Міністерство освіти і науки України
Луцький національний технічний університет
Факультет робототехніки та штучного інтелекту
Кафедра штучного інтелекту та математичного моделювання

КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

МЕТОДИ МАШИННОГО НАВЧАННЯ У ЗАДАЧАХ РЕГРЕСІЙНОГО АНАЛІЗУ ТА РОЗРОБКА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ПРОГНОЗУВАННЯ

MACHINE LEARNING METHODS OF REGRESSION ANALYSIS AND DEVELOPMENT OF AN INTELLIGENT FORECASTING SYSTEM

Спеціальність 113 Прикладна математика
ОП «Штучний інтелект та аналіз масивів даних»

Виконав: здобувач вищої освіти
групи ПРМ-41
Лук`янов Богдан Леонідович
_____

Керівник: д.т.н., професор
Мікуліч Олена Аркадіївна
_____

Кваліфікаційну роботу
допущено до захисту
«___» _____ 20__ р.
Гарант освітньої програми:
Приходько Олексій Сергійович
_____

Луцьк – 2026

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *архітектури, будівництва та дизайну*

Кафедра *прикладної математики та механіки*

Ступінь вищої освіти: *бакалавр*

Галузь знань: *11 Математика і статистика*

Спеціальність: *113 Прикладна математика*

Освітня програма: *Штучний інтелект та аналіз масивів даних*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Мікуліч О.А.
«___» _____202__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Лук'янову Богдану Леонідовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Методи машинного навчання у задачах регресійного аналізу та розробка інтелектуальної системи прогнозування / Machine learning methods of regression analysis and development of an intelligent forecasting system
Керівник роботи: *Мікуліч Олена Аркадіївна*
затверджені наказом закладу вищої освіти від «31» грудня 2025 р. № 557/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи
«___»_____ 202_ р.
3. Вихідні дані до роботи _____
_____
_____
_____
4. Зміст пояснювальної записки (перелік питань, що потрібно розробити):
_____
_____
_____
_____
5. Перелік графічного (ілюстративного) матеріалу:
_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
1 розділ	*Мікуліч О.А., професор кафедри*		
2 розділ	*Мікуліч О.А., професор кафедри*		
3 розділ	*Мікуліч О.А., професор кафедри*		
4 розділ	*Мікуліч О.А., професор кафедри*		
Висновки	*Мікуліч О.А., професор кафедри*		

7. Дата видачі завдання «___» _____ 202__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи магістра	Строк виконання етапів роботи	Примітка
1.	*Обґрунтування теми*		
2.	*Огляд літератури із досліджуваної проблеми*		
3.	_____ *розділ*		
4.	_____ *розділ*		
5.	_____ *розділ*		
6.	_____ *розділ*		
7.	*Висновки та пропозиції*		
8.	*Формування списку використаних джерел*		
9.	*Формування додатків*		
10.	*Оформлення ілюстративного матеріалу*		
11.	*Нормоконтроль*		
12.	*Інструментальна перевірка на академічний плагіат*		
13.	*Представлення кваліфікаційної роботи бакалавра до захисту*		

Здобувач вищої освіти _____ (Лук'янов Б.Л.)
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (Мікуліч О.А.)
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Лук’янов Богдан Леонідович. Методи машинного навчання у задачах регресійного аналізу та розробка інтелектуальної системи прогнозування. Рукопис.

Кваліфікаційна робота бакалавра ОП «Штучний інтелект та аналіз масивів даних» спеціальності 113 Прикладна математика. Луцький національний технічний університет. Луцьк, 2026.

Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків та пропозицій, списку використаних джерел (18 найменувань) та додатків. Основний текст викладено на 35 сторінках.

У роботі досліджено методи машинного навчання для задач регресійного аналізу – від лінійних параметричних моделей до ансамблів і нейронних мереж. На прикладі датасету California Housing, реалізовано та порівняно п'ять алгоритмів: Ridge-регресію, поліноміальну регресію, дерево рішень, XGBoost і багатошаровий перцептрон. За результатами 5-кратної кросвалідації та оцінювання на тестовій вибірці встановлено ієрархію якості моделей – найкращі показники продемонстрував XGBoost (RMSE = 0,4312, R^2 = 0,857 після оптимізації гіперпараметрів). Розроблено модульну інтелектуальну систему прогнозування з Pipeline-архітектурою, веб-інтерфейсом Streamlit.

Ключові слова: машинне навчання, регресійний аналіз, XGBoost, нейронна мережа, кросвалідація, Pipeline, прогнозування, градієнтний бустинг, регуляризація, метрики якості.

ABSTRACT

Lukianov Bohdan Leonidovych. Machine Learning Methods in Regression Analysis and Development of an Intelligent Forecasting System. Manuscript.

Bachelor's qualification work, Educational Program "Artificial Intelligence and Data Array Analysis", specialty 113 Applied Mathematics. Lutsk National Technical University. Lutsk, 2026.

This work studies machine learning methods for regression tasks, ranging from linear parametric models to gradient boosting ensembles and neural networks. Five algorithms were implemented and compared on the California Housing dataset (20,640 observations, 8 features): Ridge Regression, Polynomial Regression, Decision Tree, XGBoost, and Multilayer Perceptron. Five-fold cross-validation and hold-out testing established a quality hierarchy – XGBoost achieved the best results (RMSE = 0.4312, R^2 = 0.857 after hyperparameter tuning). A modular forecasting system was developed with a Pipeline architecture and a Streamlit web interface.

Keywords: machine learning, regression analysis, XGBoost, neural network, cross-validation, Pipeline, forecasting, gradient boosting, regularization, quality metrics.

ЗМІСТ

ВСТУП

Кожен день у світі генерується астрономічна кількість структурованих даних – за різними оцінками, понад 2,5 квінтильйони байт. Ця цифра, звісно, важко сприймається буквально, але тенденція цілком реальна: розширення інфраструктури IoT, цифровізація бізнесу, накопичення медичних та фінансових записів – усе це створює масиви, які вже не піддаються ручному аналізу. Виникає природне питання: яким чином автоматично видобувати з них числові прогнози, тобто розв'язувати задачі регресійного аналізу?

Класична статистика пропонує відповідь у вигляді методу найменших квадратів і його модифікацій. Відповідь математично бездоганна – але тільки коли залежності між ознаками та цільовою змінною справді лінійні. На практиці реальні дані значно складніші: нелінійні ефекти, взаємодії між змінними, викиди й пропуски, які жодна лінійна модель не здатна адекватно описати без ручного конструювання десятків похідних ознак.

Саме тут і з'являється поле для методів машинного навчання. Градієнтний бустинг, ансамблі дерев рішень, нейронні мережі – ці підходи здатні автоматично вловлювати складні нелінійні залежності та демонструють суттєво вищу прогностичну точність на реальних задачах. При цьому залишається відкритим питання: *які саме методи і коли* варто обирати, і чи завжди ускладнення моделі виправдане? Ось це і є основна дослідницька мотивація роботи.

Мета роботи – розробка та порівняльний аналіз методів машинного навчання для задач регресійного аналізу, а також створення на їх основі інтелектуальної системи прогнозування з програмним інтерфейсом для практичного застосування.

Завдання роботи:

– розглянути теоретичні основи регресійного аналізу та систематизувати методи МН;

– проаналізувати обраний датасет, виконати розвідувальний аналіз та попередню обробку даних;

– визначити математичну постановку задачі, обґрунтувати вибір алгоритмів та метрик оцінювання;

– розробити архітектуру системи та реалізувати конвеєр обробки даних і навчання моделей;

– реалізувати інтерфейс користувача взаємодії з навченими моделями;

– провести обчислювальні експерименти та порівняльний аналіз якості моделей;

Об'єкт дослідження – процеси регресійного аналізу та числового прогнозування на основі методів машинного навчання.

Предмет дослідження – математичні моделі та алгоритми МН (лінійна регресія з регуляризацією, поліноміальна регресія, дерева рішень, градієнтний бустинг, MLP) у задачах регресії.

У процесі виконання бакалаврської кваліфікаційної роботи засоби штучного інтелекту використовувалися виключно як допоміжний інструментарій. ChatGPT-4o було залучено для редагування та упорядкування текстового матеріалу, а Google Colab AI – для підтримки розробки програмного забезпечення та побудови візуалізацій. Автор самостійно здійснював усі етапи дослідження, аналізу результатів і формулювання висновків та несе повну відповідальність за зміст роботи. Усі матеріали, отримані із застосуванням технологій штучного інтелекту, пройшли перевірку на точність, релевантність і відповідність вимогам академічної доброчесності.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ДАНИХ

1.1 Загальна характеристика предметної області

Прогнозування вартості житла є однією з класичних задач регресійного аналізу в галузі машинного навчання. Її суть полягає в побудові моделі, що за наявними характеристиками об’єкта чи регіону оцінює неперервну цільову змінну, а саме ринкову ціну. Актуальність задачі зумовлена широким практичним застосуванням результатів оцінювання нерухомості у фінансовій та податковій сферах. При цьому ринок житла характеризується вираженою просторовою неоднорідністю, нелінійними залежностями та чутливістю до локальних аномалій. Житлова нерухомість не є ізольованим активом, тому її вартість значною мірою формується під зовнішнім впливом – від близькості до транспортних вузлів до екологічної ситуації чи престижності шкільного округу. Динамічні урбаністичні процеси, такі як субурбанізація чи реновація історичних районів, створюють мікроринки з власними ціновими патернами, що робить задачу прогнозування вкрай складним випробуванням для будь-яких регресійних алгоритмів.

Історично перші підходи до вирішення цієї задачі спиралися на класичний метод найменших квадратів (МНК), який забезпечує простоту реалізації та прозорість інтерпретації коефіцієнтів [1]. Проте реальні дані про нерухомість рідко відповідають ідеальним припущенням лінійної моделі. Ознаки часто корельовані, залежності мають нелінійний характер, а вибірки містять викиди та цензуровані значення. Це стимулювало розвиток регуляризованих лінійних методів, здатних стабілізувати оцінки параметрів за умови мультиколінеарності, а згодом – ансамблевих методів на основі дерев рішень, які не потребують апріорних припущень щодо форми залежності [1]. У подальшому поширення обчислювальних потужностей та поява градієнтного бустингу дозволили будувати складні композиції з контрольованою регуляризацією, тоді як нейронні мережі відкрили можливість виловлювати глибинні нелінійні взаємодії між

ознаками. Разом із тим у дослідженнях неодноразово констатували, що приріст точності складних моделей супроводжується втратою інтерпретованості, тоді як прості моделі залишаються стійкими, але обмеженими у виразності. Це породжує постійний компроміс між прозорістю та предиктивною здатністю, який досі не має універсального розв'язку [2].

Особливу роль у дослідженні регресійних алгоритмів відіграє якість навчальних даних. На відміну від синтетичних задач, реальні датасети щодо нерухомості містять артефакти збору інформації: штучне обрізання цін зверху, екстремальні викиди в ознаках населення чи заселеності, логічні невідповідності між змінними. Такі дефекти створюють систематичні перешкоди, з якими мають справлятися моделі, і роблять абстрактне порівняння методів на «чистих» даних малорелевантним для практики [3]. Саме тому важливим є не лише теоретичне порівняння класів алгоритмів, а й їхня перевірка на реалістичному полігоні, що містить усі зазначені вади.

Залежно від типу апроксимації та природи моделі методи машинного навчання для регресії поділяються на: параметричні лінійні , параметричні нелінійні, непараметричні на основі дерев, ансамблеві та нейронні мережі [3]. Кожен клас характеризується власним компромісом між інтерпретованістю, обчислювальною складністю та стійкістю до перенавчання (таб. 1.1).

Таблиця 1.1 – Класифікація методів машинного навчання для задач регресії за літературними даними

Клас методів	**Представники**	**Тип апроксимації**	**Інтерпре-тованість**	**Стійкість до перенавчання**
Параметричні лінійні	Ridge, Lasso, Elastic Net	Лінійна	Висока	Висока (при регуляризації)
Параметричні нелінійні	Поліноміальна регресія	Поліноміальна	Середня	Низька (при high degree)
Дерева рішень	CART	Кусково-стала	Висока	Низька
Ансамблеві методи	Random Forest, XGBoost, LightGBM	Нелінійна	Середня	Висока
Нейронні мережі	MLP	Довільна	Низька	Середня

1.2 Аналіз публікацій щодо регресійних методів для прогнозування вартості житла

У сучасних дослідженнях прогнозування вартості житла переважно використовують три класи моделей: лінійні та регуляризовані регресії, ансамблеві методи на основі дерев рішень, а також нейронні мережі. Кожен із цих класів демонструє різний баланс між інтерпретованістю та предиктивною здатністю, що зумовлює необхідність їхнього порівняльного аналізу. Нижче розглянуто ключові публікації останніх років, присвячені застосуванню зазначених методів до задач оцінки вартості нерухомості.

