ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА
(повне найменування вищого навчального закладу)

Факультет математики, фізики та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра математичного забезпечення комп'ютерних систем
(повна назва кафедри (предметної, циклової комісії))

Кваліфікаційна робота

на здобуття ступеня вищої освіти «бакалавр»
(освітньо-кваліфікаційний рівень)

Методи машинного навчання для аналізу простору станів предметної області
Machine learning methods for the analyzis of the subject area state space

Виконала: студентка денної форми навчання
спеціальності 126 – Інформаційні системи та технології .
(шифр і назва напряму підготовки, спеціальності)
Освітня програма «Інформаційні системи та технології»
(назва освітньої програми)
Мациєвська Анастасія Олександрівна
(прізвище, ім'я, по-батькові)

Керівник к.т.н., доц. Пенко В.Г.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рецензент к.ф.-м.н., доц. Антоненко О.С.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
№___ від «___»_____ 2024 р.
Завідувач кафедри
_____ Євгеній МАЛАХОВ
(підпис) (ім'я, прізвище)

Захищено на засіданні ЕК № ___
протокол № __ від «___» ____ 2024 р.
Оцінка _____/_____/_____
(за національною шкалою, шкалою ECTS, бали)
Голова ЕК
_____ Володимир ВИЧУЖАНІН
(підпис) (ім'я, прізвище)

Одеса – 2024

АНОТАЦІЯ

У дипломній роботі розглядається тема «Методи машинного навчання для аналізу простору станів предметної області».

Метою роботи є використання технології машинного навчання для розв’язання задач логістики, що виникають на підприємстві, орієнтованому на перевезенні товарів.

Результати даного аналізу допоможуть краще зрозуміти структуру даних логістичної компанії та визначити, які методи машинного навчання можуть бути найбільш ефективними для практичного застосування.

Для реалізації проекту взято набори даних існуючої логістичної компанії. Для аналізу і обробки даних використовується мова програмування Python з бібліотеками машинного навчання, такими як scikit-learn, pandas, і numpy. Для створення інтерактивного веб-додатку візуалізації результатів аналізу використовується фреймворк Streamlit.

В розробленому додатку можна проаналізувати вхідні набори даних і отримати їх характеристики. За допомогою методів машинного навчання є можливість спрогнозувати вартість перевезення або інші змінні, враховуючи параметри конкретних моделей.

Завдяки гнучкості додатку можна завантажувати необхідні набори даних, порівнювати роботу різних методів між собою та експортувати результати у форматі CSV-файлу.

У результаті проведеного аналізу були виявлені недоліки та надані рекомендації для їх усунення.

ANNOTATION

This thesis deals with the topic of «Machine learning methods for the analyzis of the subject area state space».

The aim of the work is to use machine learning technology to solve logistics problems that arise in a company focused on the transportation of goods.

The results of this analysis will help to better understand the data structure of a logistics company and determine which machine learning methods can be most effective for practical application.

The project is based on the data sets of an existing logistics company. The Python programming language with machine learning libraries such as scikit-learn, pandas, and numpy is used to analyze and process the data. The Streamlit framework is used to create an interactive web application for visualizing analysis results.

The developed application allows analyzing input data sets and obtaining their characteristics. Using machine learning methods, it is possible to predict the cost of transportation or other variables, taking into account the parameters of specific models.

Due to the flexibility of the application, you can upload the necessary data sets, compare the performance of different methods with each other, and export the results in a CSV file format.

As a result of the analysis, shortcomings were identified and recommendations were made to eliminate them.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

Скорочення

CV folds – крос-валідація

ECLAT – перетворення класу еквівалентності

ER-діаграма – Entity-Relationship diagram, діаграма сутностей-зв'язків

FCL – повністю завантажений контейнер

MAE – Середня абсолютна помилка

MSE – Середньоквадратична помилка

PCA – Principal Component Analysis, метод головних компонент

R2 Score – Коефіцієнт детермінації

SVD – Singular Value Decompression, сингулярний розклад

SVM – метод опорних векторів

ШІ – Штучний інтелект

ВСТУП

Проблеми логістичних процесів залишаються актуальними в сучасному світі, оскільки ефективне управління постачанням та розподілом ресурсів має велике значення для успішної діяльності підприємств і організацій.

Логістика включає в себе не лише дані про перевезення, але й планування найефективніших маршрутів та схем зберігання товарів і послуг від місця виробництва до споживача. У сучасному світі логістичні процеси стали настільки складними, що, крім звичайного програмного забезпечення, фахівці звертаються до штучного інтелекту та нейронних мереж, зокрема алгоритмів машинного навчання.

Застосування машинного навчання у логістиці підвищує прозорість у ланцюжку постачання, відстежуючи дані на всьому шляху відправлення. Воно також персоналізує взаємодію з клієнтами, автоматично прогнозуючи їхні потреби та вирішуючи типові запити. Машинне навчання сприяє швидкому та ефективному прийняттю рішень завдяки обробці великих обсягів різнорідних даних. Однією з основних переваг є автоматизація ручних завдань, включаючи оптимізацію маршрутів, розподіл завдань і управління запасами, що суттєво знижує потребу в ручній праці та мінімізує людські помилки і затримки.

Точне прогнозування попиту допомагає компаніям ефективно управляти рівнем запасів та підвищувати продуктивність ланцюга постачання. Машинне навчання стає важливим інструментом для визначення найбільш економічно ефективних маршрутів доставки та вибору перевізників, що допомагає підприємствам скорочувати витрати на транспортування.

Конкуренція та криза ставлять перед логістичними компаніями жорсткі вимоги: необхідно заробляти більше, витрачати менше і реагувати швидше. Однак, наразі машинне навчання ще не стало стандартизованим і звичним інструментом у діяльності логістичних компаній.

Основною метою даної роботи є використання технології машинного навчання для вирішення задач логістики, що виникають на підприємстві, орієнтованому на перевезенні товарів.

Об’єктом дослідження є процес прогнозування і виявлення факторів, що впливають на ціноутворення.

Предметом дослідження є технології машинного навчання.

Для досягнення цієї мети поставлені та вирішені наступні задачі:

– виконано аналіз предметної області;

– обрано технології та засоби реалізації;

– здійснено розвідковий аналіз даних та їх підготовку до навчання моделі;

– розглянуто різні методи машинного навчання, визначено їх переваги та недоліки;

– створено зручний веб-додаток, який дозволяє аналізувати дані за різноманітними параметрами;

– створено можливість прогнозування числових змінних та порівняння прогнозів за метриками: MSE, MAE, R2 Score;

– реалізовано інтерфейс для налаштування гіперпараметрів моделей.

Результати даного аналізу допоможуть компанії краще зрозуміти структуру даних та визначити, які методи машинного навчання можуть бути найоптимальнішими для практичного застосування.

1 КЛАСИФІКАЦІЯ ЛОГІСТИЧНИХ ЗАДАЧ

Вибір методу для вирішення логістичних завдань залежить від конкретного контексту, доступних даних та вимог до точності.

Задачі логістики здебільшого вирішуються за допомогою прикладних методів, таких як лінійне програмування, функціонально-вартісний аналіз, імітаційне моделювання та інших.

До таких задач належать:

– оптимізація логістичних маршрутів вантажів;

– задача про максимальний потік;

– вибір маршруту перевезення для мінімізації пробігу автомобільного транспорту;

– вибір виду транспорту для перевезення вантажів;

– раціональний розподіл трансформаційних об'єктів (маршрути транспортування, розміщення складських приміщень, розподіл робочої сили);

– задачі оперативного реагування на змінні вимоги споживачів;

– задачі на підвищення готовності транспортних засобів;

– зниження витрат у всіх ланках логістичного ланцюга.

Детальний огляд деяких задач :

− для покращення ефективності логістичних маршрутів вантажів застосовуються методи оптимізації маршрутів, наприклад, алгоритм Флойда-Уоршелла, який знаходить найкоротші шляхи між усіма вершинами графа та їх довжину, або алгоритм Дейкстри, який знаходить оптимальні маршрути та їх довжину між однією конкретною вершиною (джерелом) та всіма іншими вершинами графа або генетичний алгоритм - алгоритм пошуку, що використовується для вирішення задач оптимізації та моделювання шляхом випадкового підбору, комбінування та варіації шуканих параметрів з використанням механізмів, аналогічних природному відбору в природі [1].

− задача про максимальний потік у мережі за допомогою алгоритма Форда-Фалкерсона. Ця задача полягає у знаходженні потоку через транспортну мережу, який максимізує суму потоків витоку або до стоку.

− для розв’язання задачі про вибір маршруту перевезення для мінімізації пробігу автомобільного транспорту застосовується метод Кларка-Райта. Алгоритм використовує три основних етапи: побудова початкового розв’язку, пошук локальних оптимумів та вибір кращого розв’язку. Також може використовуватися функція Лагранжа для включення обмежень у вигляді штрафів у цільову функцію та метод градієнта, який використовує напрям найшвидшого зростання чи спадання функції для знаходження її екстремуму.

− для оптимізації вибору виду транспорту для перевезення вантажів може бути застосовано лінійне програмування. Для видів транспорту $T_1,T_2,\ldots,T_n$, для яких критерієм вибору є, наприклад, вартість перевезення, необхідно мінімізувати обрану цільову функцію за формулою

$$Z=c_1x_1 + c_2x_2 + \ldots + c_nx_n, \quad (1.1)$$

де Z – обрана цільова функція;

c_i – вартість перевезення одиниці вантажу видом транспорту T_i;

x_i – обсяг вантажу, який буде перевезено видом транспорту T_i, тощо.

Обмеження на вантажопідйомність $a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n\leq B_1$ (обмеження вказує, що сума вантажу, яку перевозять всі види транспорту, не може перевищувати певну максимальну величину B_1), час доставки $a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n\leq B_2$ (обмеження вказує, що сумарний час доставки вантажу не може перевищувати певну максимальну величину B_2), невід'ємність обсягів вантажу $x_1,x_2,\ldots,x_n\geq 0$ і т.д.

Розв'язанням задачі лінійного програмування буде вектор $(x_1,x_2,\ldots,x_n)$, який вказує обсяги вантажів, які слід перевезти кожним видом транспорту. Ця загальна модель може бути адаптована в залежності від конкретних умов та вимог логістичної системи [2].

Якщо логістичні задачі, пов'язані з аналізом великих обсягів неструктурованих даних, виявленням складних патернів і прогнозуванням в умовах змінного та непередбачуваного середовища, то найефективніше вони будуть вирішуватися з використанням методів машинного навчання.

Діяльність логістичних компаній піддана впливу різноманітних економічних факторів, таких як обсяг попиту на транспортні послуги, вартість палива, вид транспорту, наявність внутрішніх та зовнішніх комунікацій, надійність ланцюгів постачання і т.д. Для всебічного аналізу, обліку та коригування цих факторів необхідні оброблені та структуровані великі дані. При вмілому використанні в алгоритмах машинного навчання вони стають основою для поліпшення окремих бізнес-показників і, в цілому, рентабельності підприємства [3]. Мета машинного навчання - робити прогнози, засновані на аналізі накопичених раніше фактів, та виявленні загальних тенденцій і властивостей.

Приклади логістичних задач, для яких ефективніше використовувати методи машинного навчання:

– прогнозування попиту, аналізуючи історичні дані про продажі, сезонні тренди та акції, використовуючи моделі регресії, нейронні мережі або дерева рішень;

– аналіз реальних даних про трафік і дорожні умови для оптимізації маршрутів в режимі реального часу, знижуючи витрати на паливо та час доставки за допомогою алгоритмів оптимізації на основі посилення (reinforcement learning);

– аналіз географічних даних і визначення оптимальних місць розташування складів для мінімізації логістичних витрат за допомогою кластеризаційних алгоритмів, таких як K-means;

– підтримка оптимального рівня запасів на складах, що сприяє зниженню витрат на зберігання та мінімізує втрати від надлишкових запасів, використовуючи моделі часового ряду та прогнозних алгоритмів;

− створення систем розпізнавання образів, що здатні автоматизувати процеси сканування, ідентифікації та відстеження вантажів за допомогою комп’ютерного зору.

Враховуючи ці аспекти, можна зробити висновок, що методи машинного навчання часто ефективні при обробці складних і неструктурованих даних, а також в умовах динамічного та змінного середовища, що є типовим для багатьох логістичних завдань.

Логістика з точки зору використання машинного навчання залишається однією з найменш досліджених сфер, оскільки наразі бракує рішень, які повністю переводять усі процеси в онлайн-режим.

Сучасний бізнес стає все більш динамічним, і розуміння та використання трендів і технологій є ключовими для успіху на ринку. Трансформація бізнесу на основі цих факторів стає невід'ємною частиною стратегії виживання. Росте значення даних, оскільки їх обсяг постійно збільшується, і їх роль в сучасному бізнесі посилюється.

2 МЕТОДИ МАШИННОГО НАВЧАННЯ

У цьому розділі розглядається спектр питань, що стосується використання машинного навчання: типи, моделі, ключові аспекти технологічного конвеєру машинного навчання.

2.1 Машинне навчання

Машинне навчання – це використання математичних моделей даних, які допомагають комп'ютеру навчатись без безпосередніх інструкцій, вважається однією з форм штучного інтелекту. За допомогою алгоритмів виявляються закономірності даних, а за їх підсумками створюється модель даних прогнозування. Зі збільшенням обсягу даних модель машинного навчання набуває більшої точності. Завдяки своїй адаптивності, воно ідеально підходить для сценаріїв, де дані постійно змінюються, характеристики завдань нестабільні, або коли написання коду для рішень є практично неможливим [4].

Переваги моделей машинного навчання:

– визначення тенденцій та закономірностей даних;

– робота без втручання людини після налаштування;

– результати можуть стати точнішими з часом;

– обробка різних форматів даних у динамічних, великих обсягах та складних середовищах даних.

Недоліки моделей машинного навчання:

– необхідність великої кількості даних;

– чутливість до якості даних;

– налаштування гіперпараметрів та вибір оптимальної архітектури моделі можуть бути складними та вимагати значного часу і ресурсів;

– проблеми з інтерпретованістю;

– загроза перенавчання;

– навчання може вимагати значних обчислювальних ресурсів і часу, що може бути дорогим і не завжди виправданим.

Зазвичай алгоритм машинного навчання складається з таких пунктів:

а) підготовка даних - після визначення джерел, доступні дані об'єднуються. Важливо врахувати тип даних, які використовуються, оскільки це визначить, які алгоритми машинного навчання можуть бути застосовані. Під час перевірки даних виявляються аномалії, розробляється структура та вирішуються проблеми з цілісністю даних;

б) навчання моделі – підготовлені дані поділяються на дві групи: набір для навчання та набір для перевірки. Набір для навчання складається з більшої частини даних і використовується для налаштування моделей машинного навчання з максимальною точністю;

в) перевірка моделі – оцінюється продуктивність та точність, щоб визначити найкращу модель даних для використання у кінцевому аналізі;

г) інтерпретація результатів – отримання аналітичних відомостей, формування висновків та прогнозування результатів.

Дані зазвичай подаються у вигляді таблиці, де кожен рядок представляє окрему точку даних, таку як електронний лист, клієнт або транзакція, а кожен стовпець відповідає властивостям (ознакам) цих даних, таким як вік клієнта, сума транзакції або місце здійснення.

Важливо розуміти, що жоден алгоритм машинного навчання не зможе зробити прогноз на даних, що не містять жодної корисної інформації. Наприклад, якщо єдиним показником є назва товару в логістичних даних, алгоритм може бути неефективним у прогнозуванні обсягів попиту. Додавання додаткових ознак, таких як сезонність, економічні показники чи географічні дані, може покращити точність передбачень та виправити цю ситуацію.

2.2 Навчання під наглядом

Навчання під наглядом використовується для прогнозування результатів на основі пар об'єкт-відповідь з навчального набору даних. Мета полягає в отриманні точних прогнозів для нових, раніше небачених даних. Це вимагає участі людини на етапі створення навчального набору, але автоматизує наступні трудомісткі завдання. Існують два основні завдання: класифікація та регресія.

У першому випадку, на основі інформації про вхідний зразок, прогнозується мітка класу з обмеженого списку можливих варіантів. Класифікація може бути бінарною (поділ на два класи) або мультикласовою (більше двох класів). Наприклад, класифікація поставок на 'доставлені вчасно' та 'затримані' є бінарною.

Мета регресії полягає в прогнозуванні безперервного числа. Наприклад, прогнозування часу доставки товарів на основі відстані, обсягу замовлення, часом доби, днем тижня, а також ефективністю та завантаженістю транспортних засобів [5].

Модель навчання повинна давати точні прогнози для нових даних, що вказує на її здатність узагальнювати. Якщо модель надто точно підлаштовується під навчальні дані, вона перенавчається. Якщо модель занадто проста, вона недонавчається. Існує оптимальна точка складності моделі для найкращої узагальнюючої здатності.

Оптимальна складність моделі залежить від різноманітності даних у навчальному наборі. Більші набори даних дозволяють будувати складніші моделі, але просте дублювання або збір схожих даних не допоможе.

2.3 Лінійна регресія

Лінійна регресія (англ. Linear regression) – це модель, що описує лінійну залежність між однією змінною (залежною) і однією або декількома іншими змінними (незалежними), які впливають на неї.

Функція лінійної регресії моделює зв’язок між вхідними змінними шляхом підгонки лінійного рівняння до вхідних даних. Існують два основних типи лінійної регресії: проста лінійна регресія, яка включає лише одну незалежну змінну та одну залежну змінну, і множинна лінійна регресія, яка стосується більш ніж однієї незалежної змінної та однієї залежної змінної [5].

Мета алгоритму полягає в тому, щоб знайти найкраще відповідне рівняння лінії, яке може передбачити значення на основі незалежних змінних. Похибка між прогнозованими та фактичними значеннями повинна бути мінімальною. Нахил лінії вказує на те, наскільки змінюється залежна змінна, коли незалежна змінна змінюється на одну одиницю.

2.4 Гребенева регресія Ridge

Гребнева регресія, подібно до лінійних моделей, також використовує коефіцієнти, які підбираються для максимально точного передбачення результатів на навчальних даних та врахування додаткових обмежень. Цей метод регуляризації включає штраф для значень коефіцієнтів у функції втрат в залежності від їхнього розміру.

Всі коефіцієнти повинні наближатися до нуля. Це означає, що кожна ознака повинна мати якомога менший вплив на результат (невеликий регресійний коефіцієнт) і в той же час він повинен як і раніше мати гарну прогнозну здатність. Це обмеження є одним з прикладів регуляризації, яка використовується для запобігання перенавчанню моделі і відома як L2 регуляризація.

Модель Ridge дозволяє знайти компроміс між простотою моделі (отриманням коефіцієнтів, близьких до нуля) і якістю її роботи на навчальному наборі. Цей баланс контролюється параметром alpha. Збільшення alpha змушує коефіцієнти стискатися до близьких до нуля значень, що знижує якість роботи моделі на навчальному наборі, але може поліпшити її узагальнюючу здатність. При дуже малих значеннях alpha обмеження на коефіцієнти практично не накладається і в результаті отримується модель, що нагадує лінійну регресію.

2.5 Гребенева регресія Lasso

Альтернативою Ridge як методу регуляризації лінійної регресії є Lasso. Як і у гребневій регресії, в Lasso також коефіцієнти стискаються до близьких до нуля значень, але це відбувається трохи інакше, за допомогою L1 регуляризації. Основною метою регресії Lasso є відбір ознак шляхом стимулювання розрідженості значень коефіцієнтів. Це означає, що вона схильна зменшувати менш важливі функції до нуля, практично виключаючи їх із моделі. Регресія Lasso може бути корисною при роботі з великими наборами даних, де присутні багато потенційно незначущих функцій. Зменшуючи кількість таких, регресія спрощує модель і покращує її інтерпретацію.

2.6 Дерева рішень

Основна ідея дерев рішень полягає в тому, щоб розбити дані на менші групи за допомогою певних параметрів, щоб кожна група була більш однорідною за певним критерієм. Цей метод використовується для вирішення задач класифікації і регресії.

Дерево рішень складається з наступних елементів:

– вузли, що перевіряють значення конкретної характеристики;

– гілки/ребра, які відображають результат перевірки на вузлі та вказують напрям переходу до іншого вузла;

– кінцеві вузли, що надають прогнозовані результати, тобто у випадку класифікації - визначають класи.

Дерева рішень будуються за допомогою евристики, відомої як рекурсивне розбиття. Цей процес розбиває дані на підмножини, які потім далі розбиваються на ще менші підмножини, і так далі. Процес триває до тих пір, поки не відбудеться зупинка, яка може бути обумовлена досягненням однорідності в підмножині або досягненням іншої категорії.

Алгоритм дерев рішень розпочинається з вибору кращої ознаки для розділення даних, обчислюючи міру якості розділення (ентропію, критерій Джині або середньоквадратичну помилку). Обирається ознака, яка максимізує інформаційний приріст або мінімізує помилку. Набір даних розділяється на два піднабори на основі обраної ознаки, створюючи нові вузли. Процес повторюється рекурсивно для кожного нового вузла, поки не будуть виконані умови зупинки. Коли умови зупинки виконуються, кожен кінцевий вузол (лист) дерева присвоює прогнозоване значення. У випадку класифікації, це може означати вибір найбільш частого класу в листі, а в регресії - обчислення середнього значення цільової змінної в листі.

Переваги дерев рішень полягають в тому, що вони швидко класифікують невідомі записи, легко інтерпретуються для невеликих дерев, мають доволі високу точність на невеликих та простих даних, в порівнянні з іншими методами класифікації. Також даний алгоритм відкидає не важливі характеристики набору даних.

До недоліків можна віднести їхню схильність до перенавчання, схильність розбиття по функціям з великим рівнями складності, не стійкість до невеликих змін у навчальній вибірці та складність інтерпретації великих дерев.

2.7 Випадковий ліс

Алгоритм використовує метод "bagging" для формування ансамблю дерев рішень. Основна концепція цього методу полягає в тому, що поєднання кількох навчальних моделей призводить до покращення результатів [6].

Універсальність - головна перевага випадкового лісу, оскільки він використовується як для регресії, так і для класифікації. Замість вибору найважливішої функції, цей алгоритм шукає найкраще розділення вузла у випадковій підмножині ознак та додає додаткову випадковість при створенні нового дерева, що збільшує варіативність моделі і покращує ефективність. Таким чином, на етапі розділення вузла враховуються тільки випадкові ознаки.

Ще однією перевагою алгоритму є можливість вимірювання відносної важливості кожної ознаки для прогнозу. Важливість функцій визначає, чи будуть вони збережені або видалені. Видалення несуттєвих функцій сприяє кращій узагальнюючій здатності моделі і знижує перенавчання.

Незважаючи на швидкість навчання, алгоритм випадкового лісу може бути не найкращим вибором для практичних застосувань, де швидкість прогнозування є критичною. Це пов'язано з тим, що його прогнози можуть бути досить повільними, особливо коли використовується багато дерев.

Якщо метою є вивчення складних зв'язків у даних, краще використовувати інші методи.

2.8 Градієнтний бустинг дерев

Алгоритм Gradient Boosting ґрунтується на послідовному створенні простих моделей, які коригують помилки попередніх, щоб покращити загальну точність. Однак, існує ризик перенавчання, який потрібно уникати.

Остаточна модель комбінує результати всіх кроків, використовуючи спеціальну функцію втрат.

Важливо уважно налаштовувати гіперпараметри, такі як параметри регуляризації та кількість ітерацій. Однак вбудовані значення за замовчуванням часто працюють добре для багатьох завдань.

Однією з переваг методу Gradient Boosting є його робастність до різних типів даних, включаючи категоріальні, числові та текстові дані, що робить його універсальним для використання в різних галузях.

Також важливо відзначити, що Gradient Boosting ефективно вирішує завдання з нелінійними взаємозв'язками між ознаками та цільовою змінною. Він може легко адаптуватися до складних моделей та виявляти неочевидні закономірності в даних.

2.9 Навчання без нагляду

Неконтрольоване навчання – це тип машинного навчання, який не вимагає попередньо позначених чи класифікованих даних.

Несортована інформація групується за схожістю та відмінністю, навіть якщо категорії не надаються. Навчання без нагляду корисне для пошуку важливої інформації з даних, багато в чому схоже на те, як людина вчиться думати на основі власного досвіду, що робить його ближчим до справжнього ШІ.

Незважаючи на те, що неконтрольоване навчання може бути дуже корисним для задач, де є багато даних з складними взаємозв'язками та неоднорідністю, важливо пам'ятати про його чутливість до шуму в даних.

Існує два основних типи неконтрольованого навчання: кластеризація та асоціація. Цей метод корисний для аналізу даних про рух товарів, дозволяючи визначати популярні маршрути, частоту поставок та оптимальні логістичні рішення для різних регіонів.

2.10 Аналіз головних компонент

Аналіз головних компонент (PCA) – це статистична методика, яка виділяє ключові дані, що пояснюють весь набір, мінімізуючи обробку зайвої інформації. PCA зменшує розмірність набору даних, залишаючи лише кількісні змінні.

Алгоритм складається з п'яти кроків: нормалізація даних, створення коваріаційної матриці, обчислення власних векторів і значень, вибір головних компонентів, трансформація даних у новому вимірному просторі [7].

PCA передбачає лінійну комбінацію між змінними, де компоненти з найвищою дисперсією вважаються більш важливими, оскільки вони містять суттєву інформацію, тоді як компоненти з низькою дисперсією можуть містити шум.

Використання PCA перед кластеризацією допомагає зменшити розмірність даних, зберігаючи важливу інформацію, що полегшує процес кластеризації, особливо при роботі з великими наборами даних.

2.11 Кластеризація

Кластеризація – це тип неконтрольованого навчання, в якому завдання полягає в тому, щоб розділити сукупність на кілька груп таким чином, щоб точки даних в тих самих групах були більш схожими одна на одну, ніж точки даних в інших групах.

Метод може бути корисним для виокремлення різних груп товарів за їх характеристиками, такими як розмір, вага, температурний режим для перевезення, що допоможе підібрати оптимальний транспорт та маршрути для кожної групи.

K-means – це ітеративний процес призначення кожної точки даних групам, і повільно ці точки кластеризуються на основі схожих характеристик.

Ідея полягає в тому, щоб мінімізувати суму відстаней між точками даних і центроїдом кластера, щоб визначити правильну групу, до якої має належати кожна точка [8].

Алгоритм кластеризації починається з визначення кількості кластерів. Потім ініціалізуються центроїди, вибираючи випадкові точки даних. Далі точки даних призначаються найближчому кластеру, обчислюючи відстань між точкою і центроїдом за допомогою евклідової метрики, а кластер обирається на основі мінімальної відстані. Після цього обчислюються нові центри кластерів на основі призначених до них об'єктів. Ці кроки повторюються ітеративно, поки кластери не стабілізуються або не досягнуто максимальної кількості ітерацій.

Існує техніка ієрархічної кластеризації. Агломерація - об’єднання точок або кластерів для створення більших кластерів. Кожен з цих кластерів має більшу кількість точок. Агломерація працює таким чином (рис. 2.1): кожна точка спочатку розглядається як окремий кластер. Далі, найближчі точки групуються разом у кластер з двох точок. Потім процес продовжується з наступною найближчою точкою, яка приєднується до вже існуючого кластера. Ідеальною метою є досягнення двох кластерів, які розглядаються як єдине ціле. У крайньому випадку процес продовжується до тих пір, поки всі точки не об'єднаються в один великий кластер (рис. 2.2).

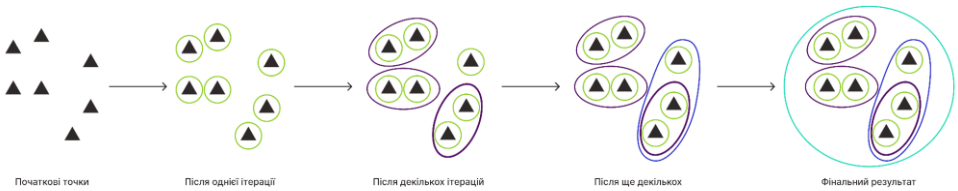

Рисунок 2.1 - Процес агломерації

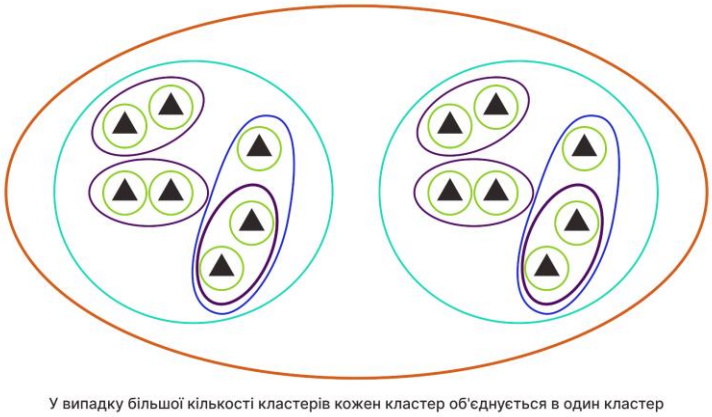

Рисунок 2.2 - Процес об’єднання кластерів

У випадку великомасштабних наборів даних, агломеративна (або) загальна ієрархічна кластеризація виходить з ладу, і дуже хорошою альтернативою є K-means.

2.12 Навчання за асоціативними правилами

Алгоритм підраховує частоту компліментарних входжень або асоціацій у дуже великому наборі даних із понад тисячами атрибутів.

Мета полягає в тому, щоб знайти асоціації, які відбуваються разом набагато частіше, ніж у випадковій вибірці можливостей. Для цього використовуються алгоритми, такі як Apriori (використовує метод генерації кандидатів та принцип апріорної властивості, шукає всі часті набори елементів, перевіряючи, чи є їх піднабори частими), ECLAT (використовує бітовий підхід та рекурсивний пошук для швидкого видобування частих наборів елементів. Не вимагає багаторазового проходження над даними, як Apriori) або FP-Growth (використовує дерево префіксів (FP-дерево) для ефективного зберігання інформації про часті набори елементів, працює в два проходи: перший - для побудови FP-дерева, а другий – для видобування частих

наборів елементів з цього дерева, може бути ефективним на великих наборах даних).

Для вимірювання зв'язків між тисячами елементів даних існує кілька показників: підтримка (використовується для знаходження частоти появи певного набору предметів у наборі даних), впевненість (яка ймовірність придбання товару Б при придбанні товару А), підйомна сила (наскільки ймовірно, що товар А буде придбаний, одночасно контролюючи, наскільки популярним є товар Б) [9].

Під час роботи з великими наборами даних може виникати проблема високої розмірності, коли кількість потенційних правил стає занадто великою, що ускладнює їх інтерпретацію. Вибір правильних значень мінімальної підтримки та мінімальної впевненості є критичним для отримання якісних результатів. Занадто високі значення можуть призвести до втрати важливих правил, а занадто низькі – до надлишкового шуму. Часто доцільно починати з вищих порогів і поступово їх знижувати.

3 РОЗВІДКОВИЙ АНАЛІЗ ДАНИХ

У цьому розділі розглядаються актуальні задачі, процес підготовки даних для їх подальшого використання у веб додатку, а також проводиться порівняння методів машинного навчання з метою оцінки їх результативності. Під час аналізу даних досліджуються їх специфічні особливості: структура, розмір, типи ознак, а також виявляються можливі аномалії та пропуски, щоб зрозуміти, які саме дані використовувати.

3.1 Актуальні задачі

Сьогодні сфера логістики потребує постійного удосконалення та інновацій. Зростаючі обсяги товарів і попит ускладнюють конкурентну боротьбу в цій галузі. Компаніям необхідно діяти оперативно, швидко адаптуватися до змін на ринку та впроваджувати нові технології для підвищення ефективності та задоволення потреб клієнтів. Впровадження автоматизації, використання аналітики даних та машинного навчання є ключовими для забезпечення конкурентних переваг у цьому швидко змінному середовищі.

Дана робота орієнтована на розробку інформаційно-програмних інструментів підтримки технологій машинного навчання в організаціях, що спеціалізуються на вирішенні логістичних завдань. Зокрема, такі інструменти можуть бути інтегровані як підсистема інформаційного аналізу в уже існуючу інноваційну онлайн-платформу SeaRates, яка надає розширені можливості для логістичного планування, пошуку та розрахунку вартості перевезення вантажів по всьому світу.

Наразі логістичний ринок знаходиться у складній ситуації, оскільки на його динаміку впливають геополітичні відносини, зміни в торговельних відносинах між країнами та зростання обсягів торгівлі через електронну

комерцію. Для забезпечення ефективної роботи компанії вкрай важливо визначити пріоритети моніторингу попиту та приділяти увагу ціноутворенню.

Сировина, фрахт, робоча сила та енергія зросли в ціні у всьому світі, що змушує компанії приділяти більше уваги контролю витрат, якщо вони хочуть переконатися у відсутності збоїв у своїх операціях та процесах. Це підштовхує власників бізнесу шукати способи заощадити гроші [10].

Використовуючи тарифи та дані про запити, компанія може не лише встановлювати оптимальні ціни на свої транспортні послуги, але й стратегічно планувати свій бюджет на логістичні операції, що сприятиме розвитку та покращенню конкурентної ситуації на ринку.

Прогнозування попиту на конкретні маршрути та типи контейнерів дозволяє планувати ланцюги постачання та розміри запасів, уникати затримок у поставках та забезпечувати належний рівень обслуговування.

Крім того, оцінюючи взаємозв’язок між типами контейнерів, характеристиками вантажу та відповідними тарифами, компанія зможе оптимізувати процес вибору контейнерів, щоб максимізувати ефективність і мінімізувати витрати. Ця оптимізація не тільки підвищить ефективність роботи, але й посприяє загальній економії витрат і задоволенню клієнтів.

Ці знання становлять велику цінність, допомагаючи спрямовувати ресурси логістичної компанії на ключові напрямки та скорочувати час доставки. Проведений аналіз даних допомагає виявляти слабкі моменти у поточних бізнес-процесах та знайти шляхи їх вдосконалення, а також стати переможцями у конкурентній боротьбі і надавати більш персоналізовані послуги.

3.2 Аналіз та підготовка наборів даних

Для вирішення поставлених задач на покращення процесів існуючої логістичної компанії отримано чотири набори даних:

− leads (запити на перевезення);

− тарифи на перевезення rates_sea за морським напрямком;

− тарифи на перевезення rates_land за наземним напрямком;

− тарифи на перевезення rates_air за повітряним напрямком.

Набір даних leads використовується для відстеження інформації про вантажі, їх характеристики, маршрути, характеристики грузу, типи доставки та інше.

Набори даних rates_sea, rates_land та rates_air містять інформацію про угоди з перевезення вантажів, включаючи дати, перевізників, очікувану тривалість перевезення, місця походження та призначення, типи і кількості вантажу та ціни за перевезення. Вони використовується для планування та відстеження логістики перевезення вантажів в залежності від видів перевезень: морських, наземних чи повітряних.

Структура вищезазначених таблиць представлена на ER-діаграмі нижче (рис. 3.1). Ця діаграма дає змогу проаналізувати взаємозв’язки між параметрами у наборах даних.

Рисунок 3.1 ER-діаграма

У компанії є три відділи (морські, наземні та авіаперевезення) та різні застосунки, для яких дані зберігаються окремо. Такий підхід дозволяє кожному відділу та застосунку зосередитися на своєму конкретному функціоналі і ефективно виконувати свої завдання.

Кожен з наборів даних містить інформацію незалежно від іншого, тому не можна провести зв’язків між таблицями, проте можливо зробити деякі припущення, що набір даних leads пов’язаний з іншими наборами і їх можна об’єднати за допомогою декартового з’єднання, використовуючи атрибути port_from_country, port_to_country та container_type.

Завдяки такому рішенню стає можливим отримати тарифи перевезень price_usd в залежності від ваги, типу контейнера, часу перевезення, місця звідки та куди. Змінюючи вхідні параметри, можна отримати на виході інформацію щодо приблизних характеристик перевезення в залежності від попиту.

Якщо б при проектуванні бази даних у компанії, в набори даних sea, air та land були додані зовнішні ключі на таблицю leads, стало б можливим зв’язати ці таблиці відношенням 1:1.

3.2.1 Поєднання наборів даних

Для зрозумілості і зручності було перейменновано деякі ознаки. Наприклад, 'load_type' у 'container_type', 'origin_country' у 'port_from_country', 'destination_country' у 'port_to_country'.

Набори даних: ‘запити на перевезення’ і ‘тарифи перевезень’ об’єднано між собою за стовпцями 'port_from_country', 'port_to_country' та 'container_type' з використанням внутрішнього об'єднання. У результаті отримано тільки ті рядки, де збігаються значення у з'єднуваних стовпцях обох таблиць.

Виявлено, що вартості у наборі даних присутні тільки за наземні перевезення, для випадків морських і авіа багато пропущених значень.

Виведено статистичні показники для колонки вартості (рис. 3.2). Вони включають у себе: кількість непустих значень у колонці, середнє значення колонки, стандартне відхилення (ступінь розкиду значень щодо середнього значення), мінімальне значення ціни, значення, нижче за які знаходяться 25%, 50%, 75% значень в стовпці, максимальне значення. Більшість цін (50% - 75%) знаходяться у діапазоні від 9150 до 9900 доларів, що вказує на те, що більшість перевезень мають середню або високу вартість.

```
count      20.000000
mean     9255.000000
std       917.935326
min      7450.000000
25%      9150.000000
50%      9750.000000
75%      9900.000000
max      9900.000000
Name: price_usd, dtype: float64
```

Рисунок 3.2 Статистичні показники для колонки ціни

Розглянуто структуру та типи даних у стовпцях. Виявлено нульові значення в деяких з них. Для основного аналізу та побудови моделей використано частину набору даних з непорожніми значеннями.

3.2.2 Категоріальні та числові ознаки

За допомогою функції, що генерує мета-інформацію, отримано інформацію щодо кількості категоріальних та числових ознак, а також кількості пропущених значень серед них.

Для зручності у таблицю додано новий стовпчик, що виводить відсоток пропущених значень для кожної ознаки у відношенні до загальної кількості рядків. Дані для ознаки ціни показують, що вона не має жодних пропущених значень (відсоток становить 0%). Більшість ознак є категоріальними.

Стовпці з повністю пропущеними значеннями або ті, де пропущені значення перевищують 50% від загальної кількості рядків, стовпці, які мають тільки одне унікальне значення та ті, де всі значення однакові видалено. Це виключає випадки роботи з одноманітними і неінформативними даними.

Пропущені значення у категоріальних колонках заповнено значеннями ‘missing`. Це зроблено замість видалення, для збереження інформації, яка може стати у нагоді під час роботи.

Обчислено кількість унікальних значень у кожній категоріальній колонці. Коли така кількість менше 10, створено графік, що показує кількість кожного унікального значення у колонці. Коли більше або дорівнює 10, обирається топ-5 найчастіших значень, підраховується кількість значень, що не увійшли туди, виводиться графік і загальна кількість унікальних значень у колонці.

З графіку, побудованого за допомогою бібліотеки Seaborn, можна зробити висновок, що найбільш використовуваною одиницею ваги є метричні тони, з більш ніж 3000 значень. Серед типів перевезень найпопулярнішим є FCL (повністю завантажений контейнер), що свідчить про те, що клієнтами є переважно великі імпортери та торгові мережі. Більше 1550 значень припадає на 20-футові контейнери (рис. 3.3). Ці дані допоможуть під час прогнозування попиту.

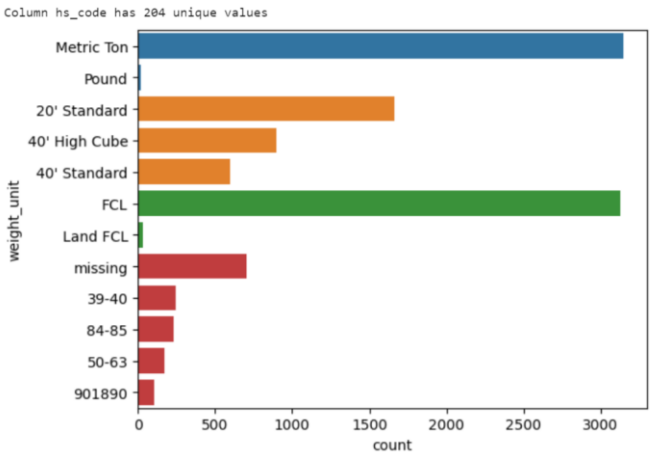

Рисунок 3.3 - Графік відношення ваги до кількості перевезень

Китай – найбільш популярна країна, звідки відправляють грузи. На другому і третьому місцях знаходяться Японія та США відповідно (рис. 3.4).

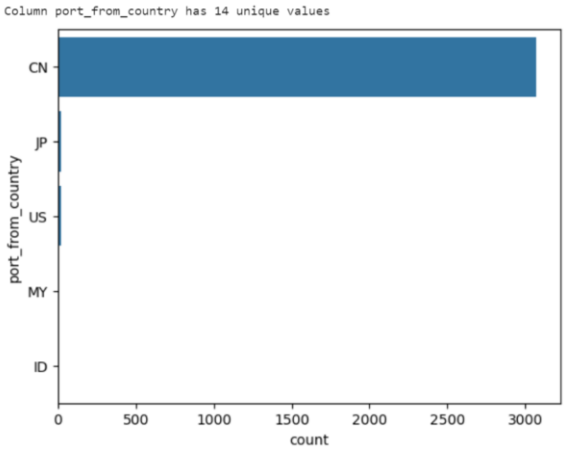

Рисунок 3.4 – Графік відношення країн, з яких відправляється вантаж до кількості перевезень

Найпопулярнішим портом є китайський Schenzhen Shi, а містом - Chiwan (рис. 3.5). Також, на жаль, датасет містить багато пустих значень, тому статистику не можна вважати повною.

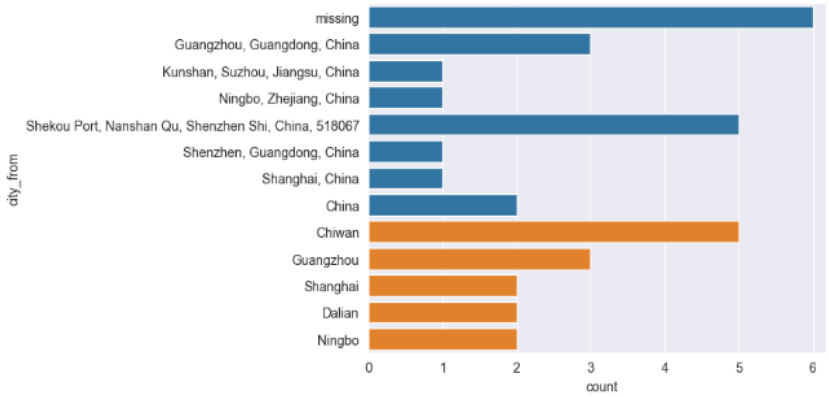

Рисунок 3.5 – Графік відношення міст і портів, з яких відправляється вантаж наземних перевезень до кількості перевезень

З даного графіка видно, що найбільш популярні міста отримання вантажа наземних перевезень переважно знаходяться у Європі (рис. 3.6).

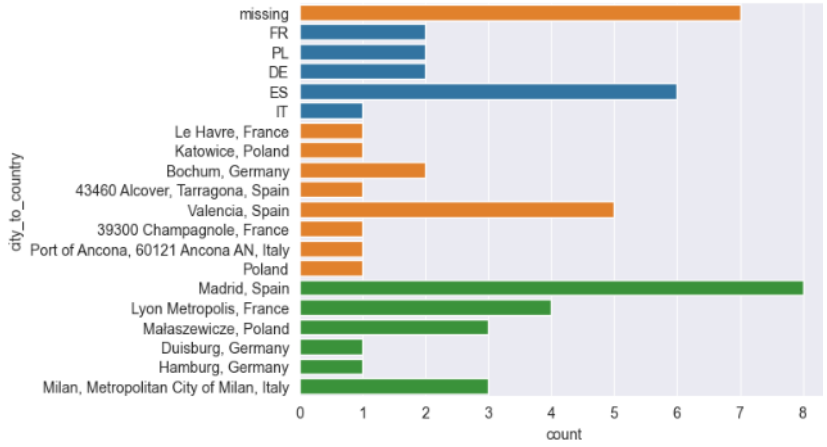

Рисунок 3.6 – Графік відношення країн і міст, в які відправляється вантаж наземних перевезень до кількості перевезень

Можна виділити три провідні перевізники, які займають лідируючі позиції з невеликим відривом: DFA, ONE, PIL та OOCL (рис. 3.7).

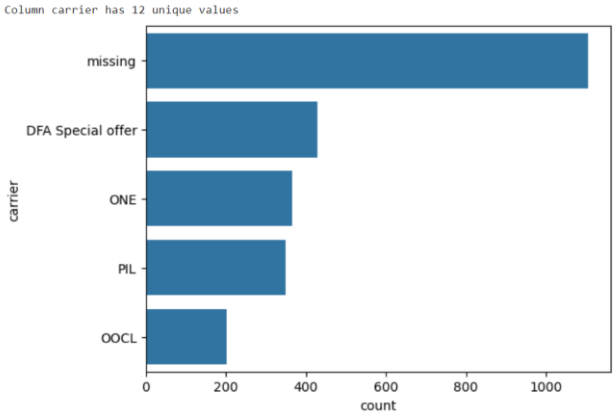

Рисунок 3.7 – Графік відношення перевізників до кількості перевезень

Китайські міста продовжують лідирувати серед місць походження товарів (рис. 3.8).

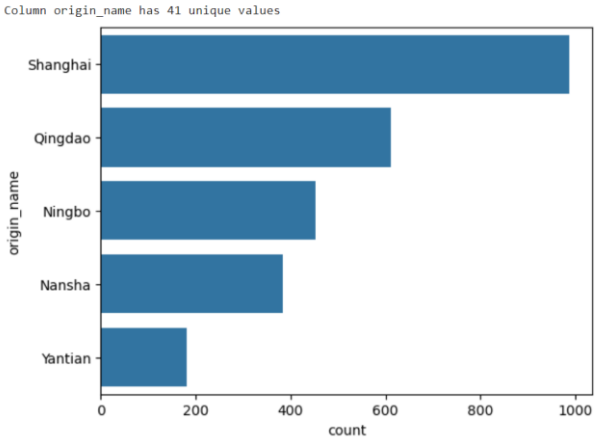

Рисунок 3.8 – Графік відношення міст походження до кількості перевезень

Для всіх числових колонок побудовано гістограми. Так як розглядається питання ціноутворення, розглянуто статистику по цінах за всіма типами перевезень. price_usd - загальна сума перевезення, count - кількість значень у наборі. Більшість перевезень попадають у ціновий діапазон від 2000 до 4000 доларів США, що вказує на стандартні тарифи для більшості відправлень. З іншого боку, перевезень з вартістю понад 4000 доларів США є значно менше. Це пояснюється тим, що такі перевезення можуть бути спеціалізованими та

включати цінний або великогабаритний вантаж, вимагати особливих умов перевезення, що робить їх менш поширеними порівняно зі стандартними відправленнями (рис. 3.9).

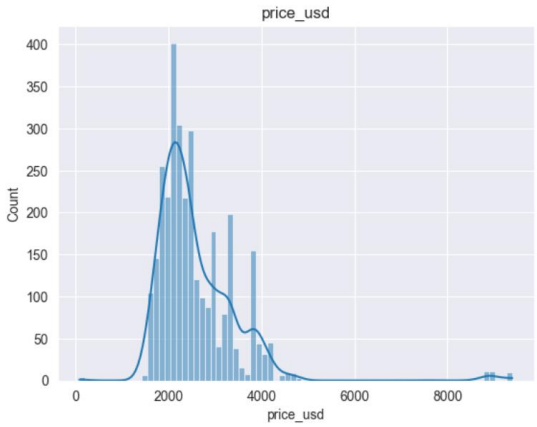

Рисунок 3.9 – Графік відношення вартості перевезення до кількості перевезень

Якщо розглянути наземні перевезення окремо, видно, що максимальна сума за перевезення не перевищує 9900 доларів США. Ціни на перевезення у діапазоні між 8300-9100 доларів США відсутні (рис. 3.10).

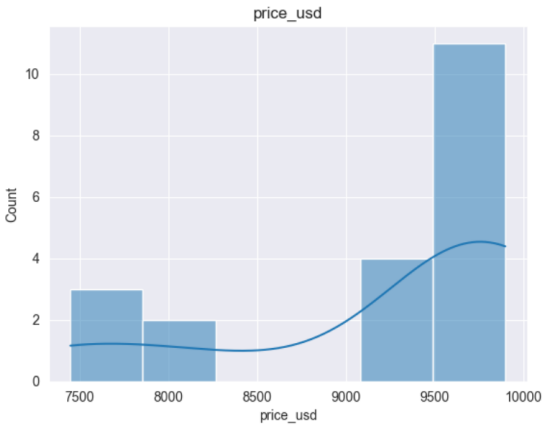

Рисунок 3.10 – Графік відношення вартості наземних перевезень до кількості перевезень

З графіку залежності ваги вантажу від кількості перевезень зрозуміло, що частіше за все перевозять вантажі вагою в діапазонах від 10 до 12.5 та від 20 до 25.5 метричних тон (рис. 3.11).

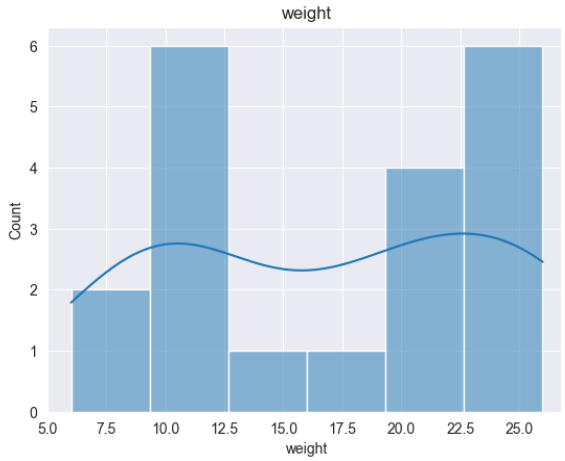

Рисунок 3.11 – Графік відношення ваги вантажу наземних перевезень до кількості перевезень

Valid days (дійсні дні) - період часу здійснення перевезення. Доставка за наземними перевезеннями найчастіше займає від 5 днів до тижня (рис.3.12).

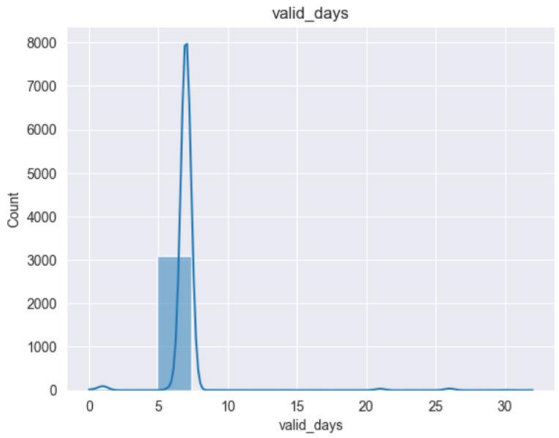

Рисунок 3.12 – Графік відношення дійсних днів перевезення до кількості перевезень

3.3 Вирішення задачі ціноутворення

Задача полягає в тому, щоб на основі тарифів і даних про запити спрогнозувати ціни для різних видів перевезень.

3.3.1 Підготовка даних для навчання моделі

Для вирішення цього завдання спочатку потрібно підготувати дані для навчання моделі. Розділено набір даних на ознаки та цільову зміну. Обрано price_usd - ціна у якості цільової, яку необхідно передбачити. Всі інші стовпці використовуються як ознаки для побудови моделі.

Важливою частиною підготовки є нормалізація даних, особливо коли справа з ознаками різного масштабу та розподілу. Нормалізація даних - це процес масштабування ознак так, щоб вони мали однаковий діапазон значень та ваговий вплив на модель. Обраним методом нормалізації є мінімаксне масштабування, яке приводить всі значення ознак до діапазону від 0 до 1.

Далі дані розбиваються на тренувальний та тестовий набори за допомогою функції train_test_split. Це дозволяє оцінити ефективність моделі на даних, які вона раніше не бачила. Параметр test_size=0.2 визначає розмір тестового набору та складатиме 20% від загальної кількості даних. Тоді як тренувальний набір 80%.

Перевірено тип даних кожної ознаки. Відокремлено цільову ознаку з числових змінних. Нормалізуємо числові ознаки. Обчислено кореляційну матрицю для всіх отриманих ознак у тренувальному наборі даних. Кореляційна матриця показує, як сильно залежать одна від одної різні ознаки у наборі. Значення кореляції можуть бути від -1 до 1, де значення близьке до 1 вказує на позитивну кореляцію, значення близьке до -1 вказує на негативну кореляцію, а значення близьке до 0 вказує на відсутність кореляції.

За допомогою бібліотеки Seaborn створено теплокарту, що візуалізує матрицю (рис. 3.13). У ній розглянуто тільки дві змінні - кількість та вага, так

як інші числові параметри видалено через їх однаковість. Кореляція 0.37 вказує на помірну позитивну лінійну залежність між ними - зміна однієї змінної пов'язана з помірною зміною в іншій змінній в тому ж напрямку. Іншими словами, коли значення однієї змінної зростають, значення іншої змінної також зазвичай зростають, і навпаки. Хоча кореляція 0.37 є позитивною, вона вважається слабкою. Це означає, що зміни в одній змінній слабко передбачають зміни в іншій.

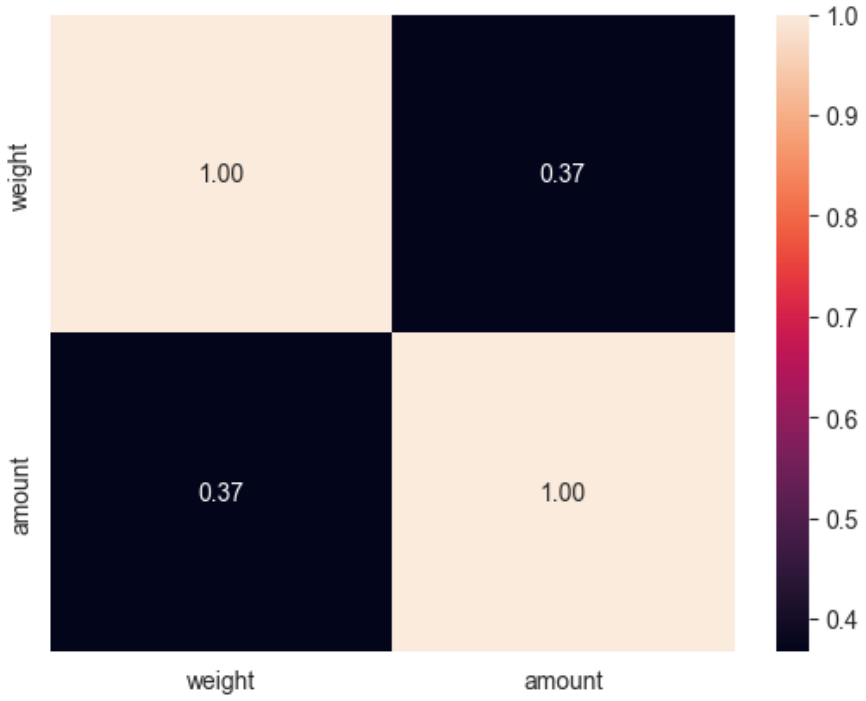

Рисунок 3.13 – Теплокарта кореляції за наземними перевезеннями

3.3.2 Тест Хі-квадрат

Для кожної унікальної пари категоріальних ознак обчислено таблицю спряженості, що дозволяє підрахувати кількість спостережень, які належать до кожної можливої комбінації категоріальних значень.

Для кожної таблиці спряженості, в свою чергу, виконано тест Хі-квадрат, що дозволив визначити, чи існує статистично значима залежність між категоріальними ознаками [11].

Результати тесту: статистика Хі-квадрат, Р-значення, ступені свободи та показник статистичної значимості зберігаються у словнику, де ключі - це пари назв категоріальних ознак.

У контексті статистичного тесту Хі-квадрат високе Р-значення свідчить про те, що немає достатніх доказів для заперечення нульової гіпотези. Нульова гіпотеза передбачає відсутність статистичного зв'язку між змінними. Таким чином, якщо Р-значення високе (зазвичай ближче до 1), це вказує на те, що немає статистично значущого зв'язку між змінними.

Більша кількість ступенів свободи зазвичай призводить до точніших результатів аналізу.

Приклади результатів:

Результат port_from vs port_to_country:

χ^2: 45.71428571428572, P Value: 0.05506266186436665, Degrees of Freedom: 32, Significant: False

Значення χ^2 (45.71) вказує на певну різницю між спостережуваними та очікуваними частотами для комбінацій "port_from" та "port_to_country". Однак, Р-значення (0.055) перевищує поріг значущості 0.05, що означає, що результати не є статистично значущими. Таким чином, неможливо зробити висновок про існування статистично значущої залежності між "port_from" та "port_to_country". Це означає, що вибір порту відправлення не має суттєвого впливу на країну порту призначення і навпаки.

Результат city_from vs city_to:

χ^2: 81.27999999999999, P Value: 2.3724543419327608e-05, Degrees of Freedom: 36, Significant: True

Високе значення χ^2 (81.28) вказує на суттєву різницю між спостережуваними та очікуваними частотами для комбінацій "city_from" та "city_to". Дуже низьке Р-значення (0.0000237) підтверджує, що ймовірність

випадкового виникнення такої великої різниці надзвичайно мала. Це означає, що вибір міста відправлення має вплив на вибір міста призначення і навпаки.

Результат request_date vs city_to:

Chi2 Stat: 96.0, P Value: 0.0021782051808705116, Degrees of Freedom: 60, Significant: True

Високе значення χ^2 (96.0) вказує на суттєву різницю між спостережуваними та очікуваними частотами для комбінацій "request_date" та "city_to". Дуже низьке P-значення (0.002178) підтверджує, що ймовірність випадкового виникнення такої великої різниці дуже мала.

Дата запиту (request_date) та місто призначення (city_to) не є незалежними, і є взаємозв'язок між цими двома змінними. Це може означати, що певні дати запитів корелюють з певними містами призначення, що може бути важливим для планування та оптимізації логістичних процесів.

За допомогою цього тесту виявлено, що залежності між портами відправлення і призначення, а також між портами та містами свідчать про тісний взаємозв'язок у виборі логістичних маршрутів через певні порти. Це може бути використано для оптимізації маршрутів та поліпшення ефективності доставки.

Залежність між датою запиту і містом призначення може вказувати на сезонні або тимчасові тренди у запитах, що допоможе у прогнозуванні попиту і плануванні пікових періодів. Існує значуща залежність між кодом HS та портом призначення і містом призначення. Це вказує на те, що певні товари частіше доставляються до конкретних портів або міст.

3.3.3 Порівняння алгоритмів машинного навчання для прогнозування

Для автоматизації процесу вибору найбільш ефективного методу машинного навчання використано підхід порівняння алгоритмів (функція `compare_algorithms`). Функція приймає на вхід датафрейм, назву стовпця з

цільовою змінною, розмір тестового набору у відсотках (за замовчуванням 20%), параметр для генерації випадкових чисел, параметр для сортування результатів порівняння (R2 Score - коефіцієнт детермінації) та порогове значення кардинальності для визначення категоріальних змінних.

Дані поділяються на числові та категоріальні ознаки.

Для числових ознак застосовується імпутація (заповнення) медіанними значеннями (центральне значення колонки), що дозволяє уникнути викривлення даних, яке може виникнути при використанні середнього (mean), особливо якщо дані містять викиди (outliers). Після імпутації пропущених значень, числові дані стандартизуються - для кожного значення віднімається середнє значення колонки, а потім ділиться на стандартне відхилення колонки, що призводить до того, що всі числові дані мають середнє значення 0 і стандартне відхилення 1.

Стандартизація допомагає моделям краще навчатися, оскільки приводить всі числові характеристики до одного масштабу.

Для колонок з низькою кардинальністю (тобто, колонка, що має невелику кількість унікальних значень), кожне значення цієї колонки перетворюється в окрему бінарну (0 або 1).

Використання параметра handle_unknown='ignore' дозволяє ігнорувати невідомі категорії під час трансформації нових даних, що важливо для забезпечення стабільності роботи моделі на нових даних.

Для категоріальних змінних пропущені дані заповнюються константою 'missing' для того, щоб розрізняти випадки, де дані були відсутні, а також це є важливим, оскільки більшість алгоритмів машинного навчання не можуть обробляти пропущені значення.

За допомогою частотного кодування категоріальні дані перетворено на числові, замінюючи кожне значення категорії частотою його появи в колонці. Частотне кодування зменшує розмірність даних і дозволяє моделям ефективніше працювати з категоричними даними. Це особливо важливо для колонок з великою кількістю унікальних значень (високою кардинальністю).

На підготовлених даних використано різні моделі машинного навчання, де price_usd є цільовою колонкою. Лінійні моделі - LinearRegression, Ridge, Lasso. Дерева прийняття рішень - DecisionTreeRegressor, RandomForestRegressor. Моделі підсилення - XGBRegressor, AdaBoostRegressor, GradientBoostingRegressor.

Після навчання моделей обчислюються ключові метрики продуктивності, такі як середньоквадратична помилка (MSE), середня абсолютна помилка (MAE) та коефіцієнт детермінації (R2 Score).

Ці метрики дозволяють оцінити якість моделей та порівняти їх між собою (рис. 3.14).

	Model	MSE	MAE	R2 Score
4	DecisionTreeRegressor	0.000000e+00	0.000000	1.000000
0	LinearRegression	4.569377e-07	0.000568	1.000000
5	XGBRegressor	1.239777e-06	0.001074	1.000000
7	GradientBoostingRegressor	9.660988e-04	0.027511	1.000000
2	Lasso	2.612017e+01	4.000246	0.999967
6	AdaBoostRegressor	6.795918e+01	2.571429	0.999915
1	Ridge	7.519275e+03	62.929633	0.990606
3	RandomForestRegressor	2.706361e+04	91.175000	0.966191

Рисунок 3.14 - Порівняння методів машинного навчання

Оцінка MSE – середнє значення квадратів різниці між прогнозованими значеннями та справжніми значеннями цільової змінної, що вимірює середньо квадратичну помилку моделі за формулою

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(yi - \widehat{yi})^2 \quad (3.1)$$

де n – кількість спостережень;

yi – фактичне значення;

$\widehat{yi}$ – передбачене значення.

MAE – середнє значення абсолютних різниць між прогнозованими значеннями та справжніми значеннями цільової змінної. Вимірює середню абсолютну помилку моделі за формулою

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} | yi - \widehat{yi} | \quad (3.2)$$

де n – кількість оцінок;

yi – фактична оцінка;

$\widehat{yi}$ – передбачена оцінка.

Чим менше значення MSE та MAE, тим краще модель.

R2 Score – вимірює пропорцію дисперсії у вихідних даних, яка пояснюється моделлю. Це значення знаходиться в межах від 0 до 1, де 1 вказує на ідеальну узгодженість моделі з даними, а 0 показує відсутність узгодженості. Коефіцієнт детермінації вимірюється за формулою

$$R^2 = 1 - \sum_{i=1}^{n}\frac{(yi - \widehat{yi})^2}{(yi - \overline{yi})^2} \quad (3.3)$$

де n – кількість спостережень;

yi – фактичне значення;

$\widehat{yi}$ – передбачене значення;

$\overline{yi}$ – середнє значення фактичних значень.

DecisionTreeRegressor, LinearRegression, XGBRegressor, GradientBoostingRegressor: R2 Score = 1.000000, що свідчить про те, що ці моделі показують ідеальний коефіцієнт детермінації і вказує на те, що вони

майже точно передбачають цільову змінну. MSE та MAE дуже малі, що також підтверджує, що помилки моделей майже відсутні.

Але такі результати можуть свідчити про перенавчання, коли модель занадто добре підлаштовується під тренувальні дані і не буде добре працювати на нових. У Lasso, AdaBoostRegressor R2 Score близьке до 1 (0.999967, 0.999915), що вказує на дуже високу точність моделей. MSE та MAE більші, ніж у перших чотирьох моделей, але все ще дуже малі.

Моделі добре підходять до даних, але можливо, з меншою ймовірністю перенавчання порівняно з першими чотирма моделями. Ridge та RandomForestRegressor показують добрі результати, але не настільки, як інші моделі. Це може свідчити про те, що вони можуть краще узагальнюватися на нових даних.

4 ОПИС МОЖЛИВОСТЕЙ СИСТЕМИ

У даному розділі представлено інструменти реалізації, застосування різних методів машинного навчання на кожному з видів перевезень та їх аналіз. Також розглянуто можливості покращення результатів.

4.1 Інструменти реалізації

Для демонстрації роботи методів машинного навчання створено веб-додаток за допомогою фреймворку Streamlit.

Streamlit – це безкоштовний фреймворк Python з відкритим вихідним кодом, який спрощує демонстрацію моделей машинного навчання та створення інтерактивних візуалізацій результатів аналізу або прогнозів [12]. Він підтримує різні типи даних, включаючи текст, зображення, аудіо та відео, і легко інтегрується з багатьма популярними бібліотеками Python, такими як Pandas, NumPy, Plotly, NetworkX, mlxtend та Scikit-learn.

Streamlit програми мають клієнт-серверну архітектуру. Серверна частина програми написана на Python. Інтерфейс користувача, який переглядається через браузер, виступає в ролі клієнта. Коли програма розробляється локально, комп’ютер одночасно виконує роль і сервера, і клієнта, а якщо перегляд програми через локальну мережу або Інтернет, сервер і клієнт працюють на різних машинах.

За допомогою команди `streamlit run your_app.py` сервер Streamlit стартує на комп'ютері і виконує всі обчислення для користувачів, які відкривають програму. Незалежно від того, чи переглядають користувачі програму локально або через Інтернет, сервер завжди працює на тій машині, де була запущена команда, і ця машина називається хостом. Хост відповідає за обчислення та пам'ять, необхідні для роботи програми для всіх

користувачів, і повинен мати достатню потужність для підтримки одночасних з'єднань.

Конкурентами Streamlit є Flask, Django, Dash і FastAPI. Streamlit відрізняється своєю простотою та інтуїтивним інтерфейсом, що дозволяє швидко створювати інтерактивні додатки. Він автоматично оновлює додатки при зміні коду Python, що полегшує процес розробки та тестування, а активне співтовариство користувачів та розробників надає підтримку і готові рішення.

За допомогою бібліотеки Pandas виконується обробка та аналіз даних. Вона надає інструменти для роботи з таблицями даних, фільтрації, об'єднання та зміни їх. NumPy використовується для ідентифікації числових типів даних.

Бібліотека Mlxtend використовується для аналізу правил асоціації. Plotly дозволяє створювати інтерактивні графіки та візуалізації, а NetworkX - аналізувати та візуалізувати графи. У методі Association_Rule за допомогою цієї бібліотеки побудовано граф з вузлами антецеденту та консеквенту з вказаною мірою lift та confidence.

Бібліотека Scikit-learn, також відома як sklearn, включає широкий спектр алгоритмів машинного навчання, які можна застосовувати для класифікації, регресії, кластеризації та виявлення аномалій [13]. Також надає різні утиліти для попередньої обробки та підготовки даних перед їх використанням для навчання моделей. Ці утиліти включають в себе методи масштабування, кодування категоріальних змінних, видалення відсутніх значень, вибір ознак. Scikit-learn надає різні інструменти для оцінки моделей, такі як метрики ефективності, методи перехресної перевірки та різні стратегії розділення даних на навчальний та тестовий набори.

4.2 Аналіз даних

При запуску веб-додатку відкривається початкова сторінка, яка надає зручний інтерфейс для перегляду та аналізу наборів даних (рис. 4.1).

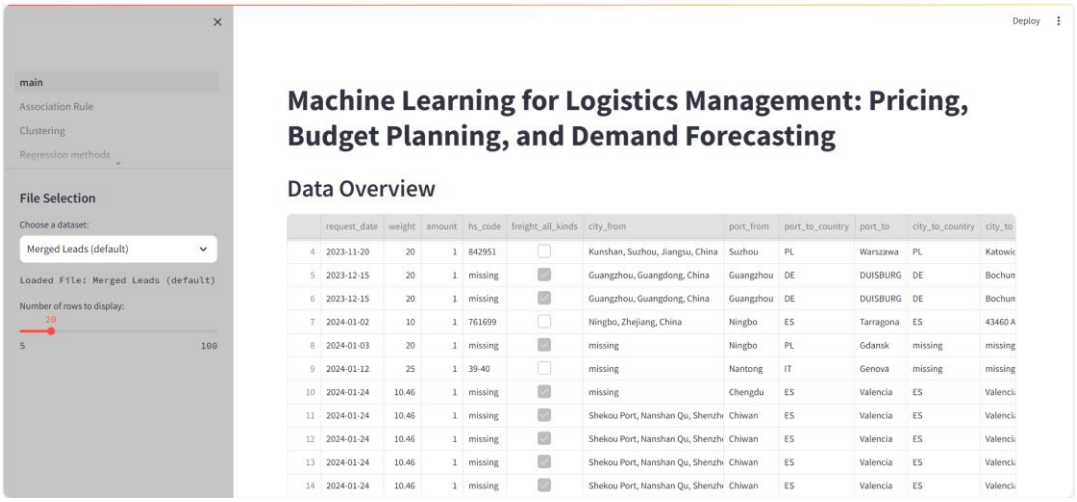

Рисунок 4.1 – Загальний вигляд додатку

Програмну реалізацію цього блоку наведено в додатку А.

За результатами об'єднання Leads (запроси на перевезення) з кожним з наборів Rates за такими параметрами як: 'port_from_country', 'port_to_country', 'container_type', виявилося, що тільки Land Rates містить у собі відповідні рядки. Це дозволило створити Merged Leads, оптимізований від дублікатів та записів з пропущеними даними.

