КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра технологій управління

Спеціальність 122 – Комп'ютерні науки,
Освітня програма «Інформаційна аналітика та впливи»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
на тему:
«Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання»

Студента 2-го курсу групи ІАВ-21

Подольського Нікіти Володимировича

(прізвище, ім'я, по батькові)

Науковий керівник:

к.т.н.

Хлевний Андрій Олександрович

(прізвище, ім'я, по батькові)

(підпис студента) *(дата)* *(підпис)*

Попередній захист:		
(Висновок: «До захисту в Екзаменаційній комісії»)		
Завідувач кафедри технологій управління ___		
(підпис)	(прізвище, ініціали)	(дата)

Київ – 2025

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Освітньо-кваліфікаційний рівень Магістр

Спеціальність 122 – Комп'ютерні науки

Освітня програма Інформаційна аналітика та впливи

ЗАТВЕРДЖУЮ

Завідувач кафедри, професор

Морозов В.В.

_____

«_____» 20___ року

ЗАВДАННЯ НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент Подольський Нікіта Володимирович

Група ІАВ-21

1. Тема кваліфікаційної роботи

«Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання»

Затверджено наказом від «___» _____ 20___ р. № _____

2. Строк подання студентом готової роботи – “___” _____ 20___ р.

3. Цільова установка та вихідні дані до роботи

Аналіз та дослідження моделей машинного навчання для прогнозування ринкової вартості автотранспортних засобів на основі технічних, експлуатаційних і ринкових параметрів. Метою є вибір та реалізація найбільш ефективного підходу до оцінки вартості автомобіля, з урахуванням складних взаємозв’язків між ознаками.

4. Зміст роботи

Аналіз і порівняння сучасних моделей машинного навчання для прогнозування ринкової вартості вживаних автомобілів на основі великого обсягу табличних даних. Дослідження охоплює оцінку точності, адаптивності та практичної придатності моделей у реальних умовах. Виокремлення підходів та алгоритмів, які використовуються для розв’язання задач регресійного прогнозування: лінійна регресія, ансамблеві методи (Random Forest, XGBoost, CatBoost), глибокі нейронні мережі з attention-механізмом, а також гібридні архітектури. Основні інструменти: Python, Scikit-learn, TensorFlow, XGBoost, CatBoost, Optuna, Google Vertex AI. Опис методів порівняння моделей на основі ключових метрик: R^2, MAE, RMSE, MAPE. Аналіз існуючих підходів до оцінки авто та обґрунтування переваг використання машинного навчання для автоматизації цінового прогнозування.

5. Перелік графічного матеріалу (слайдів)

Робота містить 3 таблиці та 50 рисунків, що ілюструють структуру даних, архітектуру моделей, процес обробки ознак, порівняння метрик якості, візуалізацію результатів прогнозування та розгортання моделі у хмарному середовищі.

6.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва частин роботи	%	Виконання роботи	
			За планом	Фактично
1.	Вибір теми дипломної роботи	3	01.10.24	01.10.24
2.	Протокол кафедри ТУ про затвердження тем дипломних робіт та призначення наукових керівників	2	27.12.24	27.12.24
3.	Формування переліку нормативних матеріалів, літератури з проблематики дипломної роботи	10	08.01.25	07.01.25

4.	Складання розгорнутого плану кваліфікаційної роботи	5	18.01.25	18.01.25
5.	Ознайомлення наукового керівника з розгорнутим планом кваліфікаційної роботи. Внесення змін.	5	19.01.25 - 20.01.25	20.01.25
6.	Підготовка розділу 1 «Історія, розвиток та аналіз існуючих рішень для емоційного аналізу»	10	12.02.25	13.02.25
7.	Підготовка розділу 2 «Формалізація методу емоційного аналізу»	14	08.03.25	08.03.25
8.	Підготовка розділу 3 «Розробка методу емоційного аналізу»	14	20.03.25	20.03.25
9.	Підготовка розділу 4 «Впровадження розробленої технології»	13	15.04.25	15.04.25
10.	Оформлення кваліфікаційної роботи. Підготовка висновків і пропозицій	15	25.04.25	25.04.25
11.	Передача кваліфікаційної роботи науковому керівникові	2	07.05.25	07.05.25
12.	Передача кваліфікаційної роботи рецензенту для рецензування	2	08.05.25	08.05.25
13.	Попередній захист кваліфікаційної роботи	5	10.05.25	10.05.25

Дата видачі завдання «___» _____ 20___ р.

Керівник роботи к.т.н., Хлевний Андрій Олександрович
(посада, прізвище, ім’я, по батькові)

_____
(підпис)

Завдання прийняв до виконання студент групи ІАВ-21
Подольський Нікіта Володимирович

_____
(прізвище, ім’я, по батькові)

_____
(підпис)

З М І С Т

АНОТАЦІЯ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Спеціальність – 122, Комп'ютерні науки, освітня

програма "Інформаційна аналітика та впливи"

Дипломна робота магістра Подольського Нікіти Володимировича

Тема роботи – «**Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання**».

Мета дипломної роботи магістра – підвищення ефективності прогнозування цін на автомобілі за рахунок створення моделі машинного навчання, яка дозволяє точніше визначати ціну авто та враховувати ключові фактори, що на неї впливають.

Об'єкт дослідження – процес формування та прогнозування ринкових цін на автотранспортні засоби.

Предмет дослідження – моделі машинного навчання та інформаційне забезпечення, що використовуються для прогнозування цін на автомобілі.

Наукова новизна роботи – поєднання нейронних мереж з механізмом уваги для прогнозування цін на автотранспортні засоби. Такий підхід дозволяє досягти високої точності, гнучкості та стійкості моделі порівняно з традиційними методами.

Більшість існуючих робіт з прогнозування цін на автомобілі використовують прості алгоритми (лінійну регресію, дерева рішень). У цій роботі використовується механізм уваги всередині нейронної мережі, який дає змогу моделі фокусуватися на найбільш важливих характеристиках, чого зазвичай немає у класичних градієнтних бустингових моделях. Також застосовується кілька методів вибору важливих ознак (mutual_info, chi², F-тест),

що забезпечує краще очищення даних перед навчанням. У багатьох існуючих дослідженнях ці етапи спрощують або ігнорують, що знижує якість моделей.

Завдяки цьому підходу робота досягає більшої стійкості та точності прогнозів на реальних, великих наборах даних, що виділяє її серед інших робіт у цій галузі.

Дипломна робота складається зі вступу, основної частини, яка включає чотири розділи, висновків та списку використаних джерел. Всього налічує 119 сторінок, перелік посилань з 54 джерел на 5 сторінках та 3 додатків.

Ключові слова: прогнозування цін, автомобілі, машинне навчання, нейронні мережі, механізм уваги, XGBoost, feature selection, взаємна інформація, середньоквадратична помилка, коефіцієнт детермінації, гібридні моделі, обробка даних.

ПЕРЕЛІК ВИКОРИСТАНИХ СКОРОЧЕНЬ

ML – Machine Learning

AI - Artificial Intelligence

NN – Neural Network

RNN - Recurrent Neural Network

КРМ – кваліфікаційна робота магістра

JSON – JavaScript Object Notation

MAE – Mean Absolute Error

MAPE - Mean Absolute Percentage Error

R^2 - Coefficient of Determination

CSV - Comma-Separated Values

API - Application Programming Interface

PaaS - Platform as a Service

IaaS - Infrastructure as a Service -

MaaS - Model as a Service

XGBoost - Extreme Gradient Boosting

TF - TensorFlow

ReLU - Rectified Linear Unit

GCS - Google Cloud Storage

NaN - Not a Number

ВСТУП

Актуальність теми **«Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання»** обумовлена сучасним розвитком технологій, постійним збільшенням обсягів даних і потребою в зручних та ефективних інструментах, які допомагають приймати раціональні управлінські рішення. Вартість автотранспортних засобів впливає на багато сфер життєдіяльності, включаючи логістику, торгівлю та послуги та є важливим елементом економіки.

У наш час, коли цифрові технології розвиваються надзвичайно швидко, а кількість доступної інформації постійно зростає, важливим є завдання прогнозування вартості автотранспортних засобів. Методи машинного навчання стають все більш популярними, бо є такими,що здатні аналізувати великі обсяги інформації, виявляти складні залежності та будувати точні прогнози. Використання цих методів у прогнозуванні цін на автотранспортні засоби дозволяє: підвищити точність оцінювання, оптимізувати витрати, планувати бюджети та приймати обґрунтовані рішення щодо придбання чи продажу транспортних засобів.

Формування ціни на автомобілі обумовлюється низкою факторів — від технічних характеристик і пробігу до ринкової кон’юнктури, сезонних коливань та макроекономічних умов. Класичні методи оцінювання часто не враховують складну структуру взаємозв’язків між цими змінними, що знижує їхню точність і надійність у динамічному середовищі.

Застосування методів машинного навчання в рамках концепції Data Science відкриває нові можливості для обробки даних, виявлення прихованих залежностей та побудови точних моделей прогнозування. Особливо перспективними є нейронні мережі, які не тільки підвищують точність результатів, а й дають змогу зрозуміти, які саме ознаки найбільше впливають на формування ціни.

Питання є актуальним і з практичної точки зору. Ретельне передбачення цін на автотранспорт є важливим для учасників ринку: автодилерів, страхових і фінансових компаній, онлайн-платформ та споживачів. Використання сучасних інструментів аналітики сприяє прозорому та обґрунтованому ціноутворенню, зменшує невизначеність і підвищує ефективність бізнес-рішень.

Отже, дослідження сучасних моделей прогнозування цін із використанням машинного навчання має високу наукову і прикладну цінність та відповідає актуальним потребам галузі.

Наукова задача даної роботи: «Створення сучасної інформаційної технології прогнозування ринкової вартості автотранспортних засобів, яка враховує велику кількість змінних і дозволяє використовувати методи машинного навчання для автоматизованої, точної та інтерпретованої оцінки. Особливу увагу приділено нейронним мережам з attention-механізмом як засобу підвищення точності та прозорості рішень.»

Отже, задача полягає у дослідженні, яке спрямовано на: аналіз результатів моделювання та їх вплив на прийняття практичних рішень у сфері ціноутворення автотранспортних засобів. Особливу увагу приділено: визначенню ключових факторів, які формують ринкову вартість автомобілів, а також оцінці точності та ефективності побудованих моделей машинного навчання.

Тематика даної кваліфікаційної роботи вписується у сучасні наукові напрями в галузі Data Science, аналітики великих даних та штучного інтелекту. Вона також співвідноситься з науковими дослідженнями, що проводяться на кафедрі технологій управління Факультету інформаційних технологій КНУ імені Тараса Шевченка. Зокрема, робота відповідає таким науковим тематикам кафедри:

- «Розробка інформаційно-аналітичних інструментів управління портфелями проєктів і програм в інтегрованих функціональних середовищах».

- «Розробка моделей, методів інтелектуального управління проєктами інноваційно орієнтованих підприємств».

У роботі досліджуються сучасні підходи до прогнозування цін на автотранспортні засоби з використанням методів машинного навчання та інтелектуального аналізу даних, що є актуальним для інформаційного забезпечення процесів прийняття управлінських рішень. Таким чином, дана тема сприяє реалізації науково-дослідної діяльності кафедри, орієнтованої на створення прикладних рішень у сфері управління інноваційними проектами.

Вона може бути пов’язана з:

- науковими програмами, спрямованими на вдосконалення методів прогнозування економічних показників, з використанням алгоритмів машинного навчання та нейронних мереж;
- проектами, пов’язаними з розробкою прикладних систем для аналітики в транспортній сфері, зокрема, щодо автоматизованого ціноутворення та оцінювання вартості активів;
- дослідженнями в області explainable AI (XAI), що дозволяють зробити процес прийняття рішень у машинному навчанні прозорим та інтерпретованим;
- ініціативами, орієнтованими на впровадження інтелектуальних технологій у комерційні сервіси: онлайн-продажі авто, фінансовий лізинг, страхування тощо.

Таким чином, виконане дослідження має не лише теоретичну,а й прикладну значущість і може бути використане як основа для подальших наукових чи інноваційних розробок.

Мета дослідження: підвищення ефективності прогнозування цін на автомобілі за рахунок створення моделі машинного навчання, яка дозволяє точніше визначати ціну авто та враховувати ключові фактори, що на неї впливають.

Задачі дослідження:

- Проаналізувати наукову літературу щодо існуючих підходів до прогнозування цін на автотранспортні засоби та використання методів машинного навчання у сфері економіки.
- Вивчити теоретичні основи методів машинного навчання, які застосовуються для задач регресійного прогнозування, зокрема нейронних мереж, дерев рішень, ансамблевих моделей тощо.
- Зібрати та підготувати набір даних, що включає технічні характеристики автомобілів, ринкові чинники та історичні ціни.
- Реалізувати модель прогнозування з використанням алгоритму машинного навчання, зокрема нейронної мережі.
- Оцінити точність і продуктивність прогнозування цін на автотранспортні засоби. У цій роботі запропоновано: модель машинного навчання, яка поєднує нейронні мережі з механізмом уваги та алгоритм XGBoost для прогнозування цін на автотранспортні засоби. Такий підхід дає змогу врахувати складні нелінійні залежності між характеристиками автомобілів і забезпечити вищу точність у порівнянні з традиційними моделями, такими як лінійна регресія або дерева рішень. Очікується, що об'єднання сильних сторін глибокого навчання, та градієнтного бустингу, дозволить побудувати ефективнішу систему цінового прогнозування.
- Проаналізувати отримані результати. Передбачено аналіз з метою з’ясувати, які саме характеристики автомобіля найбільше впливають на його вартість. Це дозволяє не лише прогнозувати ціну, а й краще зрозуміти логіку формування ціни, що є корисним для автодилерів, страхових компаній та інших учасників ринку.

Об'єкт дослідження: процес формування та прогнозування ринкових цін на автотранспортні засоби.

Предмет дослідження: моделі машинного навчання та інформаційне забезпечення, що використовуються для прогнозування цін на автомобілі.

Наукова новизна роботи – поєднання нейронних мереж з механізмом уваги для прогнозування цін на автотранспортні засоби. Такий підхід дозволяє досягти високої точності, гнучкості та стійкості моделі порівняно з традиційними методами.

Методи дослідження. У процесі дослідження було застосовано комплекс методів, які забезпечили досягнення поставленої мети та розв'язання сформульованих завдань.

По-перше, для формування теоретичної бази були використані методи аналізу та узагальнення наукової літератури з тематики прогнозування цін, машинного навчання та нейронних мереж. Це дало змогу визначити актуальні методи, які можна ефективно застосувати для вирішення задачі прогнозування вартості автотранспортних засобів.

Для роботи з даними були застосовані: методи збору, очищення, нормалізації та попередньої обробки даних. Це включало роботу з реляційними та CSV-форматами даних, відбір релевантних ознак, кодування категоріальних змінних, обробку пропущених значень та масштабування числових параметрів.

На аналітичному етапі були використані методи статистичного аналізу, що дозволило дослідити розподіли, кореляції між ознаками та виявити потенційно значущі фактори впливу на ціну автомобіля.

У результаті проведеного дослідження було реалізовано модель прогнозування цін на автотранспортні засоби з використанням методів машинного навчання. Створена модель базується на нейронній мережі з attention-механізмом, що дозволяє не лише досягти високої точності прогнозу, а й ідентифікувати ключові ознаки, які найбільше впливають на цінову оцінку, що є перевагою обраного підходу. Також було розроблено інформаційне

забезпечення, що охоплює процес збору, очищення, обробки та візуалізації даних для подальшого аналізу й застосування моделі у практичних умовах.

Мій особистий внесок у дослідження за темою: «Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання» спрямован на використання технологій Data Science і ML для розробки моделей прогнозування цін, а також створення інформаційної системи для підтримки цих моделей в умовах бізнес-процесів.

Практична цінність роботи полягає в можливості автоматизовано та з високою точністю прогнозувати ціну автомобіля на основі його характеристик. Така модель може бути інтегрована в онлайн-сервіси з продажу авто, страхові компанії, автосалони та лізингові сервіси для підвищення об’єктивності оцінки, зменшення людського фактору та оптимізації бізнес-процесів.

Апробація:

1. Nikita Podolskyi, Andrii Khlevnyi. Models and information systems for vehicle price forecasting using machine learning methods. - Information Technology and Implementation (Satellite): Conference Proceedings, November 21, 2024, Kyiv, Ukraine / Ministry of Education and Science of Ukraine, Taras Shevchenko National University of Kyiv and [etc]; Vitaliy Snytyuk (Editor). – Kyiv: Publishing House «Caravela», 2024. 343 p. – p.62.
2. Nikita Podolskyi, Andrii Khlevnyi. Development of Data Science technology in customer segmentation using support vector machine classification method. - Information Technology and Implementation (Satellite): Conference Proceedings, November 21, 2023, Kyiv, Ukraine / Ministry of Education and Science of Ukraine, Taras Shevchenko National University of Kyiv and [etc]; Vitaliy Snytyuk (Editor). – Kyiv: Publishing House «Caravela», 2023. 343 p. – p.197.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ПРОЄКТ DATA SCIENCE

1.1 Аналіз методології Data Science задля інформаційного забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання

Методологія Data Science вказує на процедуру пошуку рішень конкретної проблеми, це структурований підхід до вирішення складних проблем за допомогою даних, це найкращий спосіб організувати свою роботу, робити її краще та не втрачати часу. [5]

Data Science – машинне навчання, поєднує методи та інструменти зі статистики, машинного навчання, обробки даних, програмування та доменних знань. Використовується в різних галузях, таких як: бізнес і промисловість, державний сектор, технології та інновації, екологічні та соціальні напрямки тощо, для оптимізації процесів, прийняття рішень та розв'язання різних завдань.

Методологія Data Science складається з [44]:

1. Перший крок - це ретельне **розуміння бізнес-задачі** або проблеми, яку потрібно вирішити, бо від цього залежить вибір етапів роботи з даними в подальшому.

2. **Збір різноманітних даних** з різних джерел, таких як бази даних, файли, веб-сторінки тощо. Різноманітність джерел дуже багата й вимагає, відповідно, різного підходу для збору та обробки даних.

3. Дані можуть бути неоднорідними, містити пропуски, помилки. **Очищення даних** включає в себе обробку цих проблем, видалення пропусків та некоректних даних та перетворення їх у формат, придатний для аналізу.

4. Цей етап включає в себе **дослідження даних**, виявлення кореляцій, визначення ключових характеристик та візуалізацію даних для отримання загального розуміння їхньої структури.

5. Використання **різних алгоритмів машинного навчання** та статистичних методів для створення моделей, які можуть передбачати та аналізувати дані.

6. **Перевірка ефективності моделей** на тестових даних.

7. **Інтерпретація результатів аналізу** та їхнє використання для прийняття рішень. Наприклад, підготовка звітів, презентацій або візуалізацій для комунікації з зацікавленими сторонами.

Послідовність цих етапів відповідає підходу CRISP-DM (Cross-Industry Standard Process for Data Mining), що широко застосовується у проєктах з аналізу даних.[5]

1.2 Напрямки використання Data Science для прогнозування цін на автотранспортні засоби

Data Science є важливою частиною сучасних інструментів для оцінки вартості автомобілів. Завдяки своїм можливостям обробляти великі обсяги даних, знаходити складні зв'язки, Data Science допомагає точно та прозоро визначати ціни на транспортні засоби.

1.2.1 Автоторгівля та онлайн-ринок автомобілів

Один з основних напрямків використання Data Science — це ринок автомобілів, зокрема онлайн-платформи для їх продажу. Тут часто використовують моделі машинного навчання для прогнозування цін. У роботах [6, 7] аналізували різні алгоритми машинного навчання та можливість цінового прогнозування на основі технічних характеристик авто та ефективність алгоритмів для прогнозування цін на вживані автомобілі. Вони аналізують такі фактори, як пробіг, марка, модель, рік випуску і навіть стан автомобіля. Алгоритми можуть автоматично коригувати ціни в залежності від змін на ринку чи сезонних коливань попиту, допомагаючи продавцям і покупцям знайти оптимальні ціни.

1.2.2 Страхування та фінансові послуги

Data Science також активно застосовується в страхуванні та фінансових послугах. У роботі [8] приділяється увага ефективності прогнозування страхових ризиків і тарифів через поєднання класичних підходів, таких як: лінійна та логістична регресія, з більш новими методами машинного навчання,

такими як: дерева рішень та ансамблеві моделі. Страхові компанії зацікавлені в прогнозуванні вартості автомобілів і визначення ризиків. Наприклад, аналіз історії страхових випадків і характеристик автомобілів дозволяє точніше оцінити ймовірність наступних подій. Це допомогає знижувати витрати для страховиків і забезпечувати більш справедливе ціноутворення для клієнтів.

1.2.3 Лізинг та фінансування

У лізингу автомобілів Data Science допомагає точніше визначати вартість лізингу. Це дозволяє фінансовим установам оцінювати реальну ціну автомобіля на основі технічних характеристик і попиту на ринку. Моделі машинного навчання допомагають знижувати фінансові ризики для лізингових компаній, прогнозуючи лізингові платежі на основі даних про конкретний автомобіль. Наприклад, у роботі [7] було показано, що використання XGBoost дозволяє підвищити точність передбачення залишкової вартості авто, а це, звісно, впливає на фінансові рішення при укладанні лізингових контрактів.

1.2.4 Сегментація та аналіз поведінки споживачів

Ще один важливий аспект — це сегментація споживачів та персоналізація ціноутворення. Використовуючи Data Science, компанії можуть створювати персоналізовані стратегії для різних груп покупців. Для цього аналізуються дані про покупки, поведінку на вебсайті (що досліджено у роботі [9]), чи в додатках, а також демографічні й географічні особливості користувачів. Це дозволяє точніше прогнозувати, який продукт буде популярний у певних категорій споживачів.

1.2.5 Технічні та економічні фактори

Data Science також дає можливість детально аналізувати технічні та економічні фактори, які впливають на ціноутворення. Наприклад, дослідження змін економічної ситуації (робота [10]), коливань валютних курсів чи змін у податковій політиці дозволяє передбачити, як ці фактори можуть змінити ціни на автомобілі. Це допомагає точніше планувати бюджети і правильно управляти фінансами.

1.2.6 Прогнозування змін на ринку та стратегічне планування

Data Science дає змогу прогнозувати зміни на ринку автомобілів. За допомогою аналізу сезонних коливань попиту та цін можна розробити стратегії для закупівлі, знижок або рекламних акцій. В цьому посібнику [11] наведені сучасні методи прогнозування, й для ринку, який постійно змінюється. Моделі прогнозування можуть показати, де і для яких автомобілів ціни ймовірно зростатимуть, що дозволяє компаніям оптимізувати свої запаси і розподіляти ресурси більш ефективно.

1.3 Вибір методоогії Data Science та наповнення проєкту прогнозування цін на автотранспорнтні засоби

У цьому проєкті ми розробили систему, яка дозволяє прогнозувати ціни на автомобілі. Робота починалася зі збору необхідних даних, їх аналізу та підготовки до моделювання. Основну увагу ми приділили використанню нейронних мереж, оскільки ці моделі ефективно працюють з великими масивами інформації та здатні виявляти приховані залежності між технічними характеристиками машин і їхньою вартістю.

Наша модель побудована з використанням методів глибокого навчання. Це дає змогу враховувати не лише очевидні, а й комплексні взаємозв’язки між параметрами. Наприклад, ми аналізуємо, як поєднання року випуску, пробігу та стану авто впливає на його ринкову ціну. Головна мета — створити гнучку модель, яка точно оцінює авто навіть за умов змін на ринку.

Щоб отримати якісні прогнози, потрібні релевантні дані. Ми використовували інформацію з баз оголошень, історію змін цін, статистику сезонних коливань. Зокрема, нас цікавили такі показники: марка й модель, рік виробництва, пробіг, технічний стан, ринкова динаміка.

Далі йшла обробка даних — важливий етап, адже вихідна інформація часто містить помилки або неповну інформацію. Ми:

- видаляли дублікати й зайві стовпці,
- заповнювали пропущені значення або обґрунтовано їх ігнорували,

- виправляли некоректні записи (наприклад, нереальні пробіги),
- уніфікували одиниці виміру (наприклад, кілометри й милі).

Завдяки цьому модель отримала «чисті» й узгоджені дані для навчання.

Наступним кроком став відбір найважливіших ознак. Для цього ми застосовували кореляційний аналіз і метод головних компонент (PCA). Це дозволило скоротити обсяг вхідної інформації, зберігаючи її інформативність. Наприклад, якщо пробіг і рік випуску мають сильний вплив на ціну, а модель авто — слабкий, останній параметр можна виключити.

Коли дані були готові, ми побудували нейронну мережу. Вона здатна навчатися на історичних даних і робити прогнози щодо вартості машин, яких вона ще не бачила. Її перевага — в можливості враховувати складні взаємозв’язки між численними параметрами.

Для оцінки якості моделі ми використовували такі метрики:

- **Середня квадратична помилка (MSE)** — показує, наскільки в середньому наш прогноз відрізняється від фактичної ціни.
- **Коефіцієнт детермінації (R^2)** — відображає, наскільки добре модель пояснює варіації цін.

Ми перевіряли модель на тестовому наборі даних і порівнювали результати з реальними значеннями. Якщо показники були задовільними, переходили до впровадження моделі. Інакше — поверталися до етапів корекції та вдосконалення.

У підсумку створено систему, яка допомагає точно прогнозувати ринкову ціну автомобілів на основі технічних і ринкових параметрів. Вона може бути корисною як для продавців, так і для покупців, оскільки сприяє прийняттю зважених рішень при купівлі або продажу авто.

Інформаційне забезпечення прогнозування цін на автомобілі за допомогою машинного навчання — це комплекс рішень, що охоплює дані, цифрові інструменти, технології та процедури, які дозволяють збирати,

зберігати, обробляти й аналізувати інформацію для створення ефективних моделей оцінки вартості.

Застосування таких підходів дає бізнесу низку важливих переваг:

- забезпечує обґрунтоване ціноутворення, базоване на фактичних даних, а не лише на припущеннях або досвіді;
- дозволяє краще орієнтуватися в ринкових умовах і швидко реагувати на зміни попиту чи пропозиції;
- знижує ризики завищеної або заниженої оцінки, що безпосередньо впливає на прибутковість;
- автоматизує процес оцінювання транспортних засобів, полегшуючи роботу персоналу та покращуючи клієнтський досвід;
- допомагає виділити найвпливовіші чинники формування ціни — наприклад, пробіг, вік, марку авто;
- дає змогу заздалегідь планувати закупівлі або кампанії, прогнозуючи попит і цінові коливання.

Інструменти Data Science поступово змінюють традиційні підходи в ціноутворенні. Завдяки машинному навчанню компанії можуть аналізувати великі обсяги інформації, враховувати сезонні тренди, технічні особливості автомобілів і загальну ринкову ситуацію. Це дозволяє досягти більшої точності у прогнозах і формувати справедливі ціни для кожного конкретного випадку.

Окрім цього, сучасні моделі здатні брати до уваги індивідуальні параметри, як-от географічне розташування або характер експлуатації транспортного засобу. Візуалізація результатів аналізу (за допомогою інструментів пояснення, таких як SHAP або attention-механізми) дозволяє краще зрозуміти, чому система приймає ті чи інші рішення.

Використання аналітики також допомагає прогнозувати зміни ринку, оцінювати попит на окремі моделі авто й формувати більш ефективні стратегії щодо асортименту, реклами чи логістики.

1.4 Огляд моделей Data Science

Огляд моделей Data Science охоплює широкий спектр підходів і алгоритмів, які активно використовуються для аналізу даних, побудови прогнозів і підтримки прийняття рішень. На основі аналізу наукових і практичних джерел [12–20] нижче наведено ключові групи моделей разом із прикладами їх використання в різних галузях.

1. Регресійні моделі

До найвідоміших належить *лінійна регресія* (Linear Regression), що використовується для виявлення лінійних залежностей між незалежними змінними та цільовою ознакою [12]. Цей метод може базуватися як на одному предикторі, так і на кількох.

Сфера застосування: прогнозування продажів, моделювання економічних показників.

Іншим представником є *логістична регресія* (Logistic Regression), яка широко використовується для задач бінарної класифікації — вона дозволяє визначати ймовірність належності об’єкта до одного з двох класів [12].

Приклади використання: виявлення ризику відтоку клієнтів, медичні прогнози, оцінка фінансової надійності позичальників.

2. Класифікаційні методи

Метод опорних векторів (Support Vector Machines, SVM) допомагає побудувати гіперплощину, яка найкращим чином розділяє об'єкти різних класів у багатовимірному просторі [13].

Типові приклади: обробка текстів, біоінформатика, розпізнавання образів.

Дерева рішень (Decision Trees) — ще один ефективний підхід, що працює за принципом послідовного поділу даних на основі значень ознак [14].

Застосування: оцінювання кредитних ризиків, медична діагностика, поведінковий аналіз клієнтів.

3. Кластеризація

K-means — популярний алгоритм групування, що дозволяє розбити дані на кластери, мінімізуючи внутрішньокластерну дисперсію [14].

Використання: сегментація клієнтів, групування документів, поведінковий аналіз у ритейлі.

Ієрархічна кластеризація (Hierarchical Clustering) будує дерево зв'язків між об'єктами, показуючи, як дані можуть об'єднуватися на різних рівнях [15].

Сфери застосування: маркетингові дослідження, обробка зображень, вивчення ринку.

4. Методи зниження розмірності

Principal Component Analysis (PCA) — класичний лінійний метод, який дозволяє зменшити кількість змінних без значної втрати інформації [12].

Основні напрями використання: очищення даних, їх візуалізація, підготовка до кластеризації.

T-SNE (t-distributed Stochastic Neighbor Embedding) — техніка для нелінійного зменшення розмірності, зручна для візуального подання структур у високовимірних просторах [16].

Приклади: аналіз результатів глибокого навчання, візуалізація даних геноміки або NLP.

5. Нейронні мережі та Deep Learning

Штучні нейронні мережі (ANN) — базова форма нейромереж, яка може моделювати складні функціональні залежності між змінними [17].

Застосування: обробка мовлення, розпізнавання зображень, короткострокове прогнозування.

Глибокі нейронні мережі (DNN) — складніші архітектури, які містять багато шарів і дозволяють моделі вчитися на багаторівневих ознаках [17].

Сфери: машинне перекладання, аналіз емоцій, розпізнавання мови.

6. Ансамблеві підходи

Random Forest — метод, що об'єднує кілька дерев рішень у «ліс», що зменшує варіативність прогнозів і підвищує їхню точність [18].

Типові кейси: діагностика, фільтрація спаму, класифікація ризиків.

Gradient Boosting — алгоритм, який формує потужну модель на основі послідовного зменшення помилок попередніх моделей [19].

Приклади: передбачення ціни, скоринг позичальників, виявлення підозрілої активності.

Окремо варто згадати *XGBoost* — одну з найефективніших реалізацій градієнтного бустингу, яка відзначається високою продуктивністю в задачах класифікації та регресії.

7. Асоціативний аналіз

Алгоритм Apriori використовується для пошуку частих комбінацій елементів і виявлення асоціативних правил у транзакційних наборах [20].

Сфери застосування: аналіз споживчих кошиків, рекомендаційні системи, виявлення зв'язків між продуктами.

Таким чином, сучасні методи Data Science охоплюють широке коло інструментів, кожен з яких має своє місце в задачах прогнозування, аналізу та оптимізації. Їх застосування в сфері оцінювання вартості транспортних засобів дозволяє враховувати складні взаємозв'язки між характеристиками автомобілів, ринковими умовами та поведінкою користувачів, що є важливою складовою ефективного інформаційного забезпечення процесу прийняття рішень.

1.5 Аналіз літературних джерел щодо застосування моделей аналізу даних у проєкті «Моделі та інформаційне забезпечення прогнозування цін на автотранспортні засоби методами машинного навчання»

У даному розділі здійснено аналіз наукових публікацій, які мають безпосередній або дотичний стосунок до тематики дослідження — а саме, до прогнозування вартості автотранспортних засобів із застосуванням алгоритмів машинного навчання. Розділ містить послідовний опис ключових ідей із релевантних джерел, приклади практичних підходів, критичну оцінку застосованих методів та обґрунтування постановки дослідницької задачі.

Одним із джерел, що заслуговує на увагу, є стаття Коваленка І. під назвою «Застосування Data Science у комунальному господарстві» [21], опублікована у журналі «Економіка України». У цьому дослідженні розглядаються перспективи впровадження сучасних інструментів аналізу даних у галузях життєзабезпечення міської інфраструктури. Автор зазначає, що на поточному етапі комунальні підприємства стикаються з низкою структурних проблем, зокрема із застарілими технологіями управління, надмірним споживанням енергетичних ресурсів, низькою ефективністю розподілу ресурсів і браком системного моніторингу.

Метою дослідження було вивчення можливостей застосування методів Data Science для подолання зазначених викликів у таких сферах, як теплопостачання, водозабезпечення та енергетика. Автор виділяє чотири ключові напрями, де сучасні технології аналізу даних можуть мати практичне застосування: виявлення аномалій, формування прогнозів, оптимізація використання ресурсів та автоматизація управлінських процесів. Наприклад, за допомогою моделювання часових рядів можна спрогнозувати майбутнє навантаження на інженерні системи або витрати на технічне обслуговування. Це дозволяє створювати більш точні бюджети, зменшувати втрати та запобігати аваріям.

Особливу увагу в дослідженні приділено опису конкретних практичних кейсів. Так, у сфері теплових мереж розроблено моделі, що передбачають оптимальні режими роботи котелень залежно від погодних умов, рівня завантаженості споживачів та історичних витрат палива. У системах водопостачання застосовується кластерний аналіз для виявлення підозрілих зон витоків, а також для оптимізації алгоритмів роботи насосних станцій. В енергетичній галузі активно впроваджуються системи прогнозування попиту, які забезпечують більш збалансовану подачу електроенергії в періоди пікового навантаження.

Результати, отримані в межах дослідження, свідчать про високу ефективність Data Science у забезпеченні раціонального управління ресурсами. Автор наводить статистику, яка демонструє потенційне зменшення витрат підприємств на 15–20% при використанні методів прогнозування та автоматизованого контролю. Проте для повноцінного впровадження таких підходів необхідне створення відповідної технічної інфраструктури, включно з побудовою систем збирання даних та навчанням персоналу, здатного працювати з аналітичними інструментами.

У висновках автор робить наголос на тому, що в умовах жорстких фінансових обмежень для підприємств критично важливо мати змогу точніше прогнозувати витрати, зменшувати втрати та планувати технічне обслуговування на основі даних, а не інтуїтивно. Таким чином, досвід, описаний у статті, хоча й базується на прикладі іншої галузі, є методологічно релевантним і може бути адаптований до завдань прогнозування вартості транспортних засобів.

Зокрема, паралелі можна провести між аналізом енергоспоживання та прогнозуванням ринкової ціни автомобіля: в обох випадках маємо справу зі складними, багатофакторними системами, що вимагають точної оцінки численних вхідних параметрів. Використання алгоритмів машинного навчання дозволяє не лише ідентифікувати закономірності в історичних даних, але й адаптувати модель до нових умов, що особливо актуально на ринку транспортних засобів, де ситуація змінюється під впливом зовнішніх чинників (економічна кон’юнктура, сезонність, податкові зміни тощо).

Отже, публікація Коваленка І. становить важливу частину теоретичного підґрунтя для формулювання методичного підходу до моделювання цін на автотранспорт. Вона підтверджує доцільність використання Data Science як універсального інструменту для вирішення завдань, пов’язаних з прогнозуванням, оптимізацією витрат та прийняттям управлінських рішень у

різних секторах господарства. Таким чином, її положення можуть бути інтегровані у структуру дослідницької моделі в межах обраної теми практики.

Іншим важливим джерелом, що дозволяє глибше зрозуміти специфіку вітчизняного авторинку, є дослідження Петренка О. під назвою «Аналіз ринку автомобілів в Україні» [22]. У межах цієї праці автор зосередився на вивченні актуального стану автомобільного ринку України, його ключових характеристик, внутрішніх тенденцій розвитку, а також чинників, що формують ціни на транспортні засоби. Робота містить комплексний аналіз сучасних умов функціонування автомобільного сегмента, з урахуванням як внутрішньоекономічної ситуації, так і зовнішньополітичних впливів.

На початку дослідження Петренко окреслює загальну динаміку ринку, наголошуючи на його високій залежності від змін економічного середовища, регуляторної політики держави, рівня споживчої активності, а також попиту на первинному та вторинному ринках. Автор детально аналізує статистику обсягів продажів, структуру попиту, рейтинг популярних брендів і моделей, а також сегментацію ринку за рівнем вартості. У дослідженні підкреслюється, що значну частку ринку складають уживані транспортні засоби, які ввозяться з країн Європейського Союзу. Це зумовлено не лише нижчою вартістю вживаних авто, а й ширшим вибором моделей, прийнятним технічним станом та прозорішими умовами імпорту.

Окрему увагу дослідник приділяє переліку факторів, що мають визначальний вплив на формування ціни автомобіля. Серед них він виокремлює технічні характеристики, рік виготовлення, пробіг, стан кузова й двигуна, а також марку й модель. Крім того, істотну роль відіграють економічні умови, включаючи рівень інфляції, податкову політику та зміни валютного курсу. Автор доводить, що перелічені змінні можуть бути трансформовані у формат вхідних ознак для моделей машинного навчання, здатних здійснювати цінові прогнози з високою точністю.

У межах дослідження Петренко пропонує поглянути на ринок автомобілів крізь призму аналітики великих даних. Він обґрунтовує, що традиційні підходи, засновані на експертних оцінках і середньостатистичних узагальненнях, не забезпечують належного рівня точності у прогнозах. Натомість застосування Data Science, зокрема таких інструментів як регресійний аналіз, класифікація, кластеризація, дає змогу працювати з масивами інформації, виявляючи приховані залежності, які не піддаються прямій інтерпретації. Автор ілюструє це прикладами аналітичного моделювання, де комбінується вплив декількох факторів, таких як пробіг, вік авто та ринкова категорія, що дозволяє вивести точну цінову формулу.

Крім того, Петренко акцентує увагу на впливі макроекономічних і політичних чинників на авторинок. Так, наприклад, зміни у сфері митного регулювання, нововведення у системі оподаткування транспортних засобів або валютні коливання можуть викликати суттєві зрушення у структурі попиту. З огляду на це, автор наполягає на необхідності адаптації моделей прогнозування до реального ринкового контексту. Він зазначає, що здатність алгоритму враховувати подібні зовнішні впливи є важливою умовою його практичного застосування у бізнесі.

У фінальній частині публікації наведено перелік рекомендацій для учасників ринку, зокрема дилерів, імпортерів та логістичних компаній. Автор пропонує їм активно впроваджувати сучасні інструменти аналізу даних, щоб краще орієнтуватися в поточній кон’юнктурі ринку, уникати збитків, пов’язаних із помилками в оцінці, та прогнозувати зміни в структурі попиту.

Цінність цієї роботи в контексті дослідження прогнозування цін полягає у ґрунтовному описі ринкових характеристик та фактичного обґрунтування використання машинного навчання як інструменту аналітичного моделювання. Отже, наведені у праці Петренка висновки можуть бути безпосередньо інтегровані у розробку моделей прогнозування цін на транспортні засоби,

забезпечуючи їх актуальність та прив’язку до реального економічного середовища.

Значний інтерес для цілей даного дослідження становить робота Григоренко Л. під назвою «Прогнозування цін на транспортні засоби з використанням машинного навчання» [25]. Автор зосередив увагу на розробці алгоритмічних підходів до побудови моделей цінової аналітики з використанням сучасних інструментів штучного інтелекту. Відправною точкою для дослідження стала ідея, що ринок транспортних засобів є надзвичайно динамічним, і тому вимагає точних та адаптивних методів прогнозування, які можуть враховувати технічні характеристики автомобілів.

Григоренко аргументовано стверджує, що традиційні методи, зокрема регресійний аналіз, мають низьку ефективність у складних умовах, коли вплив окремих змінних не є лінійним, а взаємодія факторів створює складні кореляційні структури. Наприклад, взаємозв’язок між роком випуску, пробігом, типом палива та очікуваною вартістю авто може змінюватися залежно від регіону, поточної ринкової ситуації або навіть сезонності. Саме тому автор обґрунтовує необхідність переходу до алгоритмів машинного навчання, які демонструють здатність до самоадаптації та виявлення складних прихованих закономірностей.

У дослідженні пропонується використання низки методів, серед яких ключовими виступають моделі типу Random Forest, Gradient Boosting та штучні нейронні мережі. Автор порівнює їхню продуктивність у контексті прогнозування цін, підкреслюючи, що найбільш стабільні результати були отримані за допомогою бустингових моделей, зокрема XGBoost та LightGBM. Такі алгоритми дозволяють не лише покращити точність, але й зменшити вплив шуму, а також працювати з відсутніми або неконсистентними значеннями без необхідності складної ручної обробки.

Дослідження також включає глибокий розгляд етапу попередньої обробки даних. Зокрема, значна увага приділена очищенню записів від

аномалій, нормалізації числових параметрів, а також кодуванню категоріальних змінних, що включають, наприклад, тип кузова або паливо. У роботі чітко прописано алгоритм підготовки датасету: видалення дублікатів, обробка пропущених значень, а також побудова нових ознак (feature engineering), які можуть суттєво покращити продуктивність моделі.

Особливу частину дослідження становить аналіз ефективності побудованих моделей. Для об’єктивного порівняння алгоритмів автор використовує класичні метрики оцінювання якості регресійних прогнозів: середню квадратичну помилку (MSE), середню абсолютну помилку (MAE) та коефіцієнт детермінації (R^2). На основі проведених експериментів зроблено висновок, що Gradient Boosting стабільно демонструє найменші значення MSE та MAE, а також високий рівень R^2, що свідчить про здатність моделі пояснювати основну частину варіації залежної змінної — тобто ціни автомобіля.

Поряд із теоретичними аспектами, у роботі представлено також рекомендації щодо практичного використання створених моделей. Автор розробляє концепцію створення інтерфейсного інструменту у вигляді веб-платформи або інтерактивного сервісу, який дозволяє користувачу вводити базові характеристики транспортного засобу (рік, пробіг, стан, марка) та миттєво отримувати прогнозовану вартість. Такий інструмент може бути корисним не лише для приватних осіб, але й для комерційних структур, серед яких автосалони, лізингові компанії, страхові агенції, а також муніципальні органи, які управляють транспортними активами.

Григоренко також підкреслює важливість створення моделей, що можуть бути адаптовані до нових даних без необхідності повного перенавчання. Це дозволяє створити систему, яка зберігає актуальність за умов постійної зміни ринкової ситуації. Особливо важливо це в умовах нестабільної економіки або різких змін на валютному ринку, які миттєво впливають на цінові очікування споживачів і формують нові тренди у структурі попиту.

У підсумку автор доходить висновку, що використання машинного навчання у сфері цінового прогнозування відкриває широкі перспективи для цифровізації процесів прийняття рішень у транспортному секторі. Висока точність моделей, можливість масштабування та інтерпретованість результатів створюють передумови для широкого впровадження подібних рішень у різних галузях економіки. Дослідження не лише формує базис для академічних розвідок, але й надає конкретні технічні рекомендації для реалізації повноцінної системи оцінки вартості автомобілів, орієнтованої на практичне застосування.

Суттєвий внесок у дослідження механізмів прогнозування цін на основі машинного навчання зроблено в роботі Smith, J. «Machine Learning for Price Prediction» [31], що присвячена аналізу можливостей застосування інтелектуальних алгоритмів для моделювання вартості товарів і послуг у різноманітних галузях. Зокрема, автор детально розглядає практики використання машинного навчання в контексті прогнозування цін на нерухомість, автомобілі, споживчі товари та послуги, що мають високий рівень варіативності ціноутворення.

Основною тезою, яку Smith послідовно доводить протягом усієї роботи, є те, що точне та оперативне визначення цін є критично важливим для досягнення бізнес-успіху. В умовах високої конкуренції та постійної зміни ринкової ситуації здатність коректно спрогнозувати майбутні коливання вартості дозволяє компаніям не лише зберігати прибутковість, але й своєчасно адаптуватися до попиту, уникати перевитрат та мінімізувати ризики. Автор стверджує, що у багатьох випадках саме прогностичні моделі стають основою для стратегічного планування, закупівельної політики, побудови цінових стратегій, а також для підвищення ефективності маркетингових кампаній.

У дослідженні розглянуто два великі блоки методів — класичні статистичні та сучасні алгоритми машинного навчання. До першої групи відносяться лінійна регресія, множинна регресія, метод головних компонент та інші підходи, засновані на припущенні про лінійні або монотонні залежності між

змінними. Автор зазначає, що ці методи мають обмеження у випадках, коли взаємозв’язки між параметрами є складними або нелінійними. Саме в таких ситуаціях особливо ефективними є моделі машинного навчання, які демонструють здатність адаптуватися до складної структури даних і враховувати багатофакторну взаємодію.

Smith надає вичерпний опис побудови повноцінного аналітичного конвеєра для прогнозування цін. Процес починається зі збору вхідних даних із різноманітних джерел, серед яких онлайн-ресурси, бази даних оголошень, архіви транзакцій. Далі виконується очищення даних: видалення аномалій, усунення пропущених значень, уніфікація форматів тощо. Після цього здійснюється трансформація ознак — зокрема, нормалізація числових параметрів (наприклад, пробігу або року випуску), та перетворення категоріальних змінних у числову форму з використанням методів кодування (One-Hot Encoding, Target Encoding тощо).

У прикладах, що стосуються автомобільного ринку, автор виокремлює набір ознак, які найбільше впливають на формування остаточної ціни транспортного засобу. Це, зокрема, марка, модель, рік випуску, пробіг, тип кузова, тип палива, об’єм двигуна, а також ринкові умови, включно з індексами попиту, сезонністю, регіональними особливостями та економічною ситуацією. Побудовані моделі навчаються на історичних даних і проходять валідацію за допомогою контрольних вибірок.

Значну увагу автор приділяє практичному аспекту імплементації моделей у бізнес-процеси. Розглядаються можливості інтеграції алгоритмів у вже існуючі платформи, CRM-системи та сервіси оцінки, де прогнозована ціна використовується як підказка для працівників або клієнтів. Описано приклади розгортання моделей у форматі REST API або вбудовування в мобільні додатки для оцінки автомобілів у режимі реального часу.

Ключовою перевагою моделей машинного навчання, за висновками Smith, є їх здатність адаптуватися до нових умов — наприклад, різких змін курсу

валют або введення нових регуляторних норм. Оскільки модель продовжує «навчатися» на нових даних, вона зберігає актуальність і точність прогнозів навіть у динамічному середовищі. З технічного боку, автор рекомендує використовувати фреймворки на кшталт Scikit-learn, TensorFlow або XGBoost, які забезпечують гнучкість, швидкість та високу продуктивність при обробці великих обсягів даних.

У завершальній частині дослідження сформульовано перелік конкретних рекомендацій щодо розробки та впровадження моделей цінового прогнозування. Зокрема, Smith радить здійснювати регулярну перевірку якості моделей, застосовувати автоматизоване оновлення на основі нових транзакцій, а також інтегрувати інструменти інтерпретації (SHAP-значення, Feature Importance), що дозволяють пояснити, як саме окремі характеристики впливають на фінальний результат. Це сприяє довірі до моделей як з боку аналітиків, так і кінцевих користувачів.

У підсумках автор робить висновок, що машинне навчання є не лише високоефективним інструментом для прогнозування цін, а й необхідною умовою сучасного аналітичного підходу у сфері торгівлі, фінансів, логістики та ринку транспортних засобів. Робота Smith, J. є цінним джерелом як у теоретичному, так і в практичному вимірі, оскільки вона демонструє повний цикл розробки, тестування й застосування моделей, які дозволяють підвищити ефективність оцінювання об’єктів ринку.

Значне місце серед джерел, що були опрацьовані в межах дослідницької практики, займає книга McKinney, W. під назвою Python for Data Analysis [34]. Це одна з найбільш авторитетних публікацій, присвячених використанню мови програмування Python для аналітики даних. Її автор — один із засновників бібліотеки Pandas — розглядає не лише технічні особливості бібліотек (Pandas, NumPy, Matplotlib та SciPy), а й пропонує системне бачення організації аналітичного процесу в реальних умовах.

McKinney орієнтує свою книгу насамперед на аналітиків-практиків і дослідників, які прагнуть опанувати інструменти аналізу даних на рівні, що дозволяє ефективно застосовувати їх у повсякденній роботі. Центральне місце в книзі відведено бібліотеці Pandas, яка надає зручні структури даних (зокрема Series та DataFrame) для представлення табличної інформації. Автор детально пояснює, як працювати з різними джерелами даних — від CSV-файлів до баз даних SQL і веб-інтерфейсів (API), з акцентом на попередню обробку інформації.

Серед основних тем, що розглядаються у книзі, — методи фільтрації, сортування, об'єднання даних, заповнення пропущених значень, а також виконання групових операцій і агрегацій. Окремий блок присвячений візуалізації даних із використанням бібліотеки Matplotlib: побудова графіків, гістограм, діаграм розсіювання дозволяє краще інтерпретувати результати аналізу, що особливо важливо при моделюванні ціноутворення.

Крім стандартних прикладів, McKinney вводить читача в базові концепти машинного навчання. У тексті описуються практичні кейси з використанням інструментів Python для побудови моделей класифікації та регресії, що можна безпосередньо застосовувати у сфері прогнозування цін на товари, включаючи транспортні засоби. Аналіз часових рядів, технік тренду та сезонності також посідає важливе місце — ці навички є незамінними при обробці історичних цінових даних.

Книга містить чітко структуровані приклади коду, які дозволяють читачеві самостійно відтворити весь аналітичний процес: від імпорту даних — до побудови інтерактивних моделей. В окремому розділі автор розглядає специфіку обробки великих масивів інформації, звертаючись до таких бібліотек, як Dask та PySpark. Завдяки цьому підхід McKinney стає релевантним і для задач, пов’язаних із Big Data, що нерідко виникає при побудові моделей у Data Science-проєктах.

Особливу цінність книга становить для тих, хто працює з прогнозуванням цін — як у комерційній сфері, так і в наукових розробках. Зокрема, структура подачі матеріалу, орієнтація на гнучку обробку даних і приклади з реального життя роблять її незамінною для фахівців, які займаються аналітикою ринку транспортних засобів. Вона дозволяє читачеві не лише теоретично зрозуміти етапи роботи з даними, а й відразу адаптувати їх під конкретну предметну галузь.

Висновок:

Проаналізовані джерела підтверджують, що методологія Data Science — це не просто набір окремих технічних рішень, а цілісний, системно-орієнтований підхід до розв’язання складних завдань, заснований на обробці, моделюванні та інтерпретації даних. Вона охоплює повний цикл — від постановки задачі до практичного впровадження моделей у реальні процеси.

Сучасні інструменти аналізу, зокрема машинне навчання, дозволяють ефективно працювати з неструктурованими або багатовимірними наборами даних, автоматизувати рутинні завдання та виявляти приховані закономірності, що залишаються недоступними для класичних методів статистики. Data Science надає можливість підвищити гнучкість прийняття рішень, а також забезпечити адаптивність до змін ринку, що особливо важливо для динамічних секторів, як-от торгівля транспортними засобами.

У межах теми прогнозування цін на автомобілі, Data Science відкриває широкі перспективи. Зокрема, моделі, що базуються на нейронних мережах, показують високу здатність до генералізації, обробки великої кількості параметрів і самонавчання. Їх застосування дозволяє створювати високоточні прогнози навіть у нестабільному середовищі, враховуючи коливання попиту, сезонні зміни та зовнішньоекономічні фактори.

У ході аналізу літератури було визначено, що оптимальне вирішення задачі прогнозування цін на автотранспортні засоби передбачає не лише правильний вибір алгоритму, а й чітко налагоджений процес збору, очищення та

структурування даних. На основі отриманих висновків сформульовано мету дослідження: підвищення ефективності прогнозування цін на автомобілі за рахунок створення моделі машинного навчання, яка дозволяє точніше визначати ціну авто та враховувати ключові фактори, що на неї впливають. Метою є побудова такої системи, яка дозволить підприємствам оптимізувати витрати, вдосконалити цінову політику та покращити точність оцінювання вартості транспортних засобів у режимі реального часу.

РОЗДІЛ 2. ФОРМАЛІЗАЦІЯ МЕТОДІВ АНАЛІЗУ ДАНИХ У ПРОЄКТІ ПРОГНОЗУВАННЯ ВАРТОСТІ АВТОТРАНСПОРТНИХ ЗАСОБІВ ІЗ ВИКОРИСТАННЯМ DATA SCIENCE

2.1 Математичний опис моделі прогнозування цін на автотранспортні засоби

1. Обробка інформації перед побудовою моделей машинного навчання є ключовим етапом, від якого залежить ефективність подальшого аналізу. Йдеться про процес, що називається **препроцесингом** — підготовкою даних до автоматизованого опрацювання. Основна мета цього етапу полягає у впорядкуванні вхідного набору даних таким чином, щоб алгоритми могли коректно їх інтерпретувати і на їх основі формувати точні прогнози.

Препроцесинг зазвичай складається з декількох послідовних кроків, кожен із яких виконує окрему функцію. Зокрема, до них належать виявлення та усунення помилкових значень, приведення різнорідних даних до уніфікованого формату, а також заповнення відсутніх показників. Усі ці дії мають теоретичне підґрунтя, базоване на статистичних методах та принципах математичної логіки.

Очищення даних

Першим завданням є **очищення даних**. У більшості випадків вихідні таблиці містять порожні комірки або помилкові записи. Такі проблеми виникають як результат людського фактору, технічних збоїв або неповного збору інформації. Як підкреслюють Geron [33] та McKinney [34], обробка пропущених значень та попередня трансформація даних є основою успішного моделювання. Якщо система виявляє пропущене значення, його можна замінити на типовий показник.

Найчастіше для цього використовують середнє арифметичне або медіану відповідного стовпця.

Якщо певний елемент Xij в матриці даних відсутній (Xij = NaN), то його можна можна заповнити за допомогою середнього значення ознаки j, розрахованого на основі наявних даних:

$$X_{i}^{j} = \begin{cases} X_{i}^{j}, & \text{якщо } X_{i}^{j} \neq \text{NaN} \\ \mu^{j}, & \text{якщо } X_{i}^{j} = \text{NaN} \end{cases} \quad (2.1)$$

де X_{i}^{j} - значення елементу матриці, μ^{j} - середнє значення ознаки j,

NaN (англ. Not a Number) — це спеціальне значення, яке використовується для позначення відсутніх або некоректних даних у масивах чи таблицях.

Варто зазначити, що інколи, замість середнього значення доцільніше застосовувати медіану — особливо у випадках, коли дані містять викиди або мають асиметричний розподіл. Застосування медіани дозволяє зменшити вплив аномальних значень на модель.

Позначення NaN (англ. *Not a Number*), що вказує на порожню або нечислову комірку, необхідно врахувати на етапі обробки. Наявність таких записів без відповідної корекції може призвести до помилок під час обчислень або до спотворення результатів моделювання.

Перетворення категоріальних змінних.

Як зазначено у працях [33, 34], для ефективного навчання моделей важливо проводити кодування категорійних ознак та масштабування числових значень.

У багатьох задачах машинного навчання дані містять **категоріальні змінні** — тобто такі ознаки, значення яких є якісними категоріями, а не числовими величинами. Типовим прикладом можуть слугувати такі атрибути, як марка автомобіля, тип кузова чи тип пального. Оскільки більшість алгоритмів вимагають числового представлення ознак, постає необхідність у перетворенні категоріальних значень у форму, придатну для аналізу.

Одним із найбільш уживаних способів такого перетворення є **бінарне кодування**, відоме як **one-hot encoding**. Методика полягає в тому, що кожна унікальна категорія перетворюється на окрему змінну (стовпець) з бінарними значеннями — «1» означає наявність ознаки у зразка, а «0» — її відсутність. Наприклад, якщо категоріальна змінна C має k можливих значень, то піля

перетворення вона буде представлена матрицею $M \in R^{n \times k}$ розміром nxk, де n - кількість зразків у наборі даних, а k - кількість категорій:

$$M_{i□} = \begin{cases} 1, & \text{якщо категорія } C□ \text{ є для } i \\ 0, & \text{якщо категорія } C□ \text{ відсутня для } i \end{cases} \quad (2.2)$$

де $M_{i□}$ - елемент матриці, що представляє j-ту категорію для і-го зразка.

C_i - категорія для і-го зразка.

C_j - j-та категорія серед k можливих категорій.

У ситуаціях, коли кількість категорій є надмірною (наприклад, сотні різних моделей авто), такий підхід стає неефективним, оскільки значно зростає розмірність ознак. У таких випадках доцільно застосовувати **цільове кодування (target encoding)**. Воно передбачає заміну кожної категорії числом — середнім значенням цільової змінної (наприклад, ціни) для відповідної групи:

$$C□ = \frac{1}{n□} \sum_{i=1}^{n□} y_i \ (2.3)$$

де n_j - кількість зразків для категорії C_j,

y_i - значення цільової змінної (ціни) для кожного зразка.

Такий підхід дозволяє зберегти корисну інформацію про вплив кожної категорії на залежну змінну та значно зменшує кількість нових ознак у наборі.

Масштабування числових ознак.

Реальні дані зазвичай включають числові змінні, що мають різний масштаб. Наприклад, пробіг може сягати сотень тисяч, тоді як рік випуску знаходиться у вузькому діапазоні. Щоб уникнути домінування ознак з більшими значеннями, використовується техніка **стандартизації (Standard Scaling)**.

Суть методу полягає у приведенні кожного значення до шкали з нульовим середнім значенням та одиничним стандартним відхиленням. Формально це виражається так:

$$X' = \frac{X - \mu}{\delta} \ (2.4)$$

де X — початкове значення ознаки,

μ — середнє значення ознаки,

а δ — стандартне відхилення ознаки.

Такий підхід є особливо важливим при застосуванні алгоритмів, чутливих до масштабу. Крім того, правильно масштабовані дані сприяють швидшій і стабільнішій збіжності моделі під час навчання.

Заповнення пропущених значень

У процесі обробки реальних даних досить часто виникає проблема відсутніх значень. Такі пропуски можуть з'являтися з різних причин — помилки під час збирання інформації, технічні збої, неповні анкети або просто відсутність певних параметрів. Водночас більшість алгоритмів машинного навчання не здатні працювати з даними, що містять пропущені значення, тому їх обов'язково потрібно обробляти.

Як зазначає McKinney [34], заповнення пропущених значень середнім або модою є базовою, але необхідною процедурою підготовки набору даних до побудови моделей.

Існує декілька загальновживаних способів заповнення (імпутації) таких значень, які вибираються залежно від типу змінної.

Для числових ознак

Числові дані найчастіше заповнюють середнім значенням (mean imputation). Це дозволяє зберегти загальну структуру розподілу даних і уникнути суттєвого зміщення:

$$X' = \mu \text{ (для числових змінних)} \quad (2.5)$$

Такий підхід є особливо ефективним у випадках, коли розподіл ознаки близький до нормального.

Для категоріальних змінних

У разі обробки категоріальних ознак доцільно використовувати **моду** — найчастіше зустрічаємо значення. Це забезпечує збереження характерного профілю змінної і не вводить штучних нових категорій:

$$X' = \text{мода (для категоріальних змінних)} \quad (2.6)$$

де μ — середнє значення для числових змінних,

а мода — найбільш поширене значення для категоріальних змінних.

Цей спосіб особливо корисний у разі невеликої кількості унікальних категорій.

Заповнення пропущених даних є критично важливим, оскільки дозволяє уникнути втрати обсягів інформації при видаленні записів з порожніми значеннями. Крім того, у поєднанні з іншими методами попередньої обробки (нормалізація, кодування тощо), імпутація формує основу якісного датасету, який забезпечує коректне функціонування моделей.

2. Вибір важливих ознак (Feature Selection)

Однією з ключових умов побудови ефективної моделі машинного навчання є коректний відбір ознак. Це дозволяє зосередитися лише на тих параметрах, які дійсно впливають на цільову змінну. Як зазначено в роботах Kuhn та Johnson [43], методи відбору ознак суттєво підвищують продуктивність моделей машинного навчання, дозволяючи сфокусуватись на найінформативніших характеристиках. Надмірна кількість нерелевантних або слабко пов’язаних ознак може призвести до ускладнення моделі, зниження її точності або виникнення ефекту перенавчання. Для зменшення розмірності даних та підвищення ефективності обчислень застосовуються методи оцінки інформативності змінних.

Одним із поширених підходів є використання взаємної інформації (Mutual Information) для вимірювання залежності між кожною ознакою X_j та цільовим значенням y. Взаємна інформація визначає, наскільки знання про одну змінну зменшує невизначеність іншої:

$$I(X_j;y) = H(y) - H(y|X_j) \quad (2.7)$$

де $I(X_j;y)$ — інформаційна вигода між ознакою X_j (характеристика, наприклад, пробіг, рік випуску, колір, тощо) і цільовою змінною y (це може бути ціна автомобіля або інша залежна змінна),

H(y) — це ентропія цільової змінної y, яка вимірює невизначеність або варіативність y,

H(y|X_j) - це умовна ентропія, яка вимірює невизначеність y, за умови, що ми знаємо X_j .

Вища взаємна інформація свідчить про більший вплив ознаки на результат.

Також часто використовують метод SelectKBest, що дозволяє автоматично вибрати k найінформативніших ознак на основі статистичної значущості. Наприклад, для регресійних задач застосовується коефіцієнт кореляції, який обчислюється за формулою:

$$\text{score}(X_i,y) = \frac{\sum(X_i - \overline{X_i})(y-\bar{y})}{\sqrt{\sum(X_i - \overline{X_i})^2 \sum(y-\bar{y})^2}} \quad (2.8)$$

де X_i - значення ознаки,

y - цільова змінна.

Цей показник варіюється від -1 до 1. Значення, близькі до меж цього діапазону, вказують на сильний зв’язок між ознакою та результатом.

Застосування зазначених методів дозволяє залишити у моделі лише ті змінні, які мають дійсний вплив на вартість автомобіля: пробіг, рік випуску, марку, тип палива тощо. Це зменшує обсяг обчислень і підвищує стабільність прогнозу.

3. Побудова нейронної мережі

Для задачі прогнозування цін транспортних засобів ефективним підходом є побудова штучної нейронної мережі. В праці [17] докладно описано архітектуру нейронних мереж, функцію ReLU, механізм уваги та методи оцінювання точності в задачах регресії. Така нейронна мережа складається з вхідного шару, одного або кількох прихованих шарів та вихідного шару. Кожен з них містить нейрони, які передають дані один одному.

У якості функції активації прихованих шарів використовується ReLU (Rectified Linear Unit):

$$f(x) = \max(0,x) \quad (2.9)$$

де x — вхід до нейрону,

f(x) — вихід функції активації.

Ця функція проста у реалізації та ефективна, оскільки нівелює негативні значення, передаючи лише позитивні сигнали далі.

Останній шар формує передбачення цільової змінної у через лінійне поєднання ваг та значень ознак:

$$y = \hat{y} = \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_n x_n + b \quad (2.10)$$

де ω_1, ω_2, ω_n - ваги, x_1, x_2, x_n - вхідні ознаки,

b – зсув.

Створення механізму уваги (Attention Mechanism)

Для покращення адаптивності моделі до складних вхідних даних використовується механізм уваги. Він дозволяє мережі фокусуватись на найбільш інформативних ознаках. Кожній ознаці присвоюється ваговий коефіцієнт α_j, розрахований за формулою:

$$\alpha_j = \frac{\exp(\omega_j^T x)}{\sum_{k=1}^{n} \exp(\omega_j^T x)} \quad (2.11)$$

де α_j - вага для характеристики X_j,

ω_j - вектор ваг для характеристики X_j,

x - вхідний вектор ознак.

Таким чином, модель виділяє ознаки, які найбільше впливають на результат, знижуючи вагу менш релевантних змінних. Це покращує загальну якість прогнозу та дозволяє гнучко адаптувати модель до нових даних.

Функція втрат

Для навчання моделі використовується середньоквадратична помилка (MSE) як функція втрат, яку потрібно мінімізувати:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2 \quad (2.12)$$

де $\widehat{y_i}$ - передбачене значення,

y_i - реальне значення.

4. Оцінка моделі

Оцінка точності – для оцінки точності моделі в задачі регресії використовуються такі метрики, як R^2 (коефіцієнт детермінації) та середнє абсолютне відхилення (MAE)**.**

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y_i})^2} \quad (2.13)$$

де $\widehat{y_i}$ - передбачене значення,

y_i - реальне значення,

$\bar{y_i}$ - середнє значення цільової змінної.

$$\text{MAE} = = \frac{1}{n}\sum_{i=1}^{n}|\widehat{y_i} - y_i| \quad (2.14)$$

2.2 Мова програмування, основні бібліотеки та інструменти

У межах даного дослідження як основну мову програмування було обрано **Python** [34], яка є надзвичайно популярною у сфері аналізу даних та машинного навчання завдяки своїй гнучкості, простоті синтаксису і широкій екосистемі спеціалізованих бібліотек. Для реалізації задачі прогнозування цін на автотранспортні засоби було використано низку інструментів, серед яких ключову роль відіграє **бібліотека Pandas**.

Pandas — це потужний інструмент для роботи з табличними даними у Python. Бібліотека забезпечує зручні структури даних — **Series** (одновимірні

послідовності) та **DataFrame** (двовимірні таблиці), які дозволяють зберігати, обробляти та аналізувати великі обсяги інформації в зручному форматі, подібному до електронних таблиць.

Серед ключових можливостей Pandas:

- **Імпорт та експорт даних -** підтримується зчитування та збереження даних у різноманітних форматах: CSV, Excel, JSON, SQL та інших. Це дозволяє інтегрувати Pandas з багатьма джерелами даних.
- **Фільтрація та сортування -** дані можна відбирати за умовами, сортувати за одним або кількома критеріями, змінювати індексацію та перевпорядковувати записи.
- **Групування та агрегація** - Pandas надає функціонал для об’єднання даних за категоріями, обчислення підсумкових статистичних характеристик (сума, середнє, кількість тощо) для груп, та побудови агрегованих таблиць.
- **Обробка відсутніх значень** - бібліотека дозволяє знаходити, видаляти або замінювати пропущені дані. Зокрема, для числових полів можна підставляти середнє або медіану, для категоріальних — моду.
- **Маніпуляції зі стовпцями** - можна легко додавати нові змінні на основі існуючих, змінювати типи даних, перейменовувати або видаляти стовпці, виконувати обчислення векторизовано.
- **Часові ряди** - Pandas має вбудовані засоби для роботи з датами й часом, що робить її зручною для обробки даних із часовою прив’язкою (наприклад, аналіз динаміки цін за роками).
- **Інтеграція з візуалізацією** - через сумісність із matplotlib можна одразу будувати графіки на основі оброблених таблиць: лінійні, стовпчасті, гістограми, діаграми розсіювання тощо.
- **Об’єднання та об’єкти баз даних** - Pandas підтримує операції об'єднання кількох таблиць, що особливо зручно при роботі з кількома джерелами або розрізненими наборами даних.

Завдяки гнучкому API та зручній структурі, Pandas став центральним інструментом у процесі побудови системи прогнозування. Усі етапи — від очищення до первинного аналізу та візуалізації результатів — були реалізовані за допомогою можливостей цієї бібліотеки.

NumPy (Numerical Python) — це одна з базових бібліотек Python, яка використовується для ефективної роботи з великими обсягами числових даних. Вона є основою для багатьох інших інструментів у сфері обчислень і аналізу даних. Основним об’єктом, з яким працює NumPy, є багатовимірний масив ndarray, що дає змогу здійснювати швидкі математичні операції на всьому наборі даних без необхідності використання циклів.

Ключові переваги бібліотеки NumPy:

- **Робота з масивами.** NumPy дозволяє створювати одновимірні, двовимірні та багатовимірні масиви одного типу, що дає змогу ефективно зберігати і обробляти великі обсяги інформації.
- **Універсальні функції.** Бібліотека підтримує широкий набір математичних функцій, які застосовуються до всіх елементів масиву одночасно. Це включає обчислення тригонометричних функцій, логарифмів, експонент, а також функцій округлення, обчислення максимумів і мінімумів.
- **Можливості індексації та зрізів.** NumPy надає зручні інструменти для доступу до елементів масиву за допомогою індексів, умовних виразів або масок. Це дозволяє здійснювати швидке відбирання підмасивів за критеріями.
- **Базова статистика та аналіз.** За допомогою вбудованих функцій можна обчислювати основні статистичні характеристики: середнє значення, медіану, стандартне відхилення, дисперсію, кореляцію та інші показники.
- **Операції лінійної алгебри.** NumPy включає набір функцій для реалізації стандартних операцій над матрицями: множення, транспонування,

обчислення оберненої матриці, розв’язування систем лінійних рівнянь тощо. Це особливо важливо при створенні моделей машинного навчання, де матричні обчислення відіграють центральну роль.

- **Генерація випадкових чисел.** Модуль numpy.random дає можливість створювати масиви випадкових чисел, що слідують різним розподілам (нормальному, рівномірному, біноміальному тощо), що корисно при моделюванні та оцінці статистичних властивостей даних.

Завдяки своїй високій продуктивності, NumPy є незамінним інструментом у роботі з числовими даними. Його функціональність значно перевершує можливості стандартних засобів Python, дозволяючи проводити ефективну обробку даних навіть у великих обсягах.

Для візуального представлення результатів аналізу та моделювання в даній роботі використано бібліотеку **Matplotlib**. Це один із найпоширеніших інструментів для створення графіків у середовищі Python, який дозволяє будувати як базові, так і складні графічні структури.

Matplotlib забезпечує побудову різноманітних типів візуалізацій, включаючи:

- **лінійні графіки**, які відображають зміну показників у часі або залежність між змінними;
- **стовпчикові діаграми**, корисні для порівняння значень у категоріях;
- **гістограми**, що відображають розподіл значень;
- **точкові діаграми (scatter plots)** для оцінки кореляції між двома змінними;
- **контурні графіки** для представлення багатовимірних даних;
- **комбіновані графіки**, що поєднують кілька типів даних на одному полотні.

Серед переваг бібліотеки:

- **Інтуїтивний синтаксис.** Вона надає простий інтерфейс для створення графіків — команди логічно структуровані, що дозволяє швидко виводити дані у візуальному форматі.
- **Гнучкість налаштувань.** Matplotlib підтримує детальне редагування параметрів — можна змінювати кольори, стилі ліній, розміри шрифтів, мітки осей, додавати легенди, заголовки тощо.
- **Збереження в різні формати.** Створені візуалізації можна експортувати у такі графічні формати, як PNG, JPEG, PDF, SVG. Це дозволяє використовувати їх у друкованих матеріалах або презентаціях.
- **Сумісність з іншими бібліотеками.** Matplotlib добре інтегрується з **pandas**, що дає змогу будувати графіки безпосередньо з DataFrame. Крім того, існують розширення, наприклад **Seaborn**, які розширюють функціонал Matplotlib та додають нові стилі й шаблони для оформлення.
- **Підтримка інтерактивності.** Бібліотека може використовуватись у поєднанні з інтерактивними середовищами (наприклад, Jupyter Notebook), що дозволяє створювати інтерактивні візуалізації для дослідження й презентації даних. Також можливе використання з графічними інтерфейсами для реалізації панелей керування або інтерфейсів аналітики.

В контексті дослідження, присвяченого прогнозуванню цін на транспортні засоби, Matplotlib використовувалась для:

- побудови графіків залежності між окремими ознаками та цільовою змінною (ціною);
- аналізу розподілу даних;
- візуалізації результатів навчання моделі (наприклад, графіків втрат або точності).

Таким чином, ця бібліотека стала невід’ємною частиною процесу аналізу та візуального представлення результатів дослідження.

Для реалізації алгоритмів машинного навчання у цьому проєкті застосовується мова програмування Python, а також одна з найбільш поширених бібліотек — **Scikit-learn** (скорочено sklearn). Цей інструментарій розроблений для вирішення широкого спектра задач, пов'язаних із побудовою, навчанням та оцінюванням моделей.

Scikit-learn вирізняється інтуїтивно зрозумілим синтаксисом, добре структурованим API та сумісністю з іншими науковими бібліотеками Python, такими як NumPy, SciPy та Pandas. Завдяки цьому він підходить як для дослідницьких експериментів, так і для практичних реалізацій моделей.

Основні функціональні можливості **Scikit-learn** [53].

Класифікація: логістична регресія, дерева рішень, метод опорних векторів (SVM), нейронні мережі.

Регресія: лінійна регресія, регресія з регуляризацією (Lasso, Ridge), моделі на основі дерев.

Кластеризація: алгоритм K-середніх (K-Means), DBSCAN, агломеративне групування.

Зменшення розмірності: головні компоненти (PCA), дискримінантний аналіз (LDA).

Обробка даних: масштабування, нормалізація, кодування категоріальних змінних, заповнення пропущених значень.

Оцінювання моделей: перехресна перевірка, метрики якості (MAE, RMSE, R^2), побудова кривих навчання.

Модуль **Sklearn.model_selection** — ключовий елемент для оцінювання моделей. Цей модуль надає засоби для поділу даних, перевірки якості моделей, а також налаштування параметрів.

До найбільш важливих функцій model_selection належать:

- **train_test_split** — поділ даних на навчальну та тестову вибірки.
- **cross_val_score** — проведення оцінювання моделі на кількох підвибірках для перевірки стабільності результатів.

- **KFold, StratifiedKFold, GroupKFold** — реалізація різних стратегій крос-валідації для більш гнучкого контролю процесу навчання.
- **GridSearchCV** — пошук оптимальних гіперпараметрів за допомогою повного перебору параметрів.
- **RandomizedSearchCV** — варіант оптимізації гіперпараметрів на основі випадкової вибірки, що є менш обчислювально затратним.
- **cross_validate** — оцінка моделі за кількома метриками одночасно.

Ці функції забезпечують контрольоване тренування моделей та дозволяють вибрати найкращі конфігурації для досягнення високої точності. У проєктах, де дані мають складну структуру, а моделі потребують точного налаштування (наприклад, у задачах передбачення цін), модуль model_selection є незамінним інструментом.

Таким чином, **Scikit-learn** забезпечує повний набір засобів для підготовки, навчання та валідації моделей, а також інструменти для вдосконалення результатів шляхом оптимізації параметрів. Його застосування є критично важливим для отримання стабільних і відтворюваних прогнозів у сфері аналізу вартості автотранспортних засобів.

У процесі підготовки даних для моделей машинного навчання важливу роль відіграє модуль **sklearn.preprocessing** бібліотеки scikit-learn. Його функціональність охоплює ключові етапи трансформації даних, які дозволяють привести вхідну інформацію до узгодженого формату для ефективного навчання моделей. Ці операції спрямовані на стандартизацію, кодування категоріальних змінних, заповнення пропущених значень та інші процедури.

Основні можливості sklearn.preprocessing включають:

1. Масштабування ознак:

- StandardScaler: перетворює числові ознаки так, щоб їх середнє дорівнювало нулю, а стандартне відхилення — одиниці.
- MinMaxScaler: нормалізує дані у межах фіксованого інтервалу, зазвичай [0, 1].

- MaxAbsScaler: масштабує ознаки, враховуючи найбільше абсолютне значення, зберігаючи знак.
- RobustScaler: застосовує масштабування на основі медіани та міжквартильного діапазону, стійке до викидів.

2. Нормалізація:

- Normalizer: перетворює рядки (зразки) так, щоб їх норма (наприклад, L2) дорівнювала одиниці, корисно при обробці векторів ознак різної довжини.

3. Кодування категоріальних ознак:

- LabelEncoder: присвоює кожній категорії унікальне числове значення.
- OneHotEncoder: створює двійкові ознаки для кожного рівня категоріальної змінної.
- OrdinalEncoder: перетворює категорії у впорядковані числові значення.

4. Обробка пропущених значень:

- SimpleImputer: дозволяє автоматично замінювати відсутні значення на середнє, медіану або моду.
- KNNImputer: використовує метод найближчих сусідів для оцінки й підстановки відсутніх даних.

5. Бінаризація:

- Binarizer: перетворює числові значення у 0 або 1, залежно від заданого порогу.

Застосування цих засобів дозволяє суттєво покращити якість та стабільність роботи моделей, зменшити чутливість до масштабів ознак і зробити навчання більш узагальненим. Модуль preprocessing є невід'ємною складовою попередньої обробки у проєктах аналізу даних на основі Python.

У практиці машинного навчання зменшення розмірності є важливим кроком, що дозволяє спростити структуру даних без втрати критично важливої інформації. Для цього у бібліотеці scikit-learn передбачено модуль **sklearn.decomposition**, який об'єднує популярні алгоритми декомпозиції. Його

основна мета — зменшити кількість ознак у даних, покращити обчислювальну ефективність моделей, усунути надлишковість та підвищити інтерпретованість.

Ключові методи, що реалізовані в цьому модулі:

1. Principal Component Analysis (PCA) — метод лінійної декомпозиції, що дозволяє знайти основні компоненти, які пояснюють найбільшу частину дисперсії в даних. У випадку з числовими змінними, як-от технічні характеристики транспортних засобів, PCA допомагає зменшити кількість ознак, зберігаючи основний обсяг корисної інформації для прогнозування.
2. Truncated Singular Value Decomposition (TruncatedSVD) — метод, що використовується для роботи з великими та розрідженими матрицями, включаючи текстові або категоріальні дані. Він зменшує розмірність, зберігаючи значущі компоненти, і є корисним у задачах обробки текстів або об’єднаних ознак.
3. Non-negative Matrix Factorization (NMF) — техніка декомпозиції, яка працює з матрицями, що мають тільки невід’ємні значення. На відміну від PCA, тут компоненти зберігають лише додатні значення, що може бути корисним у прикладних завданнях, де інтерпретованість результатів є критичною, наприклад, у тематичному аналізі.
4. Factor Analysis — статистичний метод, що моделює залежності між змінними через вплив прихованих (латентних) факторів. Його використовують для виявлення структури в даних або усунення зайвої варіативності.
5. Latent Dirichlet Allocation (LDA) — алгоритм, який застосовується у текстовому моделюванні для виявлення прихованих тем у документах. LDA є незамінним у природній мовній обробці для побудови тематичних моделей.

Ці методи дозволяють скоротити кількість змінних, що використовуються у моделюванні, зберігаючи при цьому важливу інформацію.

Вони також покращують візуалізацію високовимірних даних і підвищують ефективність алгоритмів машинного навчання при роботі з великими наборами даних.

Під час побудови моделей машинного навчання важливо не лише навчити алгоритм на даних, а й об'єктивно оцінити його якість. Для цього в бібліотеці scikit-learn передбачено окремий підмодуль — **sklearn.metrics**, який містить широкий набір функцій для аналізу результатів моделювання як у задачах класифікації, так і регресії.

Ключові функції, що використовуються в рамках цього дослідження:

- accuracy_score — метрика, що вимірює частку правильних передбачень серед усіх. Застосовується у задачах класифікації як базова міра точності.
- mean_squared_error (MSE) — середнє квадратичне відхилення між прогнозованими та фактичними значеннями. Особливо корисне в регресійних задачах, де великі відхилення мають більший вплив на результат.
- mean_absolute_error (MAE) — середнє абсолютне відхилення, яке також використовується у регресії. На відміну від MSE, менш чутливе до окремих великих помилок.
- roc_auc_score — показник якості бінарної класифікації, що базується на площі під ROC-кривою. Враховує співвідношення між чутливістю (recall) та специфічністю при різних порогах.
- r2_score (R^2) — коефіцієнт детермінації, який використовується в регресії для оцінки пояснювальної здатності моделі. Значення, близьке до 1, свідчить про високу точність, тоді як значення 0 означає відсутність корисної інформації в моделі.
- confusion_matrix — таблиця, що демонструє кількість правильних і помилкових передбачень для кожного класу. Дозволяє оцінити якість класифікації по кожному окремому класу.

- precision_score, recall_score, f1_score — додаткові метрики для класифікаційних задач. Precision визначає точність позитивних передбачень, recall — повноту виявлення позитивних класів, а F1 — гармонійне середнє між ними, що є корисним у разі незбалансованих даних.

Ці інструменти дозволяють об'єктивно оцінити ефективність моделей та зробити висновки щодо їх придатності до використання на практиці.

У процесі побудови моделей машинного навчання важливою задачею є відбір найбільш інформативних ознак. Застосування процедур зменшення кількості вхідних змінних дозволяє підвищити ефективність моделі, уникнути перенавчання та скоротити час обчислень. У бібліотеці scikit-learn для цього призначено модуль **feature_selection**.

Одним із базових методів є SelectKBest, що дозволяє автоматично обрати фіксовану кількість найкращих ознак відповідно до значення обраного статистичного критерію. Цей підхід базується на попередньому аналізі кожної ознаки окремо, після чого ознаки з низькою інформативністю виключаються з подальшого аналізу.

Інший ефективний інструмент — mutual_info_regression — оцінює ступінь взаємозалежності між окремою ознакою та цільовою змінною. Метод використовує поняття взаємної інформації для виявлення навіть нелінійних залежностей, що робить його особливо корисним у випадках складних структур даних.

Ще один підхід — Recursive Feature Elimination (RFE) — полягає у поетапному вилученні найменш значущих ознак. Алгоритм навчає модель, оцінює важливість ознак, видаляє найменш впливову, і повторює процес до досягнення бажаної кількості змінних. Це забезпечує кращу інтерпретованість результатів і зменшує ризик перенавчання.

Крім відбору ознак, важливим кроком є обробка пропущених значень. Для цього в scikit-learn існує модуль impute, зокрема клас SimpleImputer, який

дозволяє автоматизовано заповнювати відсутні дані за заданими стратегіями — середнім значенням, медіаною або модою. Якісна імпутація є критично важливою, оскільки більшість алгоритмів машинного навчання не працюють з пропущеними значеннями.

Таким чином, поєднання модулів feature_selection та impute у бібліотеці scikit-learn дозволяє забезпечити якісну попередню підготовку даних, що є основою для побудови точних та надійних моделей прогнозування.

ColumnTransformer — це компонент бібліотеки scikit-learn, призначений для паралельного застосування різних методів обробки до окремих стовпців у структурі даних. Його використання є доцільним у випадках, коли ознаки мають різні типи: числові, категоріальні або текстові, і вимагають індивідуального підходу до трансформації.

Основні характеристики ColumnTransformer полягають у тому, що він дозволяє:

- застосовувати окремі перетворення до різних груп ознак (наприклад, масштабування числових ознак через StandardScaler, кодування категоріальних — через OneHotEncoder);
- поєднувати ці операції в єдиному об'єкті, що спрощує побудову ланцюгів обробки даних;
- інтегруватися з об'єктом Pipeline, що дозволяє об’єднати попередню обробку даних та навчання моделі у спільний робочий процес;
- обирати, які стовпці обробляти, які — передати без змін, а які — відкинути.

Використання ColumnTransformer забезпечує підвищену гнучкість і точність при підготовці неоднорідних наборів даних до моделювання, особливо у задачах регресії чи класифікації.

TensorFlow Keras [33]— це високорівнева бібліотека для побудови та навчання нейронних мереж, яка входить до складу платформи TensorFlow. Завдяки зручному синтаксису й широкому функціоналу вона є одним із

найпоширеніших інструментів у сфері глибокого навчання. Серед основних компонентів tensorflow.keras, які використовуються для розв’язання задач прогнозування, можна виокремити:

- Sequential — базова структура для побудови послідовної моделі, у якій шари додаються один за одним;
- Dense — шар повного з’єднання нейронів, який забезпечує передавання інформації між шарами;
- Dropout — регуляризаційна техніка, що випадковим чином «вимикає» нейрони під час тренування для запобігання перенавчанню;
- Adam — ефективний алгоритм оптимізації градієнта, який поєднує переваги методів Momentum і RMSProp;
- EarlyStopping та ReduceLROnPlateau — функціональні компоненти, що дозволяють автоматично зупинити тренування при відсутності поліпшення або змінювати швидкість навчання під час оптимізації.

XGBoost (Extreme Gradient Boosting) — це алгоритм градієнтного бустингу, який демонструє високу точність у задачах прогнозування. У контексті оцінювання вартості автомобілів, XGBoost може бути використаний для моделювання залежності між ціною та характеристиками транспортного засобу (наприклад, пробіг, рік випуску, тип пального, марка).

Ключові переваги XGBoost включають:

- вбудовану регуляризацію, що дозволяє зменшити ризик перенавчання;
- адаптивність до складних структур даних, у тому числі до нелінійних залежностей між ознаками;
- оцінювання важливості ознак, що дає змогу проаналізувати, які параметри найбільше впливають на цільову змінну. Зокрема, такі ознаки, як рік випуску та пробіг, часто виявляються найбільш значущими при прогнозуванні ціни автомобіля.

2.3 Переваги та недоліки обраних методів.

Для досягнення високої точності в задачах прогнозування цін на автотранспортні засоби необхідно розглянути кілька підходів машинного навчання та оцінити їх доцільність у конкретному контексті. У межах цієї роботи проведено аналіз різних методів, із акцентом на сильні та слабкі сторони кожного. При оцінюванні були враховані висновки, представлені в наукових джерелах [12, 14, 33, 13, 17, 19, 7].

1. Лінійна регресія

Сильні сторони:

- **Зрозумілість і простота у застосуванні.** Цей метод дозволяє швидко отримати базовий прогноз, що особливо зручно при початковому аналізі даних [12].
- **Інтерпретованість.** Завдяки відкритості структури моделі легко визначити, як саме кожна змінна впливає на кінцевий результат.
- **Швидкість.** Лінійна регресія ефективно працює при обробці невеликих наборів даних з обмеженим числом параметрів.

Обмеження:

- **Лінійність.** Метод обмежений у застосуванні, коли зв’язки між характеристиками не є лінійними, що часто трапляється на реальних даних.
- **Обмежена гнучкість.** При наявності складних взаємозв’язків між змінними або взаємодіючих факторів модель втрачає точність.

У порівнянні з нейронними мережами:

- На відміну від лінійної регресії, нейромережі здатні навчатися на складних, нелінійних залежностях, що часто зустрічаються у даних про автомобілі (наприклад, взаємозв’язок між роком випуску, пробігом і типом пального) [17].

- Крім того, глибинні моделі краще масштабуються під великі обсяги даних і дозволяють автоматично виявляти взаємодії між ознаками, які не потрібно явно задавати.

2. Дерева рішень

Методи, побудовані на базі дерев рішень, активно застосовуються у задачах прогнозування завдяки своїй прозорості та здатності ефективно працювати з різними типами даних. Їх особливість полягає у побудові логічної структури, яка приймає рішення шляхом поступового розбиття простору ознак.

Переваги:

- **Зрозуміла структура.** Моделі, створені у вигляді дерева, легко візуалізувати та інтерпретувати, що робить їх зручними для аналізу прийняття рішень [13, 14].
- **Гнучкість у роботі з різними даними.** Метод підходить для числових і категоріальних змінних без необхідності їхнього попереднього перетворення.
- **Стійкість до нелінійностей.** Алгоритм здатен моделювати складні залежності між ознаками, не потребуючи припущень щодо форми цих залежностей.

Недоліки:

- **Ризик перенавчання.** Без відповідної регуляризації (наприклад, обрізання дерева або обмеження глибини) модель може втратити здатність до узагальнення [33].
- **Менш точні прогнози на складних даних.** Індивідуальні дерева не завжди демонструють високу точність у випадках із великою кількістю перехресних взаємодій.
- **Вразливість до шумів.** Модель може переоцінити значення ознак, які випадково корелюють із цільовою змінною.

Порівняння з нейронними мережами:

- На відміну від дерев, які добре ілюструють логіку класифікації або регресії, нейронні мережі дозволяють моделювати значно складніші функціональні залежності між ознаками. Особливо це актуально для високорозмірних задач, де є велика кількість змінних та їх взаємодій [17].
- Застосування механізму уваги у нейронних мережах додатково дозволяє зосередити модель на ключових змінних, що підвищує її адаптивність до структурно складних даних.

3. Ансамблеві методи (Random Forest, XGBoost)

Ансамблеві підходи, такі як Random Forest і XGBoost, ґрунтуються на поєднанні кількох базових моделей для отримання більш стабільних і точних прогнозів. Завдяки агрегації результатів, ці методи дозволяють зменшити вплив окремих похибок та покращити узагальнення на нових даних.

Переваги:

- **Стійкість до перенавчання.** Об'єднання кількох моделей зменшує ймовірність того, що модель "запам’ятає" особливості навчального набору і втратить здатність до узагальнення [19].
- **Висока прогностична здатність.** Ансамблеві алгоритми демонструють гарні результати в задачах, де присутні складні залежності або велика кількість ознак.
- **Гнучкість щодо типу даних.** Підходить для обробки як числових, так і категоріальних змінних та може застосовуватись до великих за обсягом і складністю наборів даних.
- **Недоліки:**
- **Обчислювальні витрати.** Навчання ансамблевих моделей часто потребує значного часу та ресурсів, особливо при великій кількості дерев чи параметрів.
- **Обмежена прозорість.** Через складність архітектури, інтерпретація внутрішньої логіки моделей є ускладненою, що створює труднощі при поясненні результатів [13, 19].

Порівняння з нейронними мережами:

- У той час як ансамблеві алгоритми забезпечують високу точність, вони не завжди здатні відобразити складні нелінійні взаємозв'язки між змінними так ефективно, як це роблять глибокі нейронні мережі.
- Нейронні мережі, особливо ті, які використовують механізм уваги, здатні фокусуватись на найбільш релевантних ознаках. Це забезпечує додаткову гнучкість і точність у прогнозах, особливо в задачах із прихованими закономірностями та великою кількістю взаємодіючих факторів [7, 17].

4. Нейронні мережі з механізмом уваги

Сучасні глибокі нейронні мережі, зокрема моделі, що інтегрують механізм уваги (attention), демонструють високу ефективність у задачах прогнозування зі складною структурою вхідних даних. Завдяки своїй здатності виявляти складні та приховані взаємозв'язки, вони є перспективним підходом у випадках, коли традиційні методи, такі як лінійна регресія або дерева рішень, виявляються недостатньо гнучкими.

Переваги:

- Здатність до моделювання нелінійних взаємозалежностей між ознаками, що є критично важливим у задачах із високою складністю структури даних [17].
- Завдяки attention-механізму модель може автоматично визначати та акцентувати увагу на найбільш інформативних ознаках, що покращує загальну якість прогнозу.
- Добре масштабуються на великі набори даних, зберігаючи при цьому високу адаптивність до варіативності входу.

Недоліки:

- Високі вимоги до обчислювальних ресурсів та часу навчання.
- Ускладнена інтерпретація структури моделі в порівнянні з більш простими методами.

Порівняння з іншими підходами [7, 19]:

- У порівнянні з лінійною регресією та деревами рішень, нейромережі забезпечують значно більшу гнучкість і точність за рахунок здатності враховувати складні міжозначні залежності.
- Ансамблеві моделі (наприклад, Random Forest або XGBoost) демонструють високу точність, проте поступаються нейромережам у виявленні глибших патернів і слабко виражених залежностей. Застосування attention-механізму дає можливість цілеспрямовано підсилити обробку саме релевантних ознак, що додатково підвищує якість моделі.

Висновок

Математичний опис моделі моделі прогнозування цін на автотранспортні засоби відображає всі етапи процесу: від обробки даних до тренування нейронної мережі з використанням механізму уваги. Кожен етап має чітке математичне обґрунтування. Це дозволяє побудувати ефективну модель для прогнозування цін на автомобілі, яка може бути адаптована до різних сценаріїв. Крім того, це може забезпечити високу точність прогнозів.

Проведений порівняльний аналіз популярних методів машинного навчання для задачі прогнозування цін на автотранспортні засоби дозволяє зробити обґрунтований вибір оптимального підходу. Виявлено, що традиційні моделі, такі як лінійна регресія, відзначаються простотою та інтерпретованістю. Але є недостатньо гнучкими для моделювання складних, нелінійних залежностей.

Дерева рішень можуть бути кращими в цій ситуації, проте мають схильність до перенавчання та обмежену точність на складних наборах даних.

Нейронні мережі, особливо з механізмом уваги, є інструментом для прогнозування цін на автомобілі, який заслуговує на увагу. Це тому, що вони здатні обробляти складні взаємодії між ознаками, які важко моделювати за допомогою інших методів, таких як: лінійна регресія чи дерева рішень.

Ансамблеві методи, зокрема Random Forest та XGBoost, демонструють високу точність і стійкість до перенавчання. Але у здатності виявляти глибокі приховані закономірності в даних можуть поступатися нейронним мережам. Натомість нейронні мережі з механізмом уваги дозволяють не лише ефективно працювати з великими обсягами різнорідної інформації, а й зосереджуватись на найбільш ознаках, які здатні підвищити точність прогнозу. В задачах з великою кількістю ознак і складними нелінійними залежностями, така здатність дає їм значну перевагу.

Отже, з урахуванням сучасних вимог до точності, адаптивності та масштабованості, використання нейронних мереж з attention-механізмом у поєднанні з ансамблевими методами є доцільним та виправданим вибором для реалізації моделі прогнозування цін на автотранспортні засоби в межах даного дослідження.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ МЕТОДІВ DATA SCIENCE ДЛЯ ПРОГНОЗУВАННЯ ЦІН НА АВТОТРАНСПОРТНІ ЗАСОБИ

3.1 Підготовка даних для моделі прогнозування цін на автотранспортні засоби

У даному розділі описано повний цикл побудови, підготовки, тренування та оцінки моделей для прогнозування вартості вживаних автомобілів за допомогою нейронних мереж, XGBoost, а також гібридного підходу.

Цей набір бібліотек забезпечує повний цикл машинного навчання: від завантаження й підготовки даних до навчання моделей та аналізу результатів. Рис. 3.1.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import OrdinalEncoder, StandardScaler, LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from category_encoders import TargetEncoder
from sklearn.compose import ColumnTransformer
from sklearn.metrics import r2_score, mean_absolute_percentage_error, mean_absolute_error
from sklearn.feature_selection import mutual_info_regression, SelectKBest, chi2, f_regression
from sklearn.impute import SimpleImputer
from sklearn.utils import resample
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Layer
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint
from tensorflow.keras.regularizers import L2
from tensorflow.keras.utils import register_keras_serializable
from tensorflow.keras import Model, Input
import keras_tuner as kt
from catboost import CatBoostRegressor
import optuna
import json
import xgboost as xgb
import lightgbm as lgb
import shap
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display
✓ 0.0s
```

Рис. 3.1 – Бібліотеки, які використовувались у проєкті.

Дані для оцінки моделі отримано з платформи Kaggle [4] - однієї з найпопулярніших для розміщення наборів даних в тому числі і даних про вартість та характеристики автомобілів.

Перш за все, було завантажено датасет used_cars_data.csv, що містить інформацію про вживані автомобілі, зокрема їх технічні характеристики, колір, пробіг, вік тощо. Датасет US Used cars dataset містить інформацію про 3 мільйони справжніх б/в автівок. Містить 66 колонок за характеристиками про автомобілі, зокрема:

Vin: Ідентифікаційний номер автомобіля – це унікальний закодований рядок для кожного автомобіля.

back_legroom: кількість місця для ніг на задньому сидінні в дюймах.

dealer_zip: поштовий індекс дилера.

description: опис транспортного засобу на сторінці транспортного засобу.

engine_cylinders: конфігурація двигуна. Наприклад: I4, V6 тощо.

front_legroom: кількість місця для ніг на передньому сидінні в дюймах.

fuel_tank_volume: ємність паливного бака в галонах.

fuel_type: домінуючий тип палива, що споживається транспортним засобом.

height: висота автомобіля в дюймах.

horsepower: це потужність, яку виробляє двигун.

interior_color: колір салону транспортного засобу.

Візуалізація цього приведена на рис 3.2.

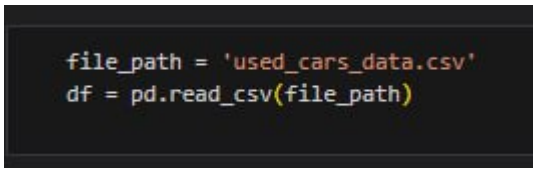


Рис. 3.2 – Завантаження датасету used_cars_data.csv

Для тренування моделі я завантажив датасет US Used cars dataset [4] у форматі CSV.

Для очищення даних було виконано наступні кроки:

- Видалено технічні стовпці, що не несуть корисної інформації (vin, listing_id, description, координати, URL тощо). Рис. 3.3.

```
columns_to_drop = [
    'bed', 'bed_height', 'bed_length', 'cabin', 'combine_fuel_economy',
    'is_certified', 'is_cpo', 'is_oemcpo', 'vehicle_damage_category',
    'vin', 'listing_id', 'trimId', 'sp_id',
    'description',
    'main_picture_url',
    'dealer_zip',
    'city',
    'latitude', 'longitude',
    'listed_date',
    'sp_name',
    'franchise_make',
    'transmission_display',
    'wheel_system_display',
    'power',
    'torque',
    'major_options',
    'make_name', 'model_name', 'trim_name'
]

df.drop(columns=columns_to_drop, inplace=True)
```

Рисунок 3.3 – Видалення стовпців

- Виключено всі записи з пропущеними значеннями (dropna) для первинного аналізу.
- Було зроблено випадкову вибірку у 300 тис. записів для зручності обчислень, що можна побачити на рис. 3.4.

```
df_cleaned = df.dropna(how='any')
sample_df = df_cleaned.sample(n=300000, random_state=42)

sample_df.to_csv('used_cars_sample_300k.csv', index=False)
```

Рис. 3.4 – Вибірка у 300 тис. записів

- Вибірку було записано у файл used_cars_sample_300k.csv, з яким вже надалі й працюю. Завантаження даних здійснюється за допомогою функції read_csv з бібліотеки pandas. Цей крок формує основну структуру даних (DataFrame) під назвою df, з якою виконуються подальші операції

з аналізу, очищення, інженерії ознак і моделювання. Перевірка змінної df підтверджує успішне зчитування набору даних. Рис. 3.5.

```
df = pd.read_csv('used_cars_sample_300k.csv')
df
✓ 1.3s
```

Рис. 3.5 – Завантаження даних з файлу used_cars_sample_300k.csv

Кожен з записів, завантаженого датасету представляє окрему пропозицію на ринку вживаних автомобілів. Таблиця містить **36 стовпців**, які охоплюють як числові, так і категоріальні ознаки. Основні з них:

- price — цільова змінна, що відображає вартість автомобіля.
- body_type, engine_type, exterior_color, transmission, wheel_system — категоріальні характеристики.
- city_fuel_economy, engine_displacement, seller_rating, savings.amount — числові показники, які можуть мати значення для прогнозування.
- frame_damaged, fleet, salvage, theft_title — бінарні змінні, що вказують на стан або історію авто.
- daysonmarket — тривалість знаходження авто в оголошенні, яка може корелювати з ціною.

Наявність деяких пропущених значень (позначених як -- або порожні комірки) передбачає необхідність подальшої обробки.

	back_legroom	body_type	city_fuel_economy	daysonmarket	engine_cylinders	engine_displacement	engine_type	exterior_color	fleet	frame_damaged	...	price	salvage	savings_amount	seller_rating	theft_title	transmission	wheel_system	wheelbase	w
0	37.2 in	SUV / Crossover	26.0	15	I4 Diesel	2000.0	I4 Diesel	Blue	False	False	...	40980.0	False	2219	4.260870	False	A	AWD	113.1 in	
1	33.7 in	Coupe	17.0	14	I6	3000.0	I6	ALPINE WHITE	False	False	...	46900.0	False	3555	4.565217	False	A	RWD	110.7 in	
2	39 in	SUV / Crossover	20.0	65	I4	2300.0	I4	Blue Metallic	True	False	...	39100.0	False	1770	4.441176	False	A	AWD	119.1 in	
3	37.9 in	SUV / Crossover	20.0	365	I4	2400.0	I4	White	False	False	...	9377.0	False	1330	4.157895	False	A	AWD	104 in	7
4	38.5 in	SUV / Crossover	20.0	71	H4	2400.0	H4	Crystal White Pearl	False	False	...	35500.0	False	727	4.571429	False	CVT	AWD	113.8 in	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
299995	38.3 in	SUV / Crossover	22.0	42	I4	2400.0	I4	Twilight Blue Metallic	False	False	...	15299.0	False	498	3.738462	False	A	AWD	103.1 in	
299996	39.1 in	Sedan	25.0	2	I4	2000.0	I4	Deep Black Pearl Metallic	False	False	...	16999.0	False	29	4.000000	False	A	FWD	110.4 in	
299997	36.1 in	SUV / Crossover	16.0	47	V6	3600.0	V6	White	True	False	...	9500.0	False	796	5.000000	False	A	AWD	113.8 in	
299998	38 in	Sedan	28.0	11	I4	2500.0	I4	Red	False	False	...	20990.0	False	369	3.833333	False	A	FWD	111.2 in	
299999	41.4 in	Sedan	30.0	130	I4	1800.0	I4	Super White	True	False	...	16990.0	False	686	3.931034	False	CVT	FWD	106.3 in	

300000 rows × 36 columns

Рис.3.6 – Перші та останні записи з набору даних, який містить 300 000 рядків і 36 стовпців, що описують характеристики вживаних автомобілів.

- У наборі даних деякі числові ознаки (наприклад, back_legroom, wheelbase, width тощо) представлені у вигляді рядків з позначенням одиниць вимірювання ("in" — дюйми, “seats” – сидіння, “gal” - галони). Для коректного використання цих ознак у моделі необхідно:

a. Замінити відсутні значення ('--') на NaN.

b. Видалити текстові позначення " in", “seats”, “gal” з рядків.

c. Перетворити очищені значення на числовий тип float.

Рис.3.7 - Попередня обробка числових ознак, які містять одиниці вимірювання ("in" — дюйми).

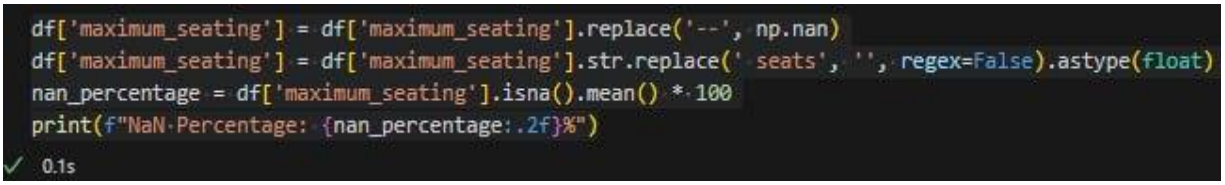


Рис. 3.8 - Попередня обробка числових ознак, які містять одиниці вимірювання ("seats" — сидіння).

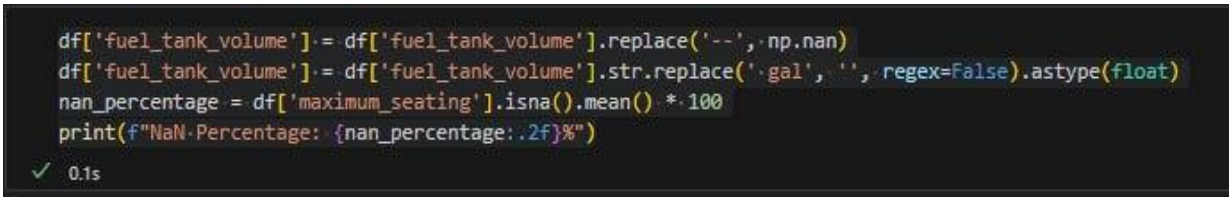


Рис. 3.9 - Попередня обробка числових ознак, які містять одиниці вимірювання ("gal" —галони).

- Після ідентифікації числових ознак, що містять пропущені значення, було виконано заміну NaN на середнє значення відповідної колонки (mean). Це стандартна стратегія, яка дозволяє зберегти обсяг вибірки без суттєвого викривлення розподілу ознак. Зокрема, до обробки були

включені як раніше підготовлені ознаки в дюймах (inch_cols), так і додаткові змінні maximum_seating і fuel_tank_volume. У результаті цього етапу усі пропуски в зазначених числових стовпцях були усунуті, про що свідчить підсумкове повідомлення з isnull().sum().

```
nan_num_cols = inch_cols + ['maximum_seating', 'fuel_tank_volume']
for col in nan_num_cols:
    mean_col = df[col].mean()
    df[col] = df[col].fillna(mean_col)
print("Missing values:", df[nan_num_cols].isnull().sum())
```
✓ 0.0s

Рис.3.10 – Усунення пропусків в зазначених числових стовпцях.

- Перетворено логічн значення (True/False) на числові (0/1). Деякі стовпці в наборі даних мають логічний тип (True або False), наприклад, frame_damaged, fleet, salvage тощо. Більшість моделей машинного навчання не підтримують логічний тип напряму, тому було виконано конвертацію таких змінних у цілочисловий формат: True → 1, False → 0. Це було реалізовано через автоматичне виявлення всіх логічних колонок і подальше приведення їх до типу int. Таке перетворення дозволяє ефективно інтегрувати логічні ознаки в загальний процес моделювання.

```
bool_columns = df.select_dtypes(include=['bool']).columns.tolist()
df[bool_columns] = df[bool_columns].astype(int)  # True → 1, False → 0
```
✓ 0.0s

Рис.3.11 – Перетворення логічних значень (True/False).

- Розраховано вік автомобіля (car_age) шляхом віднімання року виробництва від поточного року. Для підвищення інформативності моделі було створено нову ознаку - car_age, що відображає вік автомобіля. Ця змінна розраховується як різниця між поточним роком і роком випуску (year). Такий підхід дозволяє явно врахувати зношеність транспортного засобу, що має суттєвий вплив на його ціну. Етапи обробки:

a. Стовпець year перетворюється у тип int для коректних арифметичних операцій.
b. Отримується поточний рік з використанням pandas.Timestamp.now().
c. Обчислюється нова ознака car_age = current_year - year.
d. Початковий стовпець year видаляється як такий, що вже не потрібен.

```
df['year'] = df['year'].astype(int)
current_year = pd.Timestamp.now().year
df['car_age'] = current_year - df['year']
df.drop(columns=['year'], inplace=True)
```
✓ 0.0s

Рис.3.12 – Створення колонки car_age

```
df['car_age'] = df['car_age'].astype(int)
df['car_age']
```
✓ 0.0s

Рис.3.13 – Перетворення в тип int

- Оскільки модель прогнозує ціну автомобіля, важливо виключити екстремальні значення, які можуть суттєво викривити навчання. Для цього було застосовано метод обрізання за межами квантилів. Видалено аномальні значення ціни, залишивши лише дані між 0.5% та 99.5% квантилями. Цей підхід дозволяє зберегти основну масу "реалістичних" значень, водночас позбавившись лише найбільш атипових спостережень.

```
price_upper_bound = df['price'].quantile(0.995)
price_lower_bound = df['price'].quantile(0.005)
df = df[(df['price'] >= price_lower_bound) & (df['price'] <= price_upper_bound)]
```
✓ 0.0s

Рис 3.14 – Виключення екстремальних значень

Після фільтрації датасет містить лише ті автомобілі, ціна яких знаходиться у вказаному діапазоні, що сприяє більш стабільному і точному навчанню моделі.

3.2 Інформаційно-аналітичний огляд ознак для побідови моделей для прогнозування цін на автотранспортні засоби

Наступний фрагмент коду стосується підготовки ознак до кодування, а саме — розділення категоріальних змінних за кардинальністю (кількістю унікальних значень). Було здійснено розділення ознак на:

- **Цільову змінну**: price.
- **Ознаки**: всі інші стовпці, окрім price.
- Ознаки додатково класифікуються за типом:
- **Числові** (наприклад, пробіг, потужність, об'єм двигуна);
- **Категоріальні** (наприклад, колір кузова, тип двигуна);
- **Логічні** (наприклад, frame_damaged).
- Категоріальні змінні розділили на групи за кардинальністю:
- **Низька кардинальність** (≤15 унікальних значень) → One-Hot Encoding;
- **Висока кардинальність** → Target Encoding.

```
y = df['price']
X = df.drop('price', axis=1)
numerical_columns = X.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_columns = X.select_dtypes(include=['object', 'category']).columns.tolist()
bool_columns = X.select_dtypes(include=['bool']).columns.tolist()
low_cardinality_cols = []
high_cardinality_cols = []

for col in categorical_columns:
    if X[col].nunique() <= 15:
        low_cardinality_cols.append(col)
    else:
        high_cardinality_cols.append(col)

print("Low-Cardinality Features:", low_cardinality_cols)
print("High-Cardinality Features:", high_cardinality_cols)
✓ 4.4s
```

Рис.3.15 - Підготовка ознак до кодування

Для стандартизованої обробки даних було використано ColumnTransformer, який дозволяє застосовувати різні типи перетворень до різних груп ознак в одному об'єкті. Було налаштовано три окремі підходи:

1. **One-Hot Encoding** — для ознак з низькою кардинальністю (low_cardinality_cols), що створює бінарні стовпці для кожного унікального значення.
2. **Target Encoding** — для ознак з високою кардинальністю (high_cardinality_cols), які кодуються середнім значенням цільової змінної для кожної категорії.
3. **Standard Scaling** — для числових та логічних змінних, що масштабує значення до стандартного розподілу (середнє = 0, стандартне відхилення = 1). Після налаштування трансформатора виконується його застосування. Це забезпечує готовий до навчання масив ознак у числовому вигляді. Також виводиться кількість фінальних ознак та їхня розмірність. Цей етап дозволяє повністю підготувати дані до побудови моделей машинного навчання.

```
preprocessor = ColumnTransformer(
    transformers=[
        ('low_cardinality', OneHotEncoder(), low_cardinality_cols),
        ('high_cardinality', TargetEncoder(cols=high_cardinality_cols), high_cardinality_cols),
        ('numerical', StandardScaler(), numerical_columns + bool_columns)
    ])
scaler_y = StandardScaler()
X_processed = preprocessor.fit_transform(X, y)
feature_names = preprocessor.get_feature_names_out()
print(f"Total features: {len(feature_names)}")
print(f"X: {X_processed.shape[1]}")
✓ 1.6s
```

Рис.3.16 - Формування єдиного модуля обробки ознак з використанням ColumnTransformer

Далі було обчислено важливість ознак за трьома методами:

- **Information Gain** (інформаційний приріст) - використовується для виявлення найбільш інформативних змінних. Для оцінки внеску кожної ознаки у передбачення цільової змінної (price) було використано метод взаємної інформації (mutual_info_regression). Цей метод не робить припущень щодо лінійності залежності, тому особливо корисний при роботі з немонотонними або категоріальними даними. Етапи:

a. Обчислення взаємної інформації між кожною ознакою та цільовою змінною.
b. Створення таблиці з відповідними значеннями.
c. Сортування за спаданням інформативності.
d. Візуалізація Top-15 найбільш значущих ознак - проводиться через горизонтальну гістограму.

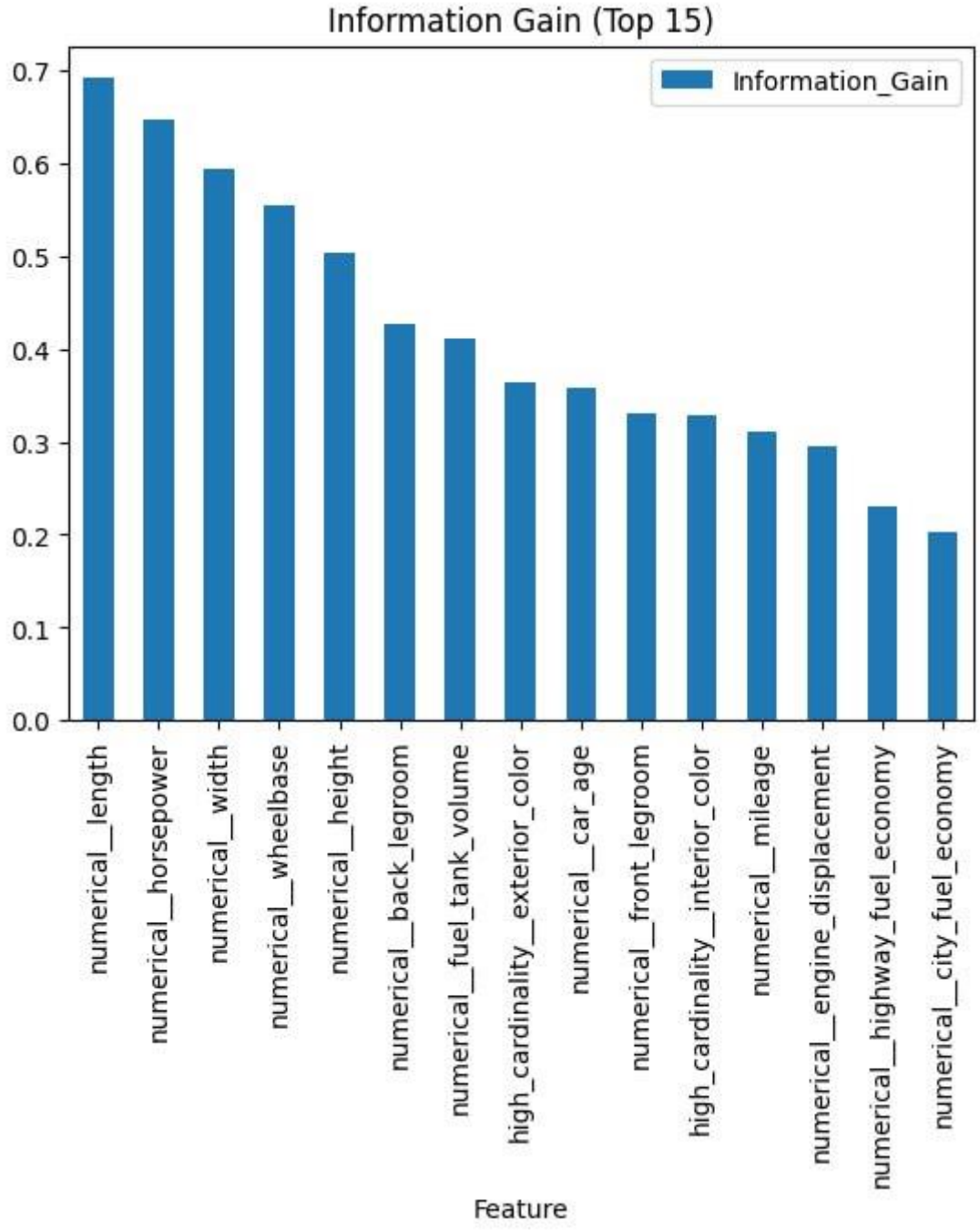


Рис.3.17 - Top-15 найбільш значущих ознак

Цей етап дозволяє не лише краще зрозуміти структуру даних, а й потенційно скоротити розмірність, виключивши найменш значущі ознаки.

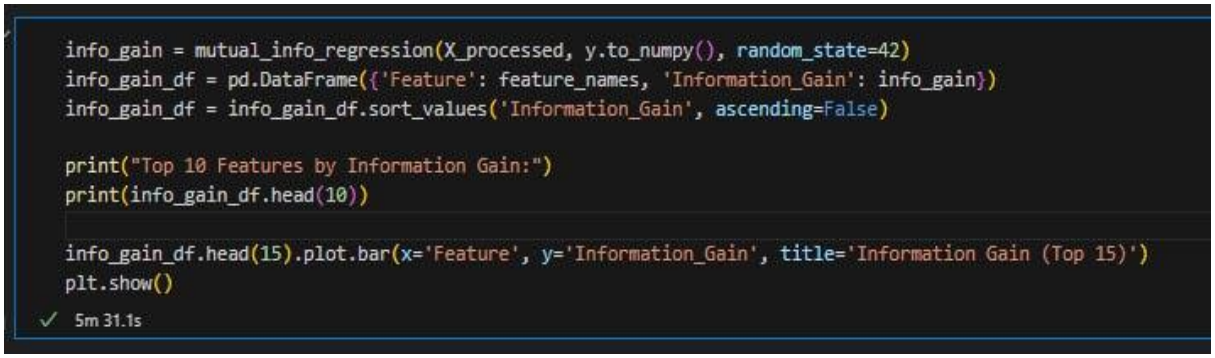


Рис.3.18 - Оцінка важливості ознак методом Information Gain (взаємна інформація)

- **Chi-square test** (хі-квадрат тест). Окрім взаємної інформації, для оцінки інформативності ознак було застосовано χ^2-критерій — класичний статистичний тест, що дозволяє визначити силу зв'язку між категоріальними ознаками та цільовою змінною. Оскільки price є числовою змінною, її попередньо було бінаризовано (q=4) — розбито на 4 рівновеликі інтервали. Після цього дані трансформуються через попередньо налаштований ColumnTransformer, у якому:

a. ознаки з високою кардинальністю кодуються через **Target Encoding**,

b. ознаки з низькою — через **One-Hot Encoding**.

Далі виконується обчислення χ^2-статистики для кожної ознаки. Отримані оцінки сортуються і виводяться, включно з Top 10 найінформативніших ознак. Для наочності побудовано гістограму Top-15:

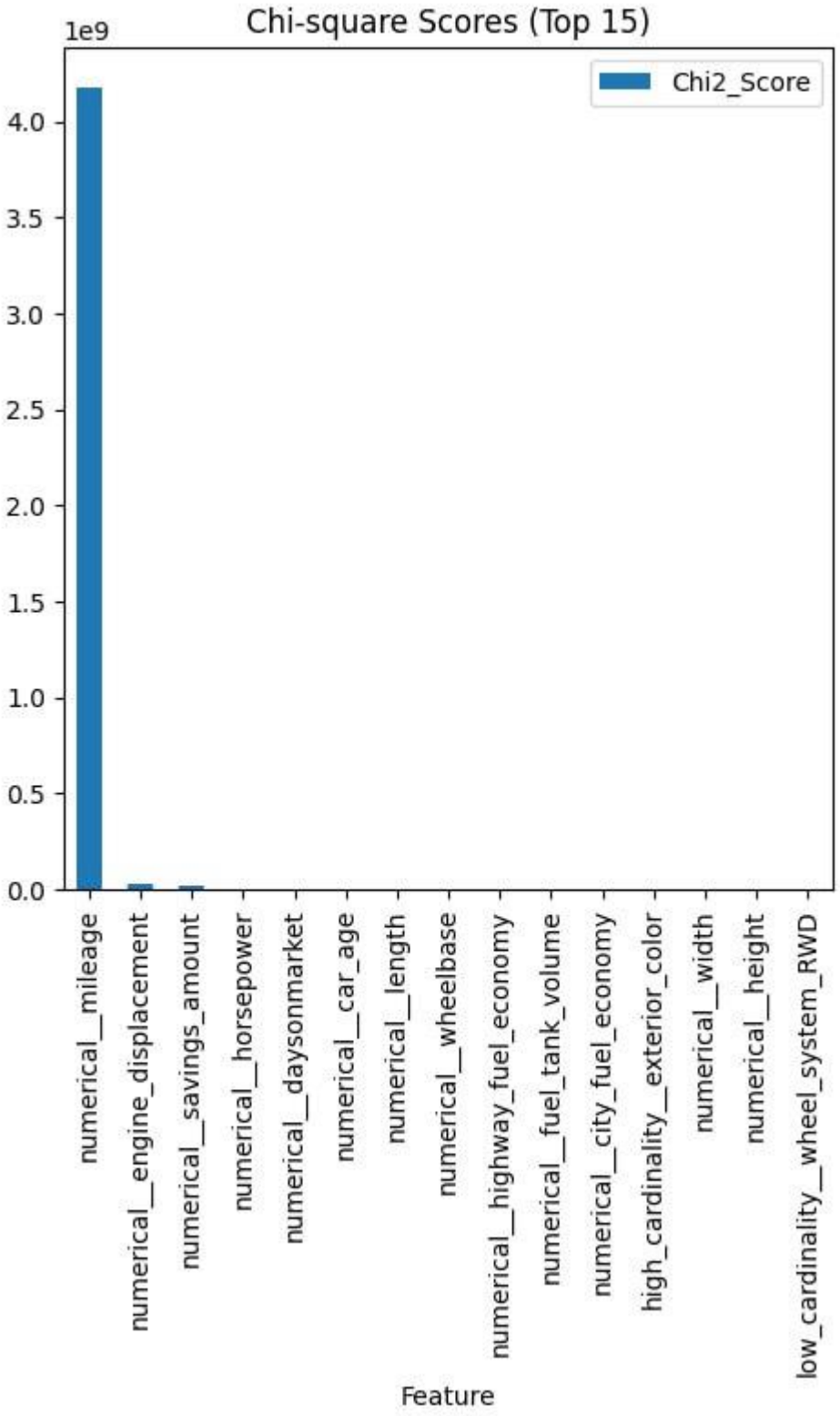
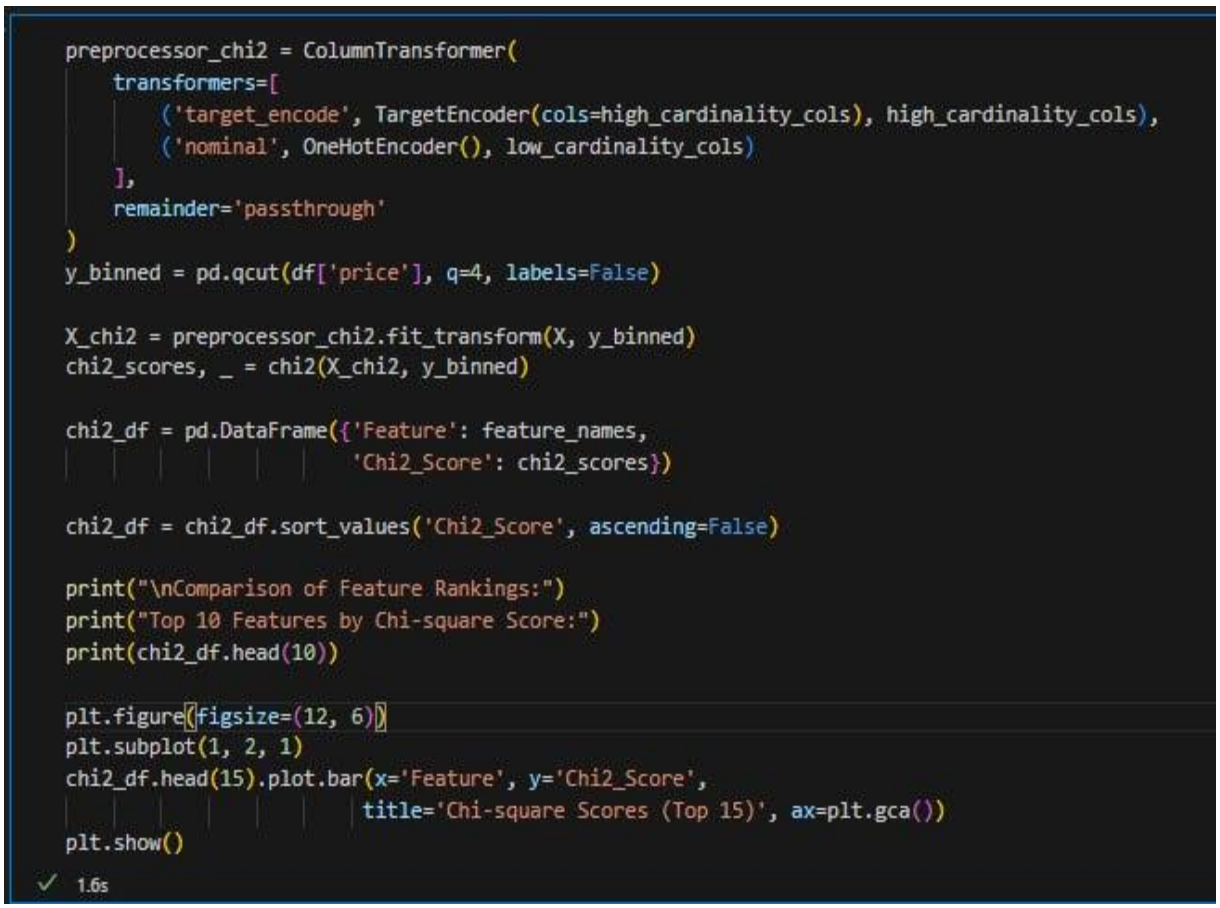


Рис.3.19 - Top-15 найінформативніших ознак

Цей метод дозволяє порівняти результати із іншими метриками важливості ознак (наприклад, Information Gain), що допомагає прийняти зважене рішення щодо скорочення розмірності.



```
preprocessor_chi2 = ColumnTransformer(
    transformers=[
        ('target_encode', TargetEncoder(cols=high_cardinality_cols), high_cardinality_cols),
        ('nominal', OneHotEncoder(), low_cardinality_cols)
    ],
    remainder='passthrough'
)
y_binned = pd.qcut(df['price'], q=4, labels=False)

X_chi2 = preprocessor_chi2.fit_transform(X, y_binned)
chi2_scores, _ = chi2(X_chi2, y_binned)

chi2_df = pd.DataFrame({'Feature': feature_names,
                        'Chi2_Score': chi2_scores})

chi2_df = chi2_df.sort_values('Chi2_Score', ascending=False)

print("\nComparison of Feature Rankings:")
print("Top 10 Features by Chi-square Score:")
print(chi2_df.head(10))

plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
chi2_df.head(15).plot.bar(x='Feature', y='Chi2_Score',
                          title='Chi-square Scores (Top 15)', ax=plt.gca())
plt.show()
✓ 1.6s
```

Рис.3.20 - фрагмент коду **chi-square test**

- **F-score** (ANOVA F-тест) - базується на аналізі дисперсії. Для доповнення аналізу ознак та перевірки їх значущості в контексті лінійної залежності від цільової змінної (price) було застосовано F-критерій через функцію f_regression. F-критерій оцінює силу зв’язку між кожною ознакою та цільовою змінною, спираючись на співвідношення між дисперсією між групами та всередині груп. Він є основою для ANOVA-аналізу. Основні кроки:
  - Обчислення F-оцінок для кожної ознаки
  - Створення DataFrame з результатами та сортування у порядку спадання
  - Виведення Top-10 ознак і побудова гістограми для Top-15

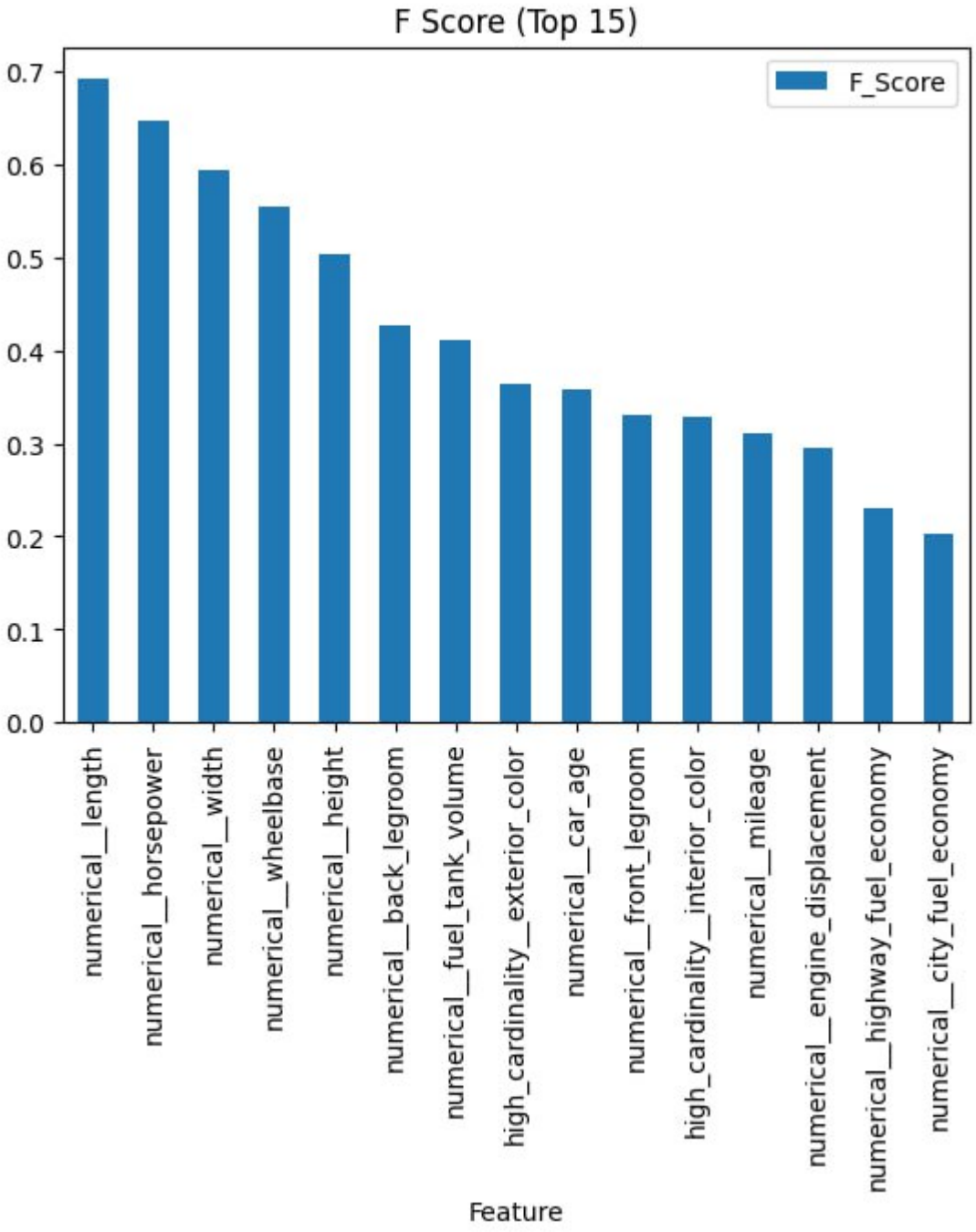


Рис.3.21 - Гістограма для Top-15 **F-score**

Цей метод особливо ефективний при роботі з числовими ознаками й використовується для відбору ознак у лінійних моделях та у методах з припущенням про нормальний розподіл.

```
f_scores, _ = f_regression(X_processed, y)
f_scores_df = pd.DataFrame({'Feature': feature_names, 'F_Score': info_gain})
f_scores_df = f_scores_df.sort_values('F_Score', ascending=False)

print("Top 10 Features by F Score:")
print(f_scores_df.head(10))

# Plot
f_scores_df.head(15).plot.bar(x='Feature', y='F_Score', title='F Score (Top 15)')
plt.show()
✓ 0.2s
```

Рис.3.22 - Метод оцінки важливості ознак **F-критерій регресії**

Для порівняння було побудовано таблицю, що містить нормалізовані значення всіх трьох метрик та середню оцінку важливості кожної ознаки. Це допомогло виділити ключові фактори, що найбільше впливають на ціну автомобіля.

Для підвищення обґрунтованості вибору ознак було здійснено порівняння трьох незалежних оцінок інформативності ознак:

- **Information Gain** — міра взаємної інформації між ознакою та ціною;
- **Chi² Score** — тест залежності між категоріальною ознакою та ціною (після бінінгу);
- **F-Score** — результат однофакторного дисперсійного аналізу.

```
comparison_df = pd.DataFrame({
    'Feature': feature_names,
    'Info_Gain': info_gain,
    'Chi2_Score': chi2_scores,
    'F_Score': f_scores
})

comparison_df['Info_Gain_Norm'] = (comparison_df['Info_Gain'] - comparison_df['Info_Gain'].min()) / (comparison_df['Info_Gain'].max() - comparison_df['Info_Gain'].min())
comparison_df['Chi2_Score_Norm'] = (comparison_df['Chi2_Score'] - comparison_df['Chi2_Score'].min()) / (comparison_df['Chi2_Score'].max() - comparison_df['Chi2_Score'].min())
comparison_df['F_Score_Norm'] = (comparison_df['F_Score'] - comparison_df['F_Score'].min()) / (comparison_df['F_Score'].max() - comparison_df['F_Score'].min())

comparison_df['Avg_Score'] = comparison_df[['Info_Gain_Norm', 'Chi2_Score_Norm', 'F_Score_Norm']].mean(axis=1)
comparison_df = comparison_df.sort_values('Avg_Score', ascending=False)

styled_table = comparison_df.head(15).style.format({
    'Info_Gain': '{:.4f}',
    'Chi2_Score': '{:.2f}',
    'F_Score': '{:.4f}',
    'Info_Gain_Norm': '{:.3f}',
    'Chi2_Score_Norm': '{:.3f}',
    'F_Score_Norm': '{:.3f}',
    'Avg_Score': '{:.3f}'
}).background_gradient(subset=['Avg_Score'], cmap='YlGn')

display(styled_table)
✓ 0.0s
```

Рис.3.23 - Порівняльний аналіз важливості ознак

Для кожної ознаки обчислено середній нормалізований рейтинг (Avg_Score). Ознаки відсортовано за середнім рейтингом, а результати представлено у вигляді таблиці (табл.3.24) з кольоровою підсвіткою значень.

	Feature	Info_Gain	Chi2_Score	F_Score	Info_Gain_Norm	Chi2_Score_Norm	F_Score_Norm	Avg_Score
60	numerical__mileage	0.3118	4176999557.29	130080.1895	0.451	1.000	0.662	0.704
55	numerical__horsepower	0.6476	3098453.09	196612.3048	0.937	0.001	1.000	0.646
58	numerical__length	0.6915	72614.52	57574.2749	1.000	0.000	0.293	0.431
67	numerical__width	0.5933	33099.67	66839.3409	0.858	0.000	0.340	0.399
66	numerical__wheelbase	0.5544	65650.21	66505.1899	0.802	0.000	0.338	0.380
41	high_cardinality__exterior_color	0.3638	34471.10	116491.3748	0.526	0.000	0.592	0.373
68	numerical__car_age	0.3570	161834.17	110071.5421	0.516	0.000	0.560	0.359
42	high_cardinality__interior_color	0.3287	3985.60	98014.4802	0.475	0.000	0.499	0.325
53	numerical__height	0.5029	28760.16	40447.9766	0.727	0.000	0.206	0.311
51	numerical__fuel_tank_volume	0.4115	61664.35	65720.7252	0.595	0.000	0.334	0.310
46	numerical__engine_displacement	0.2958	24876117.56	58091.3027	0.428	0.006	0.295	0.243
43	numerical__back_legroom	0.4271	2580.68	12748.4110	0.618	0.000	0.065	0.228
39	high_cardinality__engine_cylinders	0.1758	4112.55	77986.6084	0.254	0.000	0.397	0.217
40	high_cardinality__engine_type	0.1748	5877.83	77986.6084	0.253	0.000	0.397	0.217
37	low_cardinality__wheel_system_FWD	0.1420	1348.23	74876.0814	0.205	0.000	0.381	0.195

Рис.3.24 - Таблиця, що містить нормалізовані значення всіх трьох метрик та середню оцінку важливості кожної ознаки.

Для побудови моделі було відібрано найбільш інформативні змінні(15 ознак з найбільшим середнім нормалізованим рейтингом).

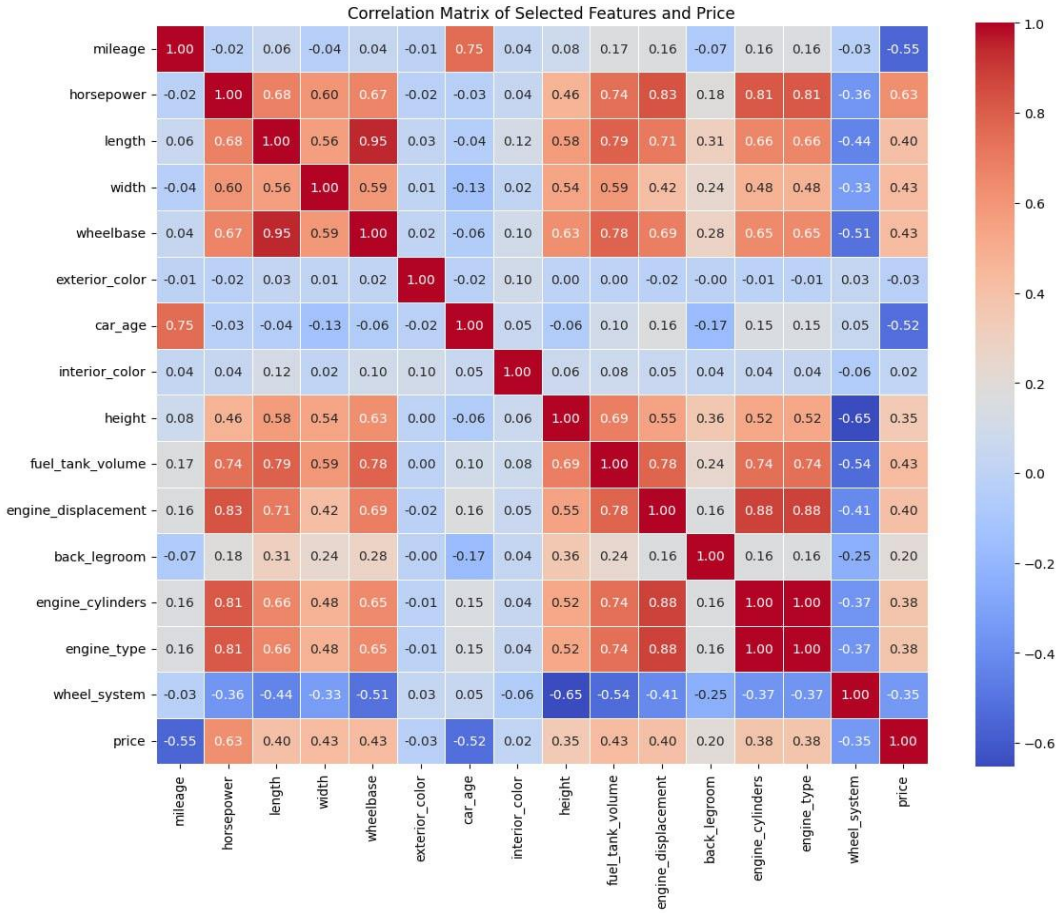


Рис.3.25 – Матриця кореляції 15 обраних ознак та ціни автомобіля

Дані було поділено у співвідношенні 80:20. Цільову змінну price було масштабовано через StandardScaler для стабільнішого навчання моделей (особливо нейронних мереж). Було ідентифіковано числові, категоріальні та логічні змінні, а також розділено категоріальні за кардинальністю (≤15 або >15 унікальних значень) для подальшого відповідного кодування. Застосовано ColumnTransformer. У результаті отримано оброблені навчальні та тестові матриці ознак, готові для подачі до моделей машинного навчання.

```
selected_columns = ['mileage', 'horsepower', 'length', 'width', 'wheelbase', 'exterior_color', 'car_age',
'interior_color', 'height', 'fuel_tank_volume', 'engine_displacement',
'back_legroom', 'engine_cylinders', 'engine_type', 'wheel_system']

X = df[selected_columns]
y = df['price'].values
X_train, X_test, y_train, y_test = train_test_split(
    X,  y, test_size=0.2, random_state=42
)
y_train_scaled = scaler_y.fit_transform(y_train.reshape(-1, 1)).ravel()
y_test_scaled = scaler_y.transform(y_test.reshape(-1, 1)).ravel()
numerical_columns = X.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_columns = X.select_dtypes(include=['object', 'category']).columns.tolist()
low_cardinality_cols = []
high_cardinality_cols = []

for col in categorical_columns:
    if X[col].nunique() <= 15:
        low_cardinality_cols.append(col)
    else:
        high_cardinality_cols.append(col)
preprocessor = ColumnTransformer(
    transformers=[
        ('low_cardinality', OneHotEncoder(), low_cardinality_cols),
        ('high_cardinality', TargetEncoder(cols=high_cardinality_cols), high_cardinality_cols),
        ('numerical', StandardScaler(), numerical_columns + bool_columns)
    ])
preprocessor.fit(X_train, y_train_scaled)
X_train_processed = preprocessor.transform(X_train)
X_test_processed = preprocessor.transform(X_test)
n_features = X_train_processed.shape[1]
✓ 1.6s
```

Рис.3.26 - Вибір ознак, поділ на навчальну й тестову вибірки, масштабування цільової змінної та трансформація вхідних даних.

Для покращення здатності моделі виявляти найбільш важливі ознаки у процесі прогнозування було реалізовано власний attention-шар FeatureAttention, заснований на нейронному ваговому механізмі. Цей шар адаптується під розмірність вхідних ознак та автоматично навчається під час оптимізації моделі. Коли шар отримує дані (наприклад: пробіг, потужність, вік авто), він обчислює “важливість” кожної ознаки — як сильно ця ознака впливає на результат.

Важливі ознаки він залишає майже без змін (множить на число близьке до 1), а менш важливі — приглушує (множить на число ближче до 0).

```python
@register_keras_serializable(package='CustomLayers')
class FeatureAttention(Layer):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)

    def build(self, input_shape):
        self.attention_weights = self.add_weight(
            name='attention',
            shape=(input_shape[-1], input_shape[-1]),
            initializer='glorot_uniform',
            trainable=True
        )
        self.attention_bias = self.add_weight(
            name='attention_bias',
            shape=(input_shape[-1],),
            initializer='zeros',
            trainable=True
        )
        super().build(input_shape)

    def call(self, inputs):
        scores = tf.nn.sigmoid(
            tf.matmul(inputs, self.attention_weights) + self.attention_bias
        )
        return inputs * scores

    def get_config(self):
        config = super().get_config()
        return config

    @classmethod
    def from_config(cls, config):
        return cls(**config)
```

✓ 0.0s

Рис.3.27 – Побудова механізму уваги для табличних даних

3.3 Модель нейронної мережі з механізмом уваги (attention)

Було побудовано нейронну мережу на основі TensorFlow Keras. Фрагмент коду представляє **архітектуру нейронної мережі з вбудованими attention-шарами**, спеціально адаптовану для задачі регресії цін на основі табличних ознак.

```python
def build_model(input_dim):
    inputs = Input(shape=(input_dim,), name='input_layer')
    x = FeatureAttention(name='attention_1')(inputs)
    x = Dense(128, activation='relu', kernel_regularizer=L2(0.001), name='dense_6')(x)
    x = FeatureAttention(name='attention_2')(x)
    x = Dropout(0.3, name='dropout_4')(x)
    x = Dense(64, activation='relu', kernel_regularizer=L2(0.001), name='dense_7')(x)
    x = FeatureAttention(name='attention_3')(x)
    x = Dropout(0.2, name='dropout_5')(x)
    x = Dense(32, activation='relu', kernel_regularizer=L2(0.001), name='dense_8')(x)
    outputs = Dense(1, name='output_layer')(x)
    return Model(inputs=inputs, outputs=outputs)
✓ 0.0s
```

Рис.3.28 - Архітектура нейронної мережі з вбудованими attention-шарами

Основні компоненти архітектури:

1. Вхідний шар (Input), розмірність якого визначається кількістю ознак після попередньої обробки (n-features).
2. Три блоки з механізмом уваги (FeatureAttention), які вставляються після кожного щільного шару. Вони адаптивно зважують вихідні ознаки, підсилюючи важливіші.
3. Три щільні шари (Dense), з кількістю нейронів 128, 64 та 32 відповідно. Всі використовують функцію активації ReLU та L2-регуляризацію для зменшення перенавчання.
4. Регуляризаційні шари типу Dropout з ймовірністю 0.3, 0.2, що сприяють зниженню переадаптації моделі.
5. Вихідний шар — один нейрон без активації (оскільки це задача регресії).

Переваги архітектури:

- Вбудовані механізми уваги покращують здатність мережі виділяти найважливіші ознаки.
- Комбінація Dropout і L2-регуляризації запобігає перенавчанню.
- Модель легко розширюється або адаптується до нових ознак.
    - Для оптимізації тренування застосовано:
- Adam-оптимізатор зі швидкістю навчання 0.001;
- Механізм ранньої зупинки (EarlyStopping) — припинення навчання при відсутності покращення протягом 10 епох;

- Механізм зниження швидкості навчання (ReduceLROnPlateau);
- ModelCheckpoint (збереження найкращої моделі).

```
model_full = build_model(X_train_processed.shape[1])
model_full.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                   loss='mse', metrics=['mae'])

callbacks = [
    EarlyStopping(monitor='val_mae', patience=10, restore_best_weights=True, verbose=1),
    ReduceLROnPlateau(monitor='val_mae', factor=0.5, patience=5, min_lr=1e-6, verbose=1),
    ModelCheckpoint('price_predictor_model_full.keras', monitor='val_mae', save_best_only=True)
]

history = model_full.fit(
    X_train_processed, y_train_scaled,
    validation_split=0.2, epochs=200, batch_size=32,
    callbacks=callbacks, verbose=1
)

with open('training_history_full.json', 'w') as f:
    json.dump({k: np.array(v).tolist() for k,v in history.history.items()}, f)
✓ 21m 55.7s
```

Рис.3.29 - повний цикл компіляції та навчання глибокої нейронної мережі для регресійної задачі прогнозування ціни автомобіля

Модель навчалась на навчальній вибірці з використанням 20% для валідації, протягом максимум 200 епох із розміром пакету навчальних прикладів 32. Валідаційна MAE використовувалась для оцінки прогресу. Після завершення тренування історія втрат та метрик зберігається у форматі JSON. Це дозволяє згодом візуалізувати процес навчання або використати метрики для порівняння моделей.

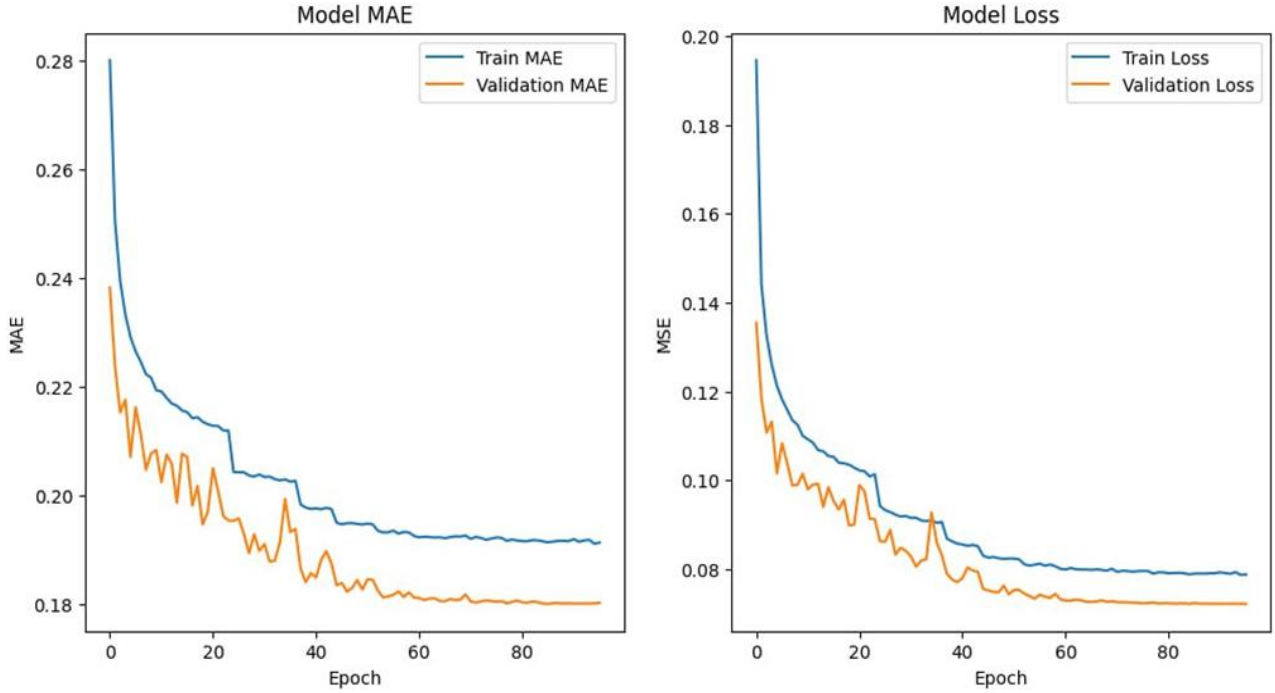


Рис.3.30 - Графіки динаміки метрик тренування (MAE, MSE) на кожній епосі

Ліва частина — Model MAE (Mean Absolute Error)

Права частина — Model Loss (Mean Squared Error, MSE)

Синя лінія — метрики тренувань на тренувальній вибірці.

Помаранчева лінія — метрики тренувань на валідаційній вибірці.

3.4 Гібридні моделі з використанням XGBoost та Random Forest

У цьому варіанті експерименту модель навчається лише на числових і логічних ознаках, тоді як категоріальні ознаки спеціально обробляються, щоб пізніше використовувати їх у традиційних ансамблевих моделях, таких як XGBoost та Random Forest.

Було використано ту ж архітектуру (build_model(...)), оптимізатор Adam, функцію втрат mse, метрику mae і набір функцій керування процесом навчання (EarlyStopping, ReduceLROnPlateau, ModelCheckpoint). Навчання виконувалося на: 200 епохах; з валідаційним розділенням 20%; batch size = 32.

```
scaler_num = StandardScaler()
X_train_num = scaler_num.fit_transform(X_train[numerical_columns + bool_columns])
X_test_num = scaler_num.transform(X_test[numerical_columns + bool_columns])

target_enc = TargetEncoder()
X_train_cat = target_enc.fit_transform(X_train[categorical_columns], y_train)
X_test_cat = target_enc.transform(X_test[categorical_columns])

scaler_y = StandardScaler()
y_train_scaled = scaler_y.fit_transform(y_train.reshape(-1, 1)).ravel()

model_embed = build_model(X_train_num.shape[1])
model_embed.compile(optimizer=Adam(0.001), loss='mse', metrics=['mae'])

callbacks_embed = [
    EarlyStopping(monitor='val_mae', patience=10, restore_best_weights=True),
    ReduceLROnPlateau(monitor='val_mae', factor=0.5, patience=5, min_lr=1e-6),
    ModelCheckpoint('price_predictor_model_embed.keras', monitor='val_mae', save_best_only=True)
]

history_embed = model_embed.fit(
    X_train_num, y_train_scaled,
    validation_split=0.2, epochs=200, batch_size=32,
    callbacks=callbacks_embed, verbose=1
)

with open('training_history_embed.json', 'w') as f:
    json.dump({k: np.array(v).tolist() for k, v in history_embed.history.items()}, f)
✓ 20m 58.9s
```

Рис.3.31 - Альтернативний сценарій підготовки та навчання моделі

З метою побудови гібридної моделі, де нейронна мережа використовується як предобробник для генерації векторних ознак (feature embeddings), було реалізовано проміжну модель (intermediate_model), яка завершується на одному з останніх шарів (dense_8) архітектури.

```
intermediate_model = Model(inputs=model_embed.input, outputs=model_embed.get_layer('dense_8').output)
✓ 0.0s
```

Рис.3.32 – Проміжна модель

```
X_train_embeddings = intermediate_model.predict(X_train_num, batch_size=32)
X_test_embeddings = intermediate_model.predict(X_test_num, batch_size=32)
✓ 10.2s
```

Рис.3.33 – Генерація векторних ознак

З метою поєднання переваг методів глибокого навчання та класичних ансамблевих підходів було реалізовано гібридну модель, яка використовує нейронну мережу для формування узагальнених представлень числових ознак,

що надалі передаються до алгоритму Random Forest для остаточного прогнозування вартості.

```
target_enc = TargetEncoder()
X_train_cat = target_enc.fit_transform(X_train[categorical_columns], y_train)
X_test_cat = target_enc.transform(X_test[categorical_columns])

X_train_hybrid = np.hstack([X_train_embeddings, X_train_cat])
X_test_hybrid = np.hstack([X_test_embeddings, X_test_cat])

rf_model = RandomForestRegressor(n_estimators=300, random_state=42, n_jobs=-1)
rf_model.fit(X_train_hybrid, y_train)
y_pred_rf = rf_model.predict(X_test_hybrid)

r2_rf, mape_rf, within10_rf, mae_rf, rmse_rf = compute_metrics(y_test, y_pred_rf)

print("\n=== Hybrid Model: NN + RF ===")
print(f"Test R²: {r2:.3f}")
print(f"Test MAPE: {mape:.2f}%")
print(f"Predictions within 10% error: {within10:.1f}%")
print(f"Test MAE (Dollars): ${mae:.2f}")
print(f"Test RMSE (Dollars): ${rmse:.2f}")
✓ 3m 32.4s
```

Рис.3.34 – Гібридна модель з використанням Random Forest

У межах гібридного підходу була реалізована модель XGBoost, яка навчається не на початкових (необроблених) даних, а на векторизованих представленнях ознак, сформованих попередньо навченою нейронною мережею. Це забезпечує моделі доступ до узагальненого представлення числових характеристик об’єкта. Підбір параметрів моделі здійснювався з використанням методу випадкового пошуку (RandomizedSearchCV).

```python
param_dist = {
    'n_estimators': [100, 300, 500, 700],
    'max_depth': [3, 5, 6, 8],
    'learning_rate': [0.01, 0.05, 0.1],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0],
    'reg_alpha': [0, 0.1, 0.5],
    'reg_lambda': [1, 1.5, 2]
}

xgb_regressor = xgb.XGBRegressor(random_state=42, n_jobs=-1)

random_search = RandomizedSearchCV(
    estimator=xgb_regressor,
    param_distributions=param_dist,
    n_iter=30,
    scoring='neg_mean_absolute_error',
    cv=3,
    verbose=2,
    random_state=42
)
random_search.fit(X_train_embeddings, y_train)

print("✅ Найкращі параметри:")
print(random_search.best_params_)

best_xgb_model = random_search.best_estimator_

y_pred_xgb_tuned = best_xgb_model.predict(X_test_embeddings)
```

Рис.3.35 – Підбір гіперпараметрів

Для підвищення якості XGBoost-моделі, яка працює на основі ембедингів, було проведено автоматичний підбір параметрів з використанням **RandomizedSearchCV**. Параметри варіювались у 30 випадкових комбінаціях, з оцінкою на основі середньої абсолютної похибки (neg_mean_absolute_error) у 3-кратній крос-валідації.

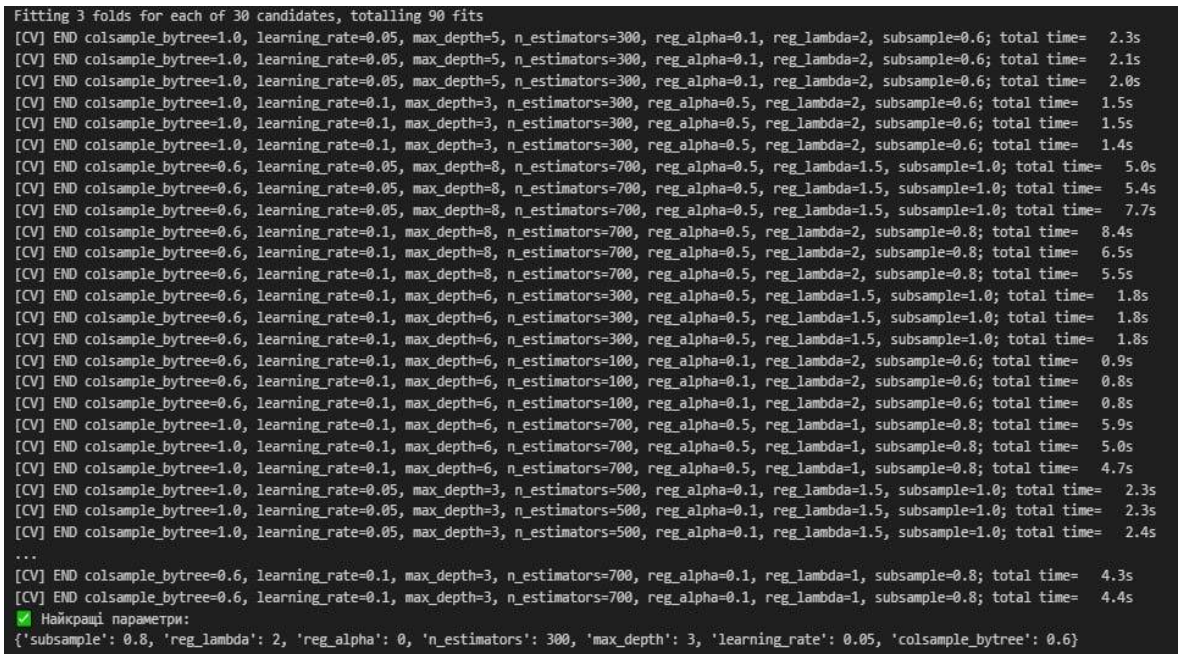


Рис.3.36 – Результат автоматичного підбору гіперпараметрів для XGBoost-моделі

Після визначення оптимальних гіперпараметрів було здійснено остаточне тренування XGBoost-регресора на гібридному наборі ознак, який об'єднує:

- Ембединги, витягнуті з попередньо навченої нейронної мережі;
- Категоріальні ознаки, перетворені за допомогою TargetEncoder.

Цей гібридний підхід дозволяє XGBoost-алгоритму векторні представлення числових ознак, сформовані за допомогою нейронної мережі, а також ефективно опрацювати категоріальні змінні без надмірного бінарного кодування, що значно покращує якість регресійної моделі.

```
X_train_cat = target_enc.fit_transform(X_train[categorical_columns], y_train)
X_test_cat = target_enc.transform(X_test[categorical_columns])

X_train_hybrid = np.hstack([X_train_embeddings, X_train_cat])
X_test_hybrid = np.hstack([X_test_embeddings, X_test_cat])

xgb_model = xgb.XGBRegressor(
    n_estimators=300, learning_rate=0.05, max_depth=3, reg_lambda = 2,
    subsample=0.8, colsample_bytree=0.6, random_state=42
)
xgb_model.fit(X_train_hybrid, y_train)

y_pred_xgb = xgb_model.predict(X_test_hybrid)
✓ 3.6s
```

Рис.3.37 - фінальне навчання XGBoost-моделі з використанням найкращих гіперпараметрів

3.5 Оцінка моделей

Для кількісного порівняння моделей було реалізовано єдину функцію compute_metrics(...), яка обчислює п’ять ключових регресійних метрик:

- **MAE** (Mean Absolute Error);
- **RMSE** (Root Mean Squared Error);
- **R^2** (коефіцієнт детермінації);
- **MAPE** (Mean Absolute Percentage Error);
- Частку прогнозів, що потрапили у 10% відхилення від реальної ціни.

```python
def compute_metrics(y_true, y_pred):
    y_true = np.array(y_true).flatten()
    y_pred = np.array(y_pred).flatten()

    mae = mean_absolute_error(y_true, y_pred)
    mse = np.mean((y_true - y_pred) ** 2)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_true, y_pred)
    mape = mean_absolute_percentage_error(y_true, y_pred) * 100
    error_percentage = np.abs((y_pred - y_true) / y_true) * 100
    within_10 = np.mean(error_percentage <= 10) * 100
    return r2, mape, within_10, mae, rmse

✓ 0.0s
```

Рис.3.38 – Функція для обчислення регресійних метрик

Було проведено підсумкову оцінку двох моделей — звичайної нейронної мережі (NN) та гібридної моделі (нейромережа + XGBoost), використовуючи однакові метрики на не масштабованих цінах.

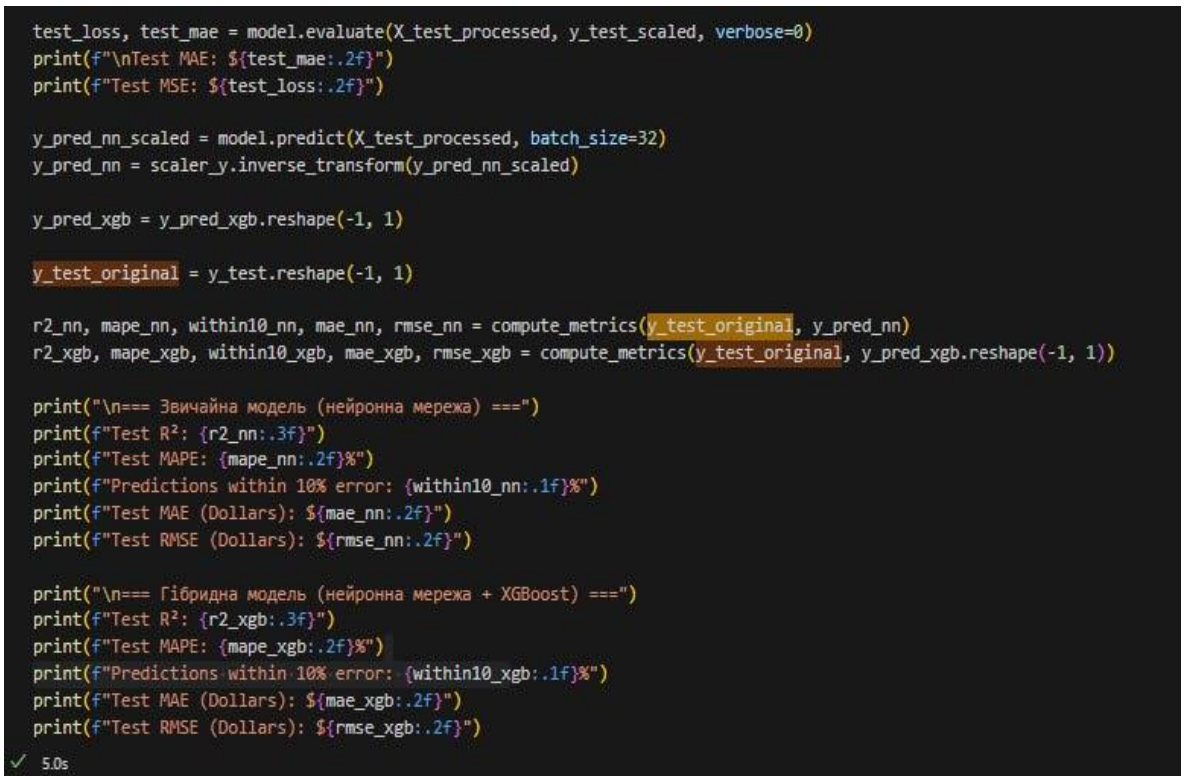


Рис.3.39 – Підсумкова оцінка двох моделей

З метою об'єктивного порівняння були навчені моделі: CatBoostRegressor, LightGBM (LGBMRegressor), "базова" XGBoost-модель, LinearRegression та "базова" Random Forest Regressor-модель. Після чого за допомогою функції compute_metrics(...) було обчислено п’ять ключових регресійних метрик для кожної моделі. Ці дані були використані для побудови підсумкової порівняльної таблиці моделей.

```python
results = pd.DataFrame({
    'Model': [
        'Neural Network', 'Hybrid (NN + XGBoost)', 'Hybrid (NN + RandomForest)', 'CatBoost',
        'LightGBM', 'RandomForest', 'XGBoost (plain)', 'Linear Regression'
    ],
    'R2': [
        r2_nn, r2_xgb, r2_rf, r2_cat, r2_lgb, r2_rf_plain,
        r2_xgb_plain, r2_linreg
    ],
    'MAPE (%)': [
        mape_nn, mape_xgb, mape_rf, mape_cat, mape_lgb, mape_rf_plain,
        mape_xgb_plain, mape_linreg
    ],
    'Within 10% (%)': [
        within10_nn, within10_xgb, within10_rf, within10_cat, within10_lgb, within10_rf_plain,
        within10_xgb_plain, within10_linreg
    ],
    'MAE ($)': [
        mae_nn, mae_xgb, mae_rf, mae_cat, mae_lgb, mae_rf_plain,
        mae_xgb_plain, mae_linreg
    ],
    'RMSE ($)': [
        rmse_nn, rmse_xgb, rmse_rf, rmse_cat, rmse_lgb, rmse_rf_plain,
        rmse_xgb_plain, rmse_linreg
    ]
})


print("\n=== Model Comparison Results ===")
display(results.style.format({
    'R2': '{:.3f}',
    'MAPE (%)': '{:.2f}%',
    'Within 10% (%)': '{:.2f}%',
    'MAE ($)': '${:,.2f}',
    'RMSE ($)': '${:,.2f}'
}))
```

✓ 0.0s

Рис.3.40 – Побудова порівняльної таблиці моделей

	Model	R2	MAPE (%)	Within 10% (%)	MAE ($)	RMSE ($)
0	Neural Network	0.935	10.73%	60.24%	$2,013.24	$2,829.62
1	Hybrid (NN + XGBoost)	0.936	10.61%	60.54%	$1,992.16	$2,796.54
2	Hybrid (NN + RandomForest)	0.934	10.79%	59.88%	$2,028.16	$2,841.92
3	CatBoost	0.950	9.24%	65.85%	$1,769.72	$2,479.08
4	LightGBM	0.933	10.85%	59.40%	$2,048.46	$2,858.08
5	RandomForest	0.947	9.23%	66.29%	$1,775.67	$2,540.15
6	XGBoost (plain)	0.899	12.94%	51.51%	$2,483.27	$3,508.14
7	Linear Regression	0.774	21.77%	35.65%	$3,824.36	$5,258.95

Рис.3.41 – Порівняльна таблиця моделей

3.6 Візуалізація та результати

Було побудовано стовпчасті діаграми порівняння моделей для кожної з метрик.

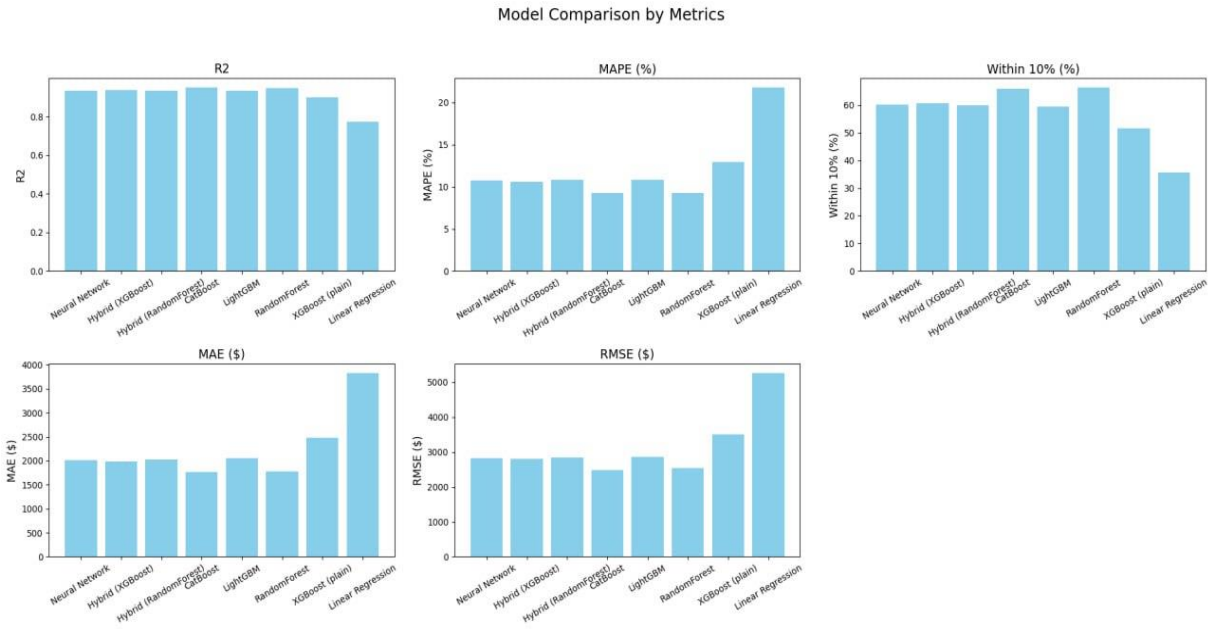


Рис.3.42 – Зведена візуалізація всіх ключових метрик для порівняння моделей

Для задачі оцінки ціни вживаного авто на основі табличних даних найкраще себе зарекомендував CatBoost, який поєднує високу точність, мінімальні помилки та простоту використання з категоріальними змінними. Проте гібридні моделі з нейронною мережею також показують високу ефективність і можуть використовуватись у більш складних ансамблевих підходах.

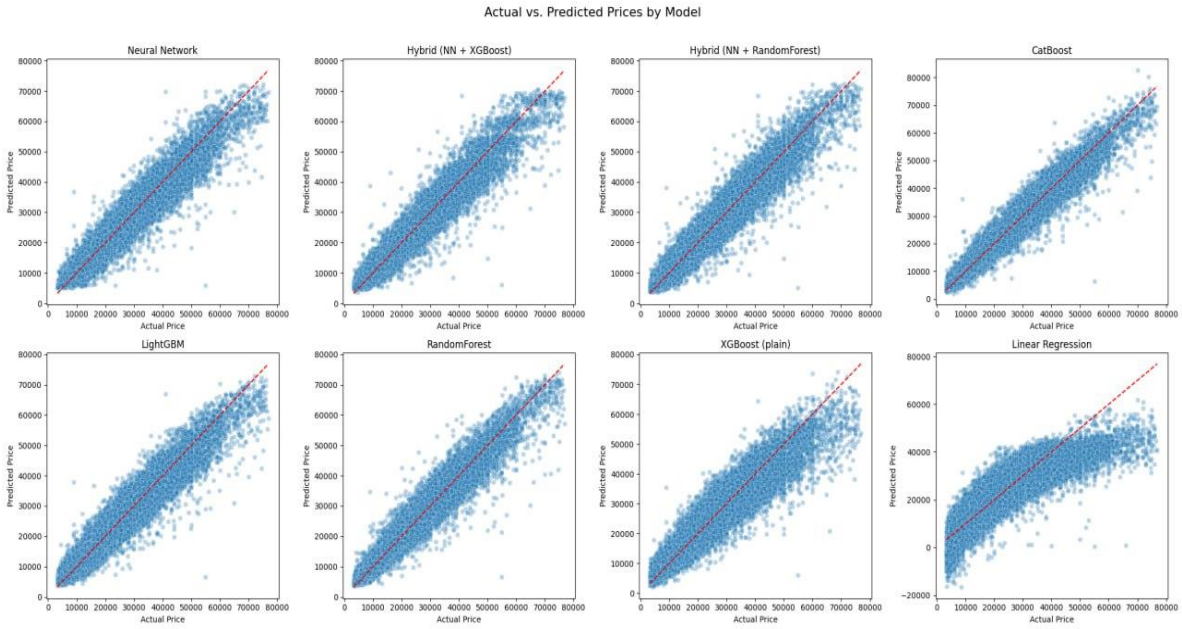


Рис.3.43 – Графіки точності моделі

Було побудовано візуалізацію, що показує наскільки кожна модель наближається до ідеального передбачення. На графіках точність моделі ілюструється тим, наскільки близько сині точки розташовані до червоної лінії, яка відображає ідеальну відповідність між фактичною та передбаченою ціною.

Висновок

У процесі побудови та тестування різних моделей прогнозування цін на вживані автомобілі було досліджено як класичні методи машинного навчання (RandomForest, XGBoost, Linear Regression), так і сучасні підходи на основі нейронних мереж та їх гібридні комбінації.

Для задачі оцінки ціни вживаного авто на основі табличних даних найкраще себе зарекомендував CatBoost, який поєднує високу точність (R^2 = 0.95), мінімальні похибки (MAPE = 9.24 %, MAE = $1769.72, RMSE = $2479.08) та зручну роботу з категоріальними змінними без необхідності додаткового кодування.

Також гібридна модель, що поєднує нейронну мережу з градієнтним бустінгом (XGBoost або RandomForest), продемонструвала високу ефективність

(NN + XGBoost: $R^2 = 0.936$, MAPE = 10.61 %, MAE = $1992.16, RMSE = $2796.54). Вона має потенціал для використання в складніших ансамблевих системах або у випадках, коли потрібна адаптація до нелінійних ознак високого рівня.

Своєю чергою, RandomForest відзначився чудовими результатами на класичних табличних ознаках ($R^2 = 0.947$, MAPE = 9.23 %, MAE = $1775.67, RMSE = $2540.15). Проте він виявився менш ефективним у поєднанні з векторними представленнями ознак, отриманими з глибокої нейронної мережі (XGBoost+ RandomForest: $R^2 = 0.934$, MAPE = 10.79 %, MAE = $2028.16, RMSE = $2841.92, що є гіршим за звичайну NN: $R^2 = 0.935$, MAPE = 10.73 %, MAE = $2013.24, RMSE = $2829.62).

Також гібридні моделі, що поєднують нейронну мережу з ансамблевими методами (XGBoost або RandomForest), продемонстрували високу ефективність. Вони мають потенціал для використання в складніших ансамблевих системах або у випадках, коли потрібна адаптація до нелінійних ознак високого рівня.

Своєю чергою, RandomForest відзначився чудовими результатами на класичних табличних ознаках (числових та категоріальних), але не є оптимальним для роботи з векторними представленнями ознак, отриманими з глибокої нейронної мережі.

РОЗДІЛ 4

4.1 Концепція впровадження моделі прогнозування цін на автотранспортні засоби

У цьому розділі розглянемо, як розроблену модель нейронної мережі з механізмом уваги можна інтегрувати в реальне бізнес-середовище. Впровадження моделі передбачає як технічні, так і організаційні етапи, а також взаємодію з кінцевими користувачами — такими як автосалони, маркетплейси, страхові компанії та фінансові установи.

1. Мета впровадження

Метою є автоматизований аналіз великої кількості історичних і поточних даних про автомобілі з подальшим прогнозуванням їх ринкової вартості. Це дозволяє:

- Скоротити час оцінки вартості авто.
- Підвищити точність визначення справедливої ціни.
- Оптимізувати пропозиції в онлайн-магазинах авто.
- Допомогти банкам і страховим компаніям у прийнятті обґрунтованих рішень.

2. Цільова аудиторія

- Автосалони та автодилери: для швидкої оцінки автомобілів при трейд-іні чи продажу.
- Онлайн-платформи: наприклад, AutoRia, OLX, Copart, які можуть інтегрувати модель у свій сайт для показу рекомендованої ціни.
- Банки та лізингові компанії: для оцінки заставного майна.
- Страхові компанії: для швидкої оцінки виплат за тотальними збитками або втратою вартості авто.
- Аналітичні агентства та маркетологи: для моніторингу цінових трендів і прогнозів.

3. Архітектура інтеграції

3.1. API-доступ до моделі

Модель можна розгорнути у вигляді REST API-сервісу. Користувач або система передає набір характеристик автомобіля у JSON-форматі, а у відповідь отримує прогнозовану ціну.

3.2. Інтеграція з базами даних

Платформа має бути підключена до актуальних баз з характеристиками автомобілів (наприклад, VIN-декодери, сайти продажу, страхові бази). Це забезпечує регулярне оновлення даних.

3.3. Обробка великих обсягів даних

Враховуючи обсяг даних (понад 3 млн записів), важливо забезпечити використання хмарних платформ (наприклад, AWS, Google Cloud) або локальних серверів із підтримкою GPU для прискореного навчання/інференсу моделі.

4. Етапи впровадження

1. Аналіз потреб компанії – виявлення цілей і бізнес-вимог.
2. Тестова інтеграція моделі – перевірка точності прогнозу в умовах реального середовища.
3. Розгортання на продакшн-сервері – запуск REST API або інтеграція у внутрішню систему.
4. Навчання персоналу – інструкції для аналітиків, менеджерів і технічного персоналу.
5. Моніторинг і оновлення моделі – регулярне оновлення даних і переобучення моделі за потребою.

5. Ризики та шляхи їх зниження

Таблиця 4.1

Потенційний ризик	Можливий вплив	Заходи для зменшення ризику
Низька точність моделі в нових умовах	Помилкові ціни на ринку	Постійне оновлення моделі, збір нових даних
Проблеми з інтеграцією	Витрати часу, затримка впровадження	Розробка API за стандартами REST, Docke
Погане сприйняття користувачами	Ігнорування моделі або недовіра	Пояснення принципів роботи, візуалізація важливих ознак
Конфіденційність даних	Порушення норм законодавства	Використання шифрування, знеособлення даних

6. Прогнозований ефект від впровадження

- Зменшення витрат для фінансових та страхових організацій
- Зниження людського фактору в оцінці ціни.
- Підвищення прозорості й довіри покупців/продавців.

7. Приклади застосування

- Онлайн-магазин пропонує одразу рекомендовану ринкову ціну при публікації оголошення.
- Автосалон автоматично формує пропозицію викупу за базою історичних цін.

Страхова компанія надає оцінку збитків на основі прогнозної моделі замість ручного огляду.

7. Оцінка економічної доцільності впровадження

Для аналізу економічного ефекту використаємо базові фінансові показники:

- ROI (Return on Investment) — рентабельність інвестицій
- TCO (Total Cost of Ownership) — загальна вартість володіння
- Payback Period — період окупності

Припущення:

- Компанія має щоденно оцінювати 1 000 авто вручну
- Один оцінювач витрачає 10 хв на авто
- Середня зарплата оцінювача: 30 000 грн/міс (≈ 200 грн/год)
- Вартість розробки та впровадження моделі: 150 000 грн (разово)
- Обслуговування та хостинг моделі: 8 000 грн/міс
- Модель обслуговує 100% запитів автоматично
- Помилки знижуються на 25%, що зменшує втрати на компенсації

7.1 Розрахунок економії на зарплаті персоналу:

- 1 000 авто × 10 хв = 10 000 хв / 60 = 166,7 годин щодня
- 166,7 год × 30 днів = 5 000 год на місяць
- Вартість праці: 5 000 × 200 грн = 1 000 000 грн/міс

Після автоматизації — зменшення затрат на персонал у 10 разів (залишається лише контроль):

- Нові витрати ≈ 100 000 грн/міс
- Економія: 900 000 грн/міс

7.2 ROI — рентабельність інвестицій

$$ROI = \frac{Доходи - Інвестиції}{Інвестиції} \times 100\% \quad (4.1)$$

$$ROI = \frac{(900000 \times 12) - 150000}{150000} = \frac{10650000}{150000} \approx 7100\ \%$$

7.3 Період окупності

$$Payback = \frac{Інвестиції}{Щомісячна\ економія} \quad (4.2)$$

$$Payback = \frac{150000}{900000} \approx 0{,}17 \text{ місяця} \approx 5 \text{ днів.}$$

Впровадження моделі машинного навчання дає змогу суттєво скоротити витрати на обробку даних та оцінку автомобілів, автоматизуючи понад 90% рутинних задач. Згідно з розрахунками, період окупності становить лише кілька днів, а річна рентабельність перевищує 7000%, що свідчить про надзвичайно високу економічну ефективність рішення.

4.2 Методи розміщення моделі в хмарному середовищі

У наш час, якщо ти створюєш модель машинного навчання, наприклад нейронну мережу з увагою або XGBoost, просто зберегти її на комп'ютері — цього мало. Щоб нею можна було користуватись у будь-який час і з будь-якого місця, потрібно розгорнути її у хмарі. Це означає, що модель працюватиме на сервері, до якого можуть підключатися інші користувачі чи програми. Є кілька способів це зробити:

1. Інфраструктура як сервіс (IaaS)

Це, коли ти сам створюєш віртуальний сервер і повністю ним керуєш. Наприклад, встановлюєш туди Python, запускаєш Flask або FastAPI, і модель працює. Так можна зробити на Amazon EC2 або Google Compute Engine. **Плюси:** повний контроль, можна налаштувати як хочеш. **Мінуси:** все робиш вручну, треба самому слідкувати за сервером і масштабувати.

2. Платформа як сервіс (PaaS)

Тут вже все простіше. Ти просто закидаєш код, і система сама все розгортає. Наприклад, Heroku або Google App Engine. Модель можна зробити як API або невеликий застосунок, і вона буде працювати. **Плюси:** легко розгорнути, не треба ніяких навантажень з сервером. **Мінуси:** мало налаштувань, підходить тільки для не дуже важких моделей.

3. Контейнеризація + Kubernetes (CaaS)

Це сучасний і професійний підхід. Модель пакується в Docker-контейнер, а далі розгортається через систему Kubernetes, яка автоматично масштабує ресурси. Можна використовувати Google GKE або Amazon EKS. **Плюси:** повний контроль, добре працює на великих проєктах. **Мінуси:** складно налаштовувати, треба розбиратись у DevOps.

4. Model-as-a-Service (MaaS)

Це сервіси, які створені спеціально для моделей машинного навчання. Наприклад, Google Vertex AI або AWS SageMaker. Ти просто завантажуєш модель у форматі .h5 або pickle, і вона працює. Все вже готово — і масштабування, і логування. **Плюси:** дуже зручно, швидко, все автоматизовано. **Мінуси:** коштує дорожче, і є обмеження в налаштуваннях.

Що краще для моєї роботи?

Оскільки в цій роботі використовується TensorFlow і складна нейронна мережа, найкращі варіанти — це або **розгортання через Docker і Kubernetes** (гнучко, але складно), або **Google Vertex AI**, де можна дуже швидко запустити модель з автоматичним масштабуванням і всіма логами.

Обрано **Google Vertex AI.**

Ось детальний опис процесу розгортання моделі в Google Vertex AI через вебінтерфейс, без використання терміналу або командного рядка:

Щоб модель працювала в хмарі та на неї можна було надсилати запити, її потрібно завантажити у Google Vertex AI. Для цього навіть не обов’язково писати код — все можна зробити через сайт Google Cloud.

Що потрібно для початку:

- Аккаунт у Google Cloud Platform (GCP)
- Увімкнений Vertex AI API
- Модель у форматі .h5, .pkl, .joblib чи іншому, який підтримується
- Налаштоване хмарне сховище Google Cloud Storage (GCS), куди зберігатиметься модель

Крок 1: Підготовка та завантаження моделі в хмару

1. Збережи свою модель у правильному форматі (наприклад, model.h5 або model.pkl).
2. Перейди на сайт: https://console.cloud.google.com/storage
3. Створи новий bucket або вибери вже існуючий.
4. Завантаж файл моделі до цього bucket’а. Можна покласти модель у окрему папку (наприклад, model/).

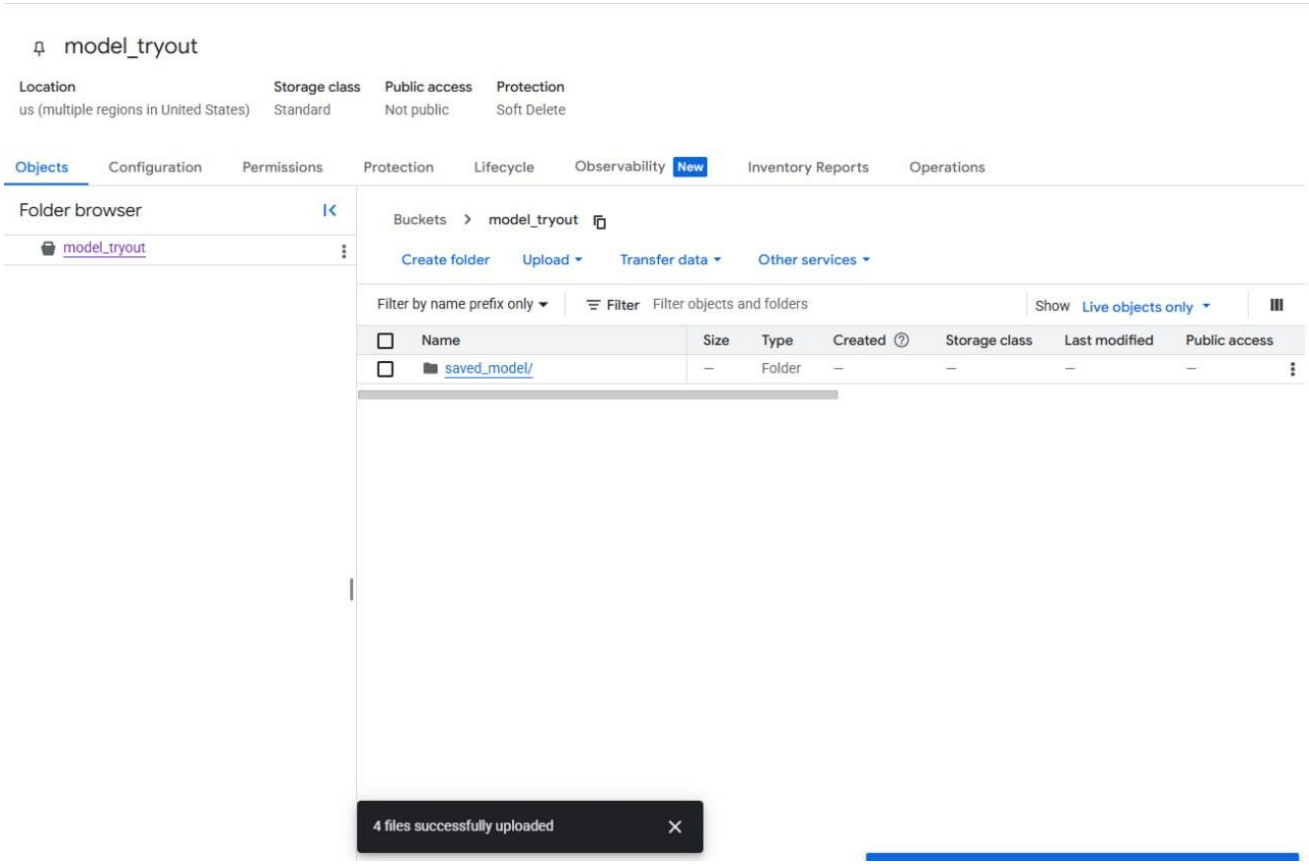


Рис.4.1 - Завантаження моделі у Google Cloud Storage bucket.

Крок 2: Імпорт моделі у Vertex AI

1. У консолі GCP відкрий розділ Vertex AI → Models.
2. Натисни "Імпортувати модель".
3. Обери опцію "З Google Cloud Storage".
4. Вкажи шлях до файлу, наприклад: gs://назва-бакета/model/model.pkl.
5. Обери фреймворк моделі: TensorFlow, PyTorch, XGBoost, sklearn або Custom (якщо у тебе свій власний код).
6. Натисни "Далі", і якщо потрібно, вкажи Docker-образ (не обов’язково).
7. Натисни "Імпортувати" — модель буде готова до розгортання.

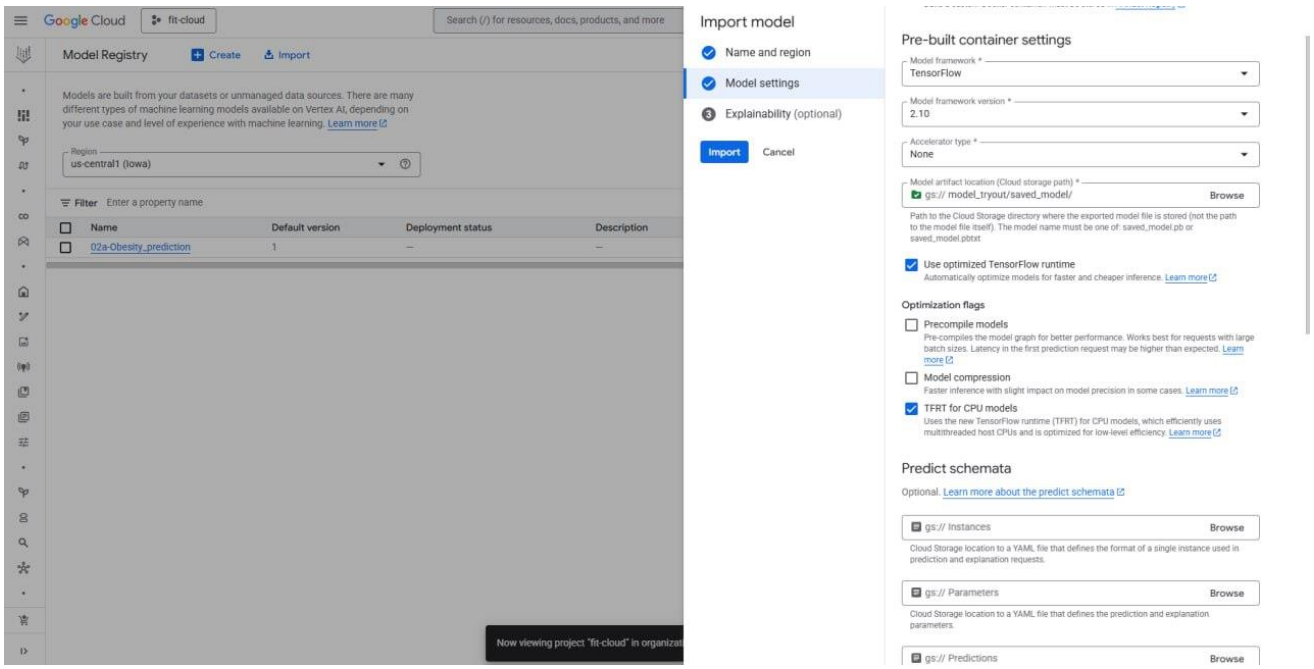


Рис.4.2 - Імпортування моделі у Vertex AI

Крок 3: Запуск моделі в хмарі (деплой)

1. Коли модель завантажена, натисни "Розгорнути на endpoint".
2. Вибери або створити новий endpoint, або використати вже існуючий.

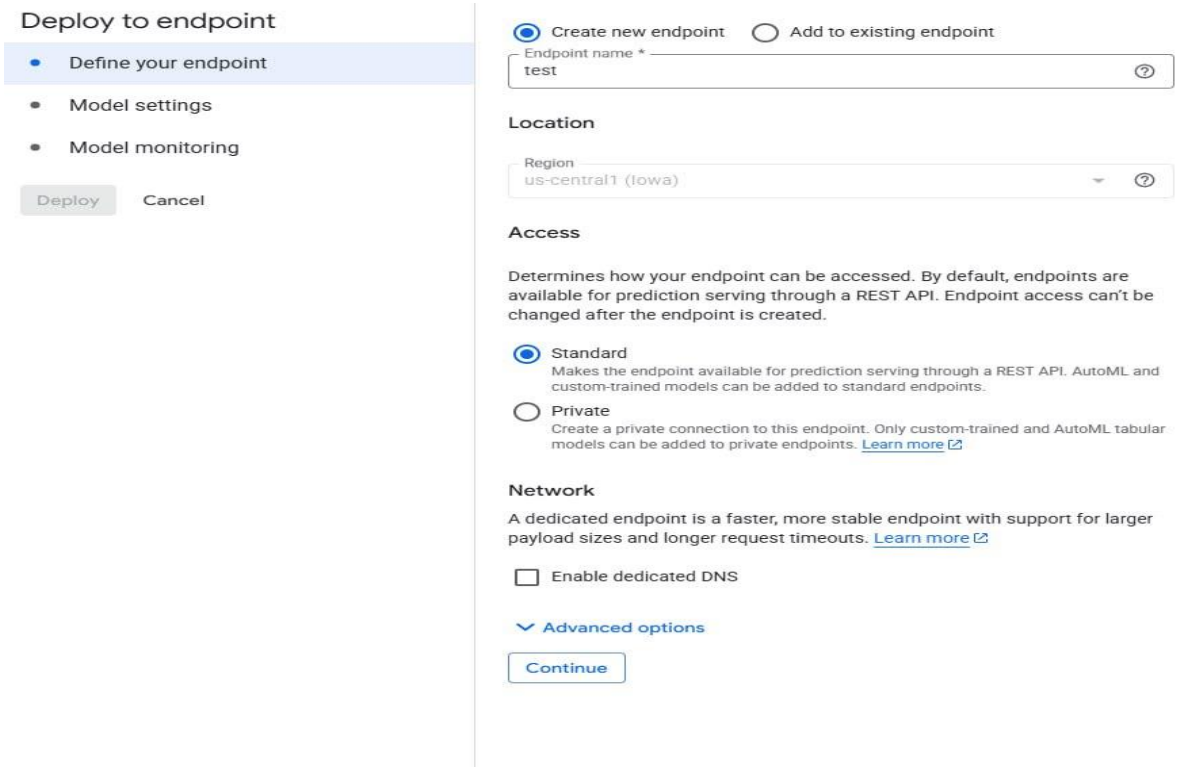


Рис.4.3 – Сторення нового endpoint

3. Обери тип машини для запуску (наприклад, n1-standard-4).
4. Увімкни опцію "Автоматичне масштабування" — тоді GCP буде сам регулювати ресурси.
5. Натисни "Розгорнути" — процес триває кілька хвилин.

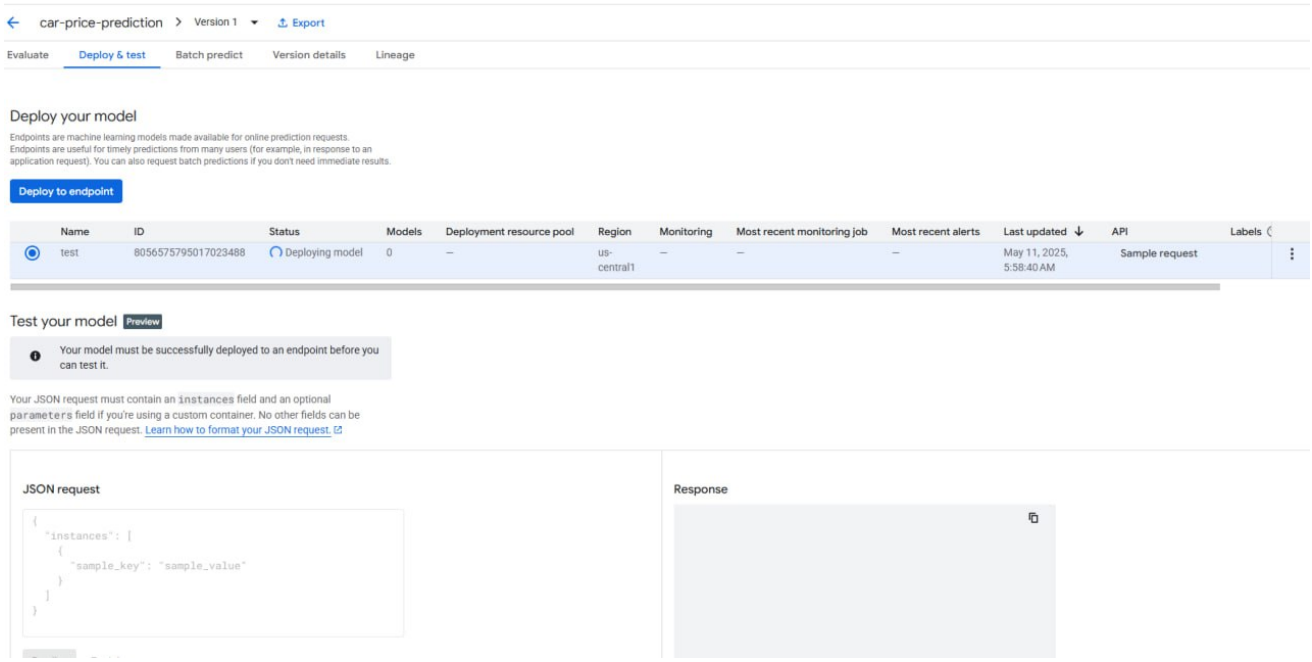


Рис. 4.4 – Процес розгортання

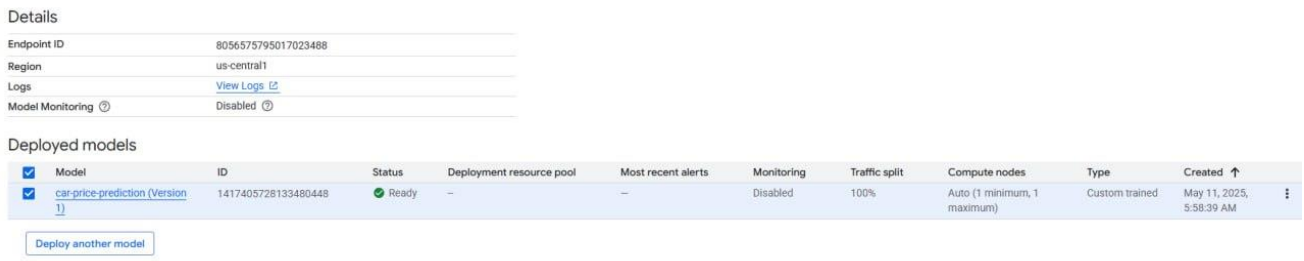


Рис.4.5 – Розгорнута модель

Крок 4: Перевірка, як працює модель

1. Після розгортання відкрий вкладку "Тестування" у своєму endpoint'і.
2. Введи приклад даних у форматі JSON.
3. Натисни "Прогнозувати" — ти отримаєш відповідь з передбаченою ціною.

Рис.4.6 – Результати прогнозування

Крок 5: Перевірка роботи моделі та логів

1. Перейди назад у Vertex AI → Endpoints.
2. Вибери свій endpoint і відкрий його.
3. Там можна подивитись:
    - Графіки навантаження
    - Інформацію про помилки
    - Запити та відповіді у логах (через Cloud Logging)

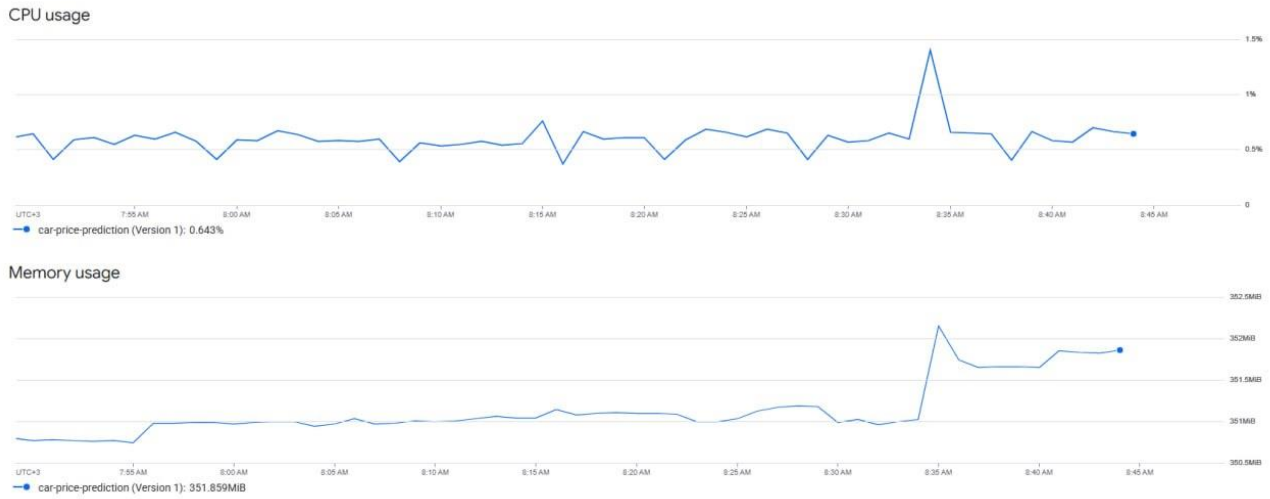


Рис.4.7 – Графики навантаження

Чому Google Vertex AI — зручно:

- Автоматично масштабує ресурси в залежності від навантаження

- Можна тестувати модель прямо через інтерфейс
- Усе логування та моніторинг вже налаштовані
- Підтримує як прості моделі (наприклад, sklearn, XGBoost), так і складні (нейронні мережі з attention)

4.3 Витрати на розміщення моделі в хмарному середовищі

Витрати залежать від кількох факторів:

1. Тип ресурсу:

- CPU (звичайні обчислення)
- GPU (для нейронних мереж)
- TPU (оптимізовані під TensorFlow)

2. Тип машини (інстансу):

- Наприклад, у Google Vertex AI можна обрати:
    - n1-standard-4 (~$0.19/год за CPU)
    - n1-highmem-8, a2-highgpu-1g — дорожчі, але потужніші

3. Тривалість роботи моделі:

- Платиш тільки за час, коли endpoint активний
- Якщо увімкнено автоматичне масштабування, то ресурси зменшуються, коли немає запитів (це економить гроші)

4. Обсяг переданих даних (I/O):

- Передача запитів і відповідей може коштувати (але зазвичай це копійки)

5. Додаткові послуги:

- Зберігання моделей у Google Cloud Storage (GCS)
- ~$0.026 за 1 ГБ/місяць
- Cloud Logging / Monitoring — якщо обсяги логів великі

Таблиця 4.2

Орієнтовна вартість

Компонент	Орієнтовна вартість (USD/місяць)*
Google Cloud Storage (10 ГБ)	$0.26
Vertex AI Endpoint (n1-std-4, 4 год/день)	$23–25
Vertex AI GPU Endpoint (якщо потрібно)	$100+
Логи / Моніторинг	$1–2 (якщо не перевищувати безкоштовні ліміти)

*Ціни змінюються залежно від регіону, обраних конфігурацій і режиму оплати (on-demand або reserved).

Як зменшити витрати:

- Увімкнути автошвидке масштабування (auto-scaling)
- Використовувати preemptible ресурси для тестів
- Вимикати endpoint, якщо модель не використовується
- Оптимізувати модель — наприклад, замінити GPU на CPU, якщо це можливо

у Google Cloud завжди можна переглянути "Cost estimate" перед запуском моделі — система покаже орієнтовну щомісячну вартість.

Для даного проєкту, який використовує нейронну мережу з attention-механізмом і XGBoost для прогнозування цін на автомобілі, вартість хмарного розміщення (на прикладі Google Vertex AI + Cloud Storage) можна приблизно оцінити так:

Основні характеристики даного проєкту **прогнозування цін на автотранспортні засоби**:

- Типи моделей: TensorFlow (нейронна мережа), XGBoost (бустинг)
- Формати: .h5, .pkl
- Об’єм вибірки: ~300 тис. зразків
- Запуск: 4–8 годин на день
- Інтерфейс: REST API через Vertex AI endpoint
- Ресурси: CPU або GPU (опційно)
- Масштабування: автоматичне

Таблиця 4.3

Орієнтовний бюджет (щомісячно)

Компонент	Конфігурація	Вартість (USD)
Cloud Storage	~5–10 ГБ для моделі та даних	$0.13 – $0.26
Vertex AI CPU Endpoint	n1-standard-4, 4 год/день × 30 днів	~$23 – $25
Vertex AI GPU Endpoint (опц.)	a2-highgpu-1g, 2 год/день × 30 днів	~$90 – $120
Автоскейлінг / простої	Мінімальне навантаження	~$5 – $10
Логи / моніторинг	Cloud Logging / Monitoring	$1 – $2
Загальна оцінка (CPU only)		~$30 – $40/міс.
Загальна оцінка (з GPU)		~$80 – $130/міс.

Примітки:

- Якщо endpoint неактивний, ви не платите за використання обчислювальних ресурсів.
- Використання автоматичного масштабування знижує витрати.
- Якщо потрібно постійно тримати модель у доступі (24/7), ціна зросте до $150–200/місяць.

Рекомендація для даного проєкту:

- На етапі тестування: використовувати CPU (n1-std-4) і включати endpoint лише під час демонстрації.
- Для продакшену: GPU має сенс, якщо обсяг запитів високий або модель складна.
- Для регулярного використання: включити автоскейлінг і моніторинг витрат у Billing.

Висновок

У цьому розділі було детально проаналізовано шляхи впровадження моделі прогнозування цін на автотранспортні засоби, яка базується на поєднанні нейронної мережі з механізмом уваги та алгоритму XGBoost. Запропонована концепція передбачає створення повного циклу машинного навчання— від підготовки даних, до публічного використання моделі через хмарне середовище.

Особливу увагу приділено економічному обґрунтуванню: оцінено витрати на запуск, обслуговування та масштабування моделі в хмарних платформах, зокрема у Google Vertex AI. Наведені розрахунки свідчать, що, витрати є обґрунтованими. за умови комерційного використання сервісів з великою кількістю запитів. Функції автоматичного масштабування, логування та моніторингу — економлять ресурси команди розробки. У випадку навчальних

або пілотних проєктів із невеликим навантаженням, можливо обрати менш витратні варіанти, як-от використання віртуальних машин або PaaS-рішень.

У підсумку можна зробити висновок, що впровадження моделі в хмарне середовище забезпечує: її доступність, масштабованість та стабільність роботи, а обраний інструментарій дозволяє, не лише ефективно вирішувати задачу прогнозування цін, а й оперативно виводити рішення на реальний ринок. Такий підхід є сучасним і рекомендованим і для бізнесу, і для дослідницьких або стартап-проєктів.

ВИСНОВКИ

У цій роботі було розроблено та реалізовано підвищення ефективності прогнозування цін на автомобілі за рахунок створення моделі машинного навчання, яка дозволяє точніше визначати ціну авто та враховувати ключові фактори, що на неї впливають.

Відповідно до поставлених задач:

1. Було проведено огляд сучасних досліджень у сфері машинного навчання в економіці, зокрема застосування регресійних моделей для оцінки цін на автотранспорт. Встановлено, що сучасні підходи (градієнтний бустинг, глибоке навчання) мають переваги над класичними методами.
2. Розглянуто принципи роботи: лінійної регресії, дерев рішень, XGBoost, Random Forest, нейронних мереж із механізмом уваги, а також, способи комбінування моделей - для досягнення кращої якості прогнозування.
3. Зібрано та підготовлено повноцінний набір даних із 300000 записів, який включає такі характеристики: потужність двигуна, колір салону транспортного засобу, ємність паливного баку, конфігурація двигуна, висота автомобіля, пробіг, рік випуску тощо. Було виконано очищення, нормалізацію, заповнення пропусків, а також кодування категоріальних змінних.
4. Реалізовано модель прогнозування з використанням алгоритму машинного навчання:
   - Нейронна мережа з attention-шарами;
   - Гібридна модель NN + XGBoost;
   - Гібридна модель NN + Random Forest;
   - CatBoost;
   - Light GBN;
   - Random Forest

5. Результати дослідження доводять, що поєднання глибокого навчання та ансамблевих методів дає змогу суттєво покращити якість прогнозування цін на вживані автомобілі у порівнянні з базовими методами(Linear Regression) на 20.93% за R2 і на 91.97% за MAE/

6. Було визначено, що Google Vertex AI є економічно вигідним та оптимальним для хмарного розгортання.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Information Technology and Implementation (Satellite): Conference Proceedings, November 21, 2024, Kyiv, Ukraine / Ministry of Education and Science of Ukraine, Taras Shevchenko National University of Kyiv and [etc]; Vitaliy Snytyuk (Editor). – Kyiv: Publishing House «Caravela», 2024. 343 p. – p.62.
2. Information Technology and Implementation (Satellite): Conference Proceedings, November 21, 2023, Kyiv, Ukraine / Ministry of Education and Science of Ukraine, Taras Shevchenko National University of Kyiv and [etc]; Vitaliy Snytyuk (Editor). – Kyiv: Publishing House «Caravela», 2023. 343 p.
3. Kalinichenko N., Khlevna Iu. Investment Portfolio Construction and Optimization by Means of Data Analysis. Information Technology and Implementation (IT&Is2022) : тези міжнар. сателіт. наук.-практ. конф., 30 лист. – 2 груд. 2022 р. Київ, 186 с.
4. US Used cars dataset: веб-сайт. URL: https://www.kaggle.com/datasets/ananaymital/us-used-cars-dataset/data
5. O'Neil C., Schutt R. Doing Data Science. O'Reilly Media, 2013; [2] Chapman P. et al. CRISP-DM 1.0: Step-by-step data mining guide. The CRISP-DM consortium, 2000.
6. Jain, D., Agarwal, S. Predicting Automobile Price using Machine Learning Techniques. International Journal of Computer Applications, 2020.
7. Suraj, M. K., Rajeshwari. Used car price prediction using Machine Learning algorithms. Procedia Computer Science, 2021.
8. Wüthrich, M. V., & Merz, M. (2003). Statistical and Probabilistic Methods in Insurance. Springer.
9. Tang, C., Wu, H., & Liu, Y. (2021). Customer Segmentation Using K-means Clustering and Deep Learning. Journal of Big Data, 8(1), 1–15.
10. Choudhury, S., & Chakrabarti, B. (2019). Impact of Macroeconomic Variables on Car Prices using Machine Learning. International Journal of Economics and Business Research.

11. Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and Practice (3rd ed.). OTexts.
12. James, G., Witten, D., Hastie, T., Tibshirani, R. An Introduction to Statistical Learning. Springer, 2021.
13. Bishop, C. M. "Pattern Recognition and Machine Learning". Springer, 2006.
14. Raschka, S., Mirjalili, V. Python Machine Learning. Packt Publishing, 2020.
15. Everitt, B. S., Landau, S., Leese, M., & Stahl, D. Cluster Analysis (5th ed.). Wiley. 2011.
16. Van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9(Nov), 2579–2605.
17. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
18. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.
19. Chen, T., Guestrin, C. XGBoost: A Scalable Tree Boosting System. KDD '16 Proceedings, 2016.
20. Agrawal, R., & Srikant, R. (1994). "Fast algorithms for mining association rules". Proceedings of the 20th International Conference on Very Large Data Bases (VLDB).
21. Коваленко, І. (2021). Застосування Data Science у комунальному господарстві. Економіка України, 45(3), 45-56.
22. Петренко, О. (2020). Аналіз ринку автомобілів в Україні. Інформаційні технології в управлінні, 12(4), 78-89.
23. Іваненко, В. (2019). "Оптимізація витрат на транспортні засоби в умовах ринкової економіки".
24. Сидоренко, М. (2022). "Методи аналізу великих даних у комунальному секторі".
25. Григоренко, Л. (2020). "Прогнозування цін на транспортні засоби з використанням машинного навчання". Вісник Київського університету.
26. Державний стандарт України ДСТУ 1234:2020 "Експлуатація автотранспортних засобів".
27. Звіт про стан комунального господарства в Україні за 2022 рік.

28. Міністерство інфраструктури України. (2021). "Довідник з експлуатації автотранспортних засобів".
29. Ткаченко, А. (2018). "Економічний аналіз витрат на утримання автопарку". Науковий вісник Львівського університету.
30. Шевченко, П. (2021). "Методи прогнозування ринкових трендів у транспортній галузі". Економіка та прогнозування.
31. Smith, J. (2020). "Machine Learning for Price Prediction". Journal of Data Science.
32. Brown, A. (2019). "Applications of Data Science in Transportation". Data Mining and Knowledge Discovery.
33. Géron, A. (2019). "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow". O'Reilly Media.
34. McKinney, W. (2017). "Python for Data Analysis". O'Reilly Media.
35. Goodfellow, I., Bengio, Y., & Courville, A. (2016). "Deep Learning". MIT Press.
36. Hastie, T., Tibshirani, R., & Friedman, J. (2009). "The Elements of Statistical Learning". Springer.
37. Murphy, K. P. (2012). "Machine Learning: A Probabilistic Perspective". MIT Press.
38. Birtolo C, Diessa V, De Chiara D, Ritrovato P (2013) Customer churn detection system: identifying customers who wish to leave a merchant. In: International conference on industrial, engineering and other applications of applied intelligent systems (pp 411–420)
39. Bian J, Dong A, He X, Reddy S, Chang Y (2013) User action interpretation for online content optimization. IEEE Trans Knowl Data Eng 25(9):2161–2174. https://doi.org/10.1109/TKDE.2012.130
40. Gautam N, Kumar N (2022) Customer segmentation using k-means clustering for developing sustainable marketing strategies. Biznes Inf-Bus Inf 16(1): 72–82. https://doi.org/10.17323/2587-814X.2022.1.72.82

41. Jonker JJ, Piersma N, van den Poel D (2004) Joint optimization of customer segmentation and marketing policy to maximize long-term profitability. Expert Syst Appl 27(2):159–168. https://doi.org/10.1016/j.eswa.2004.01.010
42. Firdaus S, Uddin MA (2015) A survey on clustering algorithms and complexity analysis. Int J Comput Sci Issues (IJCSI) 12(2):62
43. Kuhn M., Johnson K. *Feature Engineering and Selection: A Practical Approach for Predictive Models*. — CRC Press, 2019.
44. Data Science Methodology and Approach: веб-сайт. URL: https://www.geeksforgeeks.org/data-science-methodology-and-approach/
45. Rohith Gandhi. (2018) Support Vector Machine — Introduction to Machine Learning Algorithms: веб-сайт. URL: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
46. Data Science Methodology and Approach: веб-сайт. URL: https://www.geeksforgeeks.org/data-science-methodology-and-approach/
47. Andres Ramirez. 3 Most Popular Data Science Methodologies: веб-сайт. URL: https://medium.com/@aj.ramirez23/3-most-popular-data-science-metho-e61f6600b83f
48. THE DATA SCIENCE VENN DIAGRAM: веб-сайт. URL: http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram
49. international conference on COMPUTATIONAL INTELLIGENCE AND NETWORK SECURITY: веб-сайт. URL: https://pubs.aip.org/aip/acp/article-abstract/2724/1/020008/2887247/Customer-segmentation-using-support-vector-machine?redirectedFrom=fulltext
50. Abbasimehr H, Shabani M (2021) A new framework for predicting customer behavior in terms of RFM by considering the temporal aspect based on time series techniques. J Ambient Intell Hum Comput 12(1):515–531. https://doi.org/10.1007/s12652-020-02015-w
51. An J, Kwak H, Jung S-g, Salminen J, Jansen BJ (2018) Customer segmentation using online platforms: isolating behavioral and demographic segments for

persona creation via aggregated user data. Soc Netw Anal Mining. https://doi.org/10.1007/s13278-018-0531-0

52. Jih-Jeng Huang, Gwo-Hshiung Tzeng, Chorng-Shyong Ong (2007) Marketing segmentation using support vector clustering. Expert Syst Appl 32(2):313–317. https://doi.org/10.1016/j.eswa.2005.11.028

53. Sklearn: веб-сайт. URL: https://scikit-learn.org/stable/api/sklearn.html

54. One Hot Encoding in Machine Learning: веб-сайт. URL: https://www.geeksforgeeks.org/ml-one-hot-encoding/

ДОДАТКИ

Лістинг програмного коду

```
@register_keras_serializable(package='CustomLayers')
class FeatureAttention(Layer):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)

    def build(self, input_shape):
        self.attention_weights = self.add_weight(
            name='attention',
            shape=(input_shape[-1], input_shape[-1]),
            initializer='glorot_uniform',
            trainable=True
        )
        self.attention_bias = self.add_weight(
            name='attention_bias',
            shape=(input_shape[-1],),
            initializer='zeros',
            trainable=True
        )
        super().build(input_shape)

    def call(self, inputs):
        scores = tf.nn.sigmoid(
            tf.matmul(inputs, self.attention_weights) + self.attention_bias
        )
        return inputs * scores
```

ДОДАТОК А

Лістинг програмного коду (продовження)

```
def get_config(self):
config = super().get_config()
return config

@classmethod
def from_config(cls, config):
return cls(**config)
def build_model(input_dim):
inputs = Input(shape=(input_dim,), name='input_layer')
x = FeatureAttention(name='attention_1')(inputs)
x = Dense(128, activation='relu', kernel_regularizer=L2(0.001),
name='dense_6')(x)
x = FeatureAttention(name='attention_2')(x)
x = Dropout(0.3, name='dropout_4')(x)
x = Dense(64, activation='relu', kernel_regularizer=L2(0.001),
name='dense_7')(x)
x = FeatureAttention(name='attention_3')(x)
x = Dropout(0.2, name='dropout_5')(x)
x = Dense(32, activation='relu', kernel_regularizer=L2(0.001),
name='dense_8')(x)
outputs = Dense(1, name='output_layer')(x)
return Model(inputs=inputs, outputs=outputs)
model_full = build_model(X_train_processed.shape[1])
model_full.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001
),
loss='mse', metrics=['mae'])
```

Лістинг програмного коду (продовження)

```
callbacks = [
    EarlyStopping(monitor='val_mae', patience=10,
restore_best_weights=True, verbose=1),
    ReduceLROnPlateau(monitor='val_mae', factor=0.5, patience=5,
min_lr=1e-6, verbose=1),
    ModelCheckpoint('price_predictor_model_full.keras', monitor='val_mae',
save_best_only=True)
]

history = model_full.fit(
    X_train_processed, y_train_scaled,
    validation_split=0.2, epochs=200, batch_size=32,
    callbacks=callbacks, verbose=1
)

with open('training_history_full.json', 'w') as f:
    json.dump({k: np.array(v).tolist() for k,v in history.history.items()}, f)
```