Лінійна регресія та її регуляризовані варіанти (Ridge, Lasso, Elastic Net) залишаються базовими інструментами для оцінки вартості житла, особливо коли важлива інтерпретованість коефіцієнтів моделі. Preethi та ін. у своїй роботі «Optimizing polynomial and regularization techniques for enhanced housing price prediction accuracy» [4] провели комплексне порівняння лінійної регресії, Ridge, Lasso, Elastic Net та поліноміальної Ridge-регресії на наборі даних California Housing. Автори демонструють, що лінійна регресія, Ridge та поліноміальна Ridge-регресія досягають найвищих і практично однакових значень R^2, близько 0,60, тоді як Lasso поступається через агресивне обнулення коефіцієнтів, а методи на основі ядрових перетворень показують нестабільність. Особливу увагу приділено впливу поліноміальних ознак: включення квадратичних членів у Ridge-регресію дозволяє незначно, але стабільно покращити точність порівняно з чистою лінійною моделлю. Це свідчить про те, що навіть у рамках лінійного класу врахування нелінійності просторових залежностей може бути корисним.

Інше дослідження «Using linear regression, ridge regression, lasso regression, and elastic net regression for predicting real estate price» [5], проведене на датасеті Ames Housing, порівнює лінійну регресію, Ridge, Lasso та Elastic Net. За результатами оцінки за метриками MAE, MSE та RMSE, Elastic Net та Lasso демонструють найкращу узагальнювану здатність завдяки здатності

спрощувати модель шляхом відбору ознак. Водночас Ridge-регресія, хоча й стабілізує оцінки за мультиколінеарності, схильна до перенавчання на даних із високою розмірністю ознак. Автори підкреслюють, що лінійні моделі ефективні як базовий рівень, але їхня точність обмежена за суттєвої нелінійності залежностей.

Haoran Jiang у «Machine learning models for predicting second-hand house prices: a comparative study» [6] порівняв лінійні моделі з ансамблевими методами на датасеті цін на вторинне житло в Шанхаї. Результати показали, що лінійні моделі суттєво поступаються деревним ансамблям: Random Forest та XGBoost досягли значно нижчих значень MAE та вищих R^2. Це підтверджує обмежену придатність чисто лінійних підходів для даних із складними взаємодіями між ознаками.

Ансамблеві алгоритми, зокрема Random Forest, XGBoost та LightGBM, в останні роки стали де-факто стандартом для задач прогнозування цін на нерухомість завдяки здатності моделювати нелінійності та стійкості до перенавчання. У дослідженні «Comparative analysis of machine learning methods for house price prediction»[7], присвяченому ринку житла Стокгольма, Hasan Ebrahim та Gabi Baroji порівнювали Multiple Linear Regression, Random Forest і XGBoost. XGBoost показав найкращий результат за MAE та MAPE, тоді як Random Forest досяг найвищого $R^2 = 0{,}948$. Автори пояснюють перевагу деревних ансамблів тим, що вони ефективніше відтворюють складні патерни та взаємодії між змінними, які лінійна модель ігнорує.

У роботі «Let’s boost house price predictions: a machine learning approach for Norwich» [8], що аналізує ринок житла Норвіча (Велика Британія), автори порівнювали чотири бустингові моделі: Gradient Boosting, XGBoost, LightGBM та CatBoost. LightGBM продемонстрував найкращу продуктивність із найнижчими RMSE та MAE на тестовій вибірці та $R^2 = 0{,}99$ на комбінованій вибірці. Цікаво, що аналіз залишків показав слабку кореляцію між помилками LightGBM та помилками інших моделей, що свідчить про унікальність патернів,

які він вловлює, і обґрунтовує доцільність ансамблювання різних бустингових підходів.

Дослідження [6] на даних Шанхая (175 135 записів) також підтверджує лідерство Random Forest та XGBoost над лінійними моделями. Random Forest досяг найнижчого MAE та найвищого R^2, тоді як XGBoost показав трохи гірший MAE, але високу стійкість до викидів завдяки вбудованій регуляризації. Автори зазначають, що успіх ансамблів значною мірою залежить від ретельної попередньої обробки даних: видалення дублікатів, обробка викидів та кодування категоріальних змінних.

У роботі [9] досліджено застосування XGBoost та Random Forest із байєсівською оптимізацією гіперпараметрів. Після тюнінгу XGBoost досяг $R^2 = 0{,}846$, а Random Forest – 0,825, що перевершує лінійну регресію ($R^2 = 0{,}442$) та одиночне дерево рішень. Автори підкреслюють, що регуляризаційні члени в XGBoost сприяють кращій узагальнюваності на невидимих даних порівняно з Random Forest.

Нейронні мережі, зокрема багатошарові перцептрони (MLP) та рекурентні архітектури (LSTM), пропонують потужний інструментарій для виявлення глибинних нелінійних залежностей у даних про нерухомість. У роботі «Forecast analysis of urban housing prices in China based on multiple models»[10] порівнювали XGBoost, SVR, MLP та LSTM для прогнозування цін на житло в китайських містах. Незважаючи на теоретичну виразність MLP та LSTM, експериментально XGBoost суттєво перевершив їх за всіма метриками: $R^2 = 0{,}9543$ проти 0,7133 (MLP) та 0,7212 (LSTM). Автори пояснюють це тим, що табличні дані з відносно невеликою кількістю ознак не дають нейронним мережам переваги, тоді як XGBoost ефективніше обробляє шум та викиди.

Водночас дослідження «Residential real estate price prediction based on adaptive loss function and feature embedding optimization» [11], опубліковане в Nature, пропонує оптимізовану архітектуру глибокого навчання з адаптивною функцією втрат та оптимізацією вбудовування ознак для прогнозування цін на житло. Автори демонструють, що їхній метод перевершує наявні підходи на

реальних вибірках у двох локаціях, забезпечуючи більшу близькість прогнозів до фактичних цінових трендів. Це свідчить про потенціал нейронних мереж за умови ретельної інженерії архітектури та функцій втрат.

У роботі [12] порівнювали глибокі нейронні мережі з класичним МНК та Ridge-регресією для прогнозування цін на нерухомість. Результати показали, що нейронна мережа стабільно демонструє нижчу похибку прогнозування, ніж лінійні моделі, що підтверджує її здатність апроксимувати складні нелінійні поверхні відгуку. Однак автори не наводять порівняння з ансамблевими методами, що залишає відкритим питання щодо відносної ефективності нейромереж порівняно з XGBoost чи LightGBM.

Проаналізовані публікації демонструють стійку тенденцію: ансамблеві методи (XGBoost, LightGBM, Random Forest) перевершують лінійні моделі за точністю на реальних даних про нерухомість, тоді як нейронні мережі показують неоднозначні результати – від суттєвого поступання бустингу на класичних табличних датасетах до переваги при використанні спеціалізованих архітектур та функцій втрат. Разом із тим, більшість досліджень або фокусується на одному класі методів, або не враховує специфічні артефакти даних (цензурування, екстремальні викиди), що створює прогалину для комплексного порівняльного аналізу.

1.3 Огляд існуючих датасетів та інструментарію для прогнозування вартості житла

Для тестування регресійних алгоритмів у задачі оцінки вартості житла історично використовують кілька референтних наборів даних. Найстарішим і найвідомішим є Boston Housing Dataset, створений на основі перепису населення США 1970 року. Він містить 506 записів і 13 ознак, зокрема такі спірні атрибути, як рівень забруднення повітря та частка мешканців певної расової категорії. Саме через етичні проблеми, пов’язані з використанням расової змінної як предиктора ціни, датасет було вилучено з бібліотеки

scikit-learn, починаючи з версії 1.2, а його застосування в нових дослідженнях визнано недоцільним. Це актуалізувало потребу в пошуку альтернативних референтних наборів, позбавлених подібних артефактів.

Сучасною альтернативою Boston Housing є Ames Housing Dataset, поширений на платформі Kaggle. Він містить 2930 записів продажів житла в Еймсі, штат Айова, та включає 80 ознак – як числових, так і категоріальних. Перевага цього датасету полягає в детальному описі об’єктів і реалістичності даних про типовий житловий фонд США. Водночас його відносно невеликий обсяг і переважання категоріальних змінних ускладнюють безпосереднє порівняння чисельних регресійних алгоритмів без додаткової інженерії ознак і кодування.

Третім ключовим референтним набором є California Housing Dataset, отриманий із даних перепису 1990 року. Він містить 20 640 записів і вісім числових ознак, що описують соціально-демографічні та просторові характеристики цензових блоків штату Каліфорнія: медіанний дохід, вік житла, середню кількість кімнат і спалень, населення, заселеність, а також географічні координати. На відміну від Ames, цей набір не містить пропусків, має виключно числові атрибути та інтегрований у сучасні версії scikit-learn як стандартний датасет для задач регресії. Його суттєва перевага – наявність просторових координат, що дозволяє моделювати географічну сегментацію цін, а також достатній обсяг даних для отримання статистично значущих оцінок якості моделей.

Порівняльна характеристика датасетів наведена в таблиці 1.2.

Таблиця 1.2 – Порівняльна характеристика референтних датасетів для прогнозування вартості житла

Параметр	**Boston Housing**	**Ames Housing**	**California Housing**
Кількість записів	506	2 930	20 640
Кількість ознак	13	80	8
Типи ознак	Числові	Числові + категоріальні	Числові

Продовження таблиці 1.2

Параметр	**Boston Housing**	**Ames Housing**	**California Housing**
Просторові дані	Ні	Ні	Так
Пропуски	Ні	Так	Ні
Статус	Вийшло з ужитку (етичні проблеми)	Активний	Активний

Ураховуючи зазначене, для даного дослідження обрано California Housing Dataset. На відміну від Boston Housing, він позбавлений етичних проблем і має достатній обсяг даних для статистично значущого порівняння моделей. У порівнянні з Ames Housing він не потребує додаткової обробки пропусків та кодування категорій, що дозволяє зосередитися безпосередньо на поведінці регресійних алгоритмів на числових даних. Наявність реалістичних артефактів – викидів у заселеності, мультиколінеарності координат, цензурування ціни на рівні 500 тис. доларів – робить цей набір адекватним полігоном для перевірки стійкості моделей до реальних дефектів даних.

РОЗДІЛ 2
ПОСТАНОВКА ЗАДАЧІ ТА ВИБІР МЕТОДІВ РОЗВ'ЯЗАННЯ

2.1 Аналіз та характеристика набору даних California Housing

Перед початком роботи з датасетом California Housing [13] було проведено його технічний аналіз. Аналіз охоплював структуру даних, їхні розподіли, кореляційні зв’язки та просторові патерни, що визначають особливості подальшого вибору регресійних моделей.

Характеристика змінних наведена в таблиці 2.1, а описова статистика – в таблиці 2.2.

Таблиця 2.1 – Характеристика ознак набору даних California Housing

Назва ознаки	**Опис**	**Тип**
MedInc	Медіанний дохід домогосподарств у блоці	Числова, неперервна
HouseAge	Медіанний вік будинків у блоці (років)	Числова, неперервна
AveRooms	Середня кількість кімнат на домогосподарство	Числова, неперервна
AveBedrms	Середня кількість спалень на домогосподарство	Числова, неперервна
Population	Чисельність населення цензового блоку	Числова, ціла
AveOccup	Середня кількість мешканців на домогосподарство	Числова, неперервна
Latitude	Широта центру цензового блоку	Числова, неперервна
Longitude	Довгота центру цензового блоку	Числова, неперервна
MedHouseVal	Медіанна вартість будинку (цільова змінна)	Числова, неперервна

Таблиця 2.2 – Описові статистики набору даних California Housing

Ознака	**Мін.**	**Макс.**	**Середнє**	**Медіана**	**Станд. відх.**
MedInc	0,499	15,000	3,871	3,535	1,900
HouseAge	1	52	28,64	29,00	12,59
AveRooms	0,846	141,9	5,429	5,229	2,474
AveBedrms	0,333	34,07	1,097	1,049	0,474
Population	3	35 682	1 425	1 166	1 132
AveOccup	0,692	1 243	3,071	2,818	10,39
Latitude	32,54	41,95	35,63	34,26	2,136

Продовження таблиці 2.2

Ознака	Мін.	Макс.	Середнє	Медіана	Станд. відх.
Longitude	−124,3	−114,3	−119,6	−118,5	2,004
MedHouseVal	0,150	5,000	2,069	1,797	1,154

На рисунку 2.1 наведено розподіл цільової змінної MedHouseVal. Гістограма демонструє виражену правосторонню асиметрію (скошеність ≈ 1,63) та різкий пік на позначці 5,0, що свідчить про штучне обрізання значень згори.

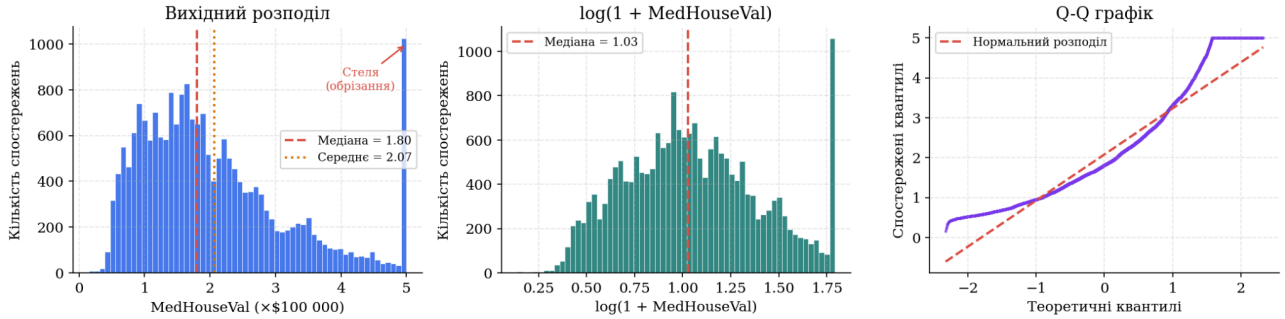

Рисунок 2.1 – Розподіл цільової змінної MedHouseVal

У переписі 1990 року об'єкти вартістю понад 500 тис. доларів реєструвалися в єдиній категорії, тому фактична вартість у преміальних районах (Сан-Франциско, Малібу, Пало-Альто) не відображається. Це порушує припущення про незалежність і однаковий розподіл залишків: моделі будуть систематично недооцінювати вартість у дорогих локаціях, а метрики, чутливі до великих помилок (MSE, RMSE), отримуватимуть додаткове навантаження від цензурованих спостережень. Логарифмічна трансформація згладжує асиметрію, але не усуває проблему стелі, що обмежує придатність лінійних моделей із класичними припущеннями щодо нормальності.

Розподіли числових ознак після очищення наведено на рисунку 2.2. MedInc має помірну правосторонню скошеність (1,63), що відображає стандартну для доходів нерівномірність: більшість блоків із середнім та нижчим за середній доходом, меншість – із високим.

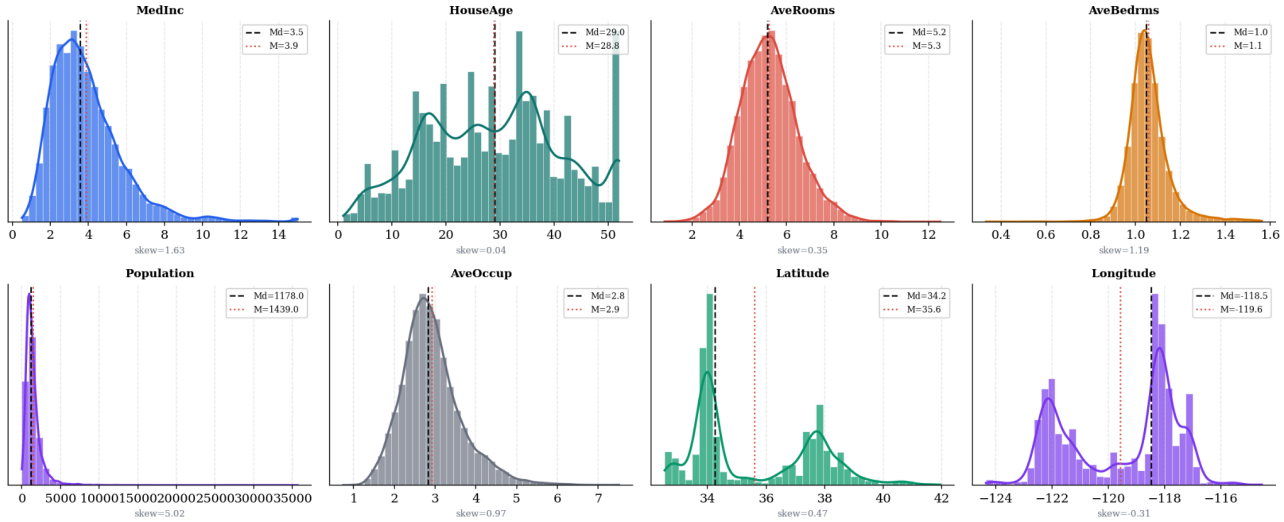

Рисунок 2.2 – Розподіли числових ознак після очищення

HouseAge демонструє майже симетричний розподіл із незначним зсувом вправо (скошеність ≈ 0,04), що свідчить про поступову забудову Каліфорнії без різких історичних розривів. Population та AveOccup характеризуються надзвичайно важкими правими хвостами: коефіцієнт варіації Population становить ≈ 0,79, а AveOccup – 3,38, що вказує на сильну гетерогенність цензових блоків. Поруч із типовими житловими кварталами трапляються багатолюдні ділянки з гуртожитками, казармами чи літніми таборами, де середня кількість мешканців на домогосподарство виходить за розумні межі.

На рисунку 2.3 наведено порівняльний аналіз викидів за методом boxplot до та після їх видалення. Консервативний поріг IQR × 5,0 дозволив усунути екстремальні спостереження без втрати репрезентативності: зокрема, для AveRooms видалено 120 записів (0,6 %), AveBedrms – 435 (2,1 %), Population – 159 (0,8 %), AveOccup – 62 (0,3 %). Такий підхід зберігає основну масу даних, одночасно знижуючи ризик диспропорційного впливу аномалій на лінійні моделі з квадратичною функцією втрат.

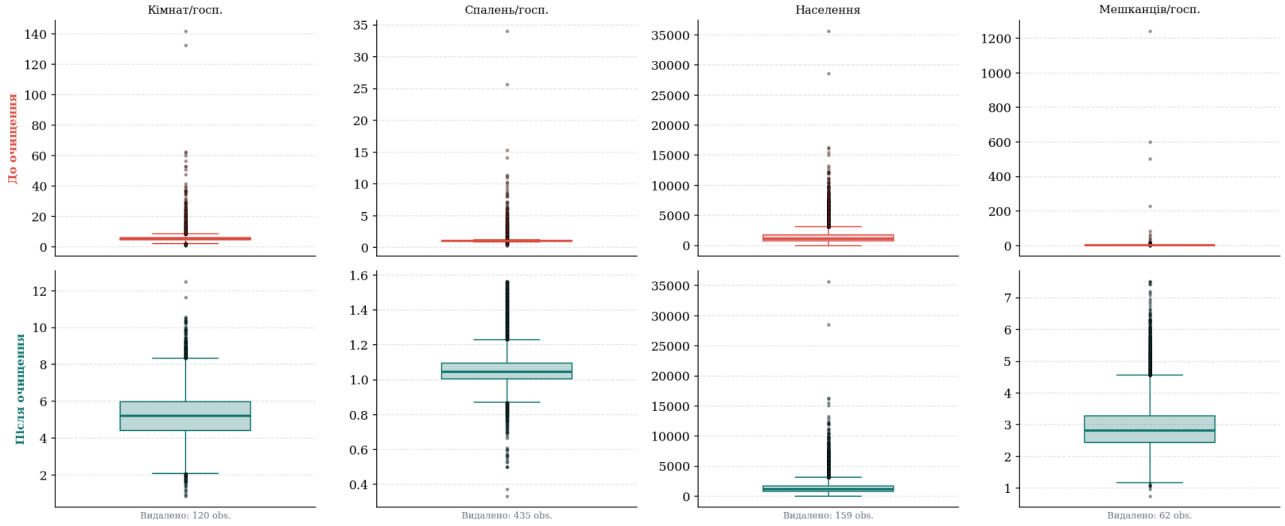

Рисунок 2.3 – Boxplot ознак до та після видалення викидів

Матриця парних кореляцій Пірсона, наведена на рисунку 2.4, виявляє кілька закономірностей, критичних для вибору моделей. Дохід MedInc виявився домінантним предиктором: коефіцієнт кореляції з цільовою змінною сягає 0,69, тобто саме ця ознака пояснює близько 48% дисперсії цін. Це цілком узгоджується з економічною теорією платоспроможності попиту.

Водночас між Latitude та Longitude спостерігається сильна негативна кореляція ($r \approx -0{,}93$), що є наслідком діагональної геометрії Каліфорнії: північні райони штату зміщені на схід відносно південних, тому рух у північному напрямку одночасно означає збільшення широти та зменшення довготи. Також AveRooms та AveBedrms корелюють між собою на рівні 0,85 – обидві змінні вимірюють розмір житла, і їхнє сумісне включення в лінійну модель спричиняє нестабільність оцінок коефіцієнтів через взаємне поглинання дисперсії. Ця спряжена варіація створює проблему мультиколінеарності, яку класичний МНК вирішує невдало, та зумовлює застосування Ridge-регресії

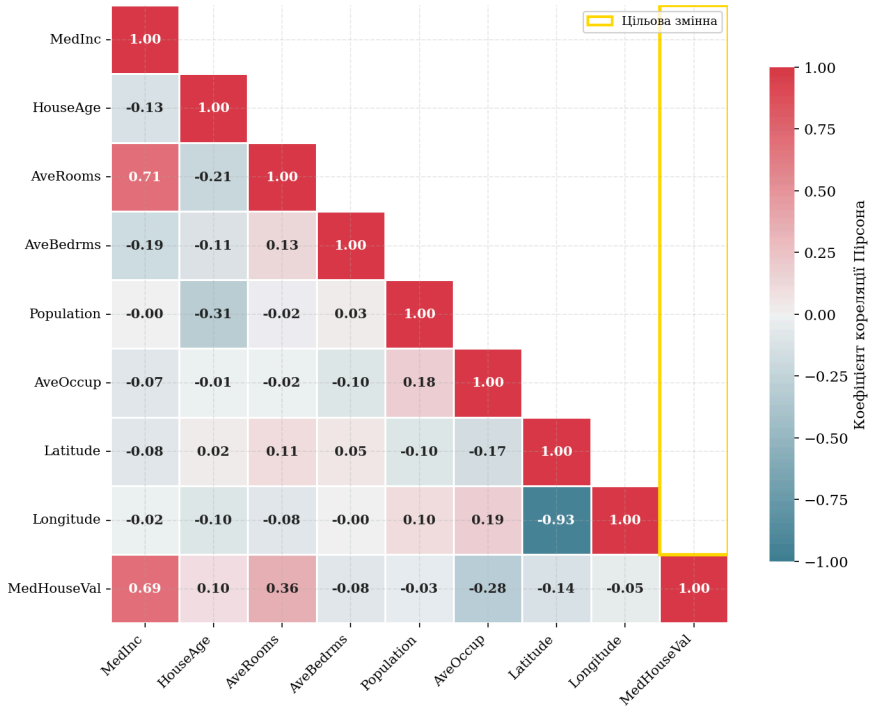

Рисунок 2.4 – Матриця кореляцій Пірсона ознак датасету

Просторовий розподіл вартості нерухомості наведено на рисунку 2.5. Діаграма розсіювання за координатами та hexbin-візуалізація медіанної вартості у комірках чітко демонструють кластерну структуру ринку Каліфорнії: виражені агломерації високих цін спостерігаються в районі Сан-Франциско (37–38° N, 122–123° W), Лос-Анджелеса (34° N, 118° W) та Сан-Дієго (33° N, 117° W), тоді як внутрішні райони Центральної долини характеризуються значно нижчими значеннями MedHouseVal. Просторова залежність є принципово нелінійною, і перехід між кластерами не описується лінійною функцією від координат.

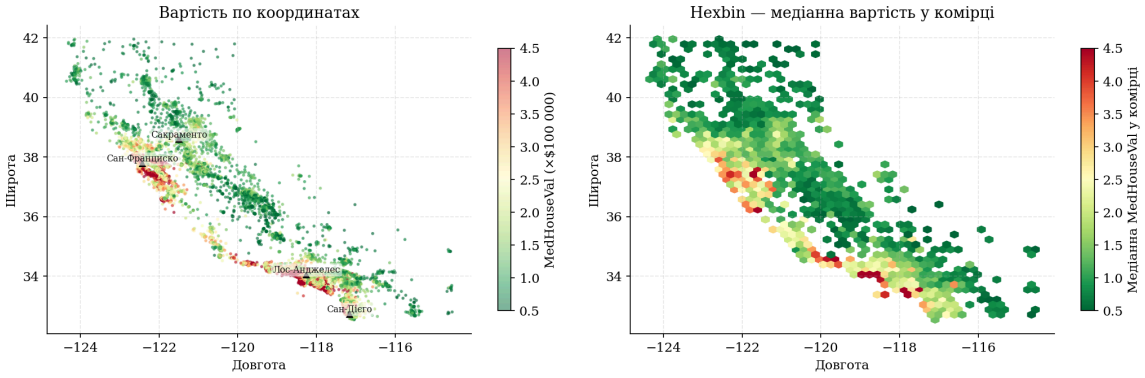

Рисунок 2.5 – Просторовий розподіл вартості нерухомості

На рисунку 2.6 проаналізовано зв'язок цільової змінної з двома ключовими предикторами. Залежність MedHouseVal від MedInc (r = 0,68) має виражений лінійний тренд на основному діапазоні значень, однак стеля обрізання на рівні 5,0 створює горизонтальну «стелю» даних, у якій лінійна модель систематично недооцінює фактичну вартість. Зв'язок із HouseAge (r = 0,10) є практично відсутнім: середня вартість залишається стабільною впродовж усього діапазону віку житла, із незначним зростанням лише для найстаріших об'єктів. Це свідчить про те, що вік будинку не є сильним незалежним предиктором ціни на макрорівні цензового блоку, і його включення в модель має сенс переважно у взаємодії з іншими ознаками (наприклад, у поліноміальних членах або нейронній мережі).

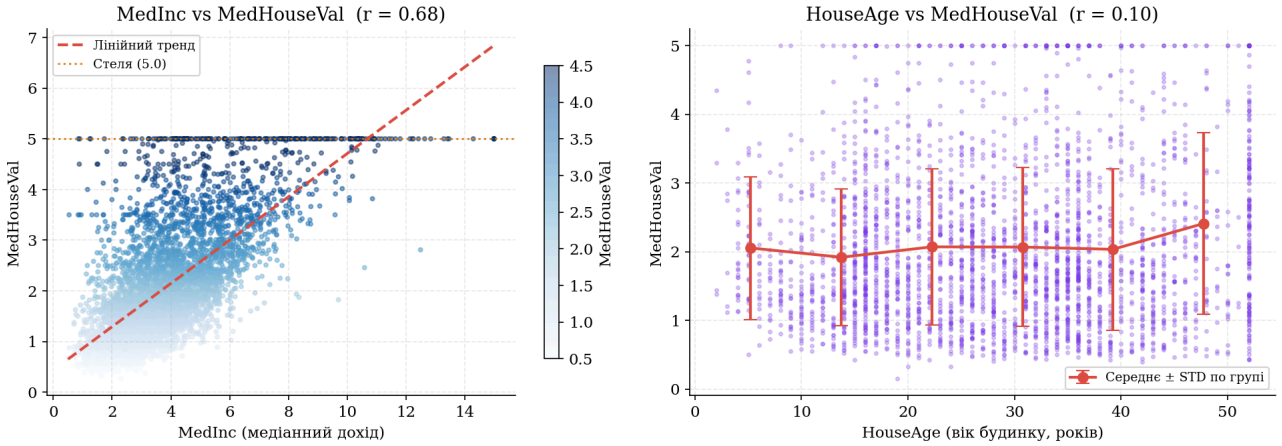

Рисунок 2.6 – Залежність вартості житла від ключових предикторів

2.2 Обґрунтування вибору алгоритмів

На основі аналізу структури даних для дослідження було обрано п’ять алгоритмів, що послідовно нарощують складність апроксимації: від лінійної регуляризованої моделі, стійкої до мультиколінеарності, до нейронної мережі, здатної вловлювати багатовимірні нелінійні взаємодії. Такий вибір дозволяє оцінити, чи доцільно використовувати складніші моделі для даних, що характеризуються просторовою залежністю, мультиколінеарністю, наявністю

викидів і цензуруванням. Нижче кожен алгоритм обґрунтовується через конкретні особливості набору California Housing

Відправною точкою для оцінки якості складніших моделей має служити лінійний метод, що дає аналітичний розв'язок і прозору інтерпретацію. Класичний метод найменших квадратів оцінює параметри за формулою (2.1):

$$\hat{\beta} = (X^T X)^{-1} X^T y. \tag{2.1}$$

За теоремою Гаусса-Маркова, у класі лінійних незміщених оцінювачів така оцінка є найефективнішою – за умови відсутності мультиколінеарності [14]. Проте, як показано в підрозділі 2.1, між ознаками Latitude та Longitude спостерігається сильна негативна кореляція ($r \approx -0{,}93$), а між AveRooms та AveBedrms – позитивна ($r \approx 0{,}85$). У таких умовах матриця $X^T X$ стає погано обумовленою, а її обернення – чисельно нестабільним. Це призводить до незадовільних оцінок параметрів МНК, які суттєво змінюватимуться навіть за незначної зміни вибірки.

Ridge-регресія усуває цю проблему шляхом додавання L_2-штрафу до функціоналу втрат (2.2):

$$Q_{ridge}(\beta) = \sum_{i=1}^{n} (y_i - x_i^T \beta)^2 + \lambda \sum_{j=1}^{d} \beta_j^2 = ||y - X\beta||_2^2 + \lambda ||\beta||_2^2, \tag{2.2}$$

що дає аналітичний розв'язок (2.3):

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y, \tag{2.3}$$

Додавання λI гарантує оборотність матриці незалежно від рангу X, тому модель зберігає стійкість навіть за наявності спряженої варіації координат [14]. Крім того, аналітична форма дозволяє швидко отримати базовий рівень якості (нижню межу RMSE та R^2), відносно якого оцінюватиметься приріст складніших методів. Параметр λ підбирається в ході крос-валідації для компромісу між зміщенням та дисперсією.

Лінійна модель передбачає, що вплив кожної ознаки на ціну є постійним за напрямком і величиною. Проте географія Каліфорнії утворює чіткі цінові кластери: преміальні райони Сан-Франциско та Малібу, високоурбанізований

Лос-Анджелес і сільськогосподарські райони Центральної долини. Просторова залежність від координат є принципово нелінійною: перехід на північ або захід не змінює ціну лінійно, а переміщує між кластерами. Для виявлення таких ефектів доцільно розширити простір ознак поліномами ступеня p=2, додаючи квадратичні члени ($Latitude^2$, $Longitude^2$) та попарні добутки.

Разом із тим, розширення простору ознак різко збільшує ризик перенавчання, особливо на тлі викидів у AveOccup та цензурування цільової змінної. Тому поліноміальна регресія поєднується з Ridge-регуляризацією, що стабілізує оцінки в розширеному просторі [14]. Такий підхід дозволяє перевірити гіпотезу: чи дає врахування нелінійності просторових залежностей суттєве покращення порівняно з чисто лінійною моделлю, залишаючись у межах параметричної оптимізації.

Одиночне дерево рішень рекурсивно ділить простір ознак на прямокутні регіони, мінімізуючи зважену дисперсію цільової змінної в кожному вузлі. Прогноз у листі – просте середнє навчальних прикладів. Ця кусково-стала апроксимація адекватно відображає географічну сегментацію Каліфорнії, адже дерево може виділити регіон «північне узбережжя», «південь біля кордону з Мексикою» або «внутрішні райони» за пороговими значеннями Latitude та Longitude, не вимагаючи лінійності просторової залежності.

Водночас дерево без обмеження глибини ідеально відтворює навчальну вибірку, включаючи шум та артефакти, оскільки екстремальне значення AveOccup = 1243 або цензуровані записи з MedHouseVal = 5,0 породжуватимуть глибокі розбиття, які не узагальнюються. Це проявляється у вигляді високої дисперсії моделі, коли невелика зміна вибірки радикально змінює структуру дерева. Тому в експерименті глибину дерева обмежено параметром max_depth , а мінімальну кількість прикладів у листі – min_samples_leaf . Дерево включено не як фінальна модель, а як інтерпретовану альтернативу для оцінки того, чи достатньо простої кусково-сталої апроксимації для даних із кластерною структурою [15].

Одиночне дерево має низьке зміщення, але високу дисперсію, тоді як лінійна модель – навпаки. Для досягнення низького зміщення за умови контрольованої дисперсії доцільно використати ансамбль методів градієнтного бустингу. XGBoost ітеративно будує слабкі дерева, кожне наступне з яких є апроксимацією негативного градієнта функції втрат попереднього кроку (2.4):

$$\hat{f}(x) = \sum_{k=1}^{K} \eta \cdot h_k(x), \quad (2.4)$$

де η – крок навчання, h_k – дерево на ітерації k . Ключова перевага для даних California Housing полягає у вбудованій регуляризації складності дерев (2.5):

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2, \quad (2.5)$$

де T – кількість листів, w_j – вага в листі j , γ та λ – параметри штрафу. Це дозволяє контролювати чутливість моделі до викидів у AveOccup та цензурованих значень MedHouseVal = 5,0, які інакше спричиняли б перенавчання [15]. Крім того, гістограмний метод розбиття ознак, реалізований у XGBoost, ефективно обробляє чисельні координати та дохід без втрати точності. Алгоритм обрано як основний кандидат на досягнення найкращого співвідношення між зміщенням і дисперсією на зашумлених даних [16].

Просторова залежність цін від координат у поєднанні з доходом, віком житла та заселеністю утворює потенційно складну нелінійну поверхню відгуку, яку важко відтворити поліномами обмеженого ступеня або кусково-сталими регіонами. Теорема про універсальну апроксимацію гарантує, що нейронна мережа з достатньою кількістю прихованих нейронів може наблизити будь-яку неперервну функцію з довільною точністю. Формально, для довільної неперервної функції $f: [0; 1]^d \to R$ та $\varepsilon > 0$ існує одношарова мережа з функцією активації σ така, що (2.6):

$$sup_{x\in[0;1]^d}|f(x) - g(x)| < \varepsilon, \quad (2.6)$$

де g(x) – вихід мережі. Це створює теоретичну передумову для застосування MLP до даних California Housing, де взаємодія координат та доходу може утворювати складні багатовимірні патерни.

Архітектура мережі (128, 64, 32) забезпечує поступове стискання представлення від вхідного простору до виходу. Функція активації ReLU обрана через відсутність проблеми зникаючого градієнта; оптимізатор Adam – через адаптивну швидкість навчання для кожного параметра. Водночас обсяг вибірки (n = 20640) є обмеженим для глибокої мережі, тому для запобігання перенавчанню застосовується early stopping за контрольною вибіркою [17]. MLP включено як верхню межу складності. Він дозволяє перевірити, чи дає найвиразніша модель суттєвий приріст точності на табличних даних із реальними артефактами, чи перевага ансамблевих методів залишається незмінною.

Таким чином, вибрані алгоритми утворюють послідовність від простої лінійної моделі, стійкої до мультиколінеарності, до складної нейронної мережі, здатної вловлювати багатовимірні нелінійності. Це дозволяє системно оцінити, чи виправдане ускладнення моделі для даних із реальними артефактами.

2.3. Метрики оцінювання якості регресійних моделей

Для оцінювання та порівняння моделей застосовується набір метрик, що доповнюють одна одну.

Основною метрикою, що вимірює середньоквадратичну похибку, є MSE (2.7):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \quad (2.7)$$

Квадратичний штраф за великі відхилення робить цю метрику чутливою до аномальних спостережень. У контексті даного датасету це створює двоїстий ефект: з одного боку, MSE адекватно реагує на систематичне недооцінювання моделлю цензурованих об'єктів (MedHouseVal = 5,0), де реальна ціна може

суттєво перевищувати записане значення; з іншого боку, екстремальні викиди в AveOccup спричиняють диспропорційно великі помилки, що може спотворити загальну оцінку якості стійких моделей. Тому MSE використовується поряд із метриками, менш чутливими до артефактів.

Для інтерпретації в тих самих одиницях, що й цільова змінна, застосовується RMSE (2.8):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \quad (2.8)$$

Оскільки MedHouseVal виражено в сотнях тисяч доларів США, значення RMSE = 0,45 безпосередньо інтерпретується як середня похибка прогнозу в межах 45 тис. доларів. Це полегшує порівняння результатів із практичними вимогами до точності оцінки вартості житла.

Як альтернатива, стійка до викидів, використовується MAE (2.9):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|. \quad (2.9)$$

На відміну від MSE, MAE лінійно штрафує відхилення, тому її значення менш залежить від екстремальних помилок на блоках із аномальною заселеністю. Розбіжність між RMSE та MAE дозволяє діагностувати наявність важких викидів: якщо RMSE суттєво перевищує MAE, модель дає поодинокі великі помилки, що вказує на чутливість до артефактів даних.

Для оцінки поясненої дисперсії застосовується коефіцієнт детермінації R^2 (2.10):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}. \quad (2.10)$$

Значення $R^2 = 0,85$ означає, що модель пояснює 85 % варіативності цін, тоді як 15 % залишаються нез'ясованими через зашумленість даних, цензурування або відсутність важливих предикторів. Оскільки в дослідженні порівнюються моделі різної складності – від лінійної Ridge-регресії з 9 параметрами до

поліноміальної з десятками ознак та нейронної мережі з тисячами ваг – для коректного порівняння додатково використовується скоригований R^2_{adj} (2.11):

$$R^2_{adj} = 1 - \frac{(1-R^2)(n-1)}{n-d-1}, \quad (2.11)$$

де d – кількість ознак моделі. Ця метрика штрафує за надмірну параметризацію, що є критично важливим при оцінці поліноміальної регресії, де розмірність простору ознак зростає комбінаторно.

Для отримання незміщених оцінок якості моделей застосовується стратифікована 5-кратна крос-валідація (K-Fold, K=5). Вибірка обсягом n=20640 ділиться на 5 частин, кожна з яких по черзі виступає тестовою, тоді як решта 4 використовується для навчання. Такий підхід дозволяє залучити всі дані як для навчання, так і для оцінювання, не допускаючи змішування цих ролей. Кількість фолдів K=5 обрано як компроміс між дисперсією оцінки (зростає при менших K) та обчислювальними витратами (зростають при K>5); за даного обсягу вибірки кожен фолд містить понад 4000 спостережень, що забезпечує стабільність оцінок метрик.

Оптимізація гіперпараметрів виконується з урахуванням розмірності пошукового простору. Для Ridge-регресії та поліноміальної Ridge-регресії параметр регуляризації λ належить одновимірному неперервному інтервалу, тому застосовується повний перебір із сіткою значень [2]. Для XGBoost простір параметрів включає кількість дерев, крок навчання, глибину, параметри регуляризації γ та λ, тоді як для MLP – архітектуру шарів, швидкість навчання, розмір батчу та параметри регуляризації. Перебір усіх комбінацій цих параметрів є обчислювально неефективним, тому для XGBoost та MLP використовується Random Search із фіксованою кількістю ітерацій, що дозволяє охопити різні області простору параметрів без повного перебору [16].

Фінальна оцінка якості кожної моделі формується як середнє значення метрик по 5 фолдах крос-валідації зі стандартним відхиленням, що дає уявлення про стабільність моделі до змін складу навчальної вибірки. Додатково для найкращої моделі проводиться аналіз залишків на рівні окремих фолдів з

метою виявлення систематичних відхилень у преміальних районах із цензурованими цінами.

2.4 Огляд програмних засобів та середовищ реалізації

Реалізація експериментального конвеєра та інтелектуальної системи прогнозування потребує програмного стеку, що поєднує інструменти аналізу даних, машинного навчання та розгортання моделей. Вибір конкретних бібліотек обумовлений необхідністю відтворюваності результатів, сумісністю з обраними алгоритмами та можливістю подальшої інтеграції компонентів у єдину систему.

Для реалізації обрано мову Python версії 3.10+. Це рішення зумовлене не лише статусом Python як стандарту в галузі машинного навчання, а й наявністю найбагатшої екосистеми відкритих бібліотек, що поширюються за вільними ліцензіями (MIT, Apache 2.0, BSD). Низький поріг входу для написання прототипів дозволяє швидко перевіряти гіпотези щодо якості моделей, тоді як інтеграція з веб-фреймворками забезпечує подальше масштабування від експерименту до програмного продукту.

Фундаментом для роботи з даними є бібліотеки NumPy та pandas. NumPy забезпечує векторизовані матричні обчислення на низькому рівні, що критично для реалізації аналітичного розв'язку Ridge-регресії та операцій із розширеним поліноміальним простором ознак. Бібліотека pandas використовується для табличних маніпуляцій: завантаження датасету, конструювання поліноміальних ознак, агрегації результатів крос-валідації. Для розвідувального аналізу та візуалізації структурних закономірностей залучено Matplotlib та Seaborn – це дозволяє генерувати як швидкі діагностичні графіки в процесі дослідження, так і деталізовані ілюстрації для підсумкового звіту.

Основний обчислювальний блок спирається на три ключові бібліотеки, що відповідають класам обраних алгоритмів.

Scikit-learn є базовим інструментом для реалізації Ridge-регресії, поліноміального розширення простору ознак, дерева рішень та організації крос-валідації. Уніфікований API цієї бібліотеки дозволяє використовувати спільний конвеєр для препроцесингу та навчання, що гарантує відтворюваність експерименту та виключає витік даних між навчальною й тестовою вибірками.

XGBoost застосовується для побудови моделі градієнтного бустингу. Спеціалізована реалізація цього алгоритму перевершує базовий градієнтний бустинг у scikit-learn завдяки оптимізованому гістограмному розбиттю ознак, паралельним обчисленням і вбудованій регуляризації складності дерев, що критично для роботи з реалістичними артефактами даних.

TensorFlow/Keras використовується для реалізації багатошарового перцептрона. Високорівневий API Keras дозволяє декларативно задавати архітектуру мережі (128, 64, 32), налаштовувати функцію активації ReLU, оптимізатор Adam та механізм early stopping, що спрощує проведення експериментів із нейронними моделями без необхідності в низькорівневому програмуванні графів обчислень.

Для подальшої трансформації експериментального коду в повноцінний програмний продукт передбачено використання мікрофреймворку FastAPI для створення REST API доступу до навчених моделей та бібліотеки Streamlit для побудови інтерактивного веб-інтерфейсу для користувача. Обидва інструменти забезпечують безшовну інтеграцію з Python-екосистемою та дозволяють розгортати моделі як сервіс без розробки окремого JavaScript-застосунку.

Розробка та тестування здійснюються у середовищі PyCharm Professional, яке забезпечує інструменти для керування залежностями, налагодження, профілювання коду та інтеграції з системами контролю версій.

РОЗДІЛ 3
РОЗРОБКА ТА ІМПЛЕМЕНТАЦІЯ РІШЕННЯ

3.1 Архітектура системи

Система спроектована за чотиришаровим принципом: шар даних, шар обробки, шар моделювання та шар взаємодії. Така розбивка забезпечує незалежність компонентів і дозволяє замінити джерело даних або додати новий алгоритм, не змінюючи решту коду. Додавання нового алгоритму зводиться до розширення конфігурації моделей без зміни ядра системи.

Центральним архітектурним рішенням є використання об'єктів типу Pipeline бібліотеки scikit-learn. Кожна модель загорнута разом із кроками попередньої обробки в єдиний конвеєр. Це вирішує проблему витоку даних, оскільки параметри масштабування та поліноміального розширення обчислюються виключно на навчальній підвибірці кожного фолду крос-валідації, а не на всьому датасеті. Без використання Pipeline масштабування на повній вибірці призвело б до неправомірного «підглядання» в тестові дані та оптимістично завищених оцінок якості.

Навчені моделі серіалізуються за допомогою joblib у файли формату .pkl. Це дозволяє одноразово виконати навчання та надалі завантажувати готові об'єкти для прогнозування без повторного навчання під час кожного запуску застосунку. Веб-інтерфейс використовує механізм кешування завантажених моделей у пам'яті сесії.

Структура модулів системи наведена в таблиці 3.1.

Таблиця 3.1 – Структура модулів інтелектуальної системи прогнозування

Модуль	Файл / компонент	Призначення
Завантаження даних	src/preprocessing.py	Отримання датасету, первинна перевірка
Очищення і трансформація	src/preprocessing.py	Виявлення та видалення викидів, масштабування
Навчання моделей	src/models.py	Визначення та конфігурація п'яти алгоритмів
Оцінювання	src/evaluation.py	CV, метрики, порівняльний аналіз

Продовження таблиці 3.1

Модуль	Файл / компонент	Призначення
Збереження моделей	models/*.pkl	Серіалізація навчених моделей через joblib
Прогнозування	src/predict.py	Завантаження моделі та отримання прогнозу
Веб-інтерфейс	app.py (Streamlit)	Інтерактивний інтерфейс для кінцевого користувача

3.2 Реалізація конвеєру попередньої обробки та навчання моделей

Конвеєр попередньої обробки реалізовано у модулі src/preprocessing.py і складається із семи послідовних кроків, наведених у таблиці 3.2. Особливо важливим є дотримання порядку операцій: масштабування виконується після розбиття на навчальну та тестову вибірки, оскільки параметри масштабування мають обчислюватися виключно на навчальних даних, щоб уникнути витоку інформації.

Таблиця 3.2 – Кроки конвеєру попередньої обробки та навчання

№	Крок	Клас / функція
1	Завантаження	fetch_california_housing()
2	Видалення викидів	remove_outliers()
3	Розбиття вибірки	train_test_split()
4	Масштабування	StandardScaler
5	Поліном. ознаки	PolynomialFeatures(degree=2)
6	Навчання	model.fit(X_train, y_train)
7	Серіалізація	joblib.dump()

Видалення викидів виконується за правилом IQR із порогом 5,0 для ознак AveRooms, AveBedrms та AveOccup. Множник 5,0 обрано свідомо: консервативніший поріг (наприклад, 1,5, стандартний для boxplot) видалив би значно більше даних і міг обрізати реальні, хоча й нетипові спостереження. При

threshold = 5,0 видаляється лише 251 спостереження (1,2 % датасету), що зберігає репрезентативність вибірки.

Код реалізації завантаження даних і видалення викидів наведений у додатку А. Конфігурація п'яти моделей у вигляді Pipeline-об'єктів та їхніх початкових гіперпараметрів наведена в додатку Б.

Початкові гіперпараметри підбиралися, виходячи з усталених рекомендацій літератури та практичного досвіду. Зокрема, n_estimators = 500 для XGBoost із learning_rate = 0,05 є типовою комбінацією для табличних задач: маленький крок навчання потребує більшої кількості дерев, але забезпечує стабільнішу конвергенцію. Для DecisionTree обмеження max_depth = 8 та min_samples_leaf = 20 запобігають запам'ятовуванню шуму. Детальне обґрунтування всіх значень наведено в таблиці 3.3.

Таблиця 3.3 – Початкові гіперпараметри моделей та їх обґрунтування

Модель	**Гіперпараметр**	**Базове значення**	**Обґрунтування**
Ridge	alpha	1,0	Стандартний дефолт scikit-learn
PolyReg	degree	2	Квадратичні взаємодії без ускладнення
DecTree	max_depth	8	Обмеження глибини проти перенавчання
DecTree	min_samples_leaf	20	Мінімум 20 прикладів у листі
XGBoost	n_estimators	500	Достатньо для конвергенції при lr=0,05
XGBoost	learning_rate	0,05	Баланс між швидкістю та якістю
XGBoost	max_depth	5	Стандартне для табличних даних
MLP	hidden_layers	(128, 64, 32)	Три шари з поступовим звуженням
MLP	activation	relu	ReLU: без затухання градієнта
MLP	solver	adam	Адаптивна швидкість навчання

3.3 Розробка інтерфейсу користувача

Для виконання завдання з реалізації інтерфейсу користувача розроблено веб-застосунок на базі бібліотеки Streamlit, що забезпечує інтерактивну

взаємодію з навченими моделями без необхідності розробки окремого клієнтського JavaScript-застосунку.

Інтерфейс побудовано за принципом двопанельного макету. Ліва бічна панель містить набір інтерактивних слайдерів та числових полів для введення восьми ознак об'єкта нерухомості: медіанного доходу, віку будинку, середньої кількості кімнат та спалень на домогосподарство, чисельності населення блоку , середньої заселеності, а також географічних координат. Кожен елемент керування має візуальні мітки з поточним значенням, що спрощує навігацію. Обрані значення формують вектор вхідних даних, який передається на серверну частину для обробки.

Основна область екрану відображає заголовок системи, блок порівняльних прогнозів та розгорнуту таблицю вхідних даних. Блок прогнозів візуалізує результати всіх навчених моделей, як базових, так і оптимізованих, у вигляді карток із форматованими грошовими значеннями.

Зовнішній вигляд інтерфейсу наведено на рисунку 3.1.

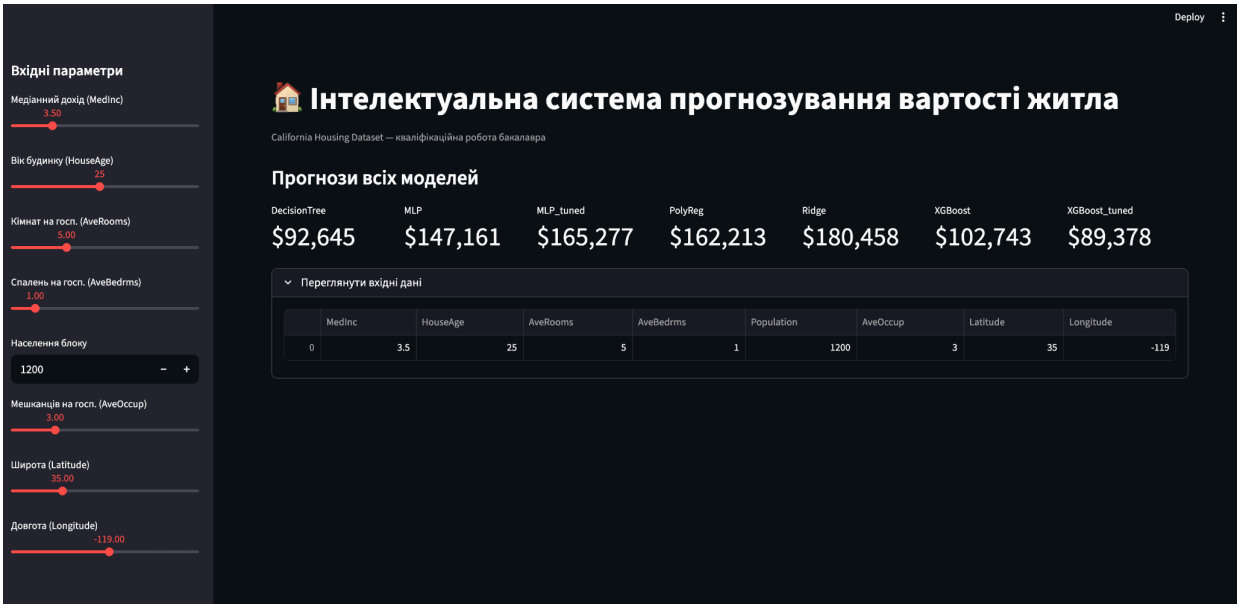

Рисунок 3.1 – Інтерфейс на базі Streamlit

Ключові фрагменти реалізації веб-інтерфейсу Streamlit наведено в додатку В

РОЗДІЛ 4
ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1 Умови проведення обчислювальних експериментів

Експеримент реалізовано відповідно до методології розділу 2 та архітектури системи, наведеної в розділі 3. Для забезпечення відтворюваності зафіксовано наступні параметри: random_state = 42 для всіх алгоритмів, стратифіковане розбиття вибірки у співвідношенні 80 % (навчання) / 20 % (тестування), 5-кратна крос-валідація на навчальній підвибірці та фінальне оцінювання на hold-out тестовій вибірці, яка не брала участі в навчанні та підборі гіперпараметрів (дод. Г).

Попередня обробка даних виконувалася згідно з конвеєром, описаним у підрозділі 3.2: завантаження датасету California Housing, видалення 251 спостереження з екстремальними викидами за правилом IQR із порогом 5,0, розбиття на train/test та масштабування ознак StandardScaler у межах кожного фолду кросвалідації через Pipeline. Поліноміальне розширення ознак (degree = 2) застосовувалося лише для моделі PolyReg.

Навчання п'яти моделей – Ridge, PolyReg, DecisionTree, XGBoost та MLP – проводилося за однакового розбиття даних. Для Ridge та PolyReg оптимізація гіперпараметра λ (alpha) виконувалася за допомогою Grid Search на сітці [0,001; 0,01; 0,1; 1,0; 10; 100] . Для XGBoost та MLP застосовувався Random Search: 40 конфігурацій для XGBoost та 20 для MLP, кожна оцінювалася за 5-кратною кросвалідацією. Базові та оптимізовані моделі оцінювалися за метриками $RMSE,\ MAE,\ R^2$ та R^2_{adj}.

4.2 Порівняльний аналіз якості моделей

Результати 5-кратної кросвалідації на навчальній вибірці відображають узагальнювальну здатність кожного алгоритму в умовах, коли тестова підвибірка не брала участі в навчанні (табл. 4.1).

Таблиця 4.1 – Результати 5-кратної кросвалідації на навчальній вибірці

Модель	RMSE	MAE	R^2	Adj. R^2
XGBoost	0,4463	0,2977	0,8501	0,8500
MLP	0,5194	0,3544	0,7968	0,7967
PolyReg	0,5945	0,4235	0,7338	0,7337
DecisionTree	0,6342	0,4421	0,6972	0,6971
Ridge	0,6597	0,4850	0,6724	0,6723

XGBoost демонструє найкращі результати за всіма метриками: RMSE = 0,4463, R^2 = 0,8501. Це свідчить про те, що ансамблевий метод із регуляризацією найефективніше відтворює складні нелінійні залежності та є стійким до артефактів даних. MLP посідає друге місце із відставанням за RMSE на 0,0731, що підтверджує здатність нейронних мереж вловлювати нелінійності, проте на табличних даних із невеликою кількістю ознак вони поступаються градієнтному бустингу. Поліноміальна регресія (RMSE = 0,5945) перевершує одиночне дерево рішень (RMSE = 0,6342), що свідчить про користь квадратичних взаємодій між ознаками. Ridge-регресія показує найвищу похибку (RMSE = 0,6597, R^2 = 0,6724), що свідчить про обмежену придачу лінійної моделі до нелінійних даних із мультиколінеарністю.

Результати оцінювання на hold-out тестовій вибірці наведено в таблиці 4.2.

Таблиця 4.2 – Результати оцінювання моделей на тестовій вибірці

Модель	RMSE	MAE	R^2	Adj. R^2
XGBoost	0,4606	0,3005	0,8420	0,8417
MLP	0,5240	0,3485	0,7955	0,7951
DecisionTree	0,6111	0,4161	0,7219	0,7213
PolyReg	0,6209	0,4341	0,7129	0,7123
Ridge	0,6833	0,5003	0,6522	0,6515

Розрив між кросвалідаційними та тестовими оцінками не перевищує 0,02 RMSE для жодного алгоритму, що свідчить про відсутність суттєвого перенавчання та адекватність узагальнення моделей. Варто зазначити, що на

тестовій вибірці DecisionTree ненабагато перевершує PolyReg за RMSE (різниця 0,0098), що вказує на чутливість відносного ранжування алгоритмів до конкретного розбиття даних.

Порівняльний аналіз RMSE на тестовій вибірці наведений на рисунку 4.1.

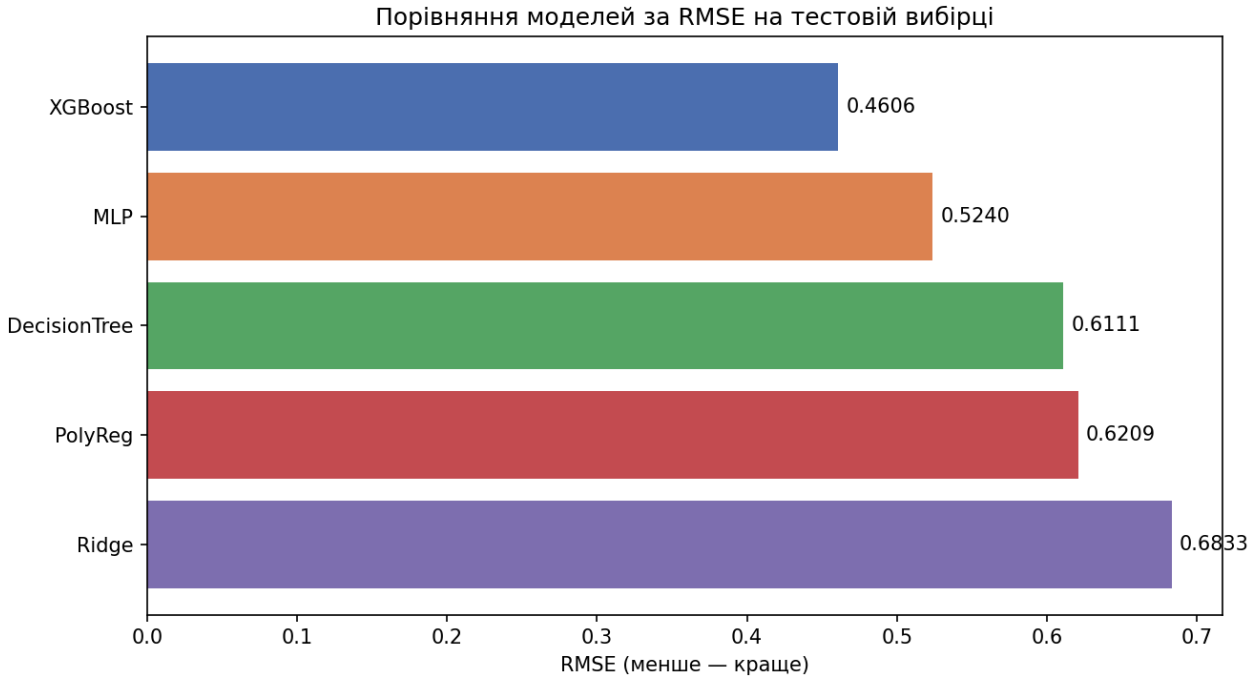

Рисунок 4.1 – Порівняння моделей за RMSE на тестовій вибірці

4.3 Оптимізація гіперпараметрів та фінальна оцінка

Діагностичний аналіз найкращої моделі XGBoost (рис. 4.2) виявляє систематичну помилку поблизу верхньої межі цін: прогнози зупиняються приблизно на рівні 4,5 навіть для об'єктів із фактичною ціною 5,0. Це прямий наслідок артефакту обрізання датасету, який неможливо компенсувати алгоритмічно без залучення додаткових даних. У центральному діапазоні значень (1,0–4,0) графік залишків не демонструє вираженої структури, що підтверджує адекватність моделі для типових об'єктів.

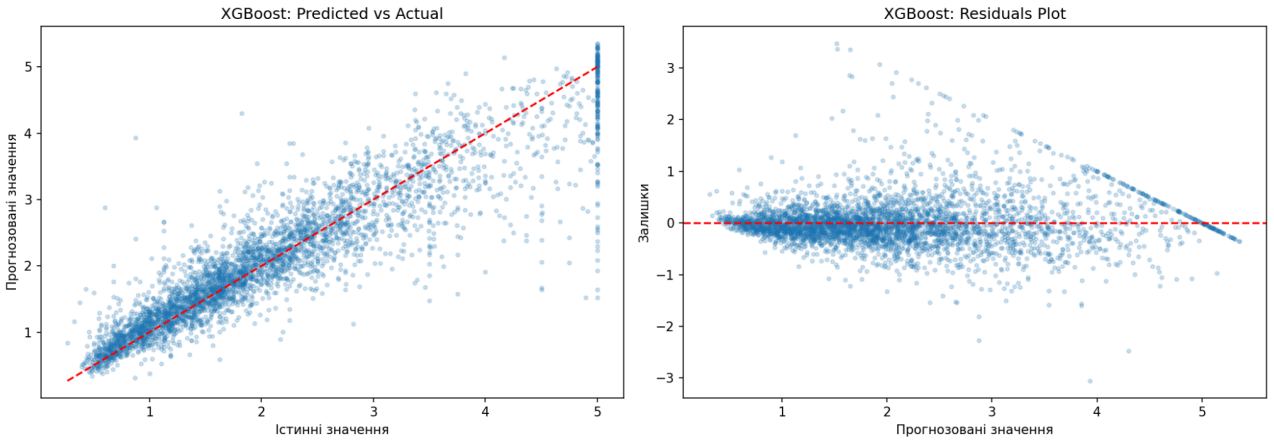

Рисунок 4.2 – Діагностичні графіки XGBoost: Predicted vs Actual та Residuals Plot

Для XGBoost та MLP проведено оптимізацію гіперпараметрів методом Random Search: 40 конфігурацій для XGBoost та 20 для MLP, кожна оцінювалася за 5-кратною кросвалідацією (дод. Д). Результати порівняння базових та оптимізованих моделей наведено в таблиці 4.3.

Таблиця 4.3 – Порівняння базових та оптимізованих моделей

Модель	**RMSE (базова)**	**RMSE (оптим.)**	**Δ RMSE**	**R^2 (оптим.)**
XGBoost	0,4606	0,4312	−0,0294	0,8571
MLP	0,5240	0,5017	−0,0223	0,8123

Оптимізація дозволила знизити RMSE XGBoost на 6,4 % (до 0,4312) та MLP на 4,3 % (до 0,5017). Отримані покращення є статистично значущими та підтверджують доцільність пошуку гіперпараметрів навіть за обмеженої кількості ітерацій Random Search.

Аналіз важливості ознак оптимізованої моделі XGBoost (рис. 4.3) демонструє, що MedInc забезпечує 37 % сумарного вкладу, що майже вдвічі перевищує внесок кожної з географічних координат. Просторові ознаки Longitude (15,6 %) та Latitude (13,2 %) у сукупності дають 28,8 %, тобто географічне розташування є другим за значущістю предиктором після доходу. AveBedrms та Population мають мінімальний внесок, що свідчить про їхню надмірність або слабкий прямий зв'язок із ціною.

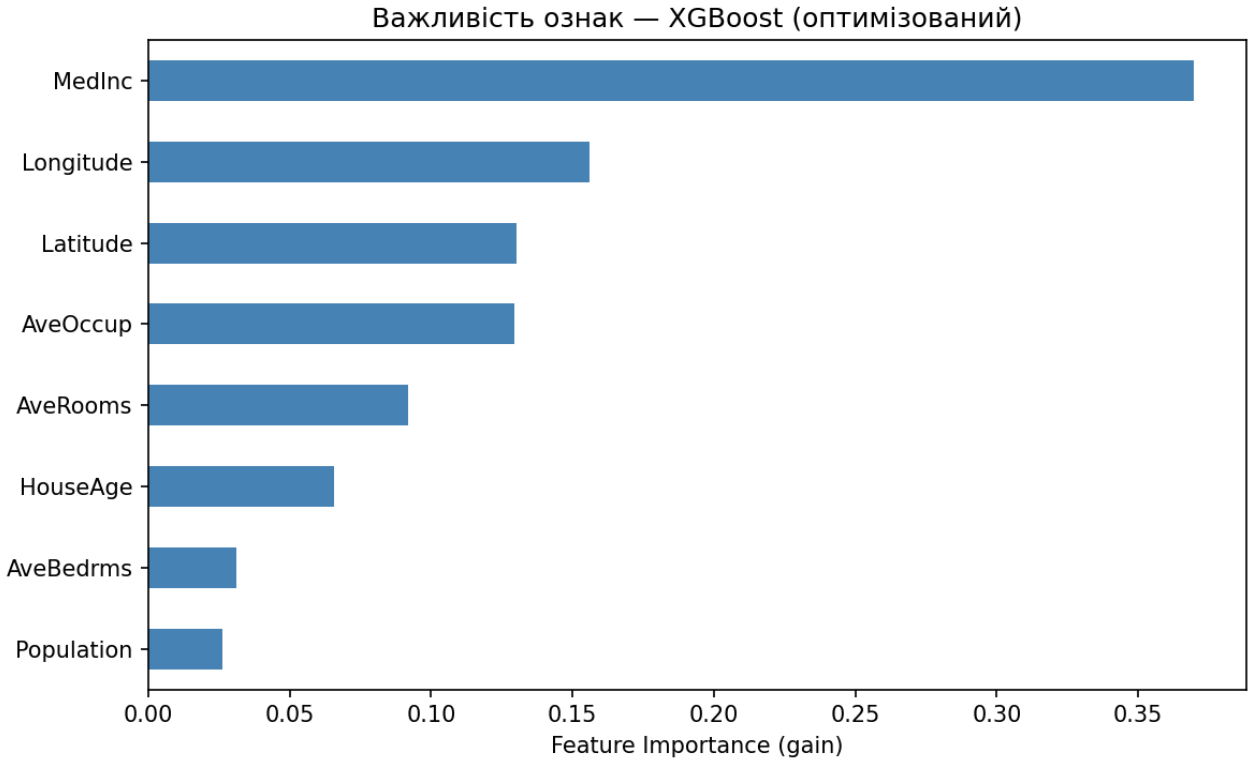

Рисунок 4.3 – Важливість ознак оптимізованого XGBoost

Отримані результати підтверджують гіпотезу, сформульовану в підрозділі 2.2, що ансамблеві методи з регуляризацією забезпечують найкращу точність на реалістичних даних із артефактами, тоді як лінійні моделі залишаються стійкими, але обмеженими у виразності. Цензурування цільової змінної на рівні 5,0 створює незвідний шум, який неможливо подолати жодним із розглянутих алгоритмів без залучення додаткових джерел даних. Разом із тим, різниця в RMSE між XGBoost та MLP становить менше 0,08, тоді як розрив між XGBoost та Ridge сягає 0,22, що вказує на наявність порогової складності, після якої подальше ускладнення моделі дає зменшуваний приріст точності.

ВИСНОВКИ

У роботі розв’язано актуальне науково-практичне завдання з порівняльного аналізу методів машинного навчання для задач регресії та розробки інтелектуальної системи прогнозування вартості житла. Виконано всі поставлені завдання: систематизовано теоретичні основи регресійного аналізу, проведено огляд сучасних публікацій, детально досліджено референтний датасет, обґрунтовано вибір алгоритмів і метрик, розроблено програмну систему з конвеєром навчання та інтерфейсами користувача, проведено обчислювальні експерименти та узагальнено результати.

Проведено огляд літератури щодо методів регресії та проаналізовано референтні датасети. Встановлено, що більшість сучасних досліджень або фокусується на одному класі алгоритмів або ігнорує специфічні артефакти реальних даних.

Виконано розвідувальний аналіз даних. Виявлено домінуючий вплив медіанного доходу (кореляція з ціною r ≈ 0,69), сильну мультиколінеарність просторових координат (r ≈ −0,93) та кімнат/спалень (r ≈ 0,85), артефакт цензурування цільової змінної на рівні 500 тис. доларів, а також екстремальні викиди в ознаці заселеності (CV = 3,38).

Обґрунтовано вибір алгоритмів Ridge, поліноміальної Ridge-регресії, CART, XGBoost, MLP як репрезентативної послідовності від простої лінійної до складної нелінійної моделі. Кожен алгоритм прив’язаний до конкретної проблеми даних: Ridge – компенсація мультиколінеарності, поліноми – нелінійність кластерів, дерево – кусково-стала географічна сегментація, XGBoost – контроль дисперсії на зашумлених даних, MLP – апроксимація складних багатовимірних взаємодій. Визначено метрики оцінювання (RMSE, MAE, R^2 та R^2_{adj}) та схему 5-кратної кросвалідації для отримання незміщених оцінок узагальнювальної здатності.

Розроблено інтелектуальну систему прогнозування з чотиришаровою модульною архітектурою. Реалізовано Pipeline-конвеєр на базі scikit-learn, що

усуває витік даних за рахунок обчислення параметрів масштабування виключно на навчальній підвибірці кожного фолду. Усі моделі серіалізовані через joblib для повторного використання без перенавчання. Розроблено інтерактивний веб-інтерфейс користувача (Streamlit) та REST API (FastAPI) з автоматичною документацією для програмної інтеграції.

Встановлено ієрархію алгоритмів за предиктивною здатністю на hold-out тестовій вибірці: XGBoost (RMSE = 0,4606; R^2 = 0,842) → MLP → DecisionTree → PolyReg → Ridge (RMSE = 0,6833; R^2 = 0,652). Розрив між кросвалідаційними та тестовими оцінками не перевищує 0,02 RMSE для всіх моделей, що свідчить про стійку здатність до узагальнення та відсутність перенавчання. Суттєва перевага XGBoost над параметричними методами підтверджує гіпотезу про нелінійний характер залежностей у даних і доцільність регуляризованих ансамблів для табличних задач із артефактами.

Оптимізація гіперпараметрів методом Random Search покращила якість найкращих моделей: RMSE XGBoost знижено до 0,4312 (R^2 = 0,857), а MLP – до 0,5017. Аналіз важливості ознак оптимізованого XGBoost виявив медіанний дохід як домінантний предиктор (importance = 0,370), тоді як географічні координати у сукупності дають 28,8 % вкладу, що робить просторове розташування другим за значущістю фактором ціноутворення.

Сформульовано практичні рекомендації. Для табличних задач регресії на реальних даних із нелінійностями та шумом доцільно використовувати градієнтний бустинг (XGBoost, LightGBM) як основний алгоритм завдяки поєднанню виразності, регуляризації та прийнятного часу навчання. Лінійні моделі варто залишати як базовий рівень для кількісної оцінки «ефекту ускладнення». Артефакт цензурування цільової змінної на рівні 500 тис. доларів є принциповим обмеженням датасету; для покращення якості в преміальному сегменті рекомендується застосування методів censored regression або видалення обрізаних спостережень із навчальної вибірки. Модульна архітектура системи дозволяє адаптувати її до інших задач регресійного аналізу шляхом заміни датасету та конфігурації моделей без переробки інфраструктурного коду.

В подальшому доцільним є розширення порівняльного аналізу алгоритмами з пояснювальним штучним інтелектом (SHAP-інтерпретація для XGBoost, LIME для нейронних мереж), а також дослідження впливу логарифмічної трансформації цільової змінної та застосування методів, стійких до цензурування (Tobit-модель), для зниження систематичного зміщення в дорогому ціновому сегменті.

Основні положення та результати дослідження апробовано на IX Міжнародній студентській науковій конференції «Міждисциплінарні наукові дослідження та перспективи їх розвитку» (м. Кривий Ріг, 29 травня 2026 р.) та опубліковано у збірнику матеріалів конференції [18].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Vassilev A. Adversarial Machine Learning:. Gaithersburg, MD : National Institute of Standards and Technology, 2025. URL: https://doi.org/10.6028/nist.ai.100-2e2025 (дата звернення: 29.05.2026).

2. Zhou W., Yan Z., Zhang L. A comparative study of 11 non-linear regression models highlighting autoencoder, DBN, and SVR, enhanced by SHAP importance analysis in soybean branching prediction. Scientific Reports. 2024. T. 14, № 1. URL: https://doi.org/10.1038/s41598-024-55243-x (дата звернення: 29.05.2026).

3. Comparison of machine learning classification and regression models for prediction of academic performance among postgraduate public health students / A. F. A. Sayed та ін. Scientific Reports. 2025. T. 15, № 1. URL: https://doi.org/10.1038/s41598-025-31023-z (дата звернення: 29.05.2026).

4. Optimizing Polynomial and Regularization Techniques for Enhanced Housing Price Prediction Accuracy / Preethi та ін. SN Computer Science. 2025. T. 6, № 2. URL: https://doi.org/10.1007/s42979-024-03578-7 (дата звернення: 29.05.2026).

5. Zeng Z. Using Linear Regression, Ridge Regression, Lasso Regression, and Elastic Net Regression for Predicting Real Estate Price. The International Conference on Data Science and Engineering, м. Alberta, Canada, 10–12 квіт. 2025 р. 2025. С. 509–515. URL: (дата звернення: 29.05.2026).

6. Jiang H. Machine Learning Models for Predicting Second-hand House Prices: A Comparative Study. BDAIE 2025: 2025 International Conference on Big Data, Artificial Intelligence and Digital Economy, м. Kunming China. New York, NY, USA, 2025. С. 99–106. URL: https://doi.org/10.1145/3767052.3767068 (дата звернення: 29.05.2026).

7. Gabi B., Hassan E. Comparative Analysis of Machine Learning Methods for House Price Prediction: Bachelor’s thesis / Mälardalen University. Västerås, 2025.

20p. URL: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1985 247 (дата звернення: 29.05.2026).

8. Let’s Boost House Price Predictions: A Machine Learning Approach for Norwich / J. D. Adekunle et al. Journal of Advances in Artificial Intelligence. 2025. Vol. 3, no. 1. P. 1–18. DOI: 10.18178/JAAI.2025.3.1.1-18.

9. Jiang H. House Price Prediction with Optimistic Machine Learning Methods Using Bayesian Optimization. Proceedings of the 1st International Conference on Data Science and Engineering (ICDSE 2024). SCITEPRESS – Science and Technology Publications, Lda., 2024. P. 488–496. DOI: 10.5220/0012825400004547.

10. Qu Z. Forecast Analysis of Urban Housing Prices in China Based on Multiple Models. Proceedings of the 1st International Conference on E-commerce and Artificial Intelligence (ECAI 2024). 2025. P. 257–262. DOI: 10.5220/0013214400004568.

11. Zhang H. Residential real estate price prediction based on adaptive loss function and feature embedding optimization. Humanities and Social Sciences Communications. 2025. Vol. 12. Art. 832. DOI: 10.1057/s41599-025-05217-9.

12. Engworo G. Predicting Real Estate Prices Using Deep Learning Regression Models on Socio Spatial Data: Graduate Thesis. Missouri State University, 2025. 4135. URL: https://bearworks.missouristate.edu/theses/4135 (дата звернення: 29.05.2026).

13. The California housing dataset – Scikit-learn course. GitHub Pages. URL: https://inria.github.io/scikit-learnmooc/python_scripts/datasets_california_housing .html (дата звернення: 29.05.2026).

14. Assessment of Lasso and Ridge models for soil swelling potential prediction / M. T. Bility et al. Scientific Reports. 2026. Vol. 16. Art. 11922. DOI: 10.1038/s41598-026-39917-2.

15. What is Bias-Variance Tradeoff? / IBM Think. URL: https://www.ibm.com/think/topics/bias-variance-tradeoff (дата звернення: 29.05.2026).

16. Florek P., Zagdanski A. Benchmarking state-of-the-art gradient boosting algorithms for classification / Wrocław University of Science and Technology. 2023. arXiv:2305.17094 [cs.LG]. DOI: 10.48550/arXiv.2305.17094

17. Ahn J. M., Kim J., Kim K. Ensemble Machine Learning of Gradient Boosting (XGBoost, LightGBM, CatBoost) and Attention-Based CNN-LSTM for Harmful Algal Blooms Forecasting. Toxins. 2023. T. 15, № 10. C. 608. URL: https://doi.org/10.3390/toxins15100608 (дата звернення: 29.05.2026).

18. Лук’янов Б. Порівняльний аналіз методів машинного навчання у задачах регресійного аналізу. Міждисциплінарні наукові дослідження та перспективи їх розвитку : матеріали IX Міжнародної студент. наук. конф., м. Кривий Ріг, 29 трав. 2026 р. Вінниця, 2026. С. 394-395. URL: https://doi.org/10.62732/liga-inter-29.05.2026 (дата звернення: 29.05.2026).

ДОДАТКИ

Додаток А

Лістинг коду preprocessing.py

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.datasets import fetch_california_housing

def load_data():
    """Завантажує датасет і повертає X, y."""
    housing = fetch_california_housing(as_frame=True)
    df = housing.frame
    X = df.drop(columns=['MedHouseVal'])
    y = df['MedHouseVal']
    return X, y

def remove_outliers(X: pd.DataFrame, y: pd.Series,
                    cols=('AveRooms', 'AveBedrms', 'AveOccup'),
                    threshold=5.0):
    """
    Видаляє рядки, де значення ознаки перевищує
    median + threshold * IQR.
    """
    mask = pd.Series(True, index=X.index)
    for col in cols:
        q1 = X[col].quantile(0.25)
        q3 = X[col].quantile(0.75)
        iqr = q3 - q1
        upper = q3 + threshold * iqr
        mask &= X[col] <= upper
    return X[mask], y[mask]

def split_data(X, y, test_size=0.2, random_state=42):
    """Розбиває на навчальну та тестову вибірки."""
    return train_test_split(X, y,
                            test_size=test_size,
                            random_state=random_state)

def get_scaler_pipeline(model):
    """Обертає модель у pipeline зі StandardScaler."""
    return Pipeline([
        ('scaler', StandardScaler()),
        ('model', model)
    ])
```

Додаток Б

Лістинг коду models.py

```
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import StandardScaler
from xgboost import XGBRegressor
from sklearn.neural_network import MLPRegressor


def get_models():
    """
    Повертає словник усіх п'яти моделей.
    Початкові гіперпараметри — розумні дефолти.
    """
    models = {

        'Ridge': Pipeline([
            ('scaler', StandardScaler()),
            ('model', Ridge(alpha=1.0))
        ]),

        'PolyReg': Pipeline([
            ('scaler', StandardScaler()),
            ('poly',   PolynomialFeatures(degree=2,
                                          include_bias=False)),
            ('model',  Ridge(alpha=1.0))
        ]),

        'DecisionTree': DecisionTreeRegressor(
            max_depth=8,
            min_samples_leaf=20,
            random_state=42
        ),

        'XGBoost': XGBRegressor(
            n_estimators=500,
            learning_rate=0.05,
            max_depth=5,
            subsample=0.8,
            colsample_bytree=0.8,
            random_state=42,
            n_jobs=-1,
            verbosity=0
        ),

        'MLP': Pipeline([
            ('scaler', StandardScaler()),
            ('model',  MLPRegressor(
```

```
            hidden_layer_sizes=(128, 64, 32),
            activation='relu',
            solver='adam',
            learning_rate_init=0.001,
            max_iter=500,
            early_stopping=True,
            validation_fraction=0.1,
            random_state=42
        ))
    ]),
}
return models
```

Додаток В

Лістинг коду app.py

```
import streamlit as st
import pandas as pd
import numpy as np
import joblib
import os

# ── Завантаження моделей ──────────────────────────────────────
@st.cache_resource
def load_models():
    model_dir = 'models'
    loaded = {}
    for fname in os.listdir(model_dir):
        if fname.endswith('.pkl'):
            name = fname.replace('.pkl', '')
            loaded[name] = joblib.load(f'{model_dir}/{fname}')
    return loaded

st.set_page_config(page_title='Система прогнозування вартості
житла',
                   page_icon='🏠', layout='wide')

st.title('🏠 Інтелектуальна система прогнозування вартості житла')
st.caption('California Housing Dataset — кваліфікаційна робота
бакалавра')

# ── Бокова панель — вхідні дані ──────────────────────────────
st.sidebar.header('Вхідні параметри')

med_inc    = st.sidebar.slider('Медіанний дохід (MedInc)', 0.5,
15.0, 3.5, 0.1)
house_age  = st.sidebar.slider('Вік будинку (HouseAge)', 1, 52,
25)
ave_rooms  = st.sidebar.slider('Кімнат на госп. (AveRooms)', 1.0,
15.0, 5.0, 0.1)
ave_bedrms = st.sidebar.slider('Спалень на госп. (AveBedrms)',
0.5, 5.0, 1.0, 0.1)
population = st.sidebar.number_input('Населення блоку', 10, 5000,
1200)
ave_occup  = st.sidebar.slider('Мешканців на госп. (AveOccup)',
1.0, 10.0, 3.0, 0.1)
latitude   = st.sidebar.slider('Широта (Latitude)', 32.5, 42.0,
35.0, 0.1)
longitude  = st.sidebar.slider('Довгота (Longitude)', -124.5,
-114.0, -119.0, 0.1)

input_df = pd.DataFrame([{
    'MedInc': med_inc, 'HouseAge': house_age,
    'AveRooms': ave_rooms, 'AveBedrms': ave_bedrms,
    'Population': population, 'AveOccup': ave_occup,
    'Latitude': latitude, 'Longitude': longitude
```

```
}])

# ── Прогнозування ──────────────────────────────────────────
models = load_models()

st.subheader('Прогнози всіх моделей')
cols = st.columns(len(models))
for col, (name, model) in zip(cols, sorted(models.items())):
    pred = model.predict(input_df)[0]
    col.metric(label=name, value=f'${pred*100_000:,.0f}')

# ── Деталі вхідних даних ──────────────────────────────────
with st.expander('Переглянути вхідні дані'):
    st.dataframe(input_df)
```

Додаток Г
Лістинг коду evaluation.py

```
import numpy as np
import pandas as pd
from sklearn.model_selection import KFold, cross_validate
from sklearn.metrics import (mean_squared_error,
mean_absolute_error, r2_score)


SCORING = {
    'MSE':  'neg_mean_squared_error',
    'MAE':  'neg_mean_absolute_error',
    'R2':   'r2',
}


def cross_val_all(models: dict, X, y, cv=5, random_state=42):
    """
    5-кратна кросвалідація для кожної моделі.
    Повертає DataFrame з усередненими метриками.
    """
    kf = KFold(n_splits=cv, shuffle=True,
random_state=random_state)
    results = []

    for name, model in models.items():
        print(f'  Навчання: {name}...')
        cv_res = cross_validate(model, X, y,
                                cv=kf,
                                scoring=SCORING,
                                n_jobs=-1,
                                return_train_score=False)
        mse  = -cv_res['test_MSE'].mean()
        mae  = -cv_res['test_MAE'].mean()
        r2   = cv_res['test_R2'].mean()
        results.append({
            'Модель':  name,
            'RMSE':    round(np.sqrt(mse), 4),
            'MAE':     round(mae, 4),
            'R²':      round(r2, 4),
            'Adj. R²': round(1 - (1 - r2) * (len(y) - 1) /
                             (len(y) - X.shape[1] - 1), 4),
        })

    return pd.DataFrame(results).sort_values('RMSE')


def evaluate_on_test(model, X_test, y_test):
    """Фінальна оцінка навченої моделі на тестовій вибірці."""
    y_pred = model.predict(X_test)
    n, d = X_test.shape
    r2 = r2_score(y_test, y_pred)
```

```
mse = mean_squared_error(y_test, y_pred)
return {
    'RMSE':    round(np.sqrt(mse), 4),
    'MAE':     round(mean_absolute_error(y_test, y_pred), 4),
    'R²':      round(r2, 4),
    'Adj. R²': round(1 - (1 - r2) * (n - 1) / (n - d - 1), 4),
}
```

Додаток Д

Лістинг коду optimization.py

```
import numpy as np
import joblib
from sklearn.model_selection import RandomizedSearchCV, KFold
from xgboost import XGBRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from src.preprocessing import load_data, remove_outliers,
split_data

X, y = load_data()
X, y = remove_outliers(X, y)
X_train, X_test, y_train, y_test = split_data(X, y)
kf = KFold(n_splits=5, shuffle=True, random_state=42)

# ── Оптимізація XGBoost ──────────────────────────────────────
xgb_grid = {
    'n_estimators':    [300, 500, 700],
    'learning_rate':   [0.01, 0.05, 0.1],
    'max_depth':       [3, 5, 7],
    'subsample':       [0.6, 0.8, 1.0],
    'colsample_bytree':[0.6, 0.8, 1.0],
    'min_child_weight':[1, 3, 5],
}

xgb_search = RandomizedSearchCV(
    XGBRegressor(random_state=42, verbosity=0, n_jobs=-1),
    param_distributions=xgb_grid,
    n_iter=40,            # 40 випадкових конфігурацій
    scoring='neg_root_mean_squared_error',
    cv=kf,
    random_state=42,
    n_jobs=-1,
    verbose=1
)
xgb_search.fit(X_train, y_train)
print('XGBoost найкращі параметри:', xgb_search.best_params_)
print('XGBoost найкращий RMSE (CV):', -xgb_search.best_score_)
joblib.dump(xgb_search.best_estimator_,
'../models/XGBoost_tuned.pkl')

# ── Оптимізація MLP ──────────────────────────────────────────
mlp_grid = {
    'model__hidden_layer_sizes': [(64,32), (128,64), (128,64,32)],
    'model__learning_rate_init': [0.001, 0.005, 0.01],
    'model__alpha':              [0.0001, 0.001, 0.01],  # L2
}

mlp_pipeline = Pipeline([
    ('scaler', StandardScaler()),
```

```
    ('model',  MLPRegressor(activation='relu', solver='adam',
                            max_iter=500, early_stopping=True,
                            random_state=42))
])

mlp_search = RandomizedSearchCV(
    mlp_pipeline,
    param_distributions=mlp_grid,
    n_iter=20,
    scoring='neg_root_mean_squared_error',
    cv=kf,
    random_state=42,
    n_jobs=-1,
    verbose=1
)
mlp_search.fit(X_train, y_train)
print('MLP найкращі параметри:', mlp_search.best_params_)
print('MLP найкращий RMSE (CV):', -mlp_search.best_score_)
joblib.dump(mlp_search.best_estimator_, '../models/MLP_tuned.pkl')

# ── Важливість ознак XGBoost ──────────────────────────────
import matplotlib.pyplot as plt
import pandas as pd

feat_imp = pd.Series(
    xgb_search.best_estimator_.feature_importances_,
    index=X_train.columns
).sort_values(ascending=True)

feat_imp.plot(kind='barh', figsize=(8, 5), color='steelblue')
plt.title('Важливість ознак — XGBoost (оптимізований)')
plt.xlabel('Feature Importance (gain)')
plt.tight_layout()
plt.savefig('../results/feature_importance_xgb.png', dpi=150)
```