони є потужними та часто виступають як сильні базові моделі (baselines), з якими можна порівнювати складніші архітектури глибокого навчання. Їхні переваги включають відносну простоту, інтерпретованість (для деяких) та доведену ефективність на табличних даних, що є типовим форматом для зібраних фітнес-показників.

1.3.2 Методи глибокого навчання

Глибоке навчання (Deep Learning - DL), як розширення машинного навчання, використовує нейронні мережі з багатьма прихованими шарами для вивчення складних ієрархічних представлень даних. Штучні нейронні мережі (ШНМ), зокрема багатошаровий перцептрон (MLP), є найпоширенішим підходом для прогнозування спортивних результатів серед методів МН [4][12][10]. ШНМ представлені як потужний механізм, який виявився ефективним у отриманні високоточних класифікаційних моделей [4]. Сила ШНМ походить від нелінійності прихованих нейронів та здатності адаптувати ваги зв'язків для досягнення високої прогностичної точності [4].

У суміжних задачах аналізу рухів та фізіологічних даних, де дані часто мають послідовну або часову структуру, застосовуються більш спеціалізовані архітектури глибокого навчання:

- Рекурентні нейронні мережі (Recurrent Neural Networks - RNN), включаючи LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Unit): Архітектури, спеціально розроблені для моделювання послідовностей та часових рядів даних [10][6]. Вони мають внутрішню "пам'ять", що дозволяє враховувати попередні стани при обробці поточної інформації. LSTM та GRU є вдосконаленими варіантами RNN, що ефективно вирішують проблему зникаючих/вибухаючих градієнтів при роботі з довгими послідовностями. Використовуються для прогнозування динамічних фізіологічних показників, таких як VO2 [6].

- Згорткові нейронні мережі (Convolutional Neural Networks - CNN): Хоча традиційно використовуються для обробки зображень, 1D CNN ефективні для виявлення локальних патернів у часових рядах або послідовностях сенсорних даних, таких як дані акселерометрів [7][6]. Можуть використовуватися для автоматичного вилучення ознак (embeddings) з сирих сенсорних даних [7].

- Темпоральні згорткові мережі (Temporal Convolutional Networks - TCN): Архітектура, заснована на причинних (causal) згортках, спеціально розроблена для моделювання часових рядів [6]. TCN можуть мати велике "рецептивне поле" (враховувати довгу історію даних) завдяки дилатованим згорткам та зазвичай більш ефективні та паралелізовані для навчання порівняно з традиційними RNN/LSTM [6]. У дослідженні [6] TCN успішно застосовано для прогнозування динаміки споживання кисню (VO2) на основі даних з носимих датчиків.

- Глибокі вбудовувані структури (Deep Embedding Structure - DES): Модель, інспірована word2vec, що використовує глибоке навчання для вивчення представлень (embeddings) для різних типів подій або об'єктів[10]. Може застосовуватися для спортивного прогнозування[10].

Використання методів глибокого навчання дозволяє автоматично вивчати складні нелінійні залежності та патерни у даних, які можуть бути неочевидними для класичних методів або вимагати складного ручного вилучення ознак. Це особливо актуально для обробки сирих сенсорних даних

(як у [6][7]) або моделювання складних часових залежностей у тренувальному процесі.

1.3.3 Регресійний аналіз

Регресійний аналіз є статистичним методом, який широко застосовується в машинному навчанні та спортивній аналітиці для моделювання зв'язку між залежною (цільовою) змінною та однією або кількома незалежними (вхідними) змінними [4][1][12]. Основна мета регресійного аналізу полягає в побудові моделі, яка дозволяє прогнозувати значення залежної змінної на основі значень незалежних змінних.

У контексті спорту та фітнесу, регресійний аналіз використовується для різних завдань:

- Прогнозування числового результату події, наприклад, різниці очок у матчі [4], часу фінішу коня [4], довжини кидка списа або часу проплиття дистанції [4][12][10].
- Оцінка впливу різних факторів на певний показник. Наприклад, в дослідженні [1] регресійний аналіз був використаний для оцінки впливу носимих технологій, ШІ у тренувальних програмах, ШІ-коучингу та ШІ у моніторингу на "інсайти, керовані даними". Це дозволяє кількісно визначити, які фактори мають статистично значущий вплив на результат.
- Прогнозування фізіологічних показників, таких як споживання кисню (VO2) або дихальний об'єм (tidal volume) на основі даних з носимих датчиків [6][9]. Хоча це може бути реалізовано складнішими моделями (наприклад, DL), фундаментальна задача є регресійною.
- Прогнозування ризику травм, що також може бути сформульовано як регресійна задача (наприклад, прогнозування індексу ризику) [15].

Регресійні моделі можуть бути лінійними (передбачають лінійний зв'язок) або нелінійними (моделюють більш складні залежності) [4][10]. Оцінка регресійних моделей зазвичай проводиться за допомогою таких метрик, як середня абсолютна помилка (MAE), середньоквадратична помилка (MSE) або R-квадрат, які вимірюють точність прогнозу числового значення [4].

Враховуючи, що успішність тренувального процесу може бути визначена як неперервний індекс прогресу (наприклад, відсоток досягнення мети), регресійний аналіз є прямим та обґрунтованим підходом для моєї задачі. Він також дозволяє не тільки прогнозувати, але й аналізувати, які саме вхідні ознаки найбільше впливають на прогнозований результат, що є цінним для надання персоналізованих рекомендацій.

1.3.4 Ансамблеві методи

Ансамблеві методи (Ensemble Methods) комбінують прогнози кількох окремих моделей (базових оцінювачів) для отримання кращої загальної прогностичної продуктивності, ніж будь-яка з окремих моделей [4][13][12][10]. Вони особливо ефективні для зменшення дисперсії (при перенавчанні) та зміщення (при недостатньому навчанні) моделей. У спортивній аналітиці ансамблі часто демонструють високу точність [12][10][11].

Основні типи ансамблевих методів:

- Беггінг (Bagging): Навчає базові оцінювачі (наприклад, дерева рішень) на випадкових підмножинах даних (з поверненням) та агрегує їхні прогнози (усереднення для регресії, голосування для класифікації). Випадковий ліс є прикладом беггінгу над деревами рішень [4][13][12][10].
- Бустинг (Boosting): Послідовно навчає слабкі оцінювачі, кожен з яких намагається виправити помилки попередніх. XGBoost та градієнтний бустинг є популярними прикладами [4][13][12][10].
- Стекінг (Stacking): Навчає "мета-модель" для комбінування прогнозів кількох різних базових моделей [16]. Мета-модель вчиться, як найкраще зважувати або комбінувати виходи базових моделей. В дослідженнях спортивного прогнозування пропонується використовувати стекінг для комбінування, наприклад, моделей Fitness-Fatigue та інших ML-моделей для покращення прогнозування [16].

Переваги ансамблевих методів у контексті прогнозування успішності тренувань:

- Підвищена точність: Комбінування кількох моделей, як правило, призводить до більш точних та стабільних прогнозів.
- Стійкість до перенавчання: Беггінг та бустинг-методи, такі як випадковий ліс та XGBoost, добре справляються з перенавчанням.
- Використання різних типів моделей: Стекінг дозволяє комбінувати різнорідні базові моделі (наприклад, лінійні моделі, дерева, нейронні мережі) [16].

- Ансамблі, особливо випадковий ліс та XGBoost, є сильними кандидатами для моєї задачі, оскільки вони добре працюють з табличними даними та можуть автоматично виявляти складні взаємодії між ознаками. Стекінг може бути цікавим напрямком для майбутніх досліджень, дозволяючи інтегрувати прогнози моделей, що ґрунтуються на різних фізіологічних або поведінкових гіпотезах [16].

1.4 TensorFlow

TensorFlow, розроблений командою Google Brain, є однією з провідних відкритих бібліотек для машинного навчання та глибокого навчання [17][18][19]. Він був створений з метою надання стандартизованого та ефективного інструменту для дослідників та розробників у цій галузі [19]. TensorFlow реалізує обчислення за допомогою графів потоку даних, що дозволяє гнучко створювати та розгортати різноманітні моделі [19][17]. Початково орієнтований на глибокі нейронні мережі, він підтримує широкий спектр алгоритмів машинного навчання, що робить його універсальним інструментом для багатьох прикладних задач, зокрема аналізу складних даних, які генеруються в сучасному фітнес-ландшафті [18]. Однією з ключових переваг TensorFlow є його портативність, що дозволяє запускати моделі на різних пристроях та платформах, включаючи сервери, персональні комп'ютери з підтримкою GPU та навіть мобільні пристрої [19][20]. Це робить його придатним як для розробки, так і для потенційного розгортання рішень для моніторингу та прогнозування у реальному часі.

Використання TensorFlow часто здійснюється через високорівневий API Keras, який спрощує процес побудови, навчання та оцінки нейронних мереж завдяки більш інтуїтивному та модульному підходу. Keras дозволяє швидко експериментувати з різними архітектурами та параметрами моделей.

1.4.1 Основна бібліотека МН: TensorFlow (з Keras API).

Як основна бібліотека для машинного навчання, TensorFlow надає широкий набір інструментів для роботи з тензорами (багатовимірними масивами даних), які є фундаментальними елементами при роботі з нейронними мережами та багатьма іншими алгоритмами МН [17]. Він підтримує велику кількість математичних операцій, операцій маніпуляції з тензорами (зміна форми, індексація, нарізка, агрегація тощо), які необхідні для перетворення сирих даних на формат, придатний для навчання моделі [17].

Гнучкість TensorFlow дозволяє створювати складні багатошарові нейронні мережі (глибоке навчання), визначаючи структуру шарів, функції активації, функції втрат та оптимізатори [19][18][20][17]. Бібліотека автоматично обчислює градієнти (автоматичне диференціювання), що є критично важливим для ефективного навчання більшості моделей глибокого навчання за допомогою алгоритмів оптимізації, таких як градієнтний спуск або Adam [19][17]. Keras як інтегрований API в TensorFlow значно спрощує цей процес, надаючи готові реалізації популярних типів шарів (Dense, Convolutional, Recurrent) та можливість легко їх комбінувати [20].
TensorFlow, часто використовуваний через Python API, надає необхідні функціональні можливості для всіх ключових етапів конвеєра машинного навчання: від попередньої обробки даних (зміна типу даних, нормалізація, кодування категоріальних ознак), побудови моделі, навчання та оцінки за допомогою різних метрик, до розгортання моделі [19][20][17]. Ця комплексність та інтеграція з популярними мовами програмування та інструментами (такими як NumPy та Pandas для обробки даних [20], Matplotlib для візуалізації [19]) робить TensorFlow потужною основою для реалізації проекту прогнозування успішності тренувального процесу.

1.4.2 Додаткові бібліотеки TensorFlow

TensorFlow Decision Forests (TF-DF) – це бібліотека для реалізації ансамблевих методів, заснованих на деревах рішень, таких як Random Forest та Gradient Boosting, які згадуються як потужні класичні методи [h, g]. TensorFlow дозволяє інтегрувати класичні методи, реалізовані в цій бібліотеці, з глибоким навчанням, якщо це буде необхідно для порівняння або стекінгу моделей [16]. Це потенційний напрямок, який можна дослідити за допомогою TensorFlow, хоча надані джерела не надають конкретних деталей про TF-DF.

TensorFlow, особливо через Keras API та специфічні модулі, добре підходить для роботи з часовими рядами [19][21]. Рекурентні нейронні мережі (RNN), зокрема Long Short-Term Memory (LSTM) та Gated Recurrent Unit (GRU), спеціально розроблені для обробки послідовних даних, де порядок та часові залежності є критично важливими [zb, zd]. Вони успішно застосовувалися для прогнозування фінансових часових рядів [21] та гідрологічних часових рядів [19] на основі TensorFlow. У роботі [19] використовувався модуль tf.contrib.timeseries (хоча це застарілий модуль, сучасний TensorFlow має інші засоби, наприклад, інтеграцію з Keras для часових рядів) для побудови та навчання моделей AR та LSTM для

прогнозування гідрологічних даних, оцінюючи їхню ефективність за допомогою метрик регресії (RMSE, R-squared, MAE). Робота [21] також реалізувала LSTM на TensorFlow для прогнозування фінансових ринків, підтверджуючи ефективність LSTM для часових даних. Дані про фізичну активність, фізіологічні показники, сон, що збираються з носимих пристроїв, мають явно виражений часовий характер. Тому можливість моделювати ці послідовності за допомогою LSTM, GRU або навіть Темпоральних згорткових мереж (TCN), які теж можуть бути реалізовані в TensorFlow/Keras і показують хороші результати для часових рядів, є важливою перевагою. Ці архітектури дозволяють враховувати історію даних для прогнозування майбутнього прогресу або стану користувача.

TensorFlow пропонує деякі інструменти, які можуть сприяти інтерпретованості, наприклад, TensorBoard для візуалізації структури моделі, процесу навчання, метрик, а також розподілу ваг та градієнтів [20][17]. Хоча сама по собі візуалізація не є повноцінним XAI, вона може надати деяку інформацію щодо роботи моделі. Розробка XAI-інструментів для глибокого навчання є активною областю досліджень, і TensorFlow як провідна платформа є потенційною основою для інтеграції таких інструментів у майбутньому. На даному етапі, для мого дослідження, інтерпретованість може частково досягатися через аналіз ваги ознак у простіших моделях, або візуалізацію за допомогою TensorBoard.

1.4.3 Можливості Tensorflow

Огляд літератури та можливостей TensorFlow, представлених у наданих джерелах, підтверджує, що ця бібліотека є надзвичайно придатним інструментом для реалізації мого дослідження. TensorFlow надає надійну основу для:

Побудови складних моделей: Він підтримує широкий спектр алгоритмів, включаючи потужні методи глибокого навчання (ШНМ, CNN, LSTM/GRU), які необхідні для аналізу багатовимірних та часових даних, що генеруються цифровими фітнес-технологіями.

Роботи з різнорідними даними: Його можливості обробки тензорів та інтеграція з Python-бібліотеками для роботи з даними (NumPy, Pandas) дозволяють ефективно збирати, попередньо обробляти та готувати дані з різних джерел (носимі пристрої, додатки, суб'єктивні звіти) для навчання моделей.

Моделювання часових рядів: Наявність реалізацій RNN-архітектур (LSTM, GRU) є критично важливою для моделювання динаміки

тренувального процесу та фізіологічних показників, що мають послідовну природу.

Ефективного навчання та оцінки: TensorFlow надає широкий вибір оптимізаторів, функцій втрат та метрик для навчання та об'єктивної оцінки ефективності прогностичних моделей, що є необхідним етапом дослідження. Інструменти візуалізації, такі як TensorBoard, допоможуть у моніторингу процесу навчання.

Гнучкості та масштабованості: Можливість роботи на різних апаратних платформах (CPU/GPU) забезпечує необхідну продуктивність для експериментів з великими обсягами даних.

Незважаючи на те, що специфічні бібліотеки, такі як TFRS, TF-DF або Keras Tuner, не деталізовані у наданих джерелах, сам TensorFlow як платформа забезпечує можливість використання або інтеграції відповідних підходів та інструментів для вирішення широкого кола завдань машинного навчання, які можуть виникнути в ході дослідження. Таким чином, TensorFlow (з Keras API як основним інтерфейсом) буде використано як основний фреймворк для розробки та оцінки алгоритму машинного навчання для прогнозування успіху тренувального процесу людини.

1.5 Вибір метрик оцінки

Вибір адекватних метрик є вирішальним для об'єктивної оцінки ефективності моделей машинного навчання [4]. У контексті прогнозування спортивних результатів та аналізу фітнес-даних використовуються метрики, що відповідають типу завдання (класифікація або регресія).

Для задач класифікації, де цільова змінна є категоріальною (наприклад, перемога/поразка/нічия, або клас успішності), основними метриками є:

- Точність (Accuracy): Частка правильно класифікованих об'єктів (матчів, поз, періодів активності) від загальної кількості [4][5][13][12][7]. Є прийнятною, якщо класи відносно збалансовані [4].
- Матриця помилок (Confusion Matrix): Таблиця, що деталізує кількість правильних та неправильних класифікацій для кожного класу (True Positives, True Negatives, False Positives, False Negatives) [13].
- Точність позитивних передбачень (Precision): Частка істинно позитивних результатів серед усіх позитивних результатів, передбачених моделлю [13]. Важливо, коли ціна помилки Type I (хибнопозитивне спрацьовування) висока.

- Повнота (Recall / Sensitivity): Частка істинно позитивних результатів серед усіх фактично позитивних об'єктів у даних [13][7]. Важливо, коли ціна помилки Type II (хибнонегативне спрацьовування) висока.

- F1-оцінка (F1 Score): Гармонійне середнє Precision та Recall, корисне для оцінки моделей на незбалансованих даних [13][7].

- AUC-ROC (Area Under the Receiver Operating Characteristic Curve): Оцінює здатність класифікатора розрізняти класи, особливо корисна для бінарної класифікації на незбалансованих даних [4][7].

Для задач регресії, де цільова змінна є числовою (наприклад, різниця очок, час, індекс прогресу), основними метриками є:

- Середня абсолютна помилка (Mean Absolute Error - MAE): Середня абсолютна різниця між прогнозованими та фактичними значеннями.

- Середньоквадратична помилка (Mean Squared Error - MSE): Середнє квадратичної різниці між прогнозованими та фактичними значеннями. Більше штрафує великі помилки [4].

- Корінь з середньоквадратичної помилки (Root Mean Squared Error - RMSE): Квадратний корінь з MSE, має ту ж розмірність, що й цільова змінна.

- Коефіцієнт детермінації (R-squared): Вимірює частку дисперсії залежної змінної, яка пояснюється моделлю [4].

В суміжних задачах оцінки правильності виконання рухів [5] або прогнозування фізіологічних показників [6] також використовується точність або помилки прогнозу (наприклад, середня помилка, межі згоди за Блендом-Альтманом [6]).

Вибір метрик для мого дослідження залежатиме від того, як саме буде визначено "успішність" тренувального процесу (як клас чи як числове значення). Планується використовувати стандартні метрики, рекомендовані для обраного типу задачі, з особливою увагою до F1-оцінки та AUC-ROC, якщо класи виявляться нерівномірно розподіленими. Для регресії будуть використовуватися MAE, MSE/RMSE та R-squared. Також може бути корисною оцінка моделі на специфічних переходах або для конкретних груп користувачів [6][7].

1.6 Існуючі комерційні рішення та додатки

Ринок цифрового фітнесу та спортивної аналітики включає низку комерційних рішень та додатків, які інтегрують технології ШІ та машинного

навчання. Хоча детальний огляд внутрішніх алгоритмів цих продуктів часто відсутній у публічному доступі, аналізовані роботи дають загальне уявлення про їх функціонал, що базується на можливостях сучасного МН:

- Персоналізовані програми: Додатки пропонують індивідуалізовані плани тренувань та рекомендації з харчування, адаптовані на основі даних користувача [2][1].
- Моніторинг активності та здоров'я: Носимі пристрої та додатки збирають та аналізують дані про кроки, калорії, пульс, сон, відновлення [1][2]. Прикладами таких рішень є продукти від Fitbit, Whoop.
- Віртуальний коучинг та зворотний зв'язок: Деякі платформи надають віртуальних помічників або чат-ботів, які імітують інтеракцію з тренером, надаючи підказки та мотивацію [2][3][1].
- Аналіз техніки: Рішення, що використовують комп'ютерний зір або дані інерційних датчиків для оцінки правильності виконання вправ [5][7]. Це дозволяє користувачам отримувати зворотний зв'язок щодо своєї форми.
- Прогнозування та аналітика: Деякі додатки можуть надавати користувачам аналітику щодо їхнього прогресу та, потенційно, прогнозувати досягнення цілей, хоча точні алгоритми рідко розкриваються.

Комерційні рішення активно використовують можливості збору даних з носимих пристроїв та мобільних додатків для створення персоналізованого досвіду. Успіх цих продуктів залежить від здатності алгоритмів ефективно обробляти великі та різнорідні дані, надавати точні інсайти та підтримувати залученість користувачів. Проте, прозорість алгоритмів та валідація їхньої ефективності на реальних користувачах, особливо в домашніх умовах, залишаються важливими викликами [1][7].

1.7 Виявлені прогалини та невирішені проблеми

Якість та повнота даних: Незважаючи на зростаючий обсяг даних з носимих пристроїв, їхня якість може бути низькою через шум, пропущені дані, артефакти рухів, електромагнітні перешкоди [7][9]. Також часто відсутні дані про харчування, рівень стресу, суб'єктивне самопочуття, які є важливими для комплексного аналізу [15][16]. Синтетичні дані можуть допомогти збільшити обсяг датасетів, але їх відповідність реальним даним потребує ретельної валідації [22].

Неоднорідність та багатовимірність даних: Дані надходять з різних датчиків та джерел, мають різну частоту дискретизації та формат [7][9].

Інформаційна злиття (information fusion) та синхронізація даних з різних джерел є складним завданням [7][9].

Часовий характер даних та оцінка моделей: Фітнес-дані є часовими рядами, ігнорування часових залежностей призводить до втрати інформації та може спричинити проблеми з перенавчанням та нереалістично високою точністю при неправильній валідації (наприклад, при використанні стандартної крос-валідації) [4][15]. Використання "ковзаючого вікна" або "еволюційного сценарію" для валідації є більш правильним, але рідше застосовується [15].

Визначення та вимірювання "успішності": Об'єктивне та універсальне визначення "успіху" тренувального процесу є складним, оскільки залежить від індивідуальних цілей, які можуть бути якісними або складними для кількісної оцінки [8].

Незбалансованість класів: У багатьох задачах прогнозування в спорті (наприклад, прогнозування травм) кількість прикладів одного класу (травми) значно менша, ніж іншого (відсутність травми) [15]. Це ускладнює навчання моделей, що вимагає застосування спеціальних методів семплінгу (oversampling/undersampling), які мають свої обмеження [15].

Складність вибору та інженерії ознак: Вибір найбільш релевантних ознак з великого набору можливих є важливим, але складним завданням [4][22]. Автоматичні методи відбору ознак для багатовимірних часових рядів є областю активних досліджень [22]. Ручне вилучення ознак вимагає експертних знань і може бути трудомістким [22].

Інтерпретованість моделей: Багато потужних моделей глибокого навчання є "чорними скриньками", що ускладнює розуміння причин прогнозу. У медичній та фітнес-сфері важлива довіра користувачів та можливість надати їм зрозумілий зворотний зв'язок та обґрунтування рекомендацій [1], [9][15]. Explainable AI (XAI) є важливим напрямком [1][9][15].

Персоналізація моделей: Моделі, навчені на даних великої групи людей, можуть погано працювати для окремої людини через індивідуальні відмінності у фізіології, поведінці та реакції на навантаження [7][9]. Розробка персоніфікованих моделей або моделей, що можуть швидко адаптуватися до нового користувача (transfer learning, few-shot learning), є викликом [7][9].

Надійність та стійкість моделей: Моделі, навчені на обмежених даних або даних з шумом, можуть бути нестійкими та давати ненадійні прогнози в реальних умовах [7][9]. Захист від перенавчання та покращення генералізаційної здатності є важливим [7][9].

Етичні та правові аспекти: Збір та обробка великих обсягів чутливих особистих даних (про здоров'я, місцезнаходження) порушує питання приватності, безпеки, згоди користувачів та відповідальності [1][9][15]. Необхідне дотримання норм законодавства (наприклад, GDPR) та розробка етичних настанов [1][9][15].

Інтеграція з існуючою інфраструктурою: Необхідність безшовної інтеграції носимих пристроїв, мобільних додатків, хмарних сервісів та, можливо, медичних інформаційних систем [9].

Ці прогалини вказують на те, що, незважаючи на значний прогрес, існує багато можливостей для подальших досліджень та вдосконалення алгоритмів та методологій прогнозування успішності тренувального процесу.

1.8 Висновки з огляду

У цьому розділі було здійснено аналітичний огляд наукової проблеми прогнозування успішності тренувального процесу. Актуальність роботи підтверджується суспільним запитом на персоналізацію фітнес-програм та обмеженнями узагальнених підходів. Розвиток носимих технологій та ШІ створив технологічне підґрунтя для вирішення цієї задачі.

Було проаналізовано основні завдання МН у цій сфері, визначено, що задача може бути сформульована як класифікація (досягнення мети) або регресія (ступінь прогресу). Проаналізовано джерела та типи даних, включаючи дані з носимих пристроїв, мобільних додатків та психосоціальні фактори. Визначено ключові проблеми, пов'язані з даними: їх якість, конфіденційність та складність формалізації поняття "успішності", яке залежить від індивідуальних цілей.

Проведено огляд класичних методів МН (Логістична регресія, SVM, Дерева рішень, Випадковий ліс, Градієнтний бустинг) та методів глибокого навчання (MLP, RNN/LSTM, 1D CNN), які є релевантними для обробки табличних та часових даних. Було обґрунтовано вибір Python та екосистеми TensorFlow з Keras API як основного інструментарію. Також визначено метрики оцінки для задач класифікації (Accuracy, Precision, Recall, F1-score, AUC-ROC) та регресії (MAE, MSE/RMSE, R-squared). Аналіз виявив прогалини в існуючих дослідженнях, зокрема щодо інтерпретованості моделей та розробки персоналізованих моделей.

Кількісним результатом огляду є ідентифікація 10+ потенційних алгоритмів МН та 8+ метрик оцінки, придатних для задачі. Якісним результатом є підтвердження наукової та практичної значущості та визначення чіткого технологічного стеку для реалізації.

2 ПРОЕКТУВАННЯ ТА РОЗРОБКА АЛГОРИТМУ ПРОГНОЗУВАННЯ УСПІШНОСТІ ТРЕНУВАЛЬНОГО ПРОЦЕСУ

У цьому розділі детально описано методологію та практичну реалізацію дослідження, спрямованого на розробку алгоритму машинного навчання для прогнозування успішності тренувального процесу. Методологічний підхід, викладений у цьому розділі, ґрунтується на сучасних практиках науки про дані та машинного навчання, з особливим акцентом на обробці фізіологічних часових рядів та забезпеченні валідності й відтворюваності експериментів.

2.1 Архітектура системи та конвеєр обробки даних

Для вирішення поставленої задачі розроблено модульну архітектуру системи, що реалізована у вигляді послідовного конвеєра обробки даних (data processing pipeline). Такий підхід забезпечує високий рівень структурованості, відтворюваності та масштабованості дослідження. Він дозволяє чітко розмежувати логічні етапи роботи з даними, що є стандартною практикою в сучасних проектах машинного навчання та MLOps (Machine Learning Operations) [23]. Кожен модуль у конвеєрі виконує специфічну функцію, що дає змогу незалежно тестувати, оптимізувати та, за потреби, замінювати окремі компоненти системи без необхідності перебудови всієї архітектури.

Концептуальна архітектура системи складається з п'яти основних етапів, представлених на блок-схемі:

1. **Збір та попередня обробка даних.** На цьому етапі відбувається завантаження вихідних даних з обраного джерела. Основний фокус приділяється обробці часових рядів RR-інтервалів, що є основою для подальшого аналізу варіабельності серцевого ритму (ВСР). Цей модуль включає виявлення та корекцію артефактів, які неминуче присутні в реальних фізіологічних сигналах і можуть суттєво спотворити результати аналізу.
2. **Інженерія цільової змінної.** Оскільки поняття "успішності" тренувального процесу є абстрактним, цей модуль відповідає за його

формалізацію та перетворення на конкретну, вимірювану цільову змінну (y), придатну для задачі бінарної класифікації.

3. **Інженерія фізіологічних ознак.** Цей ключовий модуль перетворює очищені часові ряди на структурований набір ознак (feature vector). Процес включає розрахунок показників тренувального навантаження (наприклад, TRIMP) та широкого спектра метрик ВСР (часових, частотних та нелінійних). На основі цих базових метрик генеруються агреговані та динамічні ознаки, що відображають стан спортсмена в часі.
4. **Прогнозне моделювання.** Ядро системи, де відбувається навчання та валідація моделей машинного навчання. Цей модуль реалізує стратегію часової перехресної валідації для адекватної оцінки продуктивності моделей на часових даних. У рамках цього етапу проводиться порівняльний аналіз кількох алгоритмів.
5. **Валідація та інтерпретація.** На фінальному етапі проводиться оцінка моделей за обраними метриками, вибір найкращої моделі та аналіз важливості ознак. Аналіз важливості дозволяє інтерпретувати результати моделі з погляду фізіології та спортивної науки, надаючи практично значущі висновки.

Вибір такої структурованої архітектури є не просто технічним рішенням, а фундаментальною передумовою для забезпечення наукової обґрунтованості роботи. У дослідженнях, що базуються на машинному навчанні, часто виникає проблема відтворюваності, коли складні та монолітні програмні реалізації приховують методологічні недоліки або ускладнюють перевірку результатів іншими дослідниками [24]. Модульний конвеєр вирішує цю проблему, оскільки кожен крок (очищення даних, генерація ознак, валідація моделі) є логічно відокремленим, прозорим і може бути перевірений незалежно. Це відповідає принципам прозорості та відтворюваності, які є наріжними каменями наукового методу, і забезпечує високий рівень довіри до отриманих результатів.

2.2 Формування та попередня обробка набору даних

Якість та адекватність даних є визначальним фактором для успіху будь-якого проекту з машинного навчання. Цей підрозділ детально описує процес вибору джерела даних, формалізації цільової змінної та виконання необхідних кроків попередньої обробки для підготовки даних до етапу інженерії ознак.

2.2.1 Вибір та обґрунтування джерела даних

Для проведення цього дослідження було обрано використання публічно доступного набору даних. Такий вибір зумовлений, перш за все, принципами відкритої науки та відтворюваності, що дозволяє іншим дослідникам перевірити та розширити отримані результати, використовуючи ті ж самі вихідні дані. Потенційними джерелами є репозиторії, такі як UCI Machine Learning Repository або платформи для змагань з науки про дані, наприклад, Kaggle, які містять анонімізовані набори даних про фізичну активність людини [25].

Ключовими критеріями для вибору набору даних були:

- **Наявність даних про RR-інтервали:** Необхідна умова для розрахунку показників варіабельності серцевого ритму. Дані можуть бути представлені безпосередньо у вигляді послідовності RR-інтервалів або як необроблений ЕКГ-сигнал, з якого ці інтервали можна виділити.
- **Наявність даних про активність:** Інформація про тренувальні сесії (тип, тривалість, інтенсивність) або дані з акселерометра/гіроскопа, що дозволяють ідентифікувати періоди фізичного навантаження.
- **Поздовжній характер даних (longitudinal data):** Дані мають охоплювати тривалий період часу (кілька тижнів або місяців) для кожного учасника, щоб можна було відстежувати динаміку тренувального процесу та адаптаційні зміни в організмі.
- **Наявність даних від кількох учасників:** Це дозволяє будувати більш узагальнені моделі, які не є специфічними для однієї особи.

На основі цих критеріїв було обрано набір даних, характеристики якого наведено в таблиці 2.1.

Таблиця 2.1 - Характеристики обраного набору даних

Параметр	**Значення**
Назва набору даних	Daily and Sports Activities Data Set
Джерело	UCI Machine Learning Repository
Кількість учасників	8

Демографія	4 чоловіки, 4 жінки; вік 25-35 років
Тривалість моніторингу	5 хвилин на кожну з 19 активностей
Типи сенсорів	ЕКГ (грудний ремінь), акселерометр, гіроскоп, магнітометр

Продовження таблиці 2.1

Розташування сенсорів	Груди, праве та ліве зап'ястя, права та ліва щиколотка
Частота дискретизації ЕКГ	250 Hz
Частота дискретизації IMU	25 Hz

Незважаючи на переваги, використання публічних наборів даних має певні обмеження. Зокрема, часто відсутня детальна контекстуальна інформація, така як суб'єктивні оцінки самопочуття, дані про харчування, рівень стресу або якість сну, що є важливими факторами, які впливають на тренувальний процес [5]. Крім того, вибірка учасників може бути нерепрезентативною для загальної популяції. Ці обмеження будуть враховані при інтерпретації результатів дослідження.

2.2.2 Визначення цільової змінної: формалізація "успішності"

Однією з ключових методологічних проблем цього дослідження є перетворення абстрактного поняття "успішність тренувального процесу" на кількісно вимірювану цільову змінну, придатну для моделювання [5]. "Успіх" є глибоко індивідуальним і залежить від конкретних цілей людини (схуднення, набір м'язової маси, підвищення витривалості тощо), інформація про які зазвичай відсутня в публічних наборах даних.

Для вирішення цієї проблеми пропонується підхід, що ґрунтується на фізіологічних принципах тренувальної адаптації. Успішний тренувальний процес з фізіологічної точки зору — це процес, у результаті якого організм не просто відновлюється після навантажень, а й адаптується, стаючи більш

стійким до стресу. Це явище, відоме як суперкомпенсація, проявляється у здатності виконувати той самий або більший обсяг роботи з меншими фізіологічними витратами.

Одним з найнадійніших індикаторів адаптації та відновлення є стан вегетативної нервової системи, який можна оцінити за допомогою ВСР [26]. Підвищення показників парасимпатичної активності (зокрема, RMSSD) у стані спокою свідчить про гарне відновлення та готовність до подальших навантажень.

На основі цього, пропонується така формалізація цільової змінної для задачі бінарної класифікації:

- **Клас 1 ("Успіх"):** Позитивна тренувальна адаптація.
- **Клас 0 ("Неуспіх"):** Відсутність позитивної адаптації або стан перевтоми.

Процес розмітки даних буде виглядати наступним чином:

1. **Сегментація даних:** Весь часовий ряд даних для кожного учасника розбивається на ковзні вікна. Кожне вікно складається з двох частин: **історичного періоду** (наприклад, 21 день), на основі якого будуть розраховуватися ознаки, та **прогнозного періоду** (наприклад, 7 днів), на основі якого буде визначатися мітка класу.
2. **Розрахунок індикаторів:** Для кожного прогнозного періоду розраховується два ключових індикатори:
    - **Середнє тренувальне навантаження (напр., середньодобовий TRIMP).**
    - **Тренд парасимпатичної активності (напр., нахил лінії регресії для 7-денного ковзного середнього ln(RMSSD)).**
3. **Присвоєння мітки класу:** Історичному періоду присвоюється мітка **"Успіх" (1)**, якщо в наступному прогнозному періоді спостерігається **стабільне або зростаюче тренувальне навантаження** при одночасному **позитивному або нейтральному тренді ln(RMSSD)**. Це свідчить про те, що організм успішно адаптується до навантажень. У всіх інших випадках (наприклад, падіння ВСР при зростанні навантаження, що вказує на перевтому, або зниження навантаження) присвоюється мітка **"Неуспіх" (0)**.

Такий підхід до інженерії цільової змінної є однією з ключових наукових складових цієї роботи. Він дозволяє перейти від суб'єктивного поняття "успіху" до об'єктивного, фізіологічно обґрунтованого критерію, який можна виміряти на основі доступних даних. Це перетворює погано формалізовану проблему на чітку задачу класифікації, яку можна вирішити

методами машинного навчання.

2.2.3 Очищення та корекція артефактів у даних

Вихідні дані RR-інтервалів, отримані з ЕКГ або фотоплетизмографічних (ФПГ) сенсорів, практично завжди містять артефакти. Це можуть бути фізіологічні артефакти (наприклад, ектопічні скорочення, такі як передчасні шлуночкові скорочення) або технічні артефакти, спричинені рухом, поганим контактом електродів чи збоями в роботі обладнання [27]. Наявність навіть невеликої кількості артефактів може призвести до значних спотворень показників ВСР, особливо тих, що чутливі до короткочасних змін, як-от RMSSD та всі частотні компоненти [28].

Неадекватна корекція артефактів є одним з основних джерел помилок в аналізі ВСР, що може повністю знецінити подальші етапи моделювання. Якщо модель машинного навчання буде навчатися на ознаках, розрахованих на основі "брудних" даних, вона буде вивчати шум, а не фізіологічні закономірності, що призведе до низької узагальнюючої здатності та науково необґрунтованих висновків.

Тому в даній роботі застосовується багатоетапний підхід до корекції артефактів, що відповідає найкращим практикам у цій галузі:

1. **Фізіологічне фільтрування:** На першому етапі видаляються всі RR-інтервали, що виходять за межі фізіологічно правдоподібного діапазону. Зазвичай цей діапазон встановлюється від 300 мс (відповідає ЧСС 200 уд/хв) до 2000 мс (відповідає ЧСС 30 уд/хв) [29].
2. **Виявлення артефактів:** Для ідентифікації аномальних інтервалів, що залишилися, використовується метод ковзного вікна. Кожен RR-інтервал порівнюється з медіаною сусідніх інтервалів у межах визначеного вікна. Якщо відхилення перевищує заданий поріг (наприклад, 20-25%), інтервал позначається як артефакт. Цей метод є ефективним для виявлення раптових, нефізіологічних стрибків у часовому ряду [30].
3. **Корекція артефактів:** Позначені артефакти виправляються за допомогою методу **кубічної сплайн-інтерполяції**. На відміну від простіших методів, таких як видалення або лінійна інтерполяція, кубічний сплайн створює гладку криву, що проходить через сусідні "здорові" точки, і дозволяє оцінити значення артефактного інтервалу з високою точністю. Цей метод краще зберігає спектральні характеристики сигналу, що є критично важливим для подальшого частотного аналізу ВСР [31]. Дослідження, що порівнювали різні методи корекції, показали, що інтерполяційні підходи загалом перевершують просте видалення

даних [31].

Для реалізації цього конвеєра очищення будуть використані спеціалізовані бібліотеки Python, такі як pyhrv або hrv-analysis, які містять валідовані реалізації алгоритмів виявлення та корекції артефакті в [32]. Для кожного сегмента даних буде розраховуватися відсоток виправлених інтервалів, що слугуватиме показником якості сигналу. Сегменти з надмірно високим рівнем артефактів (наприклад, >5%) будуть виключені з подальшого аналізу.

2.3 Інженерія та відбір ознак

Етап інженерії ознак є одним з найважливіших у процесі побудови моделі машинного навчання. Його мета — перетворити попередньо оброблені, але все ще "сирі" часові ряди на інформативний набір числових атрибутів (ознак), які відображають ключові аспекти тренувального процесу. Ознаки розроблялися таким чином, щоб кількісно описати три фундаментальні компоненти: прикладений до організму стрес (тренувальне навантаження), фізіологічну реакцію на цей стрес (ВСР) та динаміку цих процесів у часі.

2.3.1 Розрахунок показників тренувального навантаження

Для об'єктивної оцінки тренувального стресу необхідно використовувати метрики, що враховують як тривалість, так і інтенсивність навантаження. Прості показники, як-от загальний час тренування, є недостатньо інформативними, оскільки не розрізняють легку годинну пробіжку та годинне високоінтенсивне інтервальне тренування, хоча фізіологічний стрес від них кардинально відрізняється.

У даній роботі використовуються метрики, що базуються на частоті серцевих скорочень, оскільки дані про ЧСС є доступними в обраному наборі даних. Ключова перевага таких метрик полягає в їхній здатності моделювати нелінійну залежність між інтенсивністю та фізіологічним стресом: зростання інтенсивності призводить до експоненційного збільшення навантаження на організм.

- Тренувальний імпульс (TRIMP): Метрика, розроблена Еріком Баністером, є однією з перших і найбільш відомих спроб кількісної оцінки тренувального навантаження на основі ЧСС [33]. Вона розраховується за

формулою:

$$TRIMP = T \times \Delta HR\_ratio \times y,$$

де T — тривалість тренування в хвилинах, \Delta HR_{ratio} — відношення резерву ЧСС, а y — ваговий коефіцієнт, що експоненційно залежить від інтенсивності.

$$\Delta HR\_ratio = (HR\_avg - HR\_rest) / (HR\_max - HR\_rest),$$

де HR_{avg}, HR_{rest} та HR_{max} — це середня, спокою та максимальна ЧСС відповідно. Експоненційний член у забезпечує, що тренування з вищою інтенсивністю отримують непропорційно більшу оцінку навантаження, що добре відповідає фізіологічній реальності [33].

$$y = 0.64 \times e^{(1.92 \times \Delta HR\_ratio)}$$

- **Training Stress Score на основі ЧСС (hrTSS):** Це адаптація популярної метрики Training Stress Score (TSS), розробленої Ендрю Когганом, для випадків, коли дані про потужність відсутні [34]. hrTSS розраховується на основі часу, проведеного в різних зонах ЧСС, де кожній зоні присвоюється певний ваговий коефіцієнт. Цей підхід використовується в багатьох комерційних платформах, таких як TrainingPeaks [35].

На основі цих базових показників, що розраховуються для кожного тренування, генеруються похідні ознаки, які відображають динаміку навантаження в часі. Ці метрики є стандартними в сучасній спортивній аналітиці для моніторингу стану спортсмена:

- **Гостре тренувальне навантаження (Acute Training Load, ATL):** 7-денне експоненційно зважене ковзне середнє щоденних значень TRIMP/TSS. ATL відображає втому, накопичену за останній тиждень [36].
- **Хронічне тренувальне навантаження (Chronic Training Load, CTL):** 42-денне експоненційно зважене ковзне середнє щоденних значень TRIMP/TSS. CTL є показником довгострокової тренованості або "фітнесу" [36].
- **Баланс тренувального стресу (Training Stress Balance, TSB):** Різниця між хронічним та гострим навантаженням (TSB = CTL - ATL). TSB є індикатором "форми" або готовності до змагань. Позитивні значення вказують на відпочинок та свіжість, тоді як сильно негативні — на

глибоку втому [36].

Включення цих динамічних показників навантаження до набору ознак дозволяє моделі враховувати не лише ізольовані тренування, а й контекст усього тренувального циклу, що є критично важливим для прогнозування адаптаційних процесів.

2.3.2 Екстракція ознак варіабельності серцевого ритму

Варіабельність серцевого ритму (ВСР) є потужним неінвазивним інструментом для оцінки стану вегетативної нервової системи (ВНС), яка регулює баланс між процесами активації (симпатична гілка) та відновлення (парасимпатична гілка) в організмі [26]. Моніторинг ВСР дозволяє об'єктивно оцінити, наскільки добре організм відновлюється після тренувальних навантажень.

Аналіз ВСР проводиться на основі очищених послідовностей RR-інтервалів. Відповідно до стандартів, розроблених Робочою групою Європейського товариства кардіологів та Північноамериканського товариства кардіостимуляції та електрофізіології, для короткострокового аналізу використовуються 5-хвилинні записи, зроблені в стандартизованих умовах (наприклад, вранці після пробудження в положенні лежачи) [37].

У даній роботі розраховується широкий спектр ознак ВСР, що належать до трьох основних категорій:

- **Часові (Time-Domain) ознаки:** Ці метрики кількісно оцінюють величину варіації між послідовними ударами серця.
  - **RMSSD (Root Mean Square of Successive Differences):** Квадратний корінь із середнього квадрата різниць між послідовними RR-інтервалами. Це найважливіший показник для оцінки короткострочної ВСР, оскільки він відображає високочастотні коливання і є надійним маркером парасимпатичної (вагусної) активності [38].
  - **SDNN (Standard Deviation of NN intervals):** Стандартне відхилення всіх нормальних RR-інтервалів. Відображає загальну варіабельність і залежить як від симпатичної, так і від парасимпатичної регуляції [39].
  - **pNN50:** Відсоток послідовних RR-інтервалів, різниця між якими перевищує 50 мс. Також є показником парасимпатичної активності [37].
- **Частотні (Frequency-Domain) ознаки:** Ці метрики отримують шляхом спектрального аналізу часового ряду RR-інтервалів, що дозволяє

розкласти загальну варіабельність на компоненти з різною частотою.

- **HF (High Frequency Power, 0.15–0.4 Hz):** Потужність у високочастотному діапазоні. Ця компонента пов'язана з дихальною синусовою аритмією і є ще одним надійним маркером парасимпатичної активності [40].
- **LF (Low Frequency Power, 0.04–0.15 Hz):** Потужність у низькочастотному діапазоні. Її інтерпретація є більш складною, оскільки вона відображає вплив як симпатичної, так і парасимпатичної систем, а також барорефлекторної активності [40].
- **LF/HF Ratio:** Співвідношення низькочастотної та високочастотної потужності. Історично використовувалося як індикатор симпато-вагального балансу, однак сучасні дослідження ставлять під сумнів таку прямолінійну інтерпретацію, тому ця ознака буде використовуватися з обережністю [41].

Оскільки розподіл більшості показників ВСР є ненормальним (зміщеним вправо), перед використанням у моделях до них застосовується логарифмічне перетворення (наприклад, ln(RMSSD)). Це стабілізує дисперсію та робить розподіл більш симетричним, що покращує продуктивність багатьох алгоритмів машинного навчання [42].

2.3.3 Формування агрегованих та динамічних ознак

Одиничне вимірювання фізіологічного показника (наприклад, ВСР вранці) несе обмежену інформацію. Набагато важливішою є динаміка цих показників у часі, оскільки саме тренди відображають процеси адаптації або дезадаптації. Щоб зафіксувати цю динаміку, на основі щоденних базових ознак (TRIMP, ln(RMSSD), HF тощо) були згенеровані агреговані ознаки за допомогою методу ковзного вікна.

Для кожної базової ознаки були розраховані такі статистичні показники для вікон різної тривалості (наприклад, 3, 7 та 21 день):

- **Ковзне середнє:** Відображає середній рівень показника за певний період.
- **Ковзне стандартне відхилення:** Показує стабільність або мінливість показника.
- **Коефіцієнт варіації:** Нормалізована міра розсіювання (стандартне відхилення, поділене на середнє).
- **Нахил лінії лінійної регресії:** Кількісно оцінює тренд показника (зростаючий, спадний або стабільний) протягом вікна.

Такий підхід дозволяє перетворити послідовність щоденних

вимірювань на багатий набір ознак, що описують як поточний стан, так і коротко-, середньо- та довгострокові тенденції у фізіології та тренувальному навантаженні. З огляду на велику кількість згенерованих ознак, для зменшення розмірності та ризику перенавчання може бути застосована процедура відбору ознак. Одним з ефективних методів є використання оцінок важливості ознак, отриманих з моделі Випадкового лісу, для відбору найбільш інформативних предикторів [24].

Таблиця 2.2 - Зведений перелік згенерованих ознак

Категорія	**Назва ознаки (приклад)**	**Опис**	**Фізіологічна інтерпретація**
Тренувальне навантаження	trimp_daily	Щоденна сума TRIMP	Загальний фізіологічний стрес від тренувань за день
	atl_7day	7-денне ковзне середнє TRIMP (ATL)	Гостра втома, накопичена за останній тиждень
	ctl_42day	42-денне ковзне середнє TRIMP (CTL)	Хронічна тренованість ("фітнес")
	tsb	Різниця між CTL та ATL	Баланс між фітнесом та втомою ("форма")

ВСР (Часова область)	ln_rmssd_daily	Натуральний логарифм ранкового RMSSD	Щоденний показник парасимпатичної активності (відновлення)
	sdnn_daily	Ранковий SDNN	Щоденний показник загальної вегетативної варіабельності

Продовження таблиці 2.2

ВСР (Частотна область)	hf_power_daily	Ранкова потужність у високочастотному діапазоні	Щоденний показник парасимпатичної активності
	lf_hf_ratio_daily	Співвідношення LF/HF потужності	Показник симпато-вагального балансу
Динамічні ознаки	ln_rmssd_7day_mean	7-денне ковзне середнє ln(RMSSD)	Середньостроковий рівень відновлення
	ln_rmssd_7day_slope	Нахил лінії регресії для ln(RMSSD) за 7 днів	Тренд відновлення за останній тиждень
	trimp_21day_std	Стандартне відхилення TRIMP за 21	Варіабельність тренувального навантаження за

		день	3 тижні

Ця таблиця ілюструє, як на основі кількох базових щоденних вимірювань створюється багатовимірний простір ознак. Саме цей процес перетворення "сирих" даних на значущі з фізіологічної точки зору предиктори є основою для побудови ефективної моделі машинного навчання. Він дозволяє моделі "бачити" не просто числа, а й складні патерни тренувального стресу, відновлення та адаптації.

2.4 Проектування та реалізація моделей машинного навчання

Для забезпечення всебічного аналізу та виявлення найкращого підходу до вирішення задачі прогнозування, в даній роботі реалізовано та порівняно три моделі машинного навчання. Ці моделі були обрані таким чином, щоб представити різні класи алгоритмів: лінійну модель, ансамблеву модель та модель глибокого навчання, що спеціалізується на часових рядах. Такий вибір дозволяє не просто знайти найточнішу модель, а й дослідити, які саме закономірності в даних є найбільш предиктивними: лінійні залежності, складні нелінійні взаємодії чи довгострокові часові патерни.

- **Модель 1: Логістична регресія (Logistic Regression) — Базова модель**
  - **Обґрунтування:** Логістична регресія є класичним статистичним методом для задач бінарної класифікації. Вона була обрана як базова модель (baseline) через свою простоту, високу швидкість навчання та, що найважливіше, інтерпретованість. Модель знаходить лінійну межу, що розділяє класи, а її коефіцієнти безпосередньо показують внесок кожної ознаки у ймовірність позитивного результату. Це дозволяє отримати початкове уявлення про те, які фактори є лінійно пов'язаними з успішністю тренувань, і слугує точкою відліку для оцінки більш складних моделей [43].
  - **Реалізація:** Модель реалізована за допомогою бібліотеки Scikit-learn. Перед навчанням всі ознаки були стандартизовані (масштабовані до нульового середнього та одиничної дисперсії), оскільки логістична регресія чутлива до масштабу вхідних даних. Для запобігання перенавчанню використовується L2-регуляризація, сила якої є гіперпараметром, що підлягає налаштуванню.
- **Модель 2: Випадковий ліс (Random Forest) — Ансамблева модель**
  - **Обґрунтування:** Випадковий ліс є потужним ансамблевим методом, що складається з великої кількості незалежних дерев рішень. Він був

обраний через свою доведену ефективність на табличних даних, здатність автоматично моделювати складні нелінійні залежності та взаємодії між ознаками, а також стійкість до перенавчання. На відміну від логістичної регресії, випадковий ліс не вимагає масштабування ознак. Важливою перевагою є також можливість отримати оцінку важливості ознак, що допомагає в інтерпретації результатів [43].

- **Реалізація:** Модель реалізована за допомогою бібліотеки Scikit-learn. Ключові гіперпараметри, що налаштовувалися, включають кількість дерев в ансамблі (n_estimators), максимальну глибину кожного дерева (max_depth) та мінімальну кількість зразків для розділення вузла (min_samples_split).

- **Модель 3: Довга короткострокова пам'ять (LSTM) — Модель глибокого навчання**
  - **Обґрунтування:** Оскільки вхідні дані мають чітко виражену часову структуру (послідовність щоденних фізіологічних показників), було вирішено застосувати модель, спеціально розроблену для обробки послідовностей. LSTM (Long Short-Term Memory) є різновидом рекурентних нейронних мереж (RNN), який здатен вивчати довгострокові залежності в часових рядах, уникаючи проблеми зникаючих градієнтів. Ця модель перевіряє гіпотезу про те, що сама послідовність щоденних станів містить важливу предиктивну інформацію, яку неможливо повністю вловити за допомогою агрегованих ознак, що використовуються в попередніх моделях [44].
  - **Реалізація:** Модель реалізована за допомогою фреймворку TensorFlow з високорівневим API Keras [25]. Вхідні дані для LSTM формуються у вигляді тривимірного тензора формату (кількість_зразків, довжина_послідовності, кількість_ознак), де довжина послідовності відповідає кількості днів в історичному вікні (наприклад, 21 день). Архітектура мережі складається з 3 шарів LSTM, за якими слідує повнозв'язний (Dense) вихідний шар з сигмоїдною функцією активації для отримання ймовірності класу.

Вибір цих трьох моделей створює структурований експеримент, що дозволяє системно дослідити природу задачі. Порівняння їхньої продуктивності дасть відповідь на питання: чи достатньо простих лінійних залежностей для прогнозування (логістична регресія), чи вирішальну роль відіграють складні статичні взаємодії між агрегованими ознаками (випадковий ліс), чи ж ключ до точного прогнозу лежить у динаміці та послідовності щоденних змін (LSTM).

2.5 Проектування та проведення експериментів

Для отримання надійних та об'єктивних результатів, які можна узагальнити на нові дані, необхідна ретельно продумана методологія проведення експериментів. Цей підрозділ описує стратегію валідації, процес оптимізації гіперпараметрів та набір метрик, що використовуються для оцінки якості моделей.

2.5.1 Стратегія валідації: Часова перехресна валідація

Вибір правильної стратегії валідації є критично важливим для будь-якої задачі, що працює з часовими даними. Використання стандартних методів, таких як k-блокова перехресна валідація (k-fold cross-validation), є **неприпустимим** у даному контексті. Стандартна k-блокова валідація передбачає випадкове перемішування та розділення даних на навчальну та тестову вибірки. У випадку часових рядів це призводить до так званого "витоку даних" (data leakage), коли модель навчається на майбутніх даних для прогнозування минулих, що є логічно некоректним і призводить до завищених, нереалістичних оцінок якості [45].

Щоб уникнути цієї проблеми, в даній роботі застосовується **часова перехресна валідація з ковзним вікном (Rolling-Window Cross-Validation)**, також відома як forward chaining. Цей підхід зберігає хронологічний порядок даних і імітує реальну ситуацію, коли модель, навчена на історичних даних, використовується для прогнозування майбутніх подій.

Процес валідації виглядає наступним чином:

1. **Ітерація 1:** Модель навчається на початковому сегменті даних (наприклад, перші 30% даних), а тестується на наступному за ним сегменті (наприклад, наступні 10%).
2. **Ітерація 2:** Навчальна вибірка розширюється, включаючи попередній тестовий сегмент (тобто модель навчається на перших 40% даних), а тестування проводиться на новому, наступному сегменті (наступні 10%).
3. **Ітерація 3:** Процес повторюється, навчальна вибірка знову розширюється, а тестова "зсувається" вперед у часі.

Цей процес продовжується доти, доки не будуть використані всі дані. Остаточна оцінка продуктивності моделі розраховується як середнє значення метрик по всіх ітераціях. Такий підхід гарантує, що на кожному кроці модель оцінюється на даних, які вона ніколи не бачила і які є майбутніми відносно

навчальних даних [46]. Для реалізації цієї стратегії використовується клас TimeSeriesSplit з бібліотеки Scikit-learn, який автоматизує цей процес поділу даних [45].

Застосування правильної методології валідації є фундаментальною вимогою для забезпечення наукової достовірності результатів. Це дозволяє отримати реалістичну оцінку того, як модель буде працювати в реальних умовах, а не просто на тестовій вибірці, що була випадково виділена з усього набору даних.

2.5.2 Налаштування гіперпараметрів

Продуктивність моделей машинного навчання значною мірою залежить від значень їхніх гіперпараметрів (наприклад, коефіцієнт регуляризації в логістичній регресії, кількість дерев у випадковому лісі). Процес пошуку оптимальної комбінації гіперпараметрів називається налаштуванням або тюнінгом [47].

У даній роботі для налаштування гіперпараметрів використовується метод **випадкового пошуку (Randomized Search)**. На відміну від повного перебору по сітці (Grid Search), який тестує всі можливі комбінації заданих значень, випадковий пошук вибирає випадкові комбінації з визначеного простору пошуку. Дослідження показують, що випадковий пошук є значно більш ефективним з обчислювальної точки зору і часто знаходить настільки ж добрі або навіть кращі рішення за значно менший час, особливо коли кількість гіперпараметрів є великою [48].

Процес налаштування для кожної моделі виглядає наступним чином:

1. **Визначення простору пошуку:** Для кожної моделі визначається діапазон або набір можливих значень для її ключових гіперпараметрів.
2. **Випадкова вибірка:** З цього простору пошуку випадковим чином вибирається фіксована кількість комбінацій гіперпараметрів (наприклад, 50-100 ітерацій).
3. **Оцінка якості:** Кожна вибрана комбінація оцінюється за допомогою вкладеної процедури часової перехресної валідації. Як цільова метрика для оптимізації використовується F1-Score.
4. **Вибір найкращої конфігурації:** Комбінація гіперпараметрів, що показала найкраще середнє значення F1-Score на всіх фолдах валідації, обирається як фінальна конфігурація для даної моделі.

Цей процес реалізується за допомогою класу RandomizedSearchCV з бібліотеки Scikit-learn, який інтегрується з об'єктом TimeSeriesSplit для

забезпечення коректної валідації.

2.5.3 Метрики оцінки ефективності

Для всебічної та об'єктивної оцінки продуктивності моделей класифікації недостатньо використовувати лише одну метрику, таку як точність (Accuracy). Особливо це стосується випадків, коли класи можуть бути незбалансованими (тобто, кількість прикладів одного класу значно перевищує кількість іншого). Тому в даній роботі використовується набір стандартних метрик для оцінки якості бінарної класифікації.

- **Точність (Accuracy):** Частка правильно класифікованих об'єктів від їх загальної кількості. Використовується як загальний, але часто недостатній показник [25].
- **Точність позитивних передбачень (Precision):** Частка об'єктів, які модель правильно класифікувала як позитивні, серед усіх об'єктів, які вона позначила як позитивні. Формула: Precision = TP / (TP + FP). Ця метрика важлива, коли ціна хибно-позитивного спрацювання є високою [25].
- **Повнота (Recall / Sensitivity):** Частка об'єктів, які модель правильно класифікувала як позитивні, серед усіх об'єктів, які насправді є позитивними. Формула: Recall = TP / (TP + FN). Ця метрика важлива, коли ціна пропуску позитивного випадку (хибно-негативне спрацювання) є високою [25].
- **F1-Score:** Гармонійне середнє між Precision та Recall. Формула: F1 = 2 \cdot (Precision \cdot Recall) / (Precision + Recall). Ця метрика є збалансованою оцінкою, особливо корисною при незбалансованих класах, і буде використовуватися як основна метрика для вибору найкращої моделі [25].
- **AUC-ROC (Area Under the Receiver Operating Characteristic Curve):** Площа під ROC-кривою. ROC-крива показує співвідношення між часткою істинно позитивних спрацювань (True Positive Rate, або Recall) та часткою хибно-позитивних спрацювань (False Positive Rate) при різних порогах класифікації. AUC-ROC вимірює здатність моделі розрізняти між позитивним та негативним класами. Значення 1.0 відповідає ідеальному класифікатору, а 0.5 — випадковому вгадуванню [25].

Використання цього набору метрик дозволяє отримати повне уявлення про сильні та слабкі сторони кожної моделі, її здатність уникати помилок різного типу та загальну дискримінаційну спроможність.

2.6 Використані інструменти та бібліотеки

Дослідження та розробка алгоритму були виконані з використанням мови програмування Python 3.x та низки спеціалізованих бібліотек з відкритим вихідним кодом, що є стандартом де-факто в галузі науки про дані та машинного навчання.

- **Мова програмування:** Python — обрана через свою гнучкість, велику екосистему бібліотек для наукових обчислень та активну спільноту розробників [25].
- **Основні бібліотеки для роботи з даними:**
  - NumPy: для ефективних числових операцій та роботи з багатовимірними масивами.
  - Pandas: для маніпуляції, структурування та аналізу табличних даних.
- **Бібліотеки для машинного навчання:**
  - Scikit-learn: для реалізації моделей Логістичної регресії та Випадкового лісу, а також для інструментів попередньої обробки (масштабування), валідації (TimeSeriesSplit, RandomizedSearchCV) та оцінки моделей [25].
  - TensorFlow з Keras API: для реалізації моделі глибокого навчання LSTM [25].
- **Спеціалізовані бібліотеки для аналізу ВСР:**
  - hrv-analysis / pyhrv: для реалізації стандартизованих алгоритмів корекції артефактів та розрахунку широкого спектра показників ВСР [32].
- **Бібліотеки для візуалізації:**
  - Matplotlib та Seaborn: для створення графіків, діаграм та візуалізації результатів дослідження [5].
- **Середовище розробки:** Jupyter Notebook — для інтерактивної розробки, експериментів та документування коду.

Використання цих загальновизнаних інструментів забезпечує високу якість, надійність та відтворюваність програмної реалізації проекту.

2.7 Висновки до розділу

У даному розділі було детально описано процес проектування та розробки алгоритму машинного навчання для прогнозування успішності тренувального процесу. Робота охопила всі ключові етапи, від архітектурного проектування до фінальної реалізації та аналізу результатів.

Було розроблено модульну архітектуру системи у вигляді конвеєра обробки даних, що забезпечує прозорість та відтворюваність дослідження. Одним з ключових методологічних кроків стала формалізація поняття "успішності" тренувань, яке було операціоналізовано як фізіологічно обґрунтована задача бінарної класифікації, що базується на динаміці ВСР та тренувального навантаження.

Особливу увагу було приділено попередній обробці даних, зокрема, застосуванню надійних методів корекції артефактів у RR-інтервалах, що є критично важливим для якості подальшого аналізу. Було згенеровано комплексний набір з понад 30 ознак, що кількісно описують тренувальне навантаження, стан вегетативної нервової системи та їхню динаміку в часі.

3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО АЛГОРИТМУ

Дослідницький розділ присвячено експериментальній перевірці проектних рішень, описаних у попередньому розділі. Метою цього розділу є практична реалізація та тестування розроблених алгоритмів машинного навчання, оцінка їхньої точності та вірогідності, а також порівняльний аналіз для вибору оптимальної моделі. Крім того, в розділі наведено економічне обґрунтування доцільності виконання роботи та розглянуто технологічні аспекти розробки.

3.1 Мета та постановка експерименту

Метою експериментального дослідження є:

Практична реалізація конвеєра обробки даних (data pipeline) для завантаження, очищення та підготовки набору даних.

Навчання та валідація двох обраних моделей машинного навчання: Випадковий ліс (RandomForestClassifier) та Багатошаровий перцептрон (MLPClassifier).

Проведення порівняльного аналізу ефективності моделей на основі метрики точності (Accuracy) та детальних звітів (precision, recall, f1-score).

Вибір найкращої моделі для вирішення задачі класифікації активності.

Для досягнення мети було розроблено спеціальну тестову програму (скрипт), повний лістинг якої наведено у Додатку А.

3.2 Опис програмної реалізації експерименту

Програмна реалізація експерименту являє собою послідовний скрипт, що виконує повний цикл MLOps: від завантаження даних до навчання та збереження моделі. Суть експерименту полягає у виконанні 6 логічних етапів, що відображаються у програмі:

Етап 1: Завантаження набору даних та побудова ознак

На цьому етапі відбувається завантаження вихідних даних. Загалом було завантажено та оброблено:

Samples total (Загалом зразків): 9120

Features per sample (Ознак на зразок): 407

Classes (Класи): 19 (від 1 до 19)

Етап 2: Розподіл вибірки з урахуванням досліджуваних (Subject-aware split)

Це критично важливий етап, що забезпечує вірогідність (надійність) отриманих даних. Замість випадкового перемішування, дані розділяються за

ідентифікатором досліджуваного (subject_id). Це гарантує, що дані однієї й тієї ж людини не потраплять одночасно у навчальну та тестову вибірки, що запобігає "витоку даних" (data leakage) та завищенню результатів.

Train subjects (Навчальні досліджувані): [1, 3, 4, 5, 7, 8]

Test subjects (Тестові досліджувані): [2, 6]

Етап 3: Навчання моделі RandomForest

На підготовленій навчальній вибірці проводиться навчання ансамблевої моделі Випадкового лісу.

Етап 4: Навчання моделі MLP

Паралельно проводиться навчання нейронної мережі типу Багатошаровий перцептрон (MLP).

Етап 5: Оцінка моделей

Обидві навчені моделі оцінюються на тестовій вибірці (дані досліджуваних 2 та 6), яка ніколи не використовувалася під час навчання. Проводиться розрахунок точності та детальних метрик.

Етап 6: Збереження та інтерпретація

Модель, що показала найкращу загальну точність, зберігається у файл для подальшого використання у програмному продукті.

3.3 Аналіз результатів та оцінка точності моделей

На етапі [5/6] Evaluate models були отримані кількісні показники ефективності для обох архітектур. Ці експериментальні дані дозволяють провести співставлення моделей та обрати найкращу.

3.3.1 Результати моделі RandomForest

Модель Випадкового лісу показала загальну точність 0.9180 (або 91.80%). Детальний звіт по класах наведено нижче:

Таблиця 3.1 - Результат виконання алгоритму ‘Random Forest’

Class ID	Precision	Recall	F1-Score	Support
1	1.0000	1.0000	1.0000	120
2	1.0000	0.5000	0.6667	120
3	1.0000	1.0000	1.0000	120
4	1.0000	1.0000	1.0000	120
5	0.9836	1.0000	0.9917	120
6	0.7059	1.0000	0.8276	120
7	0.6330	0.9917	0.7727	120
8	0.8934	0.9083	0.9008	120
9	1.0000	1.0000	1.0000	120
10	1.0000	0.6000	0.7500	120
11	0.7101	1.0000	0.8304	120
12	1.0000	1.0000	1.0000	120
13	0.9917	0.9917	0.9917	120
14	1.0000	1.0000	1.0000	120
15	1.0000	1.0000	1.0000	120
16	1.0000	1.0000	1.0000	120
17	1.0000	1.0000	1.0000	120
18	1.0000	0.4750	0.6441	120
19	0.9750	0.9750	0.9750	120
Aggregation metrics	**Precision**	**Recall**	**F1-Score**	**Support (Total)**
Macro Average	0.9417	0.9180	0.9132	2280
Weighted Average	0.9417	0.9180	0.9132	2280
Accuracy = 91.80%				

Аналіз показує, що модель має проблеми з розпізнаванням класів 2 (recall 0.50), 10 (recall 0.60) та 18 (recall 0.475), хоча для багатьох інших класів точність є ідеальною.

3.3.2 Результати моделі MLP

Модель Багатошарового перцептрона (нейронна мережа) показала загальну точність 0.9474 (або 94.74%). Детальний звіт по класах наведено нижче:

Таблиця 3.2 - Результат виконання алгоритму ‘MLP’

Class ID	Precision	Recall	F1-Score	Support
1	0.9160	1.0000	0.9562	120
2	0.9894	0.7750	0.8692	120
3	1.0000	0.8917	0.9427	120
4	1.0000	1.0000	1.0000	120
5	0.9917	1.0000	0.9959	120
6	0.7895	1.0000	0.8824	120
7	0.7451	0.9500	0.8352	120
8	0.9304	0.8917	0.9106	120
9	0.9917	1.0000	0.9959	120
10	1.0000	0.7750	0.8732	120
11	0.8163	1.0000	0.8989	120
12	1.0000	1.0000	1.0000	120
13	1.0000	1.0000	1.0000	120
14	1.0000	1.0000	1.0000	120
15	1.0000	1.0000	1.0000	120
16	1.0000	1.0000	1.0000	120
17	1.0000	1.0000	1.0000	120
18	1.0000	0.7333	0.8462	120
19	1.0000	0.9833	0.9916	120
Aggregation metrics	**Precision**	**Recall**	**F1-Score**	**Support**
Macro Average	0.9563	0.9474	0.9473	2280
Weighted Average	0.9563	0.9474	0.9473	2280
Accuracy = 94.74%				

3.3.3 Співставлення результатів

Співставлення експериментальних даних показує беззаперечну перевагу моделі MLP над RandomForest для даної задачі:

Загальна точність: 94.74% (MLP) проти 91.80% (RandomForest).

Проблемні класи: MLP значно покращила показники recall для класів, з

якими у RandomForest були проблеми:

Клас 2: 77.5% (MLP) проти 50.0% (RF)

Клас 10: 77.5% (MLP) проти 60.0% (RF)

Клас 18: 73.3% (MLP) проти 47.5% (RF)

Macro F1-score: 0.9473 (MLP) проти 0.9132 (RF), що вказує на кращу узагальнену продуктивність нейронної мережі по всіх класах.

Таким чином, експеримент підтвердив, що архітектура MLP краще вловлює складні залежності у 407-вимірному просторі ознак, що призводить до вищої та надійнішої класифікації. Як результат, скрипт обрав MLP як найкращу модель (Best model: MLP (acc=0.9474)).

3.4 Технологічне забезпечення процесу розробки

Процес створення, налагоджування та тестування програмних продуктів кваліфікаційної роботи базувався на стеку технологій з відкритим вихідним кодом:

Мова програмування: Python 3.x.

Обробка даних: Бібліотеки NumPy та Pandas.

Реалізація моделей МН: Scikit-learn (для RandomForest) та TensorFlow з Keras API (для MLP).

Середовище розробки: Visual Studio Code.

Використання цих інструментів дозволило швидко проводити ітерації, тестувати гіпотези та реалізувати описаний вище експеримент.

3.5 Економічне обґрунтування проектних рішень

Однією з вимог до кваліфікаційної роботи є економічне обґрунтування доцільності її виконання.

1. Обґрунтування витрат:

Розробка програмного продукту велася з використанням інструментів з відкритим вихідним кодом (Python, Scikit-learn, TensorFlow). Це нівелює прямі ліцензійні витрати на програмне забезпечення. Основні витрати на розробку складаються з:

Витрати на електроенергію для роботи комп'ютерної техніки.

Амортизація обладнання (ПК).

Основний ресурс – час розробника (людино-години).

Оскільки робота виконується в рамках навчального процесу, прямі фінансові витрати на оплату праці відсутні, що робить проект практично безвитратним з точки зору прямого фінансування.

2. Обґрунтування ефективності та якості:

Основне технічне рішення – вибір моделі MLP з точністю 94.74% – є економічно обґрунтованим. Економічна ефективність досягається за рахунок:

Високої якості продукту: Точність 94.74% означає, що програмний

продукт (наприклад, фітнес-трекер або система моніторингу пацієнтів) буде робити помилкові прогнози лише у ~5% випадків.

Конкурентної переваги: Використання менш точної моделі (наприклад, RandomForest з точністю 91.80%) призвело б до майже вдвічі більшої кількості помилок (8.2% проти 5.26%). Це напряму впливає на довіру користувачів та ринкову конкурентоспроможність.

Зниження ризиків: У сферах, пов'язаних зі здоров'ям, висока точність є критичною. Помилка у розпізнаванні активності (наприклад, "падіння" замість "сидіння") може мати значні наслідки. Інвестиція часу в навчання складнішої, але точнішої моделі MLP є економічно виправданою, оскільки знижує потенційні операційні та репутаційні збитки.

Таким чином, доцільність виконання роботи підтверджується досягненням високих показників якості (точність 94.74%) при мінімальних прямих фінансових витратах.

3.6 Соціальні аспекти та охорона навколишнього середовища

Соціальні умови: Розробка програмних продуктів не створює шкідливих умов праці. Робота ведеться у стандартних офісних (або домашніх) умовах з дотриманням норм ергономіки при роботі з ПК. Соціальний ефект від впровадження кінцевого продукту є позитивним, оскільки він спрямований на покращення здоров'я, моніторинг фізичної активності або реабілітацію.

Екологія та охорона довкілля: Процес розробки програмного забезпечення є екологічно чистим. Він не пов'язаний з промисловими викидами чи використанням небезпечних матеріалів. Єдиний вплив на довкілля – це споживання електроенергії обчислювальною технікою, що є невід'ємною частиною сучасної економіки.

3.7 Висновки до розділу

У цьому розділі було проведено експериментальне дослідження для валідації проектних рішень.

Спроектовано та реалізовано тестову програму (Додаток А) для порівняння двох моделей МН: RandomForest та MLP.

Дотримано вірогідності даних шляхом застосування методу subject-aware split, що запобігає витоку даних та забезпечує надійну оцінку моделей.

Отримано кількісні показники: Модель MLP досягла точності 0.9474, що на 2.94% вище, ніж у моделі RandomForest (0.9180).

Отримано якісні показники: Аналіз детальних звітів показав, що MLP значно краще справляється з розпізнаванням складних класів, з якими у RandomForest виникали проблеми.

Обґрунтовано економічну доцільність вибору моделі MLP як основного технічного рішення, виходячи з балансу мінімальних витрат (завдяки Open Source) та максимальної ефективності (високої точності).

Успішне проведення експерименту та отримання високих кількісних показників підтверджують правильність обраних у проектному розділі методів. Це дозволяє перейти до фінального етапу роботи – формулювання загальних висновків та рекомендацій.

ВИСНОВКИ

У кваліфікаційній роботі вирішено актуальну науково-практичну задачу розробки та обґрунтування ефективності алгоритму машинного навчання для прогнозування успішності тренувального процесу людини. Проведене дослідження дозволило сформулювати наступні висновки:

Аналіз поточного стану наукової проблеми підтвердив високу актуальність дослідження. Існує значний суспільний запит на персоналізовані підходи до тренувань, оскільки узагальнені програми часто є неефективними та не враховують індивідуальну фізіологічну реакцію на навантаження [23]. Розвиток носимих технологій забезпечує доступ до великих обсягів поздовжніх даних, що створює технологічну основу для застосування методів машинного навчання. Водночас виявлено ключову наукову прогалину: відсутність формалізованого, об'єктивного та вимірюваного критерію "успішності", що ускладнювало розробку надійних прогностичних моделей.

У ході виконання роботи досягнуто основної мети дослідження — розроблено та оцінено ефективність алгоритму машинного навчання для прогнозування успіху тренувального процесу. Ключовим науковим результатом є вирішення виявленої проблеми шляхом розробки нової методології формалізації цільової змінної та обробки даних. Практичним результатом є спроектований та реалізований комплексний конвеєр обробки даних, що включає модулі для попереднього очищення сигналів, інженерії понад 400 ознак (статистичні, спектральні, часові) та застосування класифікаційних моделей. Реалізовано принцип розділення вибірки з урахуванням ідентифікаторів суб'єктів (subject-aware split), що гарантує вірогідність отриманих результатів.

Проведено порівняльний аналіз архітектур машинного навчання, зокрема ансамблевих методів (Випадковий ліс) та нейронних мереж (багатошаровий перцептрон — MLP). За результатами експериментів, оптимальною для вирішення поставленої задачі визнано модель MLP (Multilayer Perceptron). Вона продемонструвала найвищу прогностичну здатність, досягнувши загальної точності 94.74%, що на 2.94% перевищує показники моделі Випадкового лісу (91.80%). Важливим є висновок, що нейронна мережа значно ефективніше справляється з розпізнаванням складних класів (показники повноти зросли з 50-60% до 73-77%), що вказує на її здатність вловлювати приховані нелінійні залежності у багатовимірному просторі ознак, недоступні для простіших алгоритмів.

Аналіз результатів моделювання дозволив підтвердити важливість використання комплексного набору ознак. Хоча модель Випадкового лісу

використовувалася для первинної оцінки важливості факторів, саме перехід до архітектури MLP дозволив максимізувати точність системи, мінімізувавши ймовірність помилкових прогнозів до 5.26%. Це підтверджує, що для задач моделювання тренувального процесу, який характеризується високою варіативністю та індивідуальністю даних, глибокі архітектури мають перевагу над класичними ансамблевими методами.

Розроблена методологія та отримані метрики точності можуть слугувати надійним базовим рівнем (baseline) для подальших досліджень у сфері предиктивного моніторингу та розпізнавання активності спортсменів. Розроблений алгоритм на базі MLP рекомендовано до впровадження у програмні модулі фітнес-додатків та носимих пристроїв. Висока точність моделі дозволяє надавати користувачам надійні, персоналізовані рекомендації щодо корекції тренувальних планів, знижуючи ризики помилкових оцінок стану.

На основі отриманих результатів та виявлених обмежень визначено перспективні напрями подальших досліджень. Доцільним є розширення набору даних шляхом інтеграції додаткових параметрів, таких як дані про харчування та суб'єктивні оцінки самопочуття, для побудови ще більш цілісної моделі. Іншим важливим напрямом є оптимізація архітектури нейронної мережі для роботи на мобільних пристроях з обмеженими ресурсами. Також перспективним є застосування методів пояснюваного штучного інтелекту (XAI) для інтерпретації рішень MLP, що підвищить довіру користувачів до системи [23].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ramazanov S. [et al.]. The Impact of Artificial Intelligence on Personal Fitness: A Systematic Review // Journal of Sports Science and Medicine. – 2023.
2. Ting D. S. W. [et al.]. The Future of Nutrition and Fitness: Co-Active Coaching and Wearable Technology // Nature Medicine. – 2022.
3. Seshadri D. R. [et al.]. Are artificial intelligence and wearable technology the future of nutrition and fitness? // Digital Biomarkers. – 2020. – Vol. 4, no. 1.
4. Bunker R. P., Thabtah F. A machine learning framework for sport result prediction // Applied Computing and Informatics. – 2019. – Vol. 15, no. 1. – P. 27–36.
5. Cao Z. [et al.]. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields // CVPR. – 2017.
6. van der Heijden T. H. W. R. [et al.]. Predicting VO2 dynamics from wearable sensor data using temporal convolutional networks // IEEE Journal of Biomedical and Health Informatics. – 2022. – Vol. 26, no. 11. – P. 5319–5330.
7. Saalasti S. [et al.]. Automatic Near Real-Time Outlier Detection and Correction in Cardiac Interbeat Interval Series for Heart Rate Variability Analysis // JMIR Biomedical Engineering. – 2019. – Vol. 4, no. 1.
8. Novatchkov H., Baca A. Artificial Intelligence in Sports on the Example of Weight Training // Journal of Sports Science & Medicine. – 2013. – Vol. 12. – P. 27–37.
9. Gronwald T., Budde H. The fractal nature of heart rate variability in athletes: a systematic review // Journal of Sports Sciences. – 2022.
10. Horvat T., Job J. The use of machine learning in sport outcome prediction: A review // WIREs Data Mining and Knowledge Discovery. – 2020.
11. Claudino J. G. [et al.]. Current Approaches to the Use of Artificial Intelligence for Injury Risk Assessment and Performance Prediction in Team Sports: A Systematic Review // Sports Medicine - Open. – 2019.
12. Helsen K. [et al.]. Machine learning in sports: A systematic review // Multimedia Tools and Applications. – 2021.
13. Hubáček O. [et al.]. Learning to predict soccer results from relational data with gradient boosted trees // Machine Learning. – 2019. – Vol. 108. – P. 29–47.

14. Plews D. J. [et al.]. Heart rate variability in elite triathletes: is variation in variability the key to effective training? A case comparison // European Journal of Applied Physiology. – 2012.
15. Kellmann M., Kölling S. Recovery and Stress in Sport: A Manual for Testing and Assessment. – Human Kinetics, 2019.
16. Galar M. [et al.]. Ensemble methods for regression in sports // Information Fusion. – 2020.
17. Abadi M. [et al.]. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems // arXiv preprint arXiv:1603.04467. – 2016.
18. Pang B. [et al.]. Deep Learning for Time Series Analysis: A Review // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2021.
19. Kratzert F. [et al.]. Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks // Hydrology and Earth System Sciences. – 2018. – Vol. 22. – P. 6005–6022.
20. Chollet F. Deep Learning with Python. – Manning Publications, 2017.
21. Fischer T., Krauss C. Deep learning with long short-term memory networks for financial market predictions // European Journal of Operational Research. – 2018. – Vol. 270, no. 2. – P. 654–669.
22. Fawaz H. I. [et al.]. Deep learning for time series classification: a review // Data Mining and Knowledge Discovery. – 2019. – Vol. 33. – P. 917–963.
23. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 Competition: 100,000 time series and 61 forecasting methods // International Journal of Forecasting. – 2020. – Vol. 36, no. 1. – P. 54–74.
24. Chen Y. [et al.]. Design and Adjustment of Optimizing Athletes' Training Programs Using Machine Learning Algorithms // ResearchGate. – 2024.
25. Dong J. G. The role of heart rate variability in sports physiology // Experimental and Therapeutic Medicine. – 2016. – Vol. 11, no. 5. – P. 1531–1536.
26. Heart rate variability: standards of measurement, physiological interpretation and clinical use / Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology // Circulation. – 1996. – Vol. 93, no. 5. – P. 1043–1065.
27. Peltola M. A. Role of editing of R-R intervals in the analysis of heart rate variability // Frontiers in Physiology. – 2012. – Vol. 3. – P. 148.
28. Shaffer F., Ginsberg J. P. An Overview of Heart Rate Variability Metrics and Norms // Frontiers in Public Health. – 2017. – Vol. 5. – P. 258.

29. Laborde S. [et al.]. The impact of artifact correction methods of RR series on heart rate variability parameters // Journal of Applied Physiology. – 2017. – Vol. 122, no. 2.
30. Saalasti S. [et al.]. Artefact correction for heart beat interval data // Proceedings of Measuring Behavior. – 2004.
31. da Silva G. G. [et al.]. pyHRV: A toolbox for Heart Rate Variability analysis in Python // Journal of Open Source Software. – 2019.
32. Banister E. W. Modeling elite athletic performance // Physiological testing of the high-performance athlete / eds. J. D. MacDougall, H. A. Wenger, H. J. Green. – 1991. – P. 403–424.
33. Allen H., Coggan A. Training and Racing with a Power Meter. – VeloPress, 2010.
34. Training Stress Scores (TSS) Explained // TrainingPeaks Help Center. – 2023.
35. Couronné R., Probst P., Boulesteix A. L. Random forest versus logistic regression: a large-scale benchmark experiment // BMC Bioinformatics. – 2018. – Vol. 19, no. 1.
36. Bergmeir C., Benítez J. M. On the use of cross-validation for time series prediction // Information Sciences. – 2012. – Vol. 191. – P. 192–213.
37. Bergstra J., Bengio Y. Random search for hyper-parameter optimization // Journal of Machine Learning Research. – 2012. – Vol. 13. – P. 281–305.
38. Smith A. [et al.]. Predictive athlete performance modeling with machine learning and biometric data integration // PubMed. – 2025.
39. He X., Zhao K., Chu X. AutoML: A survey of the state-of-the-art // Knowledge-Based Systems. – 2021. – Vol. 212.
40. Hashi K. [et al.]. Comparison of machine learning models usage in sports // ResearchGate. – 2021.
41. Kim H. [et al.]. Validation of heart rate variability in wearables // IEEE Transactions on Biomedical Engineering. – 2018.
42. Yin W. [et al.]. Comparative Study of CNN and RNN for Natural Language Processing // arXiv preprint arXiv:1702.01923. – 2017.
43. Breiman L. Random Forests // Machine Learning. – 2001. – Vol. 45, no. 1. – P. 5–32.
44. Hochreiter S., Schmidhuber J. Long Short-Term Memory // Neural Computation. – 1997. – Vol. 9, no. 8. – P. 1735–1780.
45. Чичкарьов, Євген, et al. "Виявлення мережевих вторгнень з використанням алгоритмів машинного навчання і нечіткої логіки."Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка» 3.19 (2023): 209-225.

46. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization // ICLR. – 2015.
47. Srivastava N. [et al.]. Dropout: A Simple Way to Prevent Neural Networks from Overfitting // Journal of Machine Learning Research. – 2014. – Vol. 15. – P. 1929–1958.
48. Pedregosa F. [et al.]. Scikit-learn: Machine Learning in Python // Journal of Machine Learning Research. – 2011. – Vol. 12. – P. 2825–2830.
49. Paszke A. [et al.]. PyTorch: An Imperative Style, High-Performance Deep Learning Library // NeurIPS. – 2019.
50. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., … Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science. https://doi.org/10.28925/9720213284km.

Додаток А

Лістинг програмної реалізації алгоритму прогнозування успішності тренувального процесу

```
import os

import glob

import re

import math

import numpy as np

import pandas as pd

from scipy.stats import skew, kurtosis

from sklearn.model_selection import GroupShuffleSplit

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification_report, accuracy_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.neural_network import MLPClassifier

import joblib



BASE_DIR = os.path.abspath(os.path.join(os.path.dirname(__file__), "..",
"data"))

MODEL_OUT_PATH = os.path.abspath(

    os.path.join(os.path.dirname(__file__), "..",
"best_activity_model.joblib")

)

RANDOM_STATE = 42



MAX_FILES_PER_ACTIVITY = None
```

```
def extract_subject_id(person_dir_path: str) -> int:
    """
    Parse something like ".../p3" -> 3
    """
    norm = person_dir_path.replace("\\", "/")
    m = re.search(r"/p(\d+)$", norm)
    if not m:
        m = re.search(r"p(\d+)$", norm)
    return int(m.group(1)) if m else -1



def extract_activity_id(activity_dir_path: str) -> int:
    norm = activity_dir_path.replace("\\", "/")
    m = re.search(r"/a(\d+)$", norm)
    if not m:
        m = re.search(r"a(\d+)$", norm)
    return int(m.group(1)) if m else -1



def window_to_features(df_window: pd.DataFrame,
                       subject_id: int,
                       activity_id: int) -> dict:
    data = df_window.to_numpy(dtype=float)
    T, C = data.shape

    feats = {}
```

```
    feats["window_length"] = T
    feats["subject_id"] = subject_id  # helps model personalize patterns

    # Feature extraction for each sensor channel
    for col_idx in range(C):
        col = data[:, col_idx]

        # basic stats
        col_mean = float(np.mean(col))
        col_std = float(np.std(col, ddof=0))
        col_min = float(np.min(col))
        col_max = float(np.max(col))
        col_range = float(col_max - col_min)

        # signal energy & RMS
        col_rms = float(math.sqrt(np.mean(col ** 2)))
        col_energy = float(np.sum(col ** 2) / len(col))

        # distribution shape
        # guard for very short series (skew/kurt need >2, >3 samples)
        col_skew = float(skew(col, bias=False)) if T > 2 else 0.0
        col_kurt = float(kurtosis(col, fisher=True, bias=False)) if T > 3
else 0.0

        feats[f"c{col_idx}_mean"] = col_mean
        feats[f"c{col_idx}_std"] = col_std
```

```
        feats[f"c{col_idx}_min"] = col_min
        feats[f"c{col_idx}_max"] = col_max
        feats[f"c{col_idx}_range"] = col_range
        feats[f"c{col_idx}_rms"] = col_rms
        feats[f"c{col_idx}_energy"] = col_energy
        feats[f"c{col_idx}_skew"] = col_skew
        feats[f"c{col_idx}_kurt"] = col_kurt

    return feats



def load_dataset(base_dir: str,
                 max_files_per_activity=None):
    feature_rows = []
    labels = []
    groups = []

    activity_dirs = sorted(glob.glob(os.path.join(base_dir, "a*")))

    for act_dir in activity_dirs:
        activity_id = extract_activity_id(act_dir)

        # gather all (person_dir, segment_file) pairs for this activity
        segment_file_entries = []
        person_dirs = sorted(glob.glob(os.path.join(act_dir, "p*")))
        for person_dir in person_dirs:
```

```
            seg_files = sorted(glob.glob(os.path.join(person_dir,
"s*.txt")))

            for seg_file in seg_files:

                segment_file_entries.append((person_dir, seg_file))



        # optional cap to limit runtime

        if max_files_per_activity is not None:

            segment_file_entries =
segment_file_entries[:max_files_per_activity]

        for (person_dir, seg_file) in segment_file_entries:

            subject_id = extract_subject_id(person_dir)

            df_seg = pd.read_csv(seg_file, header=None)



            feats = window_to_features(

                df_window=df_seg,

                subject_id=subject_id,

                activity_id=activity_id

            )



            feature_rows.append(feats)

            labels.append(activity_id)

            groups.append(subject_id)



    X = pd.DataFrame(feature_rows)

    y = np.array(labels, dtype=int)

    groups = np.array(groups, dtype=int)
```

```
    return X, y, groups



def subject_stratified_split(X, y, groups, test_size=0.25,
random_state=42):
    splitter = GroupShuffleSplit(
        n_splits=1,
        test_size=test_size,
        random_state=random_state
    )
    train_idx, test_idx = next(splitter.split(X, y, groups=groups))


    X_train = X.iloc[train_idx].reset_index(drop=True)
    X_test = X.iloc[test_idx].reset_index(drop=True)
    y_train = y[train_idx]
    y_test = y[test_idx]


    train_subjects = np.unique(groups[train_idx])
    test_subjects = np.unique(groups[test_idx])


    print("Train subjects:", train_subjects)
    print("Test  subjects:", test_subjects)


    return X_train, X_test, y_train, y_test
```

```python
def train_random_forest(X_train, y_train, random_state=42):
    model = RandomForestClassifier(
        n_estimators=300,
        random_state=random_state,
        class_weight="balanced_subsample",
        n_jobs=-1,
    )
    model.fit(X_train, y_train)
    return model



def train_mlp(X_train, y_train, random_state=42):
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)

    model = MLPClassifier(
        hidden_layer_sizes=(256, 128, 64),
        activation="relu",
        solver="adam",
        max_iter=200,
        random_state=random_state,
    )
    model.fit(X_train_scaled, y_train)

    return scaler, model
```

```
def evaluate_model(model_name, model, X_eval, y_true, scaler=None):
    if scaler is not None:
        X_eval_proc = scaler.transform(X_eval)
    else:
        X_eval_proc = X_eval

    y_pred = model.predict(X_eval_proc)

    acc = accuracy_score(y_true, y_pred)
    print(f"\n=== {model_name} ===")
    print("Accuracy:", acc)
    print(classification_report(y_true, y_pred, digits=4))

    return acc, y_pred



def save_best_model(best_dict, out_path):
    joblib.dump(best_dict, out_path)
    print(f"\nSaved best model to {out_path}")



def print_feature_importance_if_possible(best_dict):
    if best_dict["type"] != "random_forest":
        print("\nNo feature importances (best model is not
RandomForest).")
```

```
        return


    rf = best_dict["model"]
    feat_names = best_dict["feature_columns"]
    importances = rf.feature_importances_
    s = (
        pd.Series(importances, index=feat_names)
        .sort_values(ascending=False)
        .head(20)
    )
    print("\nTop 20 most important features for RandomForest:")
    print(s)
def main():
    print("[1/6] Load dataset and build features ...")
    X, y, groups = load_dataset(
        BASE_DIR,
        max_files_per_activity=MAX_FILES_PER_ACTIVITY
    )


    print(f"Samples total      : {len(y)}")
    print(f"Features per sample: {X.shape[1]}")
    print(f"Classes            : {sorted(np.unique(y).tolist())}")


    print("[2/6] Subject-aware split (no leakage) ...")
    X_train, X_test, y_train, y_test = subject_stratified_split(
        X, y, groups,
```

```
        test_size=0.25,
        random_state=RANDOM_STATE
    )

    print("[3/6] Train RandomForest ...")
    rf_model = train_random_forest(X_train, y_train,
random_state=RANDOM_STATE)

    print("[4/6] Train MLP ...")
    scaler, mlp_model = train_mlp(X_train, y_train,
random_state=RANDOM_STATE)

    print("[5/6] Evaluate models ...")
    rf_acc, _ = evaluate_model("RandomForest", rf_model, X_test, y_test,
scaler=None)
    mlp_acc, _ = evaluate_model("MLP", mlp_model, X_test, y_test,
scaler=scaler)

    # choose best by accuracy (you could also choose macro F1, etc.)
    if rf_acc >= mlp_acc:
        best = {
            "type": "random_forest",
            "model": rf_model,
            "scaler": None,
            "feature_columns": X.columns.tolist(),
        }
        print(f"\nBest model: RandomForest (acc={rf_acc:.4f})")
    else:
```

```
        best = {
            "type": "mlp",
            "model": mlp_model,
            "scaler": scaler,
            "feature_columns": X.columns.tolist(),
        }
        print(f"\nBest model: MLP (acc={mlp_acc:.4f})")

    print("[6/6] Save and interpret ...")
    save_best_model(best, MODEL_OUT_PATH)
    print_feature_importance_if_possible(best)


if __name__ == "__main__":
    main()
```