НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ
КАФЕДРА ПРОМИЛОВОЇ ЕЛЕКТРОНІКИ

«На правах рукопису»
УДК 621.311.1

«До захисту допущено»
Завідувач кафедри

_____ Ю.С. Ямненко
(підпис) (ініціали, прізвище)
“_____”_____2019 р.

Магістерська дисертація
на здобуття ступеня магістра

зі спеціальності 171 Електроніка
(код і назва)

освітня програма (спеціалізація) Електронні компоненти і системи

на тему: Машинне навчання на віртуальному ринку електроенергії

Виконав (-ла): студент (-ка) II курсу, групи ДС-81мп
(шифр групи)

Швець Михайло Юрійович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник к.т.н., доцент Хохлов Ю.В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент професор каф. ААЕ, д.т.н., проф. Найда С.А. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Консультант
по нормоконтролю к.т.н, доц. Батрак Л.М. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”

Факультет електроніки
(повна назва)

Кафедра промислової електроніки
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо - професійною програмою

Спеціальність 171 Електроніка
(шифр і назва)

Освітня програма (спеціалізація) Електронні компоненти і системи

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ Ю.С.Ямненко
(підпис) (прізвище ініціали)

«____» _____ 2019 року

1. З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Швецю Михайлу Юрійовичу
(прізвище, ім’я, по батькові)

1. Тема дисертації Машинне навчання на віртуальному ринку електроенергії

науковий керівник дисертації Хохлов Ю.В., к.т.н, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «07» листопада 2019 року № 3859-с

2. Термін подання студентом дисертації « 11 » грудня 2019 року

3. Об’єкт дослідження методи машинного навчання

4. Вихідні дані відомості про використання та генерацію електроенергії, а також погодні показники за 11,5 місяців з періодом фіксації даних 1 хвилина

5. Перелік завдань, які потрібно розробити аналітичний огляд реляційних та нереляційних баз даних, класифікація методів машинного навчання, порівняння точності для різних моделей машинного навчання, вибір найбільш вагомих критеріїв та факторів, які впливають на пропозицію і попит в електроенергії прогнозування використаної та згенерованої електроенергії

6. Орієнтовний перелік графічного (ілюстративного) матеріалу Презентація

7. Орієнтовний перелік публікацій 1. Швець М.Ю., Заруба Д.С., Хохлов Ю.В Порівняння SQL та NoSQL баз даних. Вчені записки Таврійського національного університету імені В.І. Вернадського. Серія: Технічні науки. 2018. Том 29(68) № 6, частина 2, с. 21-25. 2._ Заруба Д.С., Швець М.Ю. «Електронна система з безпроводовим каналом керування інтелектуальним протизавадним фільтром». III всеукраїнська науково-технічна конференція «Сучасні технології кіно та аудіовізуальних систем»: Тези доповідей – Київ, 9-10 грудня 2018 р. – с. 56-57. 3. Швець М.Ю., Хохлов Ю.В., Заруба Д.С. Машинне навчання для прогнозування споживання та генерації електроенергії. Мікросистеми, Електроніка та Акустика (редакція).

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання_06 листопада 2019 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Огляд літературних джерел машинного навчання	15.02.19-01.04.19	
2	Огляд та класифікація завдань машинного навчання	01.04.19-01.05.19	
3	Пошук необхідного набору даних	01.05.19-21.05.19	
4	Аналіз та обробка даних	22.06.19-01.08.19	
5	Прогнозування кіл-ті використаної та згенерованої е/е	01.08.19-01.09.19	
6	Порівняння точності моделей машинного навчання	01.09.19-01.10.19	
7	Виокремлення ваги факторів в моделі	01.10.19-01.11.19	
8	Економічний прорахунок	01.11.19-15.11.19	
9	Розробка стартап-проекту	15.11.19-01.12.19	

Студент _____ М. Ю. Швець
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Ю. В. Хохлов
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

Магістерська дисертація присвячена підготовці і аналізу даних для покращення передбачень кількості використаної та згенерованої електроенергії методами машинного навчання, а також оцінці важливості та впливу на прогнозування періоду доби, місяця, температури, вологості повітря та інших ознак. Набір даних, що використовувався в даній роботі, містить відомості про використання та генерацію електроенергії, а також погодні показники за 11,5 місяців з періодом фіксації даних 1 хвилина.

Оброблення даних ґрунтувалось на статистичних методах обробки інформації, визначенні кількості пропущених даних, лінійних залежностей між ознаками, сумісності типів даних. За допомогою мови програмування Python та бібліотек pandas, numpy, sklearn, matplotlib було написано код програми для попередньої обробки даних, порівняно точність передбачень для трьох моделей машинного навчання та визначено модель з найбільшою точністю передбачень для обраного набору інформації. Було обрано тип бази даних виходячи з критеріїв необхідних для машинного навчання.

В результаті дослідження вдалось досягнути покращення точності результатів передбачення. Для оцінки точності передбачень було використано коефіцієнт детермінації. Було прораховано собівартість згенерованої електроенергії за 1 кВт*год, та показано економічну вигоду при застосуванні прогнозування кількості використаної та згенерованої електроенергії.