Користувачу надається можливість обрати в залежності від видів перевезень (морських – Sea Rates, наземних – Land Rates, повітряних – Air Rates чи за замовчуванням – Merged Leads) необхідний датасет для аналізу (рис. 4.2).

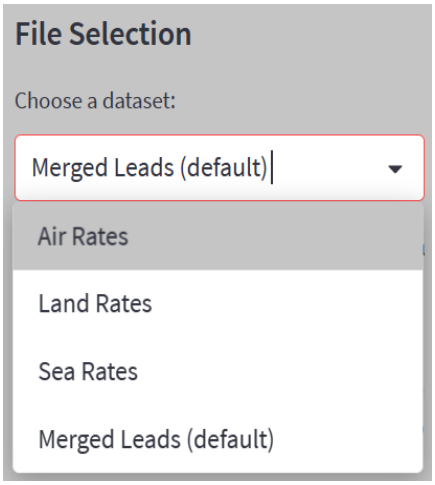

Рисунок 4.2 – Вибір набору даних

Дані конкретного набору даних представляються у вигляді таблиці, наприклад для набору даних Merged Leads (рис. 4.3).

	request_date	weight	amount	hs_code	freight_all_kinds	city_from	port_from	port_to_country	port_to	city_to_country	city_to	destination_
0	2023-09-29	25	1	94-96	☐	missing	Shanghai	ES	Barcelona	missing	missing	Madrid, Spai
1	2023-10-07	6	1	84-85	☐	Guangzhou, Guangdong, China	Guangzhou	FR	Le Havre	FR	Le Havre, France	Lyon Metrop
2	2023-10-25	7	1	482369	☐	missing	Dalian	FR	Le Havre	missing	missing	Lyon Metrop
3	2023-11-01	24	1	392062	☐	missing	Dalian	ES	Barcelona	missing	missing	Madrid, Spai
4	2023-11-20	20	1	842951	☐	Kunshan, Suzhou, Jiangsu, China	Suzhou	PL	Warszawa	PL	Katowice, Poland	Małaszewicz
5	2023-12-15	20	1	missing	☑	Guangzhou, Guangdong, China	Guangzhou	DE	DUISBURG	DE	Bochum, Germany	Duisburg, Ge
6	2023-12-15	20	1	missing	☑	Guangzhou, Guangdong, China	Guangzhou	DE	DUISBURG	DE	Bochum, Germany	Hamburg, Ge
7	2024-01-02	10	1	761699	☐	Ningbo, Zhejiang, China	Ningbo	ES	Tarragona	ES	43460 Alcover, Tarra	Madrid, Spai
8	2024-01-03	20	1	missing	☑	missing	Ningbo	PL	Gdansk	missing	missing	Małaszewicz
9	2024-01-12	25	1	39-40	☐	missing	Nantong	IT	Genova	missing	missing	Milan, Metro
10	2024-01-24	10.46	1	missing	☑	missing	Chengdu	ES	Valencia	ES	Valencia, Spain	Madrid, Spai
11	2024-01-24	10.46	1	missing	☑	Shekou Port, Nanshan Qu, Shenzh	Chiwan	ES	Valencia	ES	Valencia, Spain	Madrid, Spai
12	2024-01-24	10.46	1	missing	☑	Shekou Port, Nanshan Qu, Shenzh	Chiwan	ES	Valencia	ES	Valencia, Spain	Madrid, Spai

Рисунок 4.3 – Вивід таблиці набору даних Merged Leads у веб-додатку

Представлені основні характеристики вибраного набору даних: кількість рядків, кількість стовпців, загальна кількість пропущених значень, кількість дублікатів.

У розділі ‘Діаграма інтерактивних даних’, де візуалізуються залежності змінних між собою, є можливість отримати один з трьох типів графіків:

розсіювальний та лінійний для числових, а стовпчиковий для категоріальних змінних.

Наприклад, проаналізуємо залежність вартості та розрахункових днів (estimated_days) набору даних Sea_Rate (рис. 4.4). Це числові параметри, тому побудовано графік розсіювання. Можна зробити висновок, що між вартістю і розрахунковими днями не завжди прямо пропорційна залежність. Є випадки, коли значення розрахункових днів невелике, але вартість висока і навпаки.

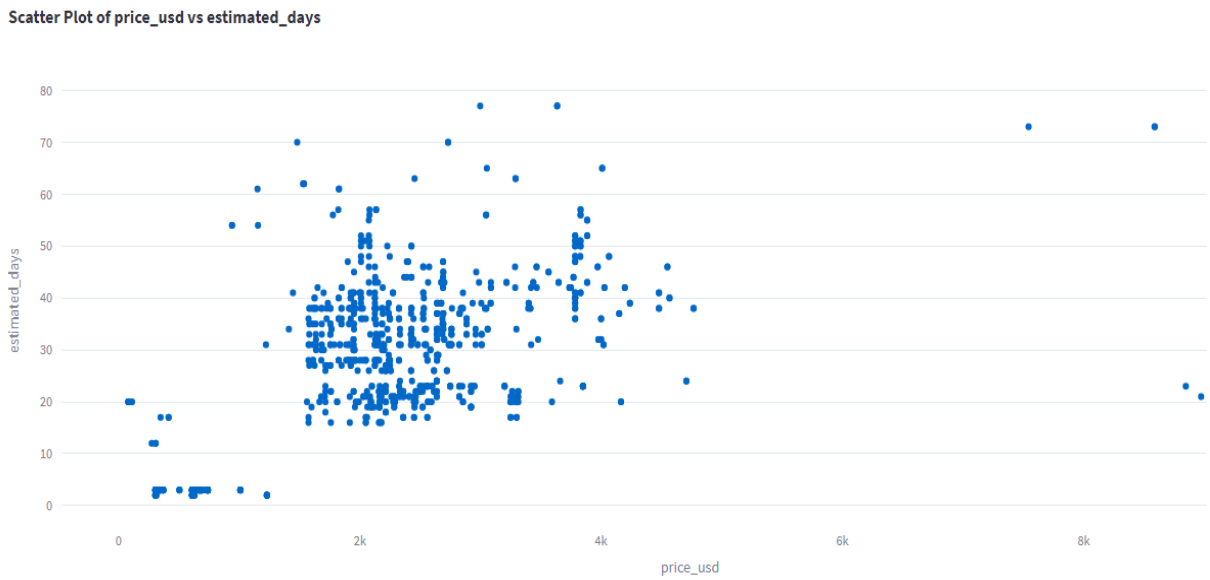

Рисунок 4.4 – Графік залежності вартості від розрахункових днів набору даних Sea_Rates

З набору даних Air_Rates наглядно ілюструється частота застосування різних авіаперевізників. Найчастіше перевезення здійснюють американські, а друге і третє місце посідають бразильські та латиноамериканські авіалінії (рис.4.5).

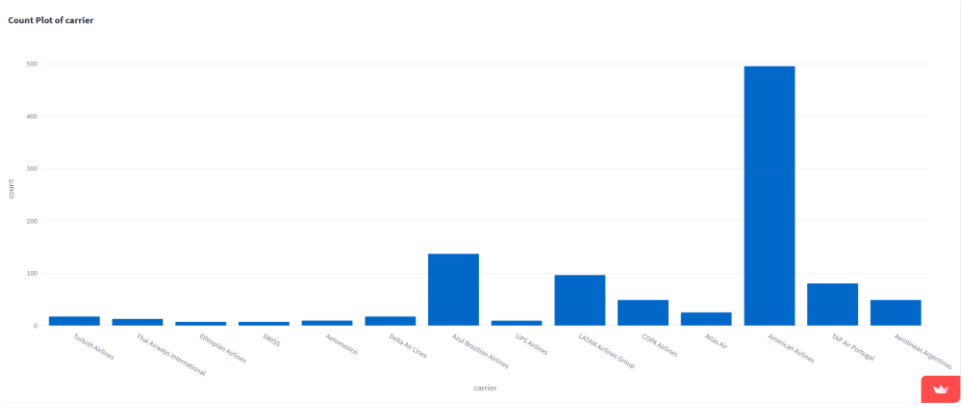

Рисунок 4.5 – Графік частоти застосування різних авіаперевізників набору даних Air_Rates

Морські перевезення здійснюються за допомогою трьох типів контейнерів. Вартості перевезень у контейнерах типу стандартний високий куб 40 футів та 40 футів стандарт - однакові, навіть, не дивлячись на те, що перший пропонує більше простору для вантажу.

Для числових змінних з набору даних можна побудувати гістограму. Наприклад, розглянемо кількісну характеристику за вартістю перевезення (рис. 4.6).

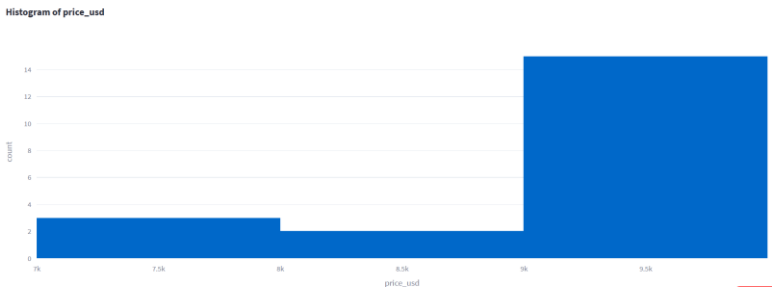

Рисунок 4.6 – Гістограма вартості перевезення набору даних Merged Leads

Найчастіше наземні перевезення здійснюються вартістю від 9000 до 9900 доларів США, що є доволі високим показником. Це свідчить про те, що перевезення відбуваються на великі відстані та включають транспортування промислового і електротехнічного устаткування. Наприклад, Нінбо (Китай) –

Таррагона (Іспанія) перевозиться вантаж з HS-кодом 761699, що свідчить про те, що це виріб з алюмінію.

Для категоріальних змінних будується кругова діаграма, якщо обраний стовпець містить у собі менше 10 унікальних категорій. Діаграма побудована за допомогою бібліотеки Plotly Express. В іншому випадку виводиться повідомлення, що обраний стовпець має занадто багато унікальних категорій для кругової діаграми. Наприклад, діаграма за даними Merged Leads city_to демонструє те, що цей стовпець містить 35% пропущених значень і може призвести до неправильних висновків або моделей, які недостатньо точно відображають реальність (рис. 4.7). Великий сегмент, що складає 25% від загальної кількості даних, з пунктом призначення Валенсія (Іспанія). Усі міста зосереджені в Європі.

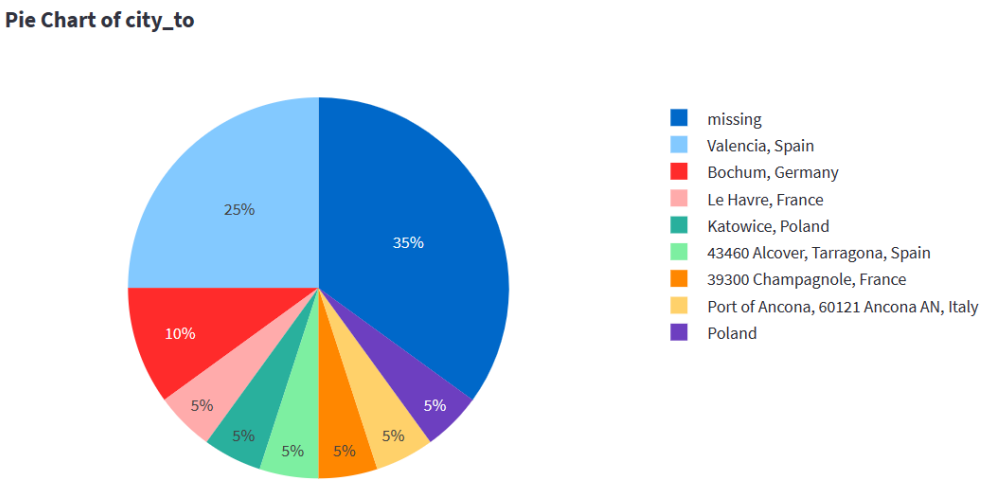

Рисунок 4.7 – Діаграма за даними Merged Leads city_to

Якщо проаналізувати міста звідки відправляються вантажі, то стає зрозумілим, що Китай є однозначним лідером (рис. 4.8). Багато китайських міст, таких як Шанхай, Гуанчжоу, Шеньчжень та Пекін, є важливими промисловими і виробничими центрами, звідки здійснюються численні відправлення вантажів. Порівнявши ці дві діаграми, можна зробити висновок, що товари, виготовлені в Китаї користуються великим попитом у Європі.

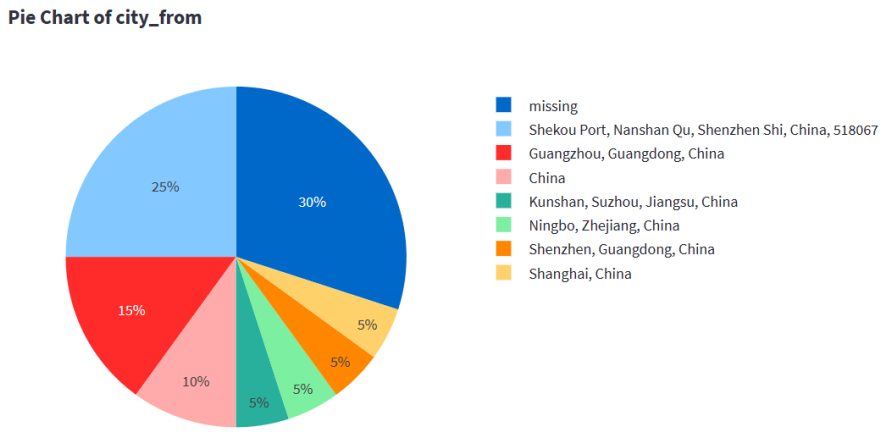

Рисунок 4.8 – Діаграма за даними Merged Leads city_from

Виведено статистику для числових даних у вигляді таблиці (рис. 4.9). Вага варіюється від 6 до 26 з середнім значенням 17.015.

	weight	amount	price_usd	valid_days
count	20	20	20	20
mean	17.015	1.05	9,255	7
std	7.0519	0.2236	917.9353	0
min	6	1	7,450	7
25%	10.46	1	9,150	7
50%	19	1	9,750	7
75%	24.25	1	9,900	7
max	26	2	9,900	7

Рисунок 4.9 – Вивід таблиці статистики числових змінних у веб-додатку

Стандартне відхилення 7.0519 вказує на значне розсіювання значень ваги. Кількість майже однакова для всіх спостережень, з мінімальним і максимальним значенням 1 та 2 відповідно, середнім значенням 1.05 і дуже низьким стандартним відхиленням 0.2236. Ціна варіюється від 7450 до 9900 доларів США з середнім значенням 9255.

Стандартне відхилення 917.9353 вказує на помірне розсіювання значень ціни, більшість знаходяться близько до середнього. Всі значення для valid_days рівні 7, що вказує на постійну дійсність протягом 7 днів для всіх спостережень.

Стандартне відхилення дійсних днів дорівнює 0, що підтверджує відсутність варіацій.

Матриця кореляції показує ступінь і напрямок лінійного зв’язку між парами змінних. Під час побудови моделей машинного навчання або регресійних моделей матриця кореляції допомагає вибрати змінні, які мають сильний зв’язок з цільовою змінною. Може виникати мультиколінеарність, коли дві або більше незалежних змінних у моделі сильно корелюють між собою. Це може вплинути на стабільність моделі і вирішується об’єднанням, регулярізацією, або видаленням однієї з корельованих змінних. Значення кореляції нуль означає, що між змінними немає лінійної залежності. Від 0 до 0,3 або -0,3 до 0.3 - зв'язок між змінними є слабким. Від 0,3 до 0,7 або -0,3 до -0,7 - помірний зв'язок. Від 0,7 до 1 або -0,7 до -1- сильний.

З представленої теплової карти (рис. 4.10) видно, що вага та кількість мають помірну позитивну кореляцію (0.2998983). Позитивне значення вказує на те, що змінні змінюються в одному напрямку - збільшення ваги вантажу зазвичай супроводжується збільшенням кількості товару. вага та вартість мають слабку негативну кореляцію (-0.1988489), зі збільшенням ваги ціна може зменшуватись, але цей зв'язок незначний. Різні товари можуть мати різні цінові категорії, незалежно від їх ваги. Наприклад, деякі легкі товари можуть мати високу вартість через їх рідкісність або специфічне призначення, тоді як деякі важкі товари можуть мати низьку вартість через їх загальну доступність або низьку популярність. Amount та price_usd зі слабкою позитивною кореляцією (0.1269271). Зі збільшенням кількості вартість трохи зростає, але цей зв'язок незначний. Більші партії товарів коштують дорожче, через додаткові витрати на обробку або транспортування.

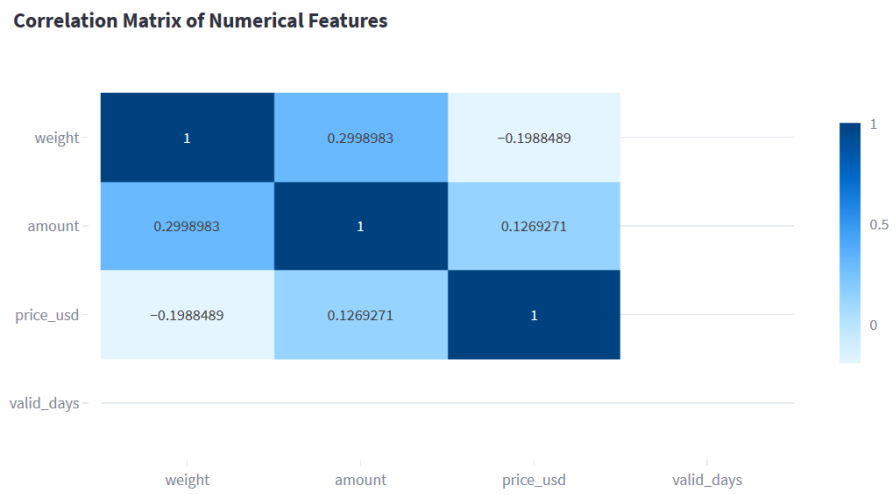

Рисунок 4.10 – Теплокарта матриці кореляції Merge_Leads

Якщо взяти, наприклад, дані з повітряних перевезень, з представленої теплової карти (рис. 4.11) видно, що вартість і кількість вантажу мають сильну позитивну кореляцію (0.9028368). Це вказує на те, що зі збільшенням обсягу завантаження ціна в доларах США також значно зростає.

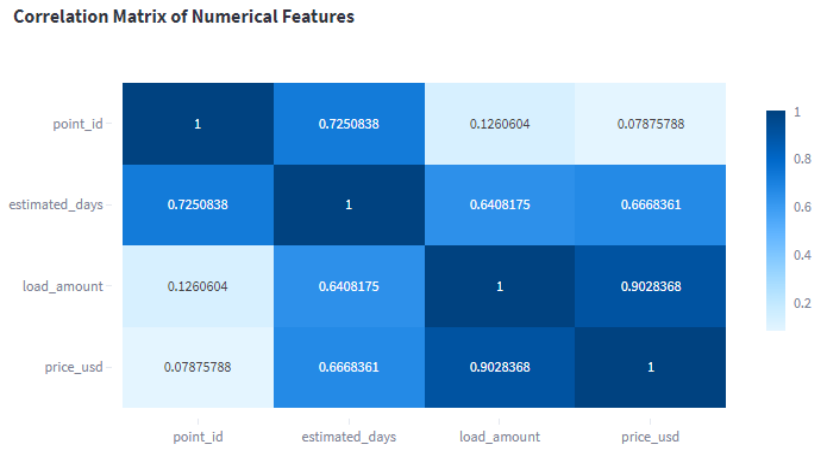

Рисунок 4.11 – Теплокарта матриці кореляції Air_Rates

Кореляція між розрахунковими днями і вартістю становить 0.6668361, що свідчить про сильну позитивну залежність.

Щоб визначати вплив категоріальних значень на числові використано аналіз дисперсії (ANOVA) [14]. F-статистика (7.0535) вказує на відношення

між варіабельністю середніх значень ціни (price_usd) у різних категоріях city_to_country та варіабельністю всередині кожної категорії.

Високе значення F-статистики свідчить про те, що існують значні відмінності між середніми значеннями price_usd для різних категорій city_to_country.

Оскільки P-значення (0.0017) є значно меншим за 0.05, можна відкинути нульову гіпотезу. Це означає, що є статистично значущі відмінності між середніми значеннями price_usd для різних категорій city_to_country.

Результати тесту (рис. 4.12) свідчать, що напрямок перевезення значно впливає на вартість.

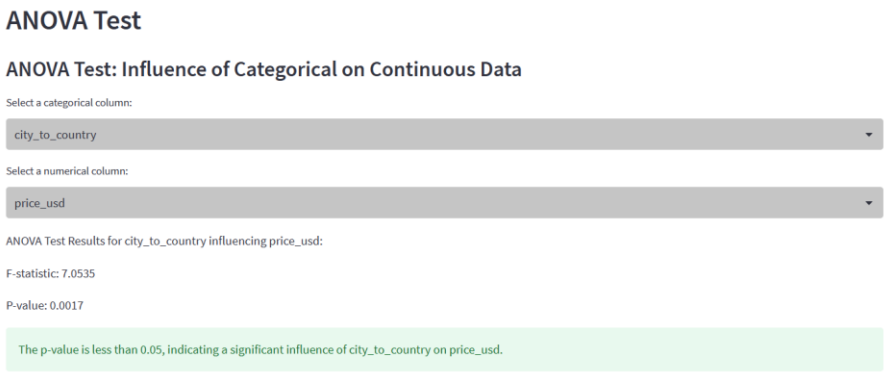

Рисунок 4.12 – ANOVA тест

4.3 Асоціативні правила

Аналіз виконано за допомогою алгоритму Apriori і бібліотеки асоціативного вивчення mlxtend (додаток Б) [15].

Надається можливість налаштування наступних параметрів:

– Minimum support (мінімальна підтримка) визначає, яка частка (відсоток) транзакцій або спостережень має містити певний набір елементів, щоб цей набір був визнаний як "частий" або "підтримуваний";

– Minimum confidence (мінімальна впевненість) визначає міру упевненості, з якою правило вважається коректним. Впевненість вимірює,

наскільки часто консеквент (наслідок) з'являється в транзакціях, що містять антецедент (попередник). Високі значення мінімальної впевненості допомагають відфільтрувати менш значущі правила, але можуть призвести до втрати важливої інформації.

Алгоритм Apriori виявляє всі можливі комбінації елементів, які з'являються разом у транзакціях, шляхом ітеративного пошуку та аналізу найпоширеніших наборів елементів. Якщо асоціативні правила знайдено, вони відображаються у вигляді таблиці.

Для візуалізації асоціативних правил використовуються графи (рис.4.13), що дозволяють наочно представити, як предмети пов'язані один з одним і які з них утворюють сильні асоціації. Для цього використано функцію draw_interactive_graph(), куди передаються наступні параметри:

– Number of top columns to consider – кількість стовпців для розгляду;

– Level Separation контролює відстань між рівнями вузлів у графі. Більше значення означає більшу відстань між рівнями;

– Node_spacing вказує на відстань між сусідніми вузлами на тому ж рівні;

– Tree_spacing контролює відстань між деревами (або гілками) у графі. Більше значення означає більшу відстань між деревами;

– Max Number of Rules дозволяє вибрати максимальну кількість асоціативних правил, що будуть відображені у графі.

В залежності від обраних параметрів на застосованому наборі даних, правила відображаються у таблиці (рис. 4.14), кожен стовпець якої містить наступну інформацію:

– Antecedents (передумови) – набір атрибутів, які передують наслідкам у правилі. Іншими словами, це "якщо" частина правила;

– Consequents (наслідки) – елементи, які слідують за передумовами. Іншими словами, це "тоді" частина правила;

– Antecedent support – частота, з якою антецедент асоціативного правила з’являється в транзакціях. Вимірюється як відсоток транзакцій, де антецедент присутній;

– Consequent support – частота, з якою з’являється консеквент;

– Support – частота, з якою асоціативне правило спостерігається в даних;

– Confidence – ймовірність, з якою консеквент відбувається у транзакціях, якщо антецедент також відбувається. Вона вимірюється як відсоток транзакцій з антецедентом, в яких також є консеквент;

– Lift – міра того, наскільки більша або менша ймовірність консеквента в транзакціях з антецедентом, ніж ймовірність консеквента в загальних транзакціях. Це дозволяє визначити, наскільки сильно залежить консеквент від антецедента. Якщо lift = 1 - антецедент і консеквент є незалежними один від одного. Якщо lift > 1 - наявність антецедента збільшує ймовірність появи консеквента;

– Leverage – міра відмінності між фактичною відносною частотою спостереження антецедента і консеквента разом у транзакціях та їх спільної частоти, яка випадково відбувалася, якби антецедент і консеквент були незалежними;

– Conviction – міра, яка визначає ступінь в якій консеквент залежить від антецедента. Вона обчислюється як відношення ймовірності випадку, коли антецедент і консеквент не з'являються разом, до фактичної ймовірності випадку, коли вони з'являються разом;

– Zhangs_metric – метрика оцінки сили асоціації між антецедентом і консеквентом, враховуючи їхню підтримку та достовірність.

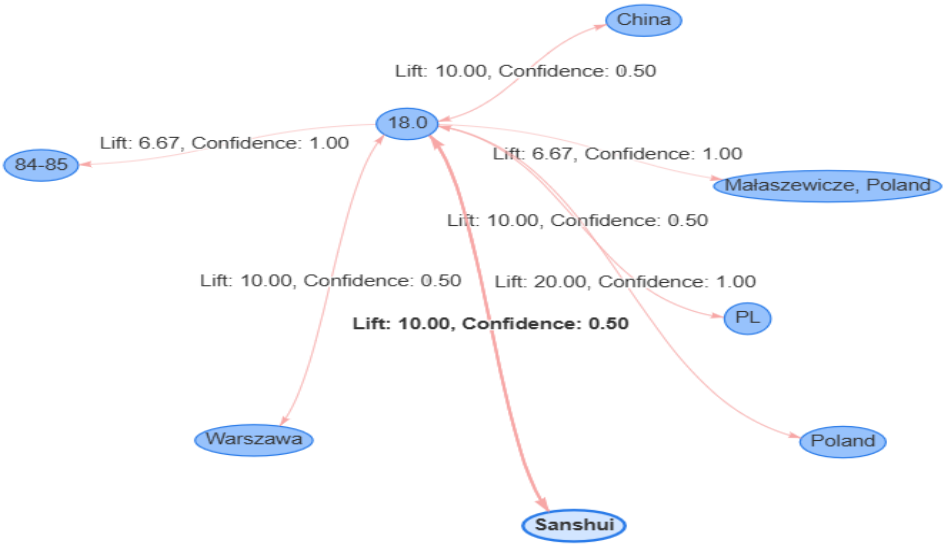

Рисунок 4.13 – Граф асоціативних правил

	antecedents	consequents	antecedent suppo
17	LATAM Airlines Group	Miami Intl	0.0
18	TAP Air Portugal	Miami Intl	0.(
19	100, Salgado Filho	American Airlines	0.0
20	100, American Airlines	Salgado Filho	0.0
21	Salgado Filho, 1000	American Airlines	0.0
22	American Airlines, 1000	Salgado Filho	0.0
23	Salgado Filho, 25	American Airlines	0.0
24	American Airlines, 25	Salgado Filho	0.0
25	250, Salgado Filho	American Airlines	0.0
26	250, American Airlines	Salgado Filho	0.0
27	300, Salgado Filho	American Airlines	0.0

Рисунок 4.14 – Вивід таблиці асоціативних правил у веб-додатку

Метод асоціативних правил найкраще спрацьовує на великих наборах даних, де залежності не так легко простежити, тому для аналізу обрано повний набір даних за авіаперевезеннями. Попередньо видалено змінні, що повторювалися і не несли цінної інформації. Мінімальну підтримку виставлено 0,05, що означає, що лише ті набори предметів, які зустрічаються у 50 і більше транзакціях (тобто 5% від 1000), будуть вважатися частими. Мінімальна впевненість 0,70 (70%), гарантує, що правила, які будуть виявлені,

мають високу ймовірність до точності. У результаті отримано таблицю і граф асоціацій. Розглянуто один з рядків (рис. 4.15). В якості передумови правила отримано комбінацію змінних: кількість навантаження у кілограмах і авіакомпанія, а як наслідок (результат використання правила) – аеропорт. Передумова (рейс 750, American Airlines) зустрічається у 6.2% всіх транзакцій, а наслідок (Salgado Filho) зустрічається у 51.1% всіх транзакцій.

Комбінація (рейс 750, American Airlines та Salgado Filho) зустрічається у 6% всіх транзакцій. У 96.77% випадків коли передумова (рейс 750, American Airlines) присутня у транзакції, наслідок (Salgado Filho) також присутній. Підйом 1.8938 вказує на те, що наслідок (Salgado Filho) у 1.8938 разів більш ймовірний у транзакціях, де присутня передумова (рейс 750, American Airlines), порівняно з випадковим збігом, що вказує на сильну асоціацію між ними. Леверидж 0.0283 підтверджує наявність позитивного зв'язку між передумовою та наслідком. Впевненість 15.159 вказує на те, наскільки надійною є асоціація. Чим більше значення, тим менш ймовірно, що наслідок виникне без передумови.

22	750,American Airlines	Salgado Fillho	0.062	0.511	0.06	0.9677	1.8938	0.0283	15.159	0.5032

Рисунок 4.15 – Рядок таблиці асоціативних правил

Асоціативні правила стануть у нагоді для виявлення частих комбінацій умов транспортування і товарів, що допоможе у створенні персоналізованих пропозицій клієнтам.

Патерни у маршрутах доставки корисні для прогнозування попиту в конкретні періоди і плануванні маркетингової стратегії.

4.4 Кластеризація

Кластеризація зосереджена на групуванні схожих об'єктів, тоді як асоціативні правила зосереджені на виявленні залежностей між елементами у великих наборах даних.

Для зменшення розмірності даних, видалення шуму, виявлення головних компонентів та їх подальшого використання для візуалізації застосовано метод PCA (додаток В).

Проаналізовано метод кластеризації на прикладі набору даних Sea Rates, так як він є більшим за розміром.

Таблиця "Processed Data" (рис. 4.16) містить результати даних після застосування методу PCA зазвичай до двух компонентів: "Component 1" і "Component 2", які є новими ознаками, отриманими в результаті зменшення розмірності даних, а значення у кожному стовпці є координатами цієї точки у новому просторі ознак. Кожен рядок відповідає одному спостереженню, а значення в кожному стовпці представляють ваги (або важливість) кожного спостереження у відповідному компоненті.

Processed Data:

	Component 1	Component 2
0	-2.0694	-0.4184
1	-2.0494	-0.3983
2	-2.0494	-0.3983
3	1.4467	-2.4689
4	2.2069	-1.7087
5	2.2069	-1.7087
6	1.0301	-1.6378
7	1.2203	-1.4476
8	1.2203	-1.4476
9	1.2807	0.0524

Рисунок 4.16 – Вивід таблиці "Processed Data" у веб-додатку

Якщо спостереження мають позитивні значення у стовпці Component 1, то це означатиме, що ці характеристики є ключовими для визначення цього напрямку варіації, негативні значення вказують на низьку важливість.

Component 2 може розкривати вплив інших факторів на вартість перевезення, вказувати на додаткові важливі взаємозв'язки або особливості в даних, які не були виявлені першою компонентою. Наприклад, вартість перевезення може залежати від інших факторів, таких як відстань між пунктами відправлення та доставки, вид транспорту, товар.

Оптимальна кількість кластерів визначається за допомогою методу "ліктя". Для кожного значення кількості кластерів (від 1 до 10) виконується кластеризація методом K-means [5], і обчислюється сума квадратів відстаней від кожної точки до найближчого центру кластера. Після цього створено графік (рис. 4.17), на якому можна визначити точку "ліктя".

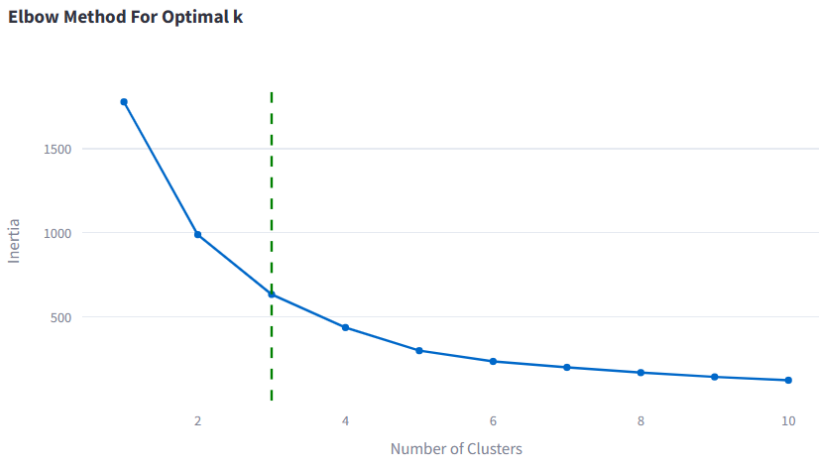

Рисунок 4.17 – Графік методу “ліктя”

Також надається можливість самостійно обрати кількість кластерів для аналізу за допомогою повзунка, щоб порівняти отримані результати з оптимальною кількістю.

Дані розподіляються на кластери на основі схожих або близьких за значенням об'єктів у стовпці Component 1.

Відбувається візуалізація кластерів за допомогою діаграми розсіювання (рис. 4.18), де кожен кластер позначений окремим кольором.

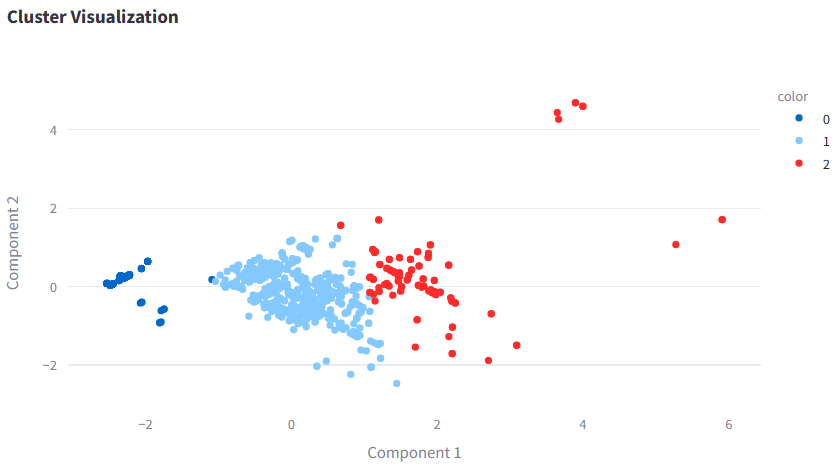

Рисунок 4.18 – Діаграма розсіювання

У вигляді стилізованої таблиці (рис. 4.19) виводяться статистичні дані для кожного кластера (середні, стандартні відхилення, мінімальні та максимальні значення кожного числового стовпця даних).

Cluster statistics

Yellow highlights the maximum values, **light green** highlights the minimum values in each column.

Cluster		25%	50%	75%	count	max	mean	min	std
0	estimated_days	3.000000	3.000000	3.000000	130.000000	20.000000	3.815385	2.000000	3.743283
0	load_amount	1.000000	1.000000	1.000000	130.000000	1.000000	1.000000	1.000000	0.000000
0	price_usd	345.900688	615.000000	690.000000	130.000000	1575.000000	594.247674	80.000000	255.303073
1	estimated_days	23.000000	31.000000	38.000000	630.000000	70.000000	31.261905	16.000000	9.381491
1	load_amount	1.000000	1.000000	1.000000	630.000000	1.000000	1.000000	1.000000	0.000000
1	price_usd	1962.000000	2240.000000	2638.000000	630.000000	3850.000000	2305.964624	941.000000	465.212979
2	estimated_days	39.000000	43.000000	50.000000	129.000000	77.000000	44.728682	20.000000	11.956139
2	load_amount	1.000000	1.000000	1.000000	129.000000	1.000000	1.000000	1.000000	0.000000
2	price_usd	3648.000000	3821.000000	4028.000000	129.000000	9394.000000	4142.072319	2454.000000	1380.000051

Рисунок 4.19 – Вивід статистики по кластерах

Кольори відмічають максимальні (жовтий) та мінімальні (світло-зелений) значення в кожному стовпці для кожного кластера.

Використовується метод groupby() для групування даних по кластерах. Після цього застосовується метод describe(), який обчислює ряд статистичних характеристик для кожної групи в окремій колонці, таких як середнє значення (mean), медіана (50%), квартилі (25%, 75%), мінімальне та максимальне значення, стандартне відхилення, кількість рядків.

Нульовий стовпчик цієї таблиці відповідає номеру кластера. В кластері 0 згруповано 130, в кластері 1 – 630, а от у кластері 2 – 129 спостережень.

Проаналізовано характеристики кластерів, а саме змінну вартості. В нульовому кластері ціни варіюються від 80 до 1575 доларів США з середнім значенням 594.25. Висока варіація цін (стандартне відхилення 255.30) вказує на те, що цей кластер може включати різні типи перевезень з різними ціновими категоріями. У першому кластері ціни варіюються від 941 до 3850 доларів США з середнім значенням 2305.96. Стандартне відхилення 465.21 вказує на те, що ціни у цьому кластері є більш концентрованими навколо середнього значення, але все ще мають значну варіацію. Ціни в другому кластері в діапазоні від 2454 до 9394 доларів США, з середнім значенням 4142.07. Високе стандартне відхилення 1380.00 вказує на значну варіацію цін у цьому кластері, що може свідчити про те, що кластер включає перевезення з дуже високими цінами або спеціалізованими пропозиціями, преміум-сегмент.

4.6 Методи регресії

Детальніше проаналізовано налаштування та роботу кожного з методів, які було обрано для навчання (додаток Г). Результати роботи моделей оцінюються за допомогою трьох метрик: MSE (середньоквадратична помилка), MAE (середньо абсолютна помилка), R2 Score (коефіцієнт детермінації). Чим менше значення MSE і MAE, тим краще. Чим ближче R2

Score до 1, тим краще. Виводиться прогнозоване значення цільової змінної, діаграма розсіювання фактичних і прогнозованих значень, графік і таблиця важливості змінних.

4.6.1 Налаштування основних параметрів для методів регресії

Спочатку роботи треба обрати набір даних в залежності від типів перевезень. Для інформації виведено таблицю унікальних комбінацій значень ‘порт звідки’ та ‘порт куди’ вибраного набору даних. Після цього обирається цільова числова змінна, яку потрібно спрогнозувати (рис. 4.20).

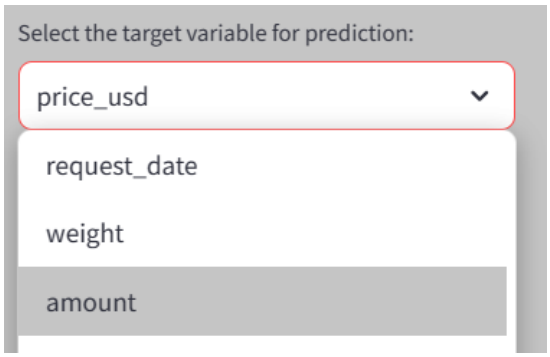

Рисунок 4.20 – Спадаючий список для вибору цільової змінної

Є можливість вибрати ознаки, які будуть використовуватися для навчання моделі (рис. 4.21). Це важливо, оскільки не всі з них можуть бути корисними або відображати вплив на цільову змінну, тому вибір правильних ознак може покращити результати моделі, зробивши її більш ефективною.

Наприклад, для прогнозування ціни, можливо, деякі ознаки, такі як кількість товару, тип, географічні дані, можуть бути важливі для прогнозування. Однак інші ознаки, які не несуть інформативності, можна виключити з моделі.

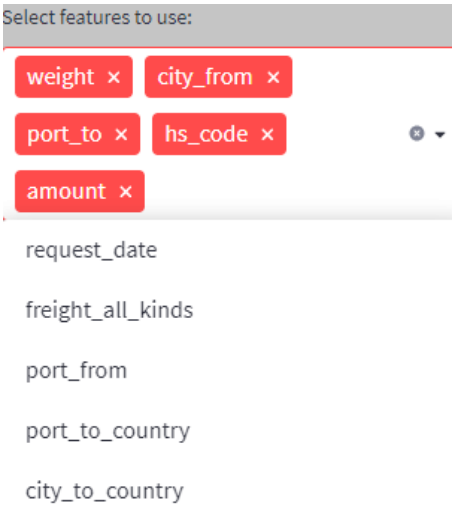

Рисунок 4.21 – Можливі ознаки впливу на цільову змінну

За допомогою радіокнопки є можливість вибору між двома варіантами кодування категоріальних змінних: "Frequency Encoding" і "Target encoding". За замовчуванням обрано "Frequency Encoding" (Кодування за частотою). Якщо обрати "Target encoding", то кожна категорія замінюється на середнє значення цільової змінної для цієї категорії. А у "Frequency Encoding" змінюється кожне унікальне значення категорії на його частоту відносно всіх значень у відповідному стовпці.

Цей метод дозволяє зберегти інформацію про розподіл категорійних значень, не збільшуючи розмірність даних шляхом створення нових стовпців.

Для прогнозування використано декілька методів машинного навчання, а саме: метод Ridge Regression, Lasso, Random Forest, Decision Tree Regressor, XGBoost, AdaBoost, Gradient Boosting (рис. 4.22).

В залежності від обраного метода з'являються необхідні параметри для налаштування.

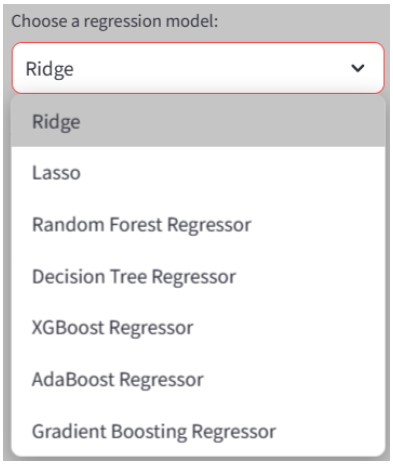

Рисунок 4.22 – Методи машинного навчання для прогнозування

Для налаштування моделей з метою досягнення кращої продуктивності та узагальнення на нових даних, використано гіперпараметри пошуку: ‘Немає’, ‘Випадковий пошук’, ‘Гратчастий пошук’ (рис. 4.23). Для випадкового можна вказати кількість ітерацій. "Number of CV folds" означає кількість поділів даних, які використовуються під час процедури крос-валідації. Крос-валідація - це техніка оцінки моделі машинного навчання, яка передбачає розбиття даних на декілька частин або "складок" (folds), щоб забезпечити більш надійну оцінку продуктивності моделі, зменшуючи вплив розподілу даних на результат. Модель навчається на деяких частинах даних і оцінюється на решті. Цей процес повторюється декілька разів, при цьому кожна частина даних один раз використовується для оцінки моделі. Для більшості задач достатньо використати 5 повторів.

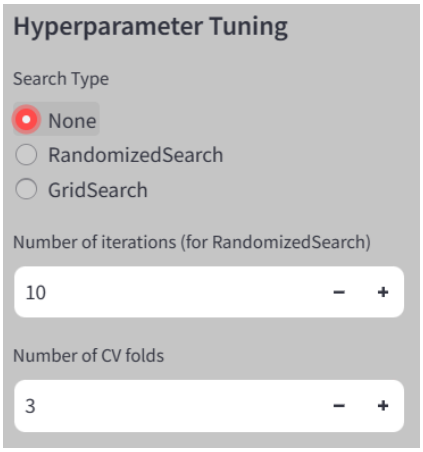

Рисунок 4.23 – Вибір параметра для прогнозування

Під час роботи гратчастого пошуку передбачається систематичний перебір комбінацій різних значень гіперпараметрів для вибору оптимальних, які забезпечують найкращу продуктивність моделі. При великій кількості комбінацій гіперпараметрів може виникнути ризик перенавчання, оскільки модель може добре пасувати до конкретних даних перевірки, але погано генералізувати на нові. Вагомим недоліком є також затрати часу і обчислювальних ресурсів. Загалом, час виконання може коливатися від кількох хвилин до кількох годин або навіть днів, залежно від вказаних параметрів. Але це є найкращим рішенням для тих, хто не розуміється на параметрах моделей.

Для прогнозування необхідно обрати значення ‘порт звідки’ і ‘порт куди’. Після встановлення всіх параметрів, щоб ініціювати роботу прогнозування, треба натиснути кнопку ‘Predict’, яка розташована у нижній лівій частині екрану.

4.6.2 Метод Ridge

Метод Ridge регресії (L2-регуляризація) використовується для вирішення проблеми мультиколінеарності та зменшення перенавчання в лінійних регресійних моделях. Основна ідея методу полягає в тому, щоб

зменшити величину коефіцієнтів регресії, зокрема шляхом додавання до функції втрат суми квадратів коефіцієнтів, помножених на параметр λ (лямбда).

Для роботи цього методу необхідно налаштувати деякі ключові параметри, такі як α (регуляризація), solver (розв'язувач) та max iterations (максимальна кількість ітерацій) (рис. 4.24).

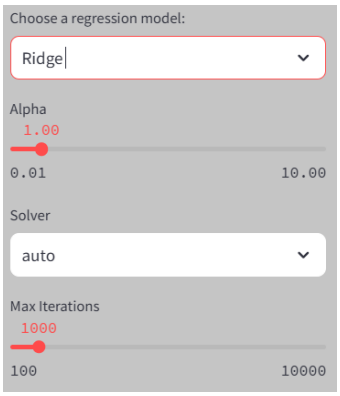

Рисунок 4.24 – Параметри моделі Ridge

Параметр alpha використовується для керування силою регуляризації моделі. Він має мінімальне значення 0.01, максимальне – 10.0, а значення за замовчуванням – 1.0. Регуляризація - це процес додавання штрафу на параметри моделі для уникнення перенавчання. Значення alpha визначає, наскільки сильно цей штраф буде застосовуватися: чим вище значення, тим сильніше регуляризація.

Параметр solver дозволяє вибрати метод розв'язку для моделі. Варіанти включають auto, svd, cholesky, lsqr, sparse_cg, sag та saga. Auto - автоматичний вибір методу, що обирає найкращий розв'язок для задачі. SVD (сингулярний розклад) підходить для невеликих наборів даних. Cholesky (розклад Холецького) ефективний для великих, щільних матриць. LSQR - ітеративний метод найменших квадратів, зазвичай використовується для розріджених матриць. Sparse_cg (розріджений метод спряжених градієнтів) знаходить розв'язок для розріджених матриць. SAG використовує стохастичний

градієнтний спуск із збереженням оновлень градієнта. SAGA - стохастичний метод, що працює з регуляризацією L1 та L2.

Параметр Max Iterations дозволяє вибрати максимальну кількість ітерацій для оптимізаційного алгоритму. Він має мінімальне значення 100, максимальне – 10 000, а значення за замовчуванням – 1000. Це важливий параметр, оскільки він впливає на точність та швидкість навчання моделі. Зазвичай велика кількість ітерацій дозволяє отримати більш точний результат, але може призвести до збільшення часу навчання моделі, особливо при роботі з великими обсягами даних або складними моделями.

Під час опрацювання методу Ridge шукалися найкращі значення параметрів. Знайдено такі α = 5,48, max iterations = 8323, за яких оцінки якості дали найліпші результати (рис. 4.25).

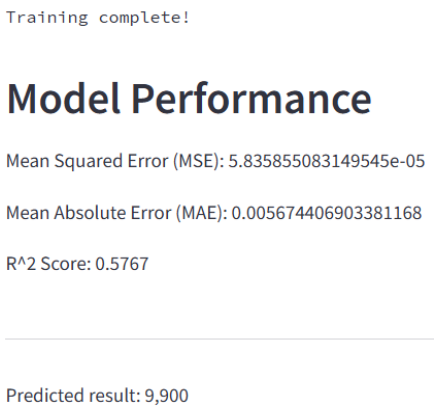

Рисунок 4.25 – Результат роботи методу Ridge

Значення MSE 5.835855083149545e-05 вказує на те, що середній квадрат помилки є досить малим, що свідчить про хорошу точність моделі. Результат MAE 0.005674406903381168 є також добрим показником, так як модель в середньому помиляється на трохи більше ніж 0.0056 в абсолютному вимірі. Значення R2 Score 0.5767 означає, що модель пояснює приблизно 57.67% варіації у вихідних даних і свідчить про середню точність моделі. Загалом ці значення показують, що модель є досить точною.

Прогнозована вартість за напрямком Шанхай - Барселона: 9,900 доларів США. Через малий діапазон вартостей, які присутні у наборі даних, прогнозована вартість близька до фактичної (рис. 4.26).

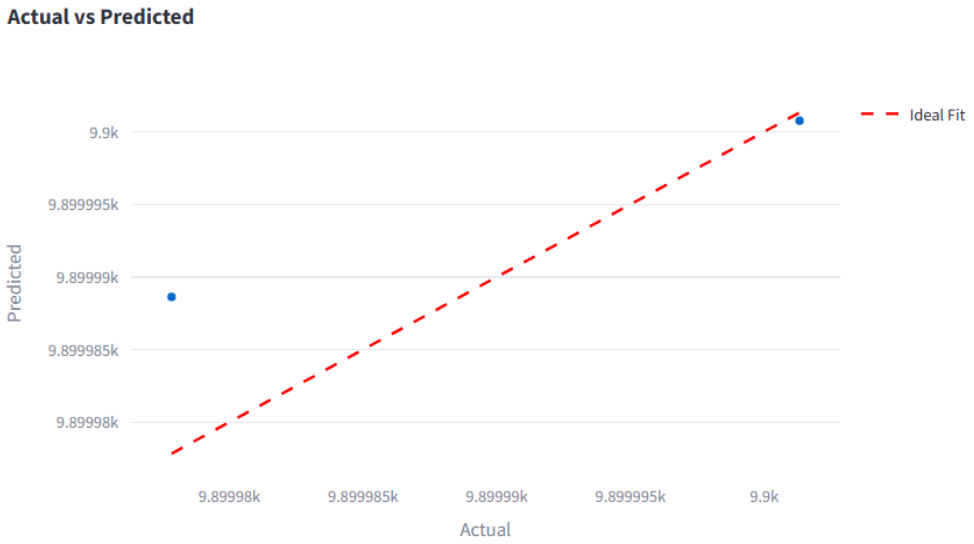

Рисунок 4.26 – Результат роботи методу Ridge

З діаграми впливу додаткових змінних (рис. 4.27) на цільову стало зрозуміло, що вага (weight) зі значенням приблизно 0.0016 робить невеликий внесок у передбаченні вартості перевезення серед обраних ознак.

Рисунок 4.27 – Діаграма впливу додаткових змінних

4.6.3 Метод Lasso

Lasso може бути менш стабільним методом, ніж Ridge, у випадках з висококорельованими змінними, але він може бути корисним у ситуаціях, де необхідно виділити найбільш значущі змінні з групи корельованих ознак.

Застосовуються параметри alpha та max iterations аналогічно методу Ridge. Для ефективної роботи Lasso потрібний досить великий розмір вибірки. У разі невеликої вибірки або нерівномірного розподілу даних, як в ситуації з Merged_Leads, Lasso може працювати менш стабільно і давати неточні результати.

Найкращі значення так і не вдалося знайти вручну, тому довелося скористатися гратчастим пошуком (рис. 4.28), який, на жаль, також не зміг виявити оптимальних параметрів і після перебору обрав значення за замовчуванням.

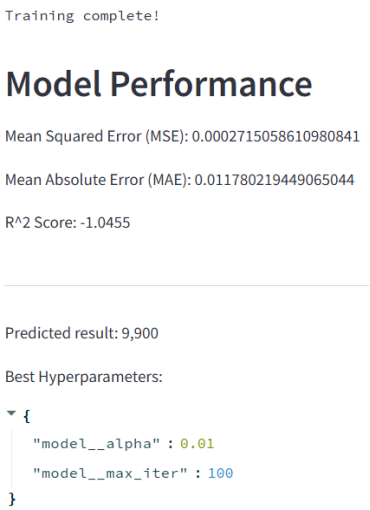

Рисунок 4.28 – Результат роботи гратчастого пошуку

На підставі отриманих метрик можна зробити висновок, що модель дає непогані значення MSE і MAE, але при цьому R2 Score є негативним значенням, що вказує на дуже низьку продуктивність моделі.

4.6.4 Метод Random Forest Regressor

Ця модель машинного навчання ґрунтується на алгоритмі випадкового лісу. Для роботи цього методу необхідно налаштувати такі параметри, як: кількість дерев, максимальна глибина, мінімальна кількість вибірок для розбиття, мінімальна кількість вибірок у листі (рис. 4.29).

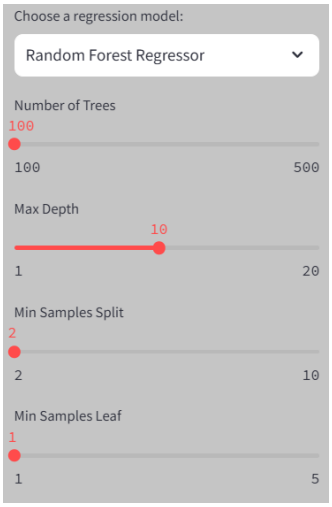

Рисунок 4.29 – Параметри моделі Random Forest Regressor

Параметр Number of Trees (Кількість дерев) визначає кількість дерев у випадковому лісі. Більше дерев може покращити стабільність моделі та зменшити схильність до перенавчання. Однак, збільшення цього параметра може збільшити час навчання та ресурси обчислення. Зазвичай рекомендується розглядати значення в діапазоні від 100 до 1000.

Параметр Max Depth (Максимальна глибина) – кількість рівнів у кожному дереві. Більша глибина дерева дозволяє моделі узагальнювати більше складніші взаємозв'язки у даних. Але глибокі дерева можуть призвести до перенавчання, особливо у випадку обмеженого набору даних. Цей параметр допомагає контролювати складність кожного дерева. Рекомендується встановлювати значення в межах від 10 до 100.

Параметр Min Samples Split (Мінімальна кількість вибірок для розбиття) – мінімальна кількість вибірок, яка необхідна для розгалуження вузла дерева. Цей параметр допомагає уникнути перенавчання, контролюючи, коли дерево може розгалужуватися далі. Зазвичай це число є часткою загальної кількості вибірок у наборі даних. Рекомендоване значення зазвичай становить від 2 до 10.

Параметр Min Samples Leaf (Мінімальна кількість вибірок у листі) – мінімальна кількість вибірок, яка потрібна для того, щоб вузол був вважаний листком (кінцевим вузлом дерева). Цей параметр допомагає уникнути перенавчання, контролюючи, коли дерево може зупинитися. Як і Min Samples Split, зазвичай це число є часткою загальної кількості вибірок у вибірці. Рекомендоване значення також становить від 2 до 10.

Під час тестування різних значень параметрів знайдено ті, що спрацьовують з найкращими результатами: 344 дерева, глибина 8, 4 вибірки для розбиття та 3 вибірки у листі.

Оцінки MSE та MAE низькі, що означає точні прогнози на тестових даних (рис. 4.30). Результат R2 Score 0.7123 говорить про те, що модель добре описує варіативність даних.

Рисунок 4.30 – Оцінки роботи методу Random Forest Regressor

Діаграма впливу додаткових змінних (рис. 4.31) продемонструвала, що кількість, вага і дата створення запиту найбільше впливають на результат прогнозування вартості.

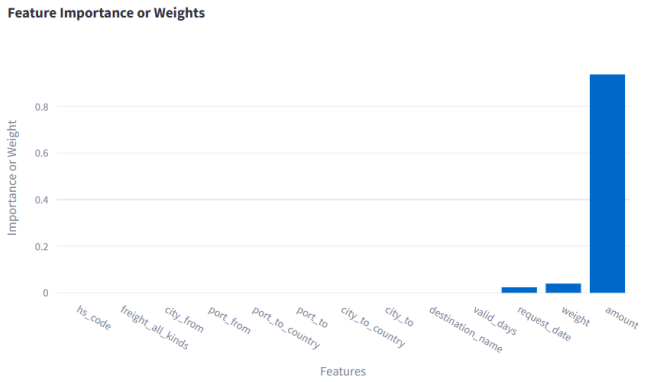

Рисунок 4.31 – Діаграма впливу додаткових змінних

4.6.5 Метод Decision Tree Regressor

Decision Tree Regressor будує дерево рішень, яке розділяє дані на групи так, щоб мінімізувати середньоквадратичну помилку (MSE) у кожному листовому вузлі. Цей метод схильний до перенавчання. Для його роботи необхідно налаштувати такі параметри (аналогічні методу Random Forest Regressor), як: максимальна глибина, мінімальна кількість вибірок для розбиття, мінімальна кількість вибірок у листі (рис. 4.32).

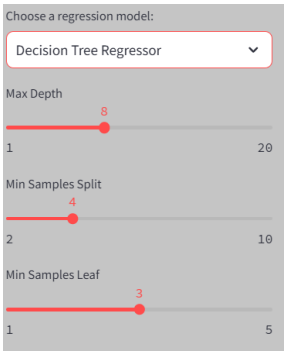

Рисунок 4.32 – Параметри методу Decision Tree Regressor

Знайдено параметри, що спрацьовують з найкращими результатами: глибина – 18, вибірок для розбиття – 7 та 4 вибірки у листі.

Значення оцінки MSE свідчить про високу точність моделі. Оцінка MAE доволі низька і свідчить про гарну якість. Значення R2 Score означає, що модель пояснює 85,11% варіації даних. Це вказує на те, що модель дуже добре справляється із завданням передбачення, пояснюючи більшу частину варіації цільової змінної (рис. 4.33).

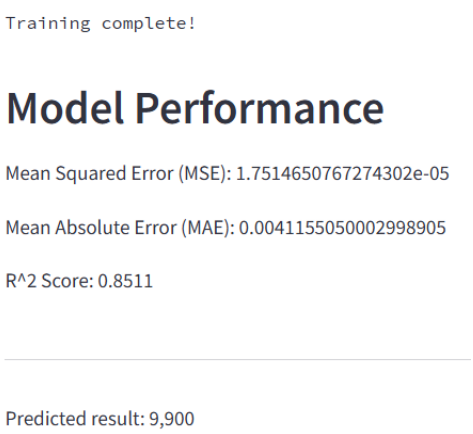

Рисунок 4.33 – Оцінки роботи методу Decision Tree Regressor

На результат найбільше впливає вага, що отримано з діаграми (рис.4.34).

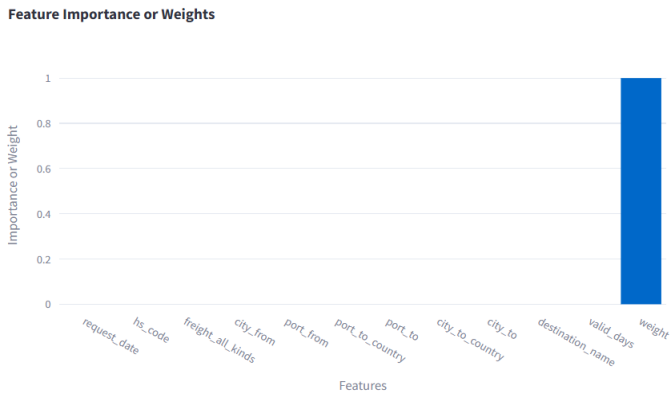

Рисунок 4.34 – Діаграма впливу додаткових змінних

4.6.6 Метод XGBoost Regressor (Extreme Gradient Boosting)

Цей метод будує ансамбль дерев рішень, де кожне нове дерево намагається виправити помилки попередніх. Він дозволяє покращити точність прогнозів та зменшити перенавантаження моделі.

Для роботи необхідно налаштувати такі параметри (рис. 4.35), як: темп навчання, кількість дерев, максимальна глубина, субвибірка, випадкові ознаки для дерева.

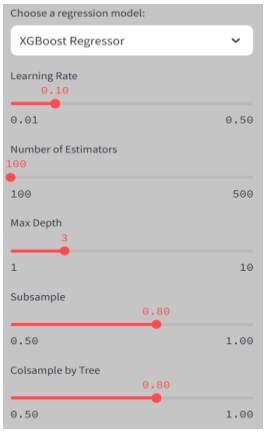

Рисунок 4.35 – Параметри методу XGBoost Regressor

Параметр Learning Rate – визначає швидкість збіжності градієнтного спуску. Він контролює величину кроку, який використовується при оновленні прогнозу моделі на кожному кроці.

Параметр Number of Estimators – кількість дерев, що будуть побудовані у процесі навчання. Більше дерев може допомогти у підвищенні точності моделі, але також може призвести до перенавчання.

Параметр Max Depth – визначає максимальну глибину кожного дерева в ансамблі. Встановлення великої глибини може призвести до перенавчання, тоді як занадто мала глибина може призвести до недонавчання.

Параметр Subsample – визначає частку випадкової підвибірки навчальних даних, яка буде використана для побудови кожного дерева.

Використання підвибірок може допомогти уникнути перенавчання і зробити модель більш стійкою до випадковості в даних.

Параметр Colsample by Tree – визначає частку ознак, які будуть випадково вибрані для побудови кожного дерева для зменшення кореляції між деревами.

Знайдено найліпші значення параметрів для роботи методу. Темп навчання встановлено – 0.17, дерев – 169, глибина – 3, субвибірка і випадкові ознаки – 0.80. Оцінки роботи моделі регресії (рис. 4.36) мають високу точність, добре передбачається значення цільової змінної. Низькі значення MSE та MAE, а також високий коефіцієнт детермінації R2 Score вказують на те, що модель якісно виконує своє завдання, пояснюючи значну частину варіації даних.

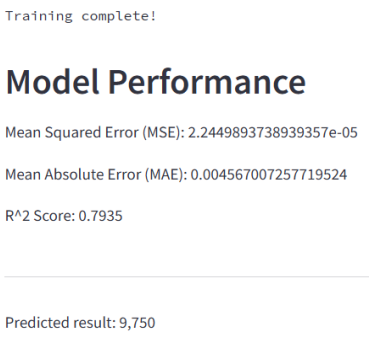

Рисунок 4.36 – Оцінки роботи методу XGBoost Regressor

Діаграма впливу додаткових змінних (рис. 4.37) продемонструвала, що вага і дата створення запиту найбільше впливають на результат прогнозування вартості. Дата створення може впливати через те, що існують сезонні коливання цін, зміна попиту та пропозицій.

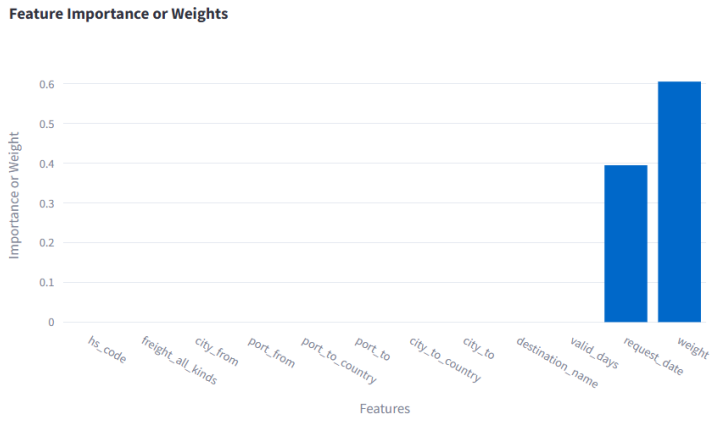

Рисунок 4.37 – Діаграма впливу додаткових змінних

4.6.7 Метод AdaBoost Regressor

Цей метод працює, "комбінуючи" декілька простих моделей регресії разом, які називаються "слабкими моделями" (не дуже точні). Кожна з цих моделей навчається на даних, а потім їх прогнози об'єднуються, щоб отримати кінцевий. Цей підхід допомагає покращити точність прогнозів, особливо коли дані мають складну структуру або шум.

Необхідні параметри (рис. 4.38) для роботи методу: кількість дерев, швидкість навчання, рівень відхилення.

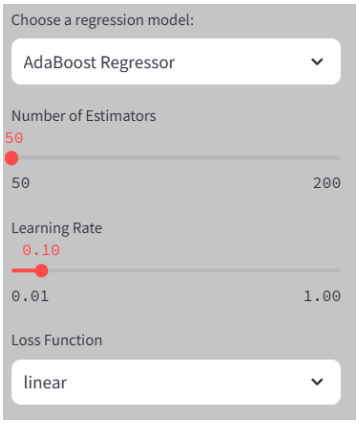

Рисунок 4.38 – Параметри методу AdaBoost Regressor

Параметр Number of Estimators – кількість дерев рішень, які будуть використовуватися для побудови ансамблю в процесі навчання.

Параметр Learning Rate – контролює швидкість, з якою модель навчається на допомогу базових моделей. Зменшення швидкості навчання може допомогти уникнути перенавчання.

Функція Loss Function – визначає рівень відхилення між прогнозованими значеннями моделі і справжніми значеннями в навчальному наборі. Для AdaBoost Regressor можна використовувати різні функції втрат:

– Linear Loss Function (Лінійна функція втрат) – вимірює абсолютне значення різниці між прогнозованими та справжніми значеннями;

– Square Loss Function (Квадратична функція втрат) – вимірює квадрат відхилення між прогнозованими та справжніми значеннями;

– Exponential Loss Function (Експоненційна функція втрат) – використовується для зменшення впливу великих помилок. Вона накладає більший штраф за більші відхилення, що може допомогти покращити поведінку моделі в разі великих помилок.

Під час тестування різних значень параметрів знайдено найбільш задовільні, в той час як гратчастий пошук видав набагато гірші.

Кількість дерев – 94, швидкість навчання – 0.56, експоненційна функція втрат.

Значення MSE та MAE низькі, що свідчить про високу точність моделі.

Значення R2 Score означає, що модель пояснює 82.82% варіації даних. Це хороший результат (рис. 4.39).

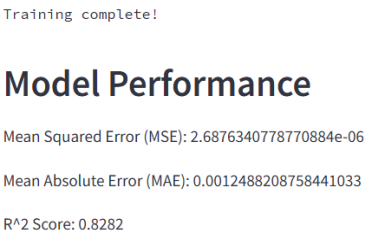

Рисунок 4.39 – Оцінки роботи методу AdaBoost Regressor

Результати гратчастого пошуку виявилися незадовільними, бо значення MSE та MAE вказують на наявність похибок у передбаченні моделі, а від'ємне значення R2 Score (-8644.3362) свідчить про те, що модель є невдалою, значно відхиляється від фактичних даних (рис. 4.40).

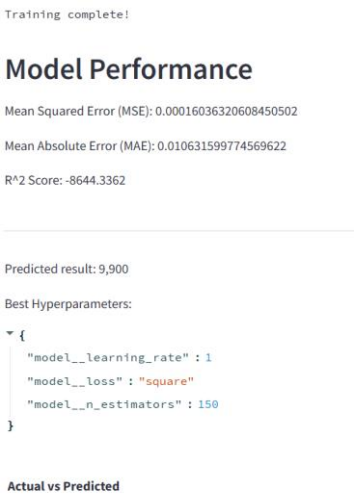

Рисунок 4.40 – Результат роботи гратчастого пошуку методу AdaBoost Regressor

На результати прогнозування вартості у цьому випадку найбільше впливають тип товару і вага (рис. 4.41).

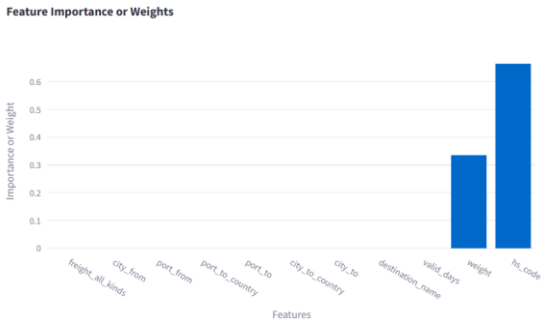

Рисунок 4.41 – Діаграма впливу додаткових змінних методу AdaBoost Regressor

4.6.8 Метод Gradient Boosting Regressor

Gradient Boosting Regressor – метод, що використовує градієнтне підсилення. Цей метод навчає послідовну серію простих моделей, кожна з яких намагається виправити помилки попередньої.

Необхідні параметри (рис. 4.42) для роботи методу: кількість дерев, швидкість навчання, максимальна глибина, мінімальна кількість вибірок для розбиття, мінімальна кількість вибірок у листі.

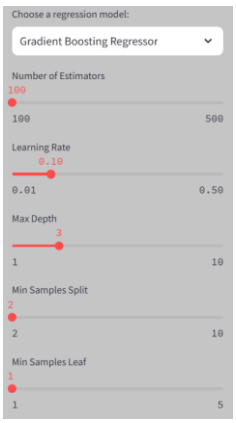

Рисунок 4.42 – Параметри методу Gradient Boosting Regressor

За напрямком Chiwan - Lyon протестовано різні значення параметрів. Виявилося, що кількість дерев (181), швидкість навчання (0.35), максимальна

глибина (5), мінімальна кількість вибірок для розбиття (5) і мінімальна кількість вибірок у листі (4) найкраще впливають на результат.

Загалом, модель демонструє хорошу ефективність і може бути корисною для практичного використання, з подальшим покращенням (рис.4.43).

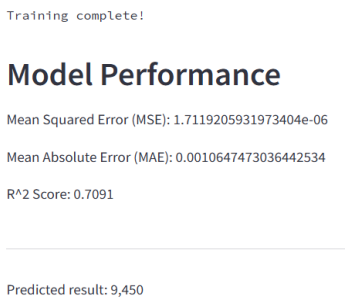

Рисунок 4.43 – Оцінки роботи методу Gradient Boosting Regressor

Тип товару найбільше впливає на результат, що значно більше за вплив ваги (рис. 4.44). Це є очевидним, бо від якості, матеріалу, попиту вартість змінюватиметься.

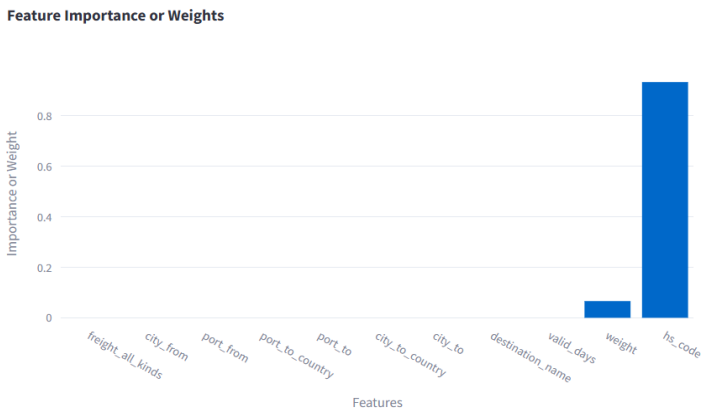

Рисунок 4.44 – Діаграма впливу додаткових змінних методу Gradient Boosting Regressor

4.6.9 Порівняння метрик оцінки якості моделей

Метрики дозволили оцінити якість прогнозування моделі. Використання кількох показників забезпечує повнішу картину продуктивності, що дозволяє врахувати всі аспекти для вибору оптимального методу. Результати оцінок якості прогнозування для різних методів на наборі даних Merged_Leads зведено в порівняльну таблицю (див. табл. 4.1).

Таблиця 4.1 – Порівняльна таблиця метрик якості прогнозування

№	Методи	MSE середньо-квадратична помилка	MAE середньо-абсолютна помилка	R2 Score коефіцієнт детермінації
1	Ridge	0.000058358508	0.005674406903	0.5767
2	Lasso	0.000271505861	0.011780219449	-1.0455
3	Random Forest Regressor	0.000004331048	0.001959434143	0.7123
4	Decision Tree Regressor	0.000017514650	0.004115505000	0.8511
5	XGBoost Regressor	0.000022449893	0.004567007258	0.7935
6	AdaBoost Regressor	0.000002687634	0.001248820876	0.8282
7	Gradient Boosting Regressor	0.000001711921	0.001064747304	0.7091

Найкращі результати показали методи Gradient Boosting Regressor та Decision Tree Regressor, які мали найнижчі значення помилок і найвищі R2 Score. Низькі результати Ridge і Lasso регресій вказують на те, що ці моделі можуть бути не найкращим вибором для даної задачі через свою лінійність і обмежену гнучкість у порівнянні з нелінійними. Ці моделі могли не враховувати всі важливі змінні, взаємодії між ними.

5 РЕКОМЕНДАЦІЇ, СПОСТЕРЕЖЕННЯ І ПЕРСПЕКТИВИ РОЗВИТКУ

Під час проведення дослідження виявлено недоліки у вихідних даних, які потребують уваги та подальшого вдосконалення. У наборах даних багато пропущених значень та дублікатів. Незважаючи на достатню кількість ознак, багато з них можна об’єднати або вилучити, оскільки вони мають однакові значення. Категоріальні дані не стандартизовані, наприклад, у деяких випадках вказано просто місто прибуття, тоді як у інших - конкретна адреса з поштовим індексом, корисніше було б використовувати координатний формат для відображення місць призначення. Назви країн також не доведено до єдиного стандарту. Неузгодженість даних призводить до складностей в об’єднанні наборів. Виявилися ознаки (стовпці), в яких було лише одне спостереження, такі стовпці довелося видалити, оскільки вони не несли жодної корисної інформації і лише створювали зайвий шум у даних, ускладнюючи аналіз.

Для уникнення виявлених проблем та забезпечення високої якості даних, компанія отримала рекомендації, деякі з них вже почали впроваджувати:

– проведення систематичного моніторингу даних;

– визначення підходів заповнення пропущених даних, наприклад використання середніх або медіанних значень для числових даних, найбільш частотних значень для категоріальних даних або застосування алгоритмів машинного навчання для прогнозування відсутніх значень;

– впровадження єдиного стандарту для назв ознак, об’єднання тих, що повторюються, визначивши одну основну назву;

– створення системи автоматичної конвертації і перевірки одиниць вимірювання та форматування даних;

– проведення навчання співробітників щодо важливості уніфікації категоріальних даних;

– визначення переліку ознак, які ще до сих пір не враховувалися, але які є значущими для повноти роботи певних процесів та їх додавання до наборів. Наприклад, для задачі ціноутворення не вистачало типу вантажу (рідина, твердий.), об’єму, відстані між пунктами, наявність додаткових послуг (страховка), інформації про перевізника і його рейтингу.

В ході проведеного дослідження було виявлено цікаві залежності. Статистичний аналіз, проведений за допомогою ANOVA тесту, продемонстрував, що вартість товару значно більше залежить від того, куди він направляється, ніж звідки. Найбільше перевезень здійснюється у Валенсію (Іспанія), що робить це місто вельми перспективним з точки зору потенційних можливостей для розвитку бізнесу в сфері логістики.

Для наземних перевезень більшість вантажів мають невеликий обсяг і не потребують використання повного або стандартного контейнера, що робить тип LTL (неповне завантаження вантажівки) більш економічно вигідним. Навпаки, вантажі, що транспортуються в повних контейнерах, зазвичай перевозяться залізницею.

За допомогою методу асоціативних правил визначилося, що маршрути між Шанхаєм (Китай), Кальяо (Перу) та Кобе (Японія) є одними з найбільш часто використовуваних для перевезення 20-футових стандартних контейнерів перевізником COSCO. Ці сполучення відіграють ключову роль у міжнародних вантажних перевезеннях, оскільки дозволяють ефективно транспортувати вантажі між Азією та Латинською Америкою, а також всередині Азії.

Виявлення двох основних сегментів у морських перевезеннях за допомогою методу кластеризації – швидкі і дешеві перевезення та найдовші і найдорожчі – дозволяє покращити логістичні процеси. Комбінуючи ці дані з інформацією про міста із запитів клієнтів, можна сконцентруватися на інвестуванні в певні рентабельні маршрути. Для клієнтів, які віддають

перевагу швидким і дешевим перевезенням, можна розробити маршрути з мінімальною кількістю зупинок та швидкою обробкою вантажів у портах, що зменшить час доставки та знизить витрати. А для тих, хто готовий платити більше, запропонувати додаткові послуги, що підвищать якість обслуговування.

Згідно з наявною інформацією, фактори, що впливають на прогнозування вартості перевезень, різняться в залежності від виду транспорту. Вибір перевізника має найбільший вплив на вартість авіаперевезень, оскільки ціни різних авіакомпаній можуть значно відрізнятися навіть на одному й тому ж маршруті. На формування остаточної вартості наземних перевезень впливають вага, кількість та тип товару. Застосування цих знань на практиці сприятиме стратегічному розвитку бізнесу та підвищенню його конкурентоспроможності на ринку логістики.

У цьому веб-додатку є потенціал для подальшого вдосконалення:

– інтеграція з базою даних замість використання csv-файлів дозволить підвищити ефективність та безпеку зберігання та обробки інформації;

– надання можливості користувачам отримувати загальну таблицю з оцінками якості всіх методів машинного навчання одночасно дозволить їм зручно порівнювати результати різних моделей та вибирати найефективніший метод для їхньої конкретної задачі;

– додавання тултипів до елементів інтерфейсу, що допоможе отримувати додаткову інформацію або пояснення щодо функціональності та параметрів і підвищить зручність використання додатку серед користувачів різного рівня підготовки;

– створення короткого навчального посібника допоможе новим користувачам швидко орієнтуватися в функціоналі додатку та зрозуміти взаємозв'язки між його основними компонентами.

Впровадження цих удосконалень сприятиме покращенню користувацького досвіду та функціональності веб-додатку.

ВИСНОВКИ

Проведено аналіз предметної області та існуючих рішень. Здійснено розвідковий аналіз даних та їх підготовку до навчання моделі. Розглянуто різні методи машинного навчання, визначено їх переваги та недоліки.

Створено зручний веб-додаток для користування працівниками існуючої логістичної компанії за допомогою мови програмування Python. Використано фреймворк Streamlit для розробки веб-додатка, однією з ключових переваг якого є те, що він підтримує багато популярних, легко інтегруємих бібліотек та інструментів Python для аналізу даних, візуалізації, машинного навчання, таких як Pandas, NumPy, Plotly, NetworkX, Scikit-learn, Mlxtend.

Додаток дозволяє аналізувати дані за різноманітними параметрами, створювати діаграми, графіки та гістограми, а також застосовувати різні методи машинного навчання для прогнозування числових змінних, зокрема вартості перевезення. Користувачі можуть порівнювати якість прогнозів за допомогою метрик середньоквадратичної помилки (MSE), середньої абсолютної помилки (MAE) та коефіцієнта детермінації (R2 Score). Процес налаштування гіперпараметрів допомагає краще зрозуміти можливості та обмеження моделі машинного навчання.

Менеджери, бізнес-аналітики, інженери з машинного навчання мають можливість використовувати додаток для глибокого аналізу даних, ефективного прийняття рішень, оптимізації логістичних процесів, виявлення тенденцій та прогнозування важливих бізнес-показників.

Дана робота протестована в компанії SeaRates та рекомендована до використання (додаток Д).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Томас Х. Кормен Алгоритмы: построение и анализ / Томас Х. Кормен, Чарльз І. Лейзерсон, Рональд Л. Ривеста та ін. / Вильямс, 2005

2) О. Мартинюк Матеріали для самостійного вивчення курсу «Математичне програмування»/ [Електронний ресурс] – Режим доступу: http://dspace.wunu.edu.ua/bitstream/316497/638/1/методичка с мп.pdf

3) Machine Learning in Logistics: Top Use Cases & Implementation [Електронний ресурс] / – Режим доступу: https:// www.itransition.com/machine-learning/logistics

4) Моделювання поведінки споживача на основі методів машинного навчання / [Електронний ресурс] / – Режим доступу: http://repository.hneu.edu.ua/bitstream/123456789/26902/1/

5) Andreas C. Müller, An Introduction to Machine Learning with Python (O'Reilly) / Andreas C. Mueller, Sarah Guido. / O'Reilly Media, Inc., 2016.

6) Gareth James An Introduction to Statistical Learning with Aplications in Python / Gareth James, Daniela Witten, Trevor Hastie / First Printing, 2023

7) Principal Component Analysis (PCA) in R Tutorial | DataCamp [Електронний ресурс] / – Режим доступу: https:// www.datacamp.com/ tutorial/ pca-analysis-r

8) Aristidis Likas The global k-means clustering algorithm/ [Електронне видання] / Pattern Recognition, 2003 – Режим доступу: https://inria.hal.science/inria-00321493/document

9) What is Association Rule Learning? [Електронний ресурс] / – Режим доступу: https://medium.com/ @ainsupriyofficial/ what-is-association- rule-learning

10) Мациєвська А. О. Вирішення задач у сфері логістики за допомогою методів машинного навчання / Пенко В. Г. // Інформатика, інформаційні

системи та технології: тези доповідей XX Всеукр. конференції студентів і молодих науковців. Одеса, 26 квітня 2024 р. – Одеса, 2024.

11) Allen B. Downey Think Stats: Probability and Statistics for Programmers / Green Tea Press, 2011

12) Streamlit documentation [Електронний ресурс] / – Режим доступу: https://docs.streamlit.io/

13) Scikit learn [Електронний ресурс] / – Режим доступу: https:// scikit-learn.org/stable/user_guide.html

14) David M. Lane Analysis of Variance Designs Chapter 15: Introduction to ANOVA / [Електронний ресурс] – Режим доступу: https://onlinestatbook.com/2/analysis_of_variance/anova.pdf

15) Є. Левус Аналіз алгоритму Apriori для структурованих і неструктурованих даних / Є. Левус, Н. Нечипір / Вид-во LAP LAMBERT Academic Publishing, 2019

ДОДАТОК А

Файл main.py

```
import pandas as pd
import numpy as np
import streamlit as st
import plotly.express as px
from scipy import stats
# Set page configuration
st.set_page_config(page_title='Logistics ML Analysis System', layout='wide')
# Title
st.title(
    "Machine Learning for Logistics Management: Pricing, Budget Planning, and
Demand Forecasting")
# Sidebar for file selection
st.sidebar.header("File Selection")
file_paths = {
    'Air Rates': 'data/rates_air.csv',
    'Land Rates': 'data/rates_land.csv',
    'Sea Rates': 'data/rates_sea.csv',
    'Merged Leads (default)': 'data/merged_leads_land_not_null.csv'
}
selected_file = st.sidebar.selectbox("Choose a dataset:",
                                     options=list(file_paths.keys()), index=3)
df = pd.read_csv(file_paths[selected_file])
st.sidebar.text(f"Loaded File: {selected_file}")
# Convert hs_code to string type if it exists in the dataframe
if 'hs_code' in df.columns:
    df['hs_code'] = df['hs_code'].astype(str)
# Display DataFrame with dynamic view control
num_rows = st.sidebar.slider('Number of rows to display:', min_value=5,
 max_value=100, value=20)
st.markdown("## Data Overview")
st.dataframe(df.head(num_rows), width=1500, height=600)

# Setup columns for stats and interactive plots
col1, col2, col3 = st.columns(3)
# Display basic data stats in a wider format
with col1:
    st.subheader("Data Summary")
    st.write("Number of rows:", df.shape[0])
    st.write("Number of columns:", df.shape[1])
    st.write("Missing values:", df.isnull().sum().sum())
    st.write("Duplicates:", df.duplicated().sum())
# Interactive plot setup with dynamic column selection
with col2:
    st.subheader("Interactive Data Plot")
    all_columns = df.columns.tolist()
    num_columns = df.select_dtypes(include=np.number).columns.tolist()
    cat_columns = df.select_dtypes(include='object').columns.tolist()

    x_axis = st.selectbox('Choose the X-axis:', all_columns, index=0,
                          key='x_axis_selectbox')
```

```
    if x_axis in num_columns:
        y_axis = st.selectbox('Choose the Y-axis:', num_columns,
                              index=1 if len(num_columns) > 1 else 0,
                              key='y_axis_num_selectbox')
    else:
        y_axis = st.selectbox('Choose the Y-axis:', cat_columns, index=0,
                              key='y_axis_cat_selectbox')

    plot_type = st.radio("Select plot type:",
                         ('Scatter Plot', 'Line Plot', 'Bar Plot'),
                         key='plot_type_radio')
    if plot_type == 'Scatter Plot' and x_axis in num_columns and y_axis in
num_columns:
        fig = px.scatter(df, x=x_axis, y=y_axis,
                         title=f'Scatter Plot of {x_axis} vs {y_axis}')
    elif plot_type == 'Line Plot' and x_axis in num_columns and y_axis in
num_columns:
        fig = px.line(df, x=x_axis, y=y_axis,
                      title=f'Line Plot of {x_axis} vs {y_axis}')
    elif plot_type == 'Bar Plot' and x_axis in cat_columns:
        fig = px.bar(df, x=x_axis, title=f'Count Plot of {x_axis}')
    else:
        st.warning('Please select appropriate axes for the chosen plot type.')
        fig = None
    if fig:
        st.plotly_chart(fig)

# Histogram for any numerical column
with col3:
    st.subheader("Data Distribution Plot")
    selected_column = st.selectbox('Select a numerical column for histogram:',
num_columns, key='hist_column_selectbox')
    if selected_column:
        hist_fig = px.histogram(df, x=selected_column,
                                title=f'Histogram of {selected_column}')
        st.plotly_chart(hist_fig)

# Pie chart for categorical data
if cat_columns:
    st.markdown("## Categorical Data Composition")
    categorical_column = st.selectbox('Select a categorical column:',
                                      cat_columns, index=0,
                                      key='pie_cat_selectbox')
    if df[
        categorical_column].nunique() < 10:  # Only make a pie chart if there
are fewer than 10 unique categories
        pie_fig = px.pie(df, names=categorical_column,
                         title=f'Pie Chart of {categorical_column}')
        st.plotly_chart(pie_fig)
    else:
        st.write(
       "Selected column has too many unique categories for a pie chart.")

# Data Report
st.markdown("## Data Report")
st.write("Descriptive Statistics:")
st.dataframe(df.describe())

# Correlation matrix for numerical data
show_corr = st.checkbox('Show correlation matrix for numerical data',
                        value=True)
if show_corr:
    st.subheader("Correlation Matrix")
```

```
    numerical_data = df.select_dtypes(include=np.number)
    if not numerical_data.empty:
        corr_matrix = numerical_data.corr()
        fig_corr = px.imshow(corr_matrix, text_auto=True, aspect='auto',
                             title='Relationships between numerical features')
        st.plotly_chart(fig_corr)
    else:
        st.write("No numerical data available for correlation.")

# ANOVA test to show influence of categorical value on continuous values
st.markdown("## ANOVA Test")
if cat_columns and num_columns:
    st.subheader("ANOVA Test: Influence of Categorical on Continuous Data")
    selected_cat = st.selectbox('Select a categorical column:', cat_columns,
key='anova_cat_selectbox')
    selected_num = st.selectbox('Select a numerical column:', num_columns,
key='anova_num_selectbox')

    if selected_cat and selected_num:
        grouped_data = df[[selected_cat, selected_num]].dropna()
        groups = [group[selected_num].values for name, group in
                  grouped_data.groupby(selected_cat)]
        if len(groups) > 1:
            anova_result = stats.f_oneway(*groups)
            st.write(
                f"ANOVA Test Results for {selected_cat} influencing
{selected_num}:")
            st.write(f"F-statistic: {anova_result.statistic:.4f}")
            st.write(f"P-value: {anova_result.pvalue:.4f}")
            if anova_result.pvalue < 0.05:
                st.success(
                    f"The p-value is less than 0.05, indicating a significant
influence of {selected_cat} on {selected_num}.")
            else:
                st.warning(
                    f"The p-value is greater than 0.05, indicating no
significant influence of {selected_cat} on {selected_num}.")
        else:
            st.warning(
                "ANOVA test requires at least two groups. Please select a
different categorical column.")

# Downloadable data report
st.markdown("## Download Data Report")
@st.cache_data
def convert_df_to_csv(d):
    return d.to_csv().encode('utf-8')

csv = convert_df_to_csv(df)
st.download_button("Download Data Report", csv, "data_report.csv", "text/csv",
key='download-csv')
```

ДОДАТОК Б

Файл Association_Rule.py

```
import pandas as pd
import streamlit as st
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
from streamlit_agraph import agraph, Node, Edge, Config
import networkx as nx

st.title("Advanced Association Rule Mining Dashboard")

@st.experimental_singleton
def load_data(path):
    """ Load data selectively with memory optimization. """
    return pd.read_csv(path)
@st.experimental_singleton
def preprocess_data(df, top_n):
    """ Convert all data to string to prevent type errors and prepare
transactions. """
    # Select top n columns with the most unique values
    col_unique_counts = df.nunique().sort_values(ascending=False)
    top_columns = col_unique_counts.head(top_n).index.tolist()
    df = df[top_columns].applymap(str)
    transactions = df.apply(lambda row: row.dropna().tolist(), axis=1).tolist()
    encoder = TransactionEncoder()
    encoded_array = encoder.fit(transactions).transform(transactions)
    return pd.DataFrame(encoded_array, columns=encoder.columns_)
@st.experimental_memo
def compute_rules(encoded_df, min_support, min_confidence, max_rules):
    """ Compute frequent itemsets and association rules based on user-defined
thresholds. """
    frequent_itemsets = apriori(encoded_df, min_support=min_support,
                                use_colnames=True)
rules = association_rules(frequent_itemsets, metric="confidence",
                          min_threshold=min_confidence)
    rules['antecedents'] = rules['antecedents'].apply(
        lambda x: ', '.join(list(x)))
    rules['consequents'] = rules['consequents'].apply(
        lambda x: ', '.join(list(x)))
    # Limit the number of rules
    rules = rules.head(max_rules)
    return rules
def draw_interactive_graph(rules, level_separation, node_spacing,
                           tree_spacing):
    """ Draw an interactive graph using streamlit-agraph. """
    G = nx.DiGraph()
    for index, row in rules.iterrows():
        antecedents = row['antecedents']
        consequents = row['consequents']
        lift = row['lift']
        confidence = row['confidence']

   G.add_edge(antecedents, consequents, lift=lift, confidence=confidence)
```

```
    nodes = []
    edges = []
    for node in G.nodes:
        nodes.append(
        Node(id=node, label=node, size=25, shape="ellipse", title=node))

    for edge in G.edges:
        source, target = edge
        lift = G[source][target]['lift']
        confidence = G[source][target]['confidence']
        edges.append(Edge(source=source,
        label=f"Lift: {lift:.2f}, Confidence: {confidence:.2f}",
        target=target))

    config = Config(
        width=950,
        height=700,
        directed=True,
        physics=False,
        hierarchical=True,
        layout={"hierarchical": {"enabled": True,
                                 "levelSeparation": level_separation,
                                 "nodeSpacing": node_spacing,
                                 "treeSpacing": tree_spacing,
                                 "blockShifting": True,
                                 "edgeMinimization": True,
                                 "parentCentralization": True,
                             "direction": "LR", "sortMethod": "hubsize"}},
                             interaction={"hover": True},
        edges={
            "smooth": {"type": "cubicBezier", "forceDirection": "horizontal",
"roundness": 0.5}}
    )

    return agraph(nodes=nodes, edges=edges, config=config)

# Sidebar - Dataset selection
st.sidebar.header("Dataset Selection")
file_paths = {
    'Air Rates': 'data/rates_air.csv',
    'Land Rates': 'data/rates_land.csv',
    'Sea Rates': 'data/rates_sea.csv',
    'Merged Leads (default)': 'data/merged_leads_land_not_null.csv'
}
selected_file = st.sidebar.selectbox("Choose a dataset:",
                                     options=list(file_paths.keys()), index=3)
path = file_paths[selected_file]

# Load data with optimizations
df = load_data(path)
columns_to_delete=['create_date','valid_from','valid_to','point_id','origin_c
ountry','origin_code','destination_country','load_type','destination_code','r
equest_date' ]
for column_to_delete in columns_to_delete:
    if(column_to_delete in df.columns):
        df=df.drop(column_to_delete,axis=1)

# User input for number of top columns to consider
top_n = st.sidebar.slider("Number of top columns to consider", 1,
len(df.columns), 5)

# User input for graph layout settings
level_separation = st.sidebar.slider("Level Separation", 100, 1000, 300, 50)
```

```
node_spacing = st.sidebar.slider("Node Spacing", 50, 500, 200, 50)
tree_spacing = st.sidebar.slider("Tree Spacing", 100, 1000, 300, 50)

# User input for rule filtering
max_rules = st.sidebar.slider("Max Number of Rules", 10, 100, 50, 10)

# Data preprocessing and association rule mining
encoded_df = preprocess_data(df, top_n)
min_support = st.sidebar.slider("Minimum Support", 0.01, 0.5, 0.05, 0.01)
min_confidence = st.sidebar.slider("Minimum Confidence", 0.0, 1.0, 0.5, 0.1)
rules = compute_rules(encoded_df, min_support, min_confidence, max_rules)

# Displaying results
if not rules.empty:
    st.write("Generated Association Rules:")
    st.dataframe(rules)

    # Visualization: Interactive Network graph visualization
    if st.button("Show Interactive Network Graph"):
        st.write("Interactive Network Graph of Item Associations:")
        draw_interactive_graph(rules, level_separation, node_spacing,
                               tree_spacing)
else:
    st.write("No rules found with the current settings.")
```

ДОДАТОК В

Файл Clustering.py

```
import streamlit as st
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA, TruncatedSVD
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import plotly.express as px
import plotly.graph_objects as go

st.set_page_config(page_title='Logistics ML Analysis System', layout='wide')
st.title('Clustering Analysis')
# Sidebar for file selection
st.sidebar.header("File Selection")
file_paths = {
    'Air Rates': 'data/rates_air.csv',
    'Land Rates': 'data/rates_land.csv',
    'Sea Rates': 'data/rates_sea.csv',
    'Merged Leads (default)': 'data/merged_leads_land_not_null.csv'
}
selected_file = st.sidebar.selectbox("Choose a dataset:",
options=list(file_paths.keys()), index=3)
df = pd.read_csv(file_paths[selected_file])
st.sidebar.text(f"Loaded File: {selected_file}")

# Sidebar for user inputs
st.sidebar.header('Settings')
n_components = st.sidebar.selectbox('Select number of components:', options=[2,
3], index=0)

# Load and preprocess data
numeric_columns = df.select_dtypes(include=[np.number]).columns.tolist()
df_numeric = df[numeric_columns].dropna()
scaler = StandardScaler()
data_scaled = scaler.fit_transform(df_numeric)
# Dimensionality reduction
#reduction_method = st.sidebar.radio('Select dimensionality reduction method:',
('PCA', 'SVD'), key='reduction_method')
reduction_method = 'PCA';
if reduction_method == 'PCA':
    reducer = PCA(n_components=n_components)
else:
    reducer = TruncatedSVD(n_components=n_components)
data_reduced = reducer.fit_transform(data_scaled)
st.write('Processed Data:', pd.DataFrame(data_reduced, columns=[f'Component
{i+1}' for i in range(n_components)]))

# Clustering
k_values = range(1, min(11, len(data_reduced)))
inertias = []
for k in k_values:
    kmeans = KMeans(n_clusters=k, random_state=0)
    kmeans.fit(data_reduced)
    inertias.append(kmeans.inertia_)
```

```
# Elbow plot for determining k
fig = go.Figure()
fig.add_trace(go.Scatter(x=list(k_values), y=inertias, mode='lines+markers',
name='Inertia'))
fig.add_vline(x=3, line_width=2, line_dash="dash", line_color="green")
fig.update_layout(title='Elbow Method For Optimal k', xaxis_title='Number of
Clusters', yaxis_title='Inertia')
st.plotly_chart(fig)

# Select number of clusters
max_clusters = min(10, len(data_reduced))
n_clusters = st.slider('Select number of clusters:', min_value=2,
max_value=max_clusters, value=3, key='cluster_slider')

# Apply K-Means and visualize clusters
kmeans = KMeans(n_clusters=n_clusters, random_state=0)
clusters = kmeans.fit_predict(data_reduced)
df_numeric['Cluster'] = clusters

# Visualization of Clusters
st.subheader('Cluster Visualization')
if n_components == 2:
    fig = px.scatter(x=data_reduced[:, 0], y=data_reduced[:, 1],
color=clusters.astype(str),
                     labels={'x': 'Component 1', 'y': 'Component 2'},
title='Cluster Visualization')
elif n_components > 2:
    fig = px.scatter_3d(x=data_reduced[:, 0], y=data_reduced[:, 1],
z=data_reduced[:, 2] if n_components > 2 else 0,
                        color=clusters.astype(str), labels={'x': 'Component
1', 'y': 'Component 2', 'z': 'Component 3' if n_components > 2 else 'Component
2'},
                        title='Cluster Visualization')
st.plotly_chart(fig)

# Displaying statistics for each cluster
cluster_stats = df_numeric.groupby('Cluster').describe().stack(level=0)

# Description of what highlight colors mean
st.markdown('### Cluster statistics')
st.markdown('**Yellow** highlights the maximum values, **light green**
highlights the minimum values in each column.')
# Improve the display of cluster statistics with formatting
def highlight_max(s):
return ['background-color: yellow' if v == s.max() else '' for v in s]

def highlight_min(s):
return ['background-color:lightgreen' if v == s.min() else '' for v in s]

styled_stats = cluster_stats.style.apply(highlight_max).apply(highlight_min)
st.dataframe(styled_stats)
```

ДОДАТОК Г

Файл Regression methods.py

```
import streamlit as st
import pandas as pd
import numpy as np
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.model_selection import train_test_split, GridSearchCV,
RandomizedSearchCV
from sklearn.preprocessing import StandardScaler, FunctionTransformer
from sklearn.compose import ColumnTransformer, make_column_selector
from sklearn.pipeline import Pipeline
from sklearn.linear_model import Ridge, Lasso
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor,
AdaBoostRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.impute import SimpleImputer
from xgboost import XGBRegressor
import plotly.express as px
import plotly.graph_objects as go
from category_encoders import TargetEncoder
from time import time
from sklearn.utils import resample
import logging

# Set page config
st.set_page_config(page_title='Regression Analysis System', layout='wide')
# Title and description
st.title('Regression Analysis for Prediction')
st.markdown('''
This application predicts prices using various regression models. Choose a
dataset from the dropdown menu for demonstration.''')
# Load and preprocess data
@st.cache_data
def load_data(filename):
    df = pd.read_csv(filename)
    # Remove leading zeros in the 'price_usd' column if it exists
    if 'price_usd' in df.columns:
        df['price_usd'] = df['price_usd'].apply(lambda x:
int(float(str(x).replace(',', ''))) if pd.notnull(x) else x)
    return df

# Sidebar - Dataset selection and reset button
st.sidebar.header("Dataset Selection")
file_paths = {
    'Air Rates': 'data/rates_air.csv',
    'Land Rates': 'data/rates_land.csv',
    'Sea Rates': 'data/rates_sea.csv',
    'Merged Leads (default)': 'data/merged_leads_land_not_null.csv'
}
selected_file = st.sidebar.selectbox("Choose a dataset:",
options=list(file_paths.keys()), index=3)
df = load_data(file_paths[selected_file])
st.sidebar.text(f"Loaded File: {selected_file}")


if st.sidebar.button('Reset App'):
    st.experimental_rerun()
```

```
# Initialize origin and destination columns
origin_col = 'port_from'
destination_col = 'port_to'

# Handle different column names for origins and destinations
if 'port_from' not in df.columns:
    if 'origin_name' in df.columns:
        origin_col = 'origin_name'
    elif 'origin_country' in df.columns:
        origin_col = 'origin_country'
    else:
        st.error("No valid origin column found in the dataset.")
        st.stop()

if 'port_to' not in df.columns:
    if 'destination_name' in df.columns:
        destination_col = 'destination_name'
    elif 'destination_country' in df.columns:
        destination_col = 'destination_country'
    else:
        st.error("No valid destination column found in the dataset.")
        st.stop()

# Display available connections
st.sidebar.header("Available Connections")
if origin_col in df.columns and destination_col in df.columns:
    available_connections                    =                   df[[origin_col,
destination_col]].drop_duplicates()
    st.sidebar.dataframe(available_connections, width=300, height=200)
else:
    st.sidebar.warning(f"Columns  {origin_col}  and/or  {destination_col}  not
found in the dataset.")

# Exclude datetime columns
datetime_columns            =            df.select_dtypes(include=['datetime',
'datetimetz']).columns.tolist()

# Select only numeric columns
numeric_columns = df.select_dtypes(include=[np.number]).columns.tolist()
#categorical_columns = df.select_dtypes(include=[object]).columns.tolist()

# Sidebar - Model setup
st.sidebar.header("Model Setup")
target = st.sidebar.selectbox('Select the target variable for prediction:',
df.columns, index=df.columns.get_loc('price_usd') if 'price_usd' in df.columns
else 0)
features    =    st.sidebar.multiselect('Select    features    to    use:',
df.columns.drop([target] + datetime_columns), default=df.columns.drop([target]
+ datetime_columns).tolist())
encode_categorical   =   st.sidebar.radio("Encode   categorical   variables:",
["Frequency Encoding", "Target encoding"], index=0)

model_type  =  st.sidebar.selectbox('Choose  a  regression  model:',  ['Ridge',
'Lasso',  'Random  Forest  Regressor',  'Decision  Tree  Regressor',  'XGBoost
Regressor', 'AdaBoost Regressor', 'Gradient Boosting Regressor'])

# Hyperparameters Sidebar
```

```
alpha_ridge, solver_ridge, max_iter_ridge, alpha_lasso, max_iter_lasso,
n_estimators_rf, max_depth_rf, min_samples_split_rf, min_samples_leaf_rf,
max_depth_dt, min_samples_split_dt, min_samples_leaf_dt, learning_rate_xgb,
n_estimators_xgb, max_depth_xgb, subsample_xgb, colsample_bytree_xgb,
n_estimators_ada, learning_rate_ada, loss_ada, n_estimators_gb,
learning_rate_gb, max_depth_gb, min_samples_split_gb, min_samples_leaf_gb =
[None] * 25

if model_type == 'Ridge':
    alpha_ridge = st.sidebar.slider('Alpha', 0.01, 10.0, 1.0)
 solver_ridge = st.sidebar.selectbox('Solver', ['auto', 'svd', 'cholesky',
'lsqr', 'sparse_cg', 'sag', 'saga'])
  max_iter_ridge = st.sidebar.slider('Max Iterations', 100, 10000, 1000)
elif model_type == 'Lasso':
    alpha_lasso = st.sidebar.slider('Alpha', 0.01, 10.0, 1.0)
  max_iter_lasso = st.sidebar.slider('Max Iterations', 100, 10000, 1000)
elif model_type == 'Random Forest Regressor':
    n_estimators_rf = st.sidebar.slider('Number of Trees', 100, 500, 100)
  max_depth_rf = st.sidebar.slider('Max Depth', 1, 20, 10)
  min_samples_split_rf = st.sidebar.slider('Min Samples Split', 2, 10, 2)
    min_samples_leaf_rf = st.sidebar.slider('Min Samples Leaf', 1, 5, 1)
elif model_type == 'Decision Tree Regressor':
    max_depth_dt = st.sidebar.slider('Max Depth', 1, 20, 10)
  min_samples_split_dt = st.sidebar.slider('Min Samples Split', 2, 10, 2)
    min_samples_leaf_dt = st.sidebar.slider('Min Samples Leaf', 1, 5, 1)
elif model_type == 'XGBoost Regressor':
   learning_rate_xgb = st.sidebar.slider('Learning Rate', 0.01, 0.5, 0.1)
n_estimators_xgb = st.sidebar.slider('Number of Estimators',100,500,100)
    max_depth_xgb = st.sidebar.slider('Max Depth', 1, 10, 3)
    subsample_xgb = st.sidebar.slider('Subsample', 0.5, 1.0, 0.8)
colsample_bytree_xgb = st.sidebar.slider('Colsample by Tree',0.5,1.0,0.8)
elif model_type == 'AdaBoost Regressor':
n_estimators_ada = st.sidebar.slider('Number of Estimators', 50, 200, 50)
learning_rate_ada = st.sidebar.slider('Learning Rate', 0.01, 1.0, 0.1)
loss_ada = st.sidebar.selectbox('Loss Function', ['linear', 'square',
'exponential'])
elif model_type == 'Gradient Boosting Regressor':
n_estimators_gb = st.sidebar.slider('Number of Estimators',100, 500, 100)
    learning_rate_gb = st.sidebar.slider('Learning Rate', 0.01, 0.5, 0.1)
    max_depth_gb = st.sidebar.slider('Max Depth', 1, 10, 3)
    min_samples_split_gb = st.sidebar.slider('Min Samples Split',2,10, 2)
    min_samples_leaf_gb = st.sidebar.slider('Min Samples Leaf', 1, 5, 1)

# Hyperparameter tuning
st.sidebar.header("Hyperparameter Tuning")
search_type = st.sidebar.radio("Search Type", ["None", "RandomizedSearch",
"GridSearch"], index=0)
n_iter_search = st.sidebar.number_input("Number of iterations (for
RandomizedSearch)", min_value=1, value=10)
cv_folds = st.sidebar.number_input("Number of CV folds", min_value=2, value=3)

# Filtering based on origin and destination
origin = st.sidebar.selectbox(f'Select {origin_col.replace("_", " ")}:',
df[origin_col].unique())
destination = st.sidebar.selectbox(f'Select {destination_col.replace("_", "
")}:', df[destination_col].unique())
filtered_df = df[(df[origin_col] == origin) & (df[destination_col] ==
destination)]

# Check if filtered_df is empty
if filtered_df.empty:
    st.error("No data available for the selected origin and destination. Please
select different options.")
```

```
else:
    # Preprocessing configuration
    numeric_cols = make_column_selector(dtype_include=np.number)
    categorical_cols = make_column_selector(dtype_include=object)
    numerical_transformer = Pipeline(steps=[
        ('imputer', SimpleImputer(strategy='median')),
        ('scaler', StandardScaler())])
    categorical_transformer = TargetEncoder() if encode_categorical == "Target
encoding" else Pipeline([
        ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
('freq_enc', FunctionTransformer(lambda X: pd.DataFrame(X).apply(lambda x:
x.map(x.value_counts(normalize=True)), axis=0).values, validate=False))])
    preprocessor = ColumnTransformer(transformers=[
        ('num', numerical_transformer, numeric_cols),
        ('cat', categorical_transformer, categorical_cols)
    ], remainder='passthrough')

    # Model Initialization with Hyperparameters
    model_dict = {
        'Ridge':         Ridge(alpha=alpha_ridge,         solver=solver_ridge,
max_iter=max_iter_ridge) if model_type == 'Ridge' else Ridge(),
        'Lasso':    Lasso(alpha=alpha_lasso,    max_iter=max_iter_lasso)    if
model_type == 'Lasso' else Lasso(),
        'Random                         Forest                         Regressor':
RandomForestRegressor(n_estimators=n_estimators_rf,    max_depth=max_depth_rf,
min_samples_split=min_samples_split_rf,  min_samples_leaf=min_samples_leaf_rf)
if model_type == 'Random Forest Regressor' else RandomForestRegressor(),
        'Decision                       Tree                         Regressor':
DecisionTreeRegressor(max_depth=max_depth_dt,
min_samples_split=min_samples_split_dt,  min_samples_leaf=min_samples_leaf_dt)
if model_type == 'Decision Tree Regressor' else DecisionTreeRegressor(),
        'XGBoost   Regressor':   XGBRegressor(learning_rate=learning_rate_xgb,
n_estimators=n_estimators_xgb,                         max_depth=max_depth_xgb,
subsample=subsample_xgb, colsample_bytree=colsample_bytree_xgb) if model_type
== 'XGBoost Regressor' else XGBRegressor(),
        'AdaBoost Regressor': AdaBoostRegressor(n_estimators=n_estimators_ada,
learning_rate=learning_rate_ada,  loss=loss_ada)  if  model_type  ==  'AdaBoost
Regressor' else AdaBoostRegressor(),
        'Gradient                       Boosting                       Regressor':
GradientBoostingRegressor(n_estimators=n_estimators_gb,
learning_rate=learning_rate_gb,                         max_depth=max_depth_gb,
min_samples_split=min_samples_split_gb,  min_samples_leaf=min_samples_leaf_gb)
if model_type == 'Gradient Boosting Regressor' else GradientBoostingRegressor()
    }

    param_distributions = {
        'Ridge':     {'model__alpha':     np.linspace(0.01,     10.0,     100),
'model__solver':  ['auto',  'svd',  'cholesky',  'lsqr',  'sparse_cg',  'sag',
'saga']},
        'Lasso':     {'model__alpha':     np.linspace(0.01,     10.0,     100),
'model__max_iter': [100, 500, 1000, 5000, 10000]},
        'Random Forest Regressor': {'model__n_estimators': [100, 200, 300, 400,
500], 'model__max_depth': [None, 10, 20, 30], 'model__min_samples_split': [2,
5, 10], 'model__min_samples_leaf': [1, 2, 4]},
        'Decision Tree Regressor': {'model__max_depth': [None, 10, 20, 30],
'model__min_samples_split': [2, 5, 10], 'model__min_samples_leaf': [1, 2, 4]},
        'XGBoost Regressor': {'model__n_estimators': [100, 200, 300, 400, 500],
'model__learning_rate': [0.01, 0.1, 0.2, 0.3], 'model__max_depth': [3, 5, 7,
9], 'model__subsample': [0.6, 0.8, 1.0], 'model__colsample_bytree': [0.6, 0.8,
1.0]},
        'AdaBoost  Regressor':  {'model__n_estimators':  [50,  100,  150,  200],
'model__learning_rate':  [0.01,  0.1,  0.5,  1.0],  'model__loss':  ['linear',
'square', 'exponential']},
```

```
        'Gradient Boosting Regressor': {'model__n_estimators': [100, 200, 300,
400, 500], 'model__learning_rate': [0.01, 0.1, 0.2, 0.3], 'model__max_depth':
[3,     5,     7,     9],     'model__min_samples_split':     [2,     5,     10],
'model__min_samples_leaf': [1, 2, 4]}
    }

    # Sidebar - Trigger predictions
    if st.sidebar.button('Predict'):
        if target in df.columns and features:
            X = filtered_df[features]
            y = filtered_df[target]

            if len(X) < 100:
                st.warning("Insufficient  data,  generating  synthetic  data  to
ensure enough samples.")
                if y.nunique() <= 1:
                    noise = np.random.normal(0, 0.01, y.shape)
                    y_noisy = y + noise
                    # X_encoded = preprocessor.fit_transform(X)
                    X_encoded = preprocessor.fit_transform(X, y)

                    df_encoded  =  pd.DataFrame(X_encoded,  columns=[f"col_{i}"
for i in range(X_encoded.shape[1])])
                    df_encoded[target] = y_noisy

                    k_neighbors = min(5, len(df_encoded) - 1)
                    try:
                        df_resampled = df_encoded
                        X_resampled = df_resampled.drop(columns=[target])
                        y_resampled = df_resampled[target]
                    except ValueError as e:
                        st.error(f"Error during SMOGN: {e}")
                        X_resampled, y_resampled = X, y_noisy
                else:
                    X_resampled,  y_resampled  =  resample(X,  y,  replace=True,
n_samples=100, random_state=42)
            else:
                X_resampled, y_resampled = X, y

            while len(X_resampled) < 10:
                X_resampled,  y_resampled  =  resample(X,  y,  replace=True,
n_samples=10, random_state=42)
                numeric_X_resampled                                          =
X_resampled.select_dtypes(include=[np.number])
                numeric_X_resampled     +=     np.random.normal(0,     0.01,
numeric_X_resampled.shape)
                X_resampled[numeric_X_resampled.columns] = numeric_X_resampled
                y_resampled += np.random.normal(0, 0.01, y_resampled.shape)

            logging.info(f"X_resampled: {X_resampled}")
            logging.info(f"y_resampled: {y_resampled}")

            X_train, X_test, y_train, y_test = train_test_split(X_resampled,
y_resampled, test_size=0.2, random_state=42)

            model = model_dict[model_type]
            pipeline  =  Pipeline([('preprocessor',  preprocessor),  ('model',
model)])

            if search_type == "GridSearch":
                search                =                GridSearchCV(pipeline,
param_distributions[model_type], cv=cv_folds, verbose=3)
            elif search_type == "RandomizedSearch":
```

```
                search = RandomizedSearchCV(pipeline,
param_distributions[model_type], n_iter=n_iter_search, cv=cv_folds,
random_state=42, verbose=3)
            else:
                search = pipeline

            progress_bar = st.progress(0)
            status_text = st.empty()
            start_time = time()
            console_output = st.empty()

            search.fit(X_train, y_train)
            progress_bar.progress(100)
            status_text.text("Training complete!")
            end_time = time()

            y_pred = search.predict(X_test)

            st.header('Model Performance')
            mse = mean_squared_error(y_test, y_pred)
            mae = mean_absolute_error(y_test, y_pred)
            r2 = r2_score(y_test, y_pred)
            st.write(f'Mean Squared Error (MSE): {mse}')
            st.write(f'Mean Absolute Error (MAE): {mae}')
            st.write(f'R^2 Score: {r2:.4f}')
            st.write('---')
            # Round predicted result to an integer and add dollar sign
            predicted_result = round(np.mean(y_pred))
            formatted_price = f"{predicted_result:,}"
            st.write(f'Predicted result: {formatted_price}')

            if search_type in ["GridSearch", "RandomizedSearch"]:
                st.write("Best Hyperparameters:")
                st.write(search.best_params_)

            fig = px.scatter(x=y_test, y=y_pred, labels={'x': 'Actual', 'y':
'Predicted'}, title='Actual vs Predicted')
            fig.add_trace(go.Scatter(x=[y_test.min(), y_test.max()],
y=[y_test.min(), y_test.max()], mode='lines', line=dict(color='red',
dash='dash'), name='Ideal Fit'))
            st.plotly_chart(fig)

            if hasattr(model, 'feature_importances_'):
                importance = model.feature_importances_
            elif hasattr(model, 'coef_'):
                importance = model.coef_
            else:
                importance = [0] * len(features)

            if len(features) != len(importance):
                st.warning(f"Length mismatch: features({len(features)}) and
importance({len(importance)})")
                feature_importance_df = pd.DataFrame({'Features':
features[:len(importance)], 'Importance':
importance}).sort_values(by='Importance', ascending=False)
            else:
                feature_importance_df = pd.DataFrame({'Features': features,
'Importance': importance}).sort_values(by='Importance', ascending=False)

            feature_importance_df =
feature_importance_df.sort_values(by='Importance',
ascending=True).head(len(importance))
```

```
            features = feature_importance_df['Features'].tolist()
            importance = feature_importance_df['Importance'].tolist()

            fig_importance = px.bar(x=features, y=importance, labels={'x':
'Features', 'y': 'Importance or Weight'}, title='Feature Importance or
Weights')
            st.plotly_chart(fig_importance)

            st.write("Feature Importances")
            st.dataframe(feature_importance_df)
            csv = feature_importance_df.to_csv().encode('utf-8')
            st.download_button(label="Download Feature Importances", data=csv,
file_name='feature_importances.csv', mime='text/csv')
        else:
            st.error("Please select a valid target variable and at least one
feature for prediction.")
```

ДОДАТОК Д

Довідка про впровадження

"04" червня 2024р.

Анастасії МАЦИЄВСЬКІЙ
65098, Одеська область Одеса,
Хаджібейський район,
вул. Блока, 64
тел. 068-252-87-82

ДОВІДКА
про впровадження інформаційної системи

Виконана студенткою-бакалавром факультету математики, фізики та інформаційних технологій Одеського національного університету ім. І.І. Мечникова, спеціальності 126 Інформаційні системи і технології, Мацієвською А.О. дипломна робота на тему "Методи машинного навчання для аналізу простору станів предметної області" має практичну значущість і рекомендована до впровадження у практику діяльності SeaRates.

Рекомендовано до впровадження запропоновані бакалавром рекомендації щодо покращення наборів вхідних даних, а також застосування виявлених корисних показників.

В.о. Генерального Директора
Стефан РОГОВСЬКИЙ

"СІРЕЙТС ЮКРЕЙН"
Ідентифікаційний код 43382202