Очікується, що результати досліджень суттєво сприятимуть подальшому розвитку передбачень методами машинного начання.

Ключові слова: машинне навчання, коефіцієнт кореляції Пірсона, коефіцієнт детермінації, Випадковий ліс.

АННОТАЦИЯ

Магистерская диссертация посвящена подготовке и анализу данных для улучшения предсказаний количества использованной и сгенерированной электроэнергии методами машинного обучения, а также оценке важности и влияния на прогнозирование времени суток, месяца, температуры, влажности воздуха и других признаков. Набор данных, используемый в данной работе, содержит сведения об использовании и генерации электроэнергии, а также погодные показатели за 11,5 месяцев с периодом фиксации данных 1 минута.

Обработка данных основывалась на статистических методах обработки информации, определении количества пропущенных данных, линейных зависимостях между признаками, совместимости типов данных. С помощью языка программирования Python и библиотек pandas, numpy, sklearn, matplotlib был написан код программы для предварительной обработки данных, проведено сравнение точности предсказаний для трех моделей машинного обучения и определена модель с наибольшей точностью. Был выбран тип базы данных исходя из критериев необходимых для машинного обучения.

В результате исследования удалось добиться улучшения точности для результатов предсказаний. Для оценки точности было использовано коэффициент детерминации. Была просчитана себестоимость сгенерированной электроэнергии за 1 кВт*ч, и показана экономическая выгода при применении прогнозирования количества использованной и сгенерированной электроэнергии.

Ожидается, что результаты исследований поспособствуют дальнейшему улучшению предсказаний методами машинного обучения.

Ключевые слова: машинное обучение, коэффициент корреляции Пирсона, коэффициент детерминации, Случайный лес.

ANNOTATION

The master's thesis is devoted to the preparation and analysis of data to improve predictions of the amount of used and generated electricity using machine learning methods. The importance and influence on predicting the time of day, month, year, temperature, humidity, atmospheric pressure, and other factors were determined. The dataset used in this article contains information on the consumption and generation of electricity and weather data for 11,5 months with a data fixing period of 1 minute.

Data processing was based on statistical methods of information processing, determination the amount of missing data, linear relationships between features, compatibility of data types. Using the Python programming language and libraries pandas, numpy, sklearn, matplotlib, a pre-processing code was written, the precision of the predictions for the three machine learning models was compared, and the model with the highest precision for the selected set of information was identified. The type of database was selected based on the criteria required for machine learning.

As a result of the research was increase accuracy for a prediction model. A determination coefficient was used to estimate the accuracy of the predictions. The cost of the 1 kWh generated electricity was calculated and the economic benefit was shown in applying the forecasted amount of used and generated electricity.

It is expected that the results of the research will significantly contribute to the further development of predictions by machine learning methods.

Keywords: machine learning, Pearson correlation coefficient, determination coefficient, Random forest.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних

ML – Machine Learning (машинне навчання)

LDA - Linear discriminant analysis (лінійний дискримінантний аналіз)

KNN - K-nearest neighbors (К-найближчих сусідів)

LVQ - Learning vector quantization (мережі векторного квантування)

MAE – mean absolute error (середня абсолютна помилка)

MAPE - mean absolute percentage error (середня абсолютна процентна похибка)

MSE – mean square error (середньоквадратична помилка)

ВСТУП

Актуальність теми. В теперешній час швидкими темпами відбувається розвиток та розповсюдження технології MicroGrid, основна задача якої забезпечення енергоефективності з використанням альтернативних джерел електроенергії як основних елементів мережі електроживлення. Тому необхідно забезпечити взаємовигідні умови для споживання і вироблення відновлювальної електроенергії.

Баланс виробництва і споживання електроенергії - це основа технологічної стійкості енергосистеми, його порушення позначається на якості електроенергії (відбувається деградація частоти і напруги в мережі), що знижує ефективність роботи обладнання. Короткострокове прогнозування навантаження (КПН) в основному націлене на прогнозування навантаження системи з випередженням часу від однієї години до семи днів, що необхідно для адекватного планування і роботи енергосистем. КПН традиційно є важливим компонентом систем управління енергоспоживанням (СУЕ) [1, 2, 3], оскільки воно надає вхідні дані для аналізу потоку навантаження і аналізу непередбачених обставин. Прогнозування навантаження також стало важливим компонентом енергетичних брокерських систем[4]. Це дає можливість керувати вартістю покупки електроенергії шляхом регулювання завантаження устаткування, переводячи, наприклад, основні обсяги споживання електроенергії в години і зони оптового ринку енергії з найменшою ціною.

Попередній обробці даних та методам прогнозування присвячені роботи таких науковців як Айвазян С.А, Бажинов А.Н., Бухтштабер В.М., Енюков И.С., Мешалкин Л.Д., Николенко С., Ричардс Д. та ін. Отже прогнозування даних, та їх попередній аналіз є актуальною темою досліджень.

Зв’язок роботи з науковими програмами, планами, темами. Дисертація була підготовлена відповідно до науково-дослідного плану

кафедри промислової електроніки Національного технічного університету України "Київський політехнічний інститут ім. Ігоря Сікорського.

Метою дослідження є покращення результатів передбачень моделей машинного навчання за допомогою попередньої обробки даних, визначення вагових коефіцієнтів, критеріїв та факторів які впливають на використання і генерацію електроенергії.

Для досягнення поставленої мети були вирішені наступні **завдання**:

- аналітичний огляд реляційних та нереляційних баз даних;
- класифікація методів машинного навчання;
- вибір найбільш вагомих критеріїв та факторів, які впливають на пропозицію і попит в електроенергії;
- прогнозування використаної та згенерованої електроенергії;
- порівняння точності для різних моделей машинного навчання.

Об'єктом дослідження є методи машинного навчання, де на основі отриманих даних система аналізує та робить передбачення майбутньої поведінки.

Предметом дослідження є залежність і попит на електроенергію при різних вагових коефіцієнтах, таких як: погода, пора року, час доби.

Наукова новизна даних досліджень полягає в автоматичному встановленні залежностей між кількістю використаної та згенерованої електроенергії від температури, вологості повітря, пори року, місяця, дня, години та виокремлення вкладу кожного із них. Підвищено точність передбачень методами машинного навчання за допомогою статистичних методів обробки інформації.

Практичне значення одержаних результатів полягає в економії коштів користувачів MicroGrid, а також в виробленні рекомендацій для:

1. Очищення і форматування даних
2. Попереднього аналізу даних
3. Вибору найбільш корисних ознак і створення нових більш репрезентативних

4. Перевірки моделі на тестовій вибірці
5. Інтерпретація результатів

Особистий внесок здобувача. Дисертаційна робота є узагальненням результатів теоретичних і експериментальних досліджень, проведених автором самостійно. У роботі, опублікованій із співавторами, дисертанту належать аналітичний огляд реляційних та нереляційних баз даних, аналіз переваг та недоліків SQL та NoSQL баз даних, тестування швидкодії систем керування SQL та NoSQL баз даних, проведення аналізу даних для передбачень, передбачення кількості використаної та згенерованої е/е методами машинного навчання, виокремлення ваги факторів при навчанні моделі машинного навчання Випадковий ліс, програмне налаштування.

Апробація результатів дисертації. Основні теоретичні положення та результати магістерського дослідження були презентовані у доповіді на науково-технічній конференції III Всеукраїнська науково-технічна конференція студентів, аспірантів та науковців "Сучасні технології кіно та аудіовізуальних систем", м. Київ, 9-10 грудня, 2019 р.

Публікації. Основний зміст дисертації відображений у 2 наукових працях, з них 1 опублікована та 1 знаходиться у редакції у наукових фахових виданнях за переліком ВАК України:

- Швець М.Ю., Заруба Д.С., Хохлов Ю.В Порівняння SQL та NoSQL баз даних. Вчені записки Таврійського національного університету імені В.І. Вернадського. Серія: Технічні науки. 2018. Том 29(68) № 6, частина 2, с. 21-25.
- Швець М.Ю., Хохлов Ю.В., Заруба Д.С. Машинне навчання для прогнозування споживання та генерації електроенергії. Мікросистеми, Електроніка та Акустика (редакція)

Структура та обсяг дисертації. Дисертація складається зі вступу, чотирьох розділів, висновків, списку використаних джерел із 50 найменувань та 1 додатка. Загальний обсяг дисертаційної роботи становить 90 сторінок, у тому числі 73 сторінки основного тексту, 28 рисунки та 17 таблиць.

1. ПРОГНОЗУВАННЯ ДАНИХ МЕТОДАМИ МАШИННОГО НАВЧАННЯ

Машинне навчання (Machine Learning) - великий підрозділ технології штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися. Розрізняють два типи навчання. Навчання по прецедентах, або індуктивне навчання, засноване на виявленні загальних закономірностей по частковим емпіричним даним. Дедуктивне навчання передбачає формалізацію знань експертів і їх перенесення в комп'ютер у вигляді бази знань. Дедуктивне навчання прийнято відносити до області експертних систем, тому терміни машинне навчання і навчання по прецедентах можна вважати синонімами.

Машинне навчання це підрозділ штучного інтелекту, який дозволяє будувати алгоритми, здатні навчатися самостійно. Зазвичай у кожної програми є якась мета, заради якої вона і створювалася. Це може бути рішення квадратного рівняння, зміна яскравості зображення і т.п. Тоді маючи вхідні дані (рівняння або картинку) і написану програму для вирішення завдання, ми можемо отримати потрібні нам вихідні значення (рішення або яскравий малюнок). У машинному навчанні ж головною метою є побудова алгоритму (програми) для того щоб перетворити вхідні значення у вихідні. Виходить, що алгоритм машинного навчання будує модель вирішення задачі, ґрунтуючись тільки на знаннях, які отримано з вхідного набору даних. І найголовніше полягає в розв’язку задачі на довільних вхідних даних, які не перетинаються з навчальними. Це називається узагальненням і саме завдяки цій властивості методи машинного навчання так ефективні.

Машинне навчання знаходиться на стику математичної статистики, методів оптимізації та класичних математичних дисциплін, але має також і власну специфіку, пов'язану з проблемами обчислювальної ефективності та перенавчання. Багато методів індуктивного навчання розроблялися як альтернатива класичним статистичним підходам. Багато методів тісно пов'язані з виокремленням інформації та інтелектуальним аналізом даних.

Машинне навчання - не тільки математична, а й практична, інженерна дисципліна. Чиста теорія, як правило, не призводить відразу до методів і алгоритмів, які можуть застосовуватися на практиці. Щоб змусити їх добре працювати, доводиться винаходити додаткові евристики, що компенсують невідповідність зроблених в теорії припущень умов реальних завдань. Практично жодне дослідження в машинному навчанні не обходиться без експерименту на модельних або реальних даних, що підтверджує практичну працездатність методу.

1.1. Підбір бази даних для машинного навчання

Для забезпечення якісних передбачень системи необхідна достатня кількість початкових даних, швидка взаємодія при внесенні та пошуку даних. Саме тому, попередньо було проведено аналітичний огляд реляційних та нереляційних баз даних, проаналізовано переваги та недоліки SQL та NoSQL баз даних[5].

База даних (БД) представляє упорядкований набір логічно взаємопов'язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб користувачів. У технічному розумінні включно й система управління БД[6].

Існує два основних типи баз даних: SQL і NoSQL – або реляційна і нереляційна база даних. Різниця полягає в тому як вони побудовані, який тип інформації зберігають і як саме.

Система керування базами даних (СКБД) - це комплекс програмних і мовних засобів, необхідних для створення баз даних, підтримання їх в актуальному стані та організації пошуку в них необхідної інформації.

СКБД відповідають за:

- пошук потрібних даних
- фізичне розміщення даних і їх описів

- відновлення і оновлення баз даних відповідно до змін у реальному світі (підтримка актуального стану)

Реляційні бази даних структуровані (рис.1.1.), їх можна порівняти з телефонною книжкою, яка зберігає номери телефонів і адреси. SQL база даних складається з двох або більше таблиць з стовпцями і рядками. Кожен рядок представляє запис (всі дані в одному рядку відносяться до одного і того ж об’єкта), і кожен стовпець збирає дуже конкретний тип інформації, таку як ім'я, вік, стать та номер телефону.

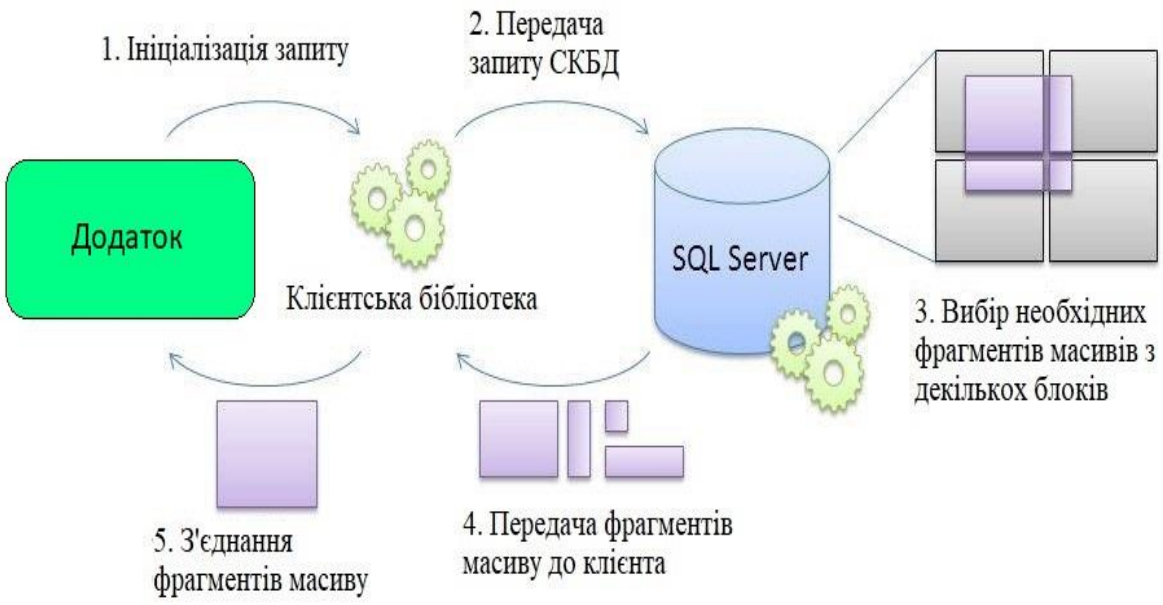

Рис. 1.1. Принцип роботи реляційної бази даних

Реляційні (SQL) сховища забезпечують найкращу суміш простоти, стійкості, гнучкості, продуктивності, масштабованості і сумісності. Але основною проблемою є те, що різноманітність додатків зростає, а отже виникає необхідність в збільшенні місця. І з ростом кількості баз даних, масштабованість починає переважати над іншими особливостями. Оскільки все більше додатків працюють в умовах високого навантаження, наприклад, таких як веб-сервіси, їх вимоги до масштабованості дуже швидко змінюються і сильно зростають. Саме тому почали з’являтися бази даних, що забезпечують інший механізм зберігання та видобування даних, ніж звичний підхід таблиць-

відношень в реляційних базах даних. Сплеск популярності був спричинений потребами Web 2.0 компаніями, такими як Facebook, Google, та Amazon.com.

Нереляційні (NoSQL) бази даних є документно-орієнтовані, їх співвідносять з папкою наповненою файлами. Однією з основних особливостей технології NoSQL є відмова від реляційної моделі. Кожне рішення в рамках технології NoSQL використовує свою власну модель. NoSQL бази даних все більше і більше використовуються в задачах із застосуванням Big Data та веб-додатках.

Переваги SQL бази даних:

1. Відповідність ACID [7] (атомарність, узгодженість, ізольованість, довговічність), що захищає цілісність бази даних, визначаючи точно, як транзакції взаємодіють з базою даних. Як правило, база даних NoSQL жертвує ACID для забезпечення гнучкості та швидкості обробки, але для електронної комерції та фінансових додатків переважним варіантом залишається база даних, сумісна з ACID.

2. Кожна таблиця містить одну або декілька категорій даних у стовпцях[8].

3. Кожен рядок містить унікальний екземпляр даних для категорій, визначених стовпцями[9].

4. Користувач може отримати доступ до даних з бази даних, не знаючи структури таблиці бази даних[8].

Недоліки SQL бази даних:

1. Масштабованість: користувачі повинні масштабувати реляційні бази даних на потужних серверах, які є дорогими та складними для обробки. Щоб масштабувати реляційну базу даних, вона повинна бути розподілена на декілька серверів[8].

2. Складність: дані SQL-сервера повинні бути вписані у таблиці. Якщо дані не вкладаються в таблиці, необхідно проектувати структуру бази даних, яка буде складною і важкою для обробки [8].

3. Вартість: реляційні системи керування базами даних (СКБД) вимагають дорогих систем зберігання та запатентованих серверів [9].

Переваги NoSQL бази даних:

1. Зберігання великих обсягів даних, які погано структуровані. БД NoSQL не обмежує типи даних, які можна зберігати разом, і дозволяє додавати нові типи. За допомогою баз даних на базі документів можна зберігати дані в одному місці, не визначаючи заздалегідь, які «типи» цих даних [10].

2. Швидша обробка даних, ніж в реляційних базах даних [8].

3. Дешевше: не реляційні бази даних використовують дешеві кластери товарних серверів для управління операціями та даними [9].

4. Краща масштабованість в порівнянні з реляційними базами даних. Проте NoSQL бази даних не повністю масштабовані у всіх ситуаціях [11].

Недоліки NoSQL бази даних:

1. Узгодженість даних: більшість NoSQL БД не виконують транзакції ACID. Натомість NoSQL покладається на принцип "кінцевої послідовності". Це забезпечує деякі переваги продуктивності, але це створює ризик того, що дані на одному вузлі бази даних можуть не синхронізуватися з даними іншого вузла [11];

2. Відсутність стандартизації: NoSQL не є специфічним типом інтерфейсу бази даних або програмування. Мова дизайну та запитів баз даних NoSQL різко відрізняється між різними продуктами NoSQL - набагато ширше, ніж серед традиційних баз даних SQL [11].

3. Деякі нереляційні БД погано розповсюджуються на декілька вузлів. Якщо БД не може відокремитись автоматично, вона не може автоматично збільшуватись або зменшуватись у відповідь на коливальний попит.

MongoDB надає декілька різних можливостей, таких як: гнучка модель даних, індексування і високошвидкісні запити, які значно спрощують навчання і використання алгоритмів машинного навчання в порівнянні з традиційними реляційними базами даних. Використання MongoDB в якості

серверної бази даних для зберігання і збільшення даних для навчання ML забезпечує сталість і підвищену ефективність.[12]

1.2. Класифікація завдань машинного навчання

Для прогнозування електроспоживання і побудови профілів клієнтів використовуються різні методи, зазвичай засновані на аналізі ретроспективної динаміки електроспоживання і діючих на нього факторів, виявленні статистичного зв'язку між ознаками і на побудові моделей. До недавнього часу найпоширенішими методами прогнозування були однофакторні прогнози по часових рядах, засновані на регресійних методах. Однак такі прогнози не здатні враховувати вплив на споживання електроенергії таких нерегулярних факторів, як погодні явища, коливання цін на паливо, поломки обладнання, тому на практиці слід застосовувати багатофакторне прогнозування, що дозволяє будувати прогноз з точністю, значно перевищує точність по часових рядах[13]. За типом задач (рис. 1.2) машинне навчання поділяють: навчання з вчителем, навчання без вчителя, навчання з частковим залученням вчителя, навчання з підкріпленням.

Рис. 1.2. Класифікація методів машинного навчання

Навчання з вчителем

Навчання з вчителем (supervised learning) передбачає наявність повного набору розмічених даних для тренування моделі на всіх етапах її побудови. [13]

При цьому типі навчання на вхід подається набір тренувальних прикладів, який зазвичай називають навчальним або тренувальним набором даних, і завдання полягає в тому, щоб продовжити вже відомі відповіді на новий досвід, виражений зазвичай у вигляді тестового набору даних. Наявність повністю розміченого датасета означає, що кожному прикладу в навчальному наборі відповідає рішення, яке алгоритм і повинен отримати. Таким чином, розмічений датасет з фотографій квітів навчить нейронну мережу, де зображена троянда, ромашка або нарциси. Основне припущення тут в тому, що дані, доступні для навчання, будуть чимось схожі на дані, на яких потім доведеться застосовувати навчену модель, інакше ніяке узагальнення буде неможливо. Коли мережа отримає нове фото, вона порівняє його з прикладами з навчального датасета, щоб передбачити відповідь.

Для «читання» тексту приклад навчання з учителем - це навчання моделі, яка будує дерева синтаксичного розбору пропозицій (які слова від яких залежать) по набору дерев, побудованих людьми для конкретних пропозицій. Припущення тут в тому, що дерева розбору будуються по одним і тим же законам, і модель, навчену на деякому наборі дерев розбору, можна буде застосувати і до нових пропозицій, що не входять в навчальний набір. Якщо це припущення порушиться, модель працювати не буде. Наприклад, якщо лінгвісти розмічали речення англійською мовою, то потім застосувати модель для читання на німецькій, де букви приблизно ті ж, але синтаксис зовсім інший, то результаті будуть зовсім некоректні.

В основному навчання з учителем застосовується для вирішення трьох типів задач:

- класифікації;
- регресії;

- ранжирування.

У задачі класифікації (рис. 1.3) потрібно поданий на вхід об'єкт визначити в один з (зазвичай кінцевого числа) класів. [13] Наприклад, розділити фотографії тварин на кішок, собак, коней і «все інше»; або по фотографії людського обличчя визначити, хто з ваших друзів у соціальній мережі на ній зображений. Якщо продовжувати приклад з мовою, то типова задача класифікації – це розмітка слів за частинами мови. Якість алгоритму оцінюється тим, наскільки точно він може правильно класифікувати нові фото з коал і черепахами.

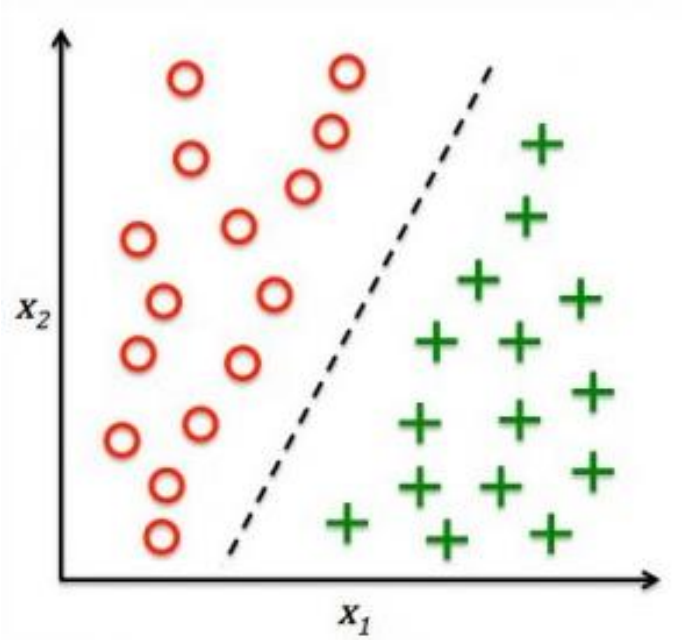

Рис. 1.3. Приклад задачі класифікації

Завдання регресії (рис. 1.4) пов'язані з безперервними даними. [14] Прикладами можуть слугувати, лінійна регресія, обчислює очікуване значення змінної *y*, враховуючи конкретні значення *x*, або по зростанню людини передбачити його вагу, зробити прогноз завтрашньої погоди, передбачити ціну акції або, скажімо, виділити на фотографії прямокутник, в котрому знаходиться людське обличчя - зробити це необхідно, щоб потім ці прямокутники подати на вхід згаданого вище класифікатора.

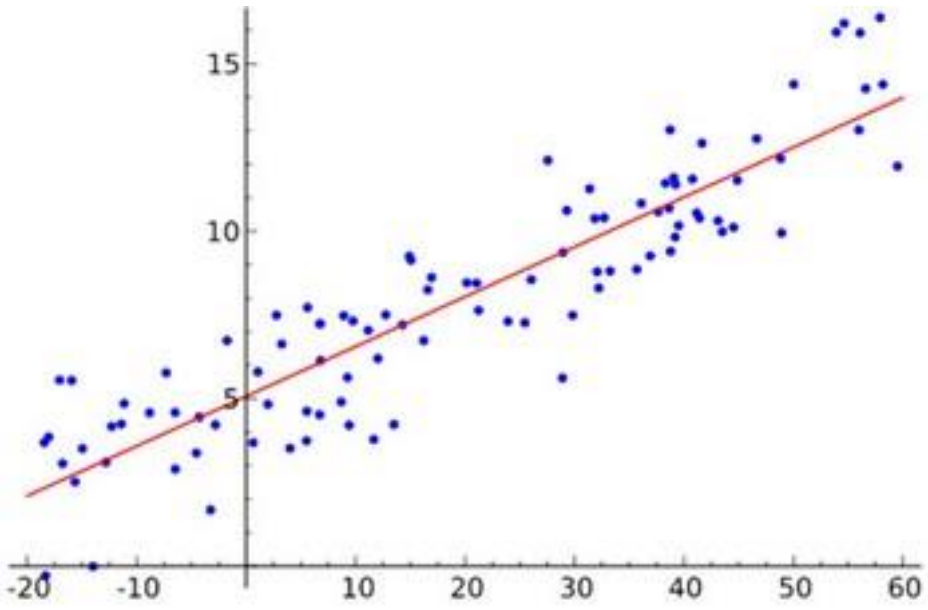

Рис. 1.4. Приклад задачі регресії

Розподіл на регресію і класифікацію, дуже умовний, можна легко придумати «проміжні» приклади. Але зазвичай ясно, яке завдання вирішується, і цей поділ має змістовний сенс: змінюються цільові функції і як наслідок, процес навчання. Є і інші види завдань, що не укладаються в цю класифікацію. Наприклад, в пошукових і рекомендаційних системах часто зустрічається завдання навчання ранжирування (learning to rank). Воно ставиться так: за наявними даними (в пошуковій системі це будуть тексти документів і минула поведінку користувачів) відранжувати, розставити наявні об'єкти в порядку зменшення цільової функції (в пошуковій системі вона називається релевантністю: наскільки даний документ підходить, щоб видати це у відповідь на даний запит).

Навчання без вчителя

У навчанні без вчителя (unsupervised learning) у моделі є набір даних, і немає явних вказівок, що з ним робити. Нейронна мережа намагається самостійно знайти кореляції в даних, витягуючи корисні ознаки і аналізуючи їх. Залежно від завдання модель систематизує дані по-різному.

Типовий приклад завдання навчання без вчителя - це кластеризація (clustering): потрібно розбити дані на заздалегідь невідомі класи по деякій мірі схожості так, щоб точки, віднесені до одного і того ж кластеру, були якомога ближче один до одного, як можна більш схожі, а точки з різних кластерів були б якомога далі один від одного, як можна менш схожі. Алгоритм підбирає схожі дані, знаходячи спільні ознаки, і групують їх разом. Наприклад, вирішивши завдання кластеризації, можна виділити сімейство генів з послідовностей нуклеотидів, або кластеризувати користувачів веб-сайту і персоналізувати його під кожен кластер, або сегментувати медичний список, щоб легко було зрозуміти, де захворювання.

Ще одне із завдань навчання без вчителя - це зниження розмірності, коли вхідні дані мають велику розмірність (наприклад, якщо у вас на вході розбитий на слова текст, розмірність буде обчислюватися десятками тисяч, якщо фотографії - мільйонами), а завдання полягає в тому, щоб побудувати уявлення даних меншої розмірності, яке тим не менше буде досить повно відображати вихідні дані. Наприклад, по представленню меншої розмірності можливо буде досить успішно реконструювати вихідні точки більшої розмірності. Це можна розглядати як окремий випадок загальної задачі виділення ознак (feature extraction). Часто використовують в автоенкодерах – вони приймають вхідні дані, кодують їх, а потім намагаються відтворити початкові дані з отриманого коду. Не так багато реальних ситуацій, коли використовують простий автоенкодер. Але варто додати шари і можливості розширюватись: використовуючи зашумлені і вихідні версії зображень для навчання, автоенкодери можуть видаляти шум з відеоданих, зображень або медичних сканів, щоб підвищити якість даних.

Третій найбільший загальний клас навчання без учителя - завдання оцінки щільності: нам дано точки даних $\{x_1, \ldots, x_N\}$ і можливо уявлення про те, звідки взялися ці точки, а потрібно оцінити розподіл $p(x)$, з якого вони вийшли.

Також існує завдання виявлення аномалій - навчання без вчителя використовують для знаходження викидів в даних (рис. 1.5).

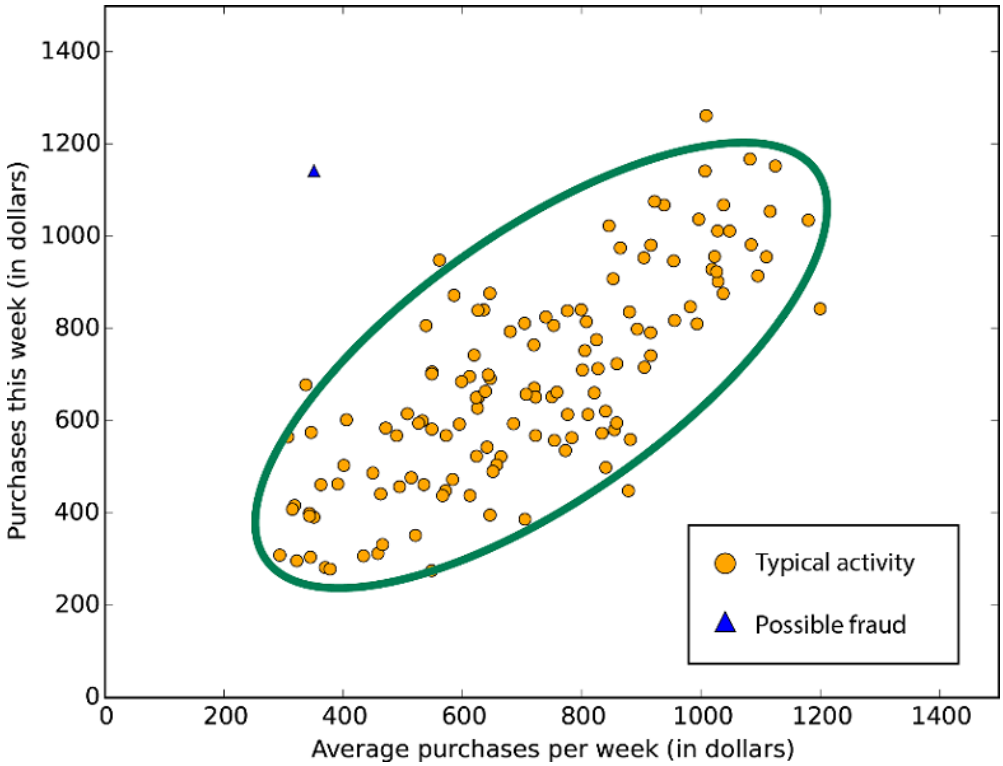

Рис. 1.5. Приклад задачі виявлення аномалій

Асоціації - деякі характеристики об'єкта корелюють з іншими ознаками. Розглядаючи пару ключових ознак об'єкта, модель може передбачити інші, з якими існує зв'язок.

У навчанні без вчителя складно обчислити точність алгоритму, тому що в цих даних відсутні «правильні відповіді» або мітки. Але розмічені дані часто ненадійні або їх занадто дорого отримати. У таких випадках, надаючи моделі свободу дій для пошуку залежностей, можна отримати хороші результати[14].

Навчання з частковим залученням вчителя

Нерідко в житті виникає щось середнє між навчанням з учителем і навчанням без учителя. Так зазвичай виходить тоді, коли нерозмічену приклади знайти дуже легко, а розмічені отримати складно. [14] Наприклад у все тому ж прикладі з синтаксичним розбором набрати скільки завгодно

нерозмічених текстів не представляє ніякої складності, а ось вручну намалювати навіть одне дерево нелегко. А розмічені дані для розпізнавання мови - то звукові файли, в яких вручну, відзначені (або хоча б перевірені) межі кожної фонеми, кожного звуку людської мови. Але це дуже складний і тривалий процес. А записати нерозмічені звуки живої людської мови набагато простіше. Таку ситуацію іноді називають навчанням з частковим залученням вчителя, або напівконтрольованним навчанням.

Навчання з частковим залученням вчителя (semi-supervised learning) - навчальний датасет містить як розмічені, так і розділені дані. Цей метод особливо корисний, коли важко отримати з даних важливі ознаки або розмітити всі об'єкти - трудомістке завдання.

Навчання з підкріпленням

Навчання з підкріпленням (reinforcement learning) - це різновид машинного навчання, при якому агент вчиться діяти в навколишньому середовищі, виконуючи дії і тим самим напрацьовує інтуїцію, після чого спостерігає результати своїх дій. При прийнятті рішення агент вивчає зворотний зв'язок, нові тактики і рішення здатні привести до більшого виграшу. Цей підхід використовує довгострокову стратегію - так само як в шахах: наступний найкращий хід може не допомогти виграти в кінцевому рахунку. Тому агент намагається максимізувати сумарну нагороду.

Це ітеративний процес. Чим більше рівнів з зворотного зв'язку, тим краще стає стратегія агента. Такий підхід особливо корисний для навчання роботів, які керують автономними транспортними засобами або інвентарем на складі. [13]

До областей в яких найбільш затребувані системи з навчання з підкріпленням можна віднести:

- безпілотні автомобілі
- ігрова індустрія
- робототехніка

- рекомендаційні системи

Агент і Середовища грають ключові ролі в алгоритмі навчання з підкріпленням. Середовище - це той світ, в якому доводиться працювати Агенту. Крім того, Агент отримує від Середовища підкріплючі сигнали (винагороду): це число, що характеризує, наскільки хорошим або поганим можна вважати поточний стан Сеедовища. Мета Агента - максимізувати сукупну винагороду, так званий «виграш». Для данного типу навчання використовується своя термінологія[15]:

- Стани: Стан - це повний опис світу, в якому не втрачено жодного фрагмента інформації, що характеризує цей світ. Це може бути позиція, фіксована або динамічна. Як правило, такі стани записуються у вигляді масивів, матриць або тензорів вищого порядку.
- Дія: Дія зазвичай залежить від умов навколишнього середовища, і в різних середовищах агент буде робити різні дії. Безліч допустимих дій агента записується в просторі, іменованому «простір дій». Як правило, кількість дій в просторі скінченне.
- Середовище: Це місце, в якому агент існує і з яким взаємодіє. Для різних середовищ використовуються різні типи винагород, стратегій, тощо
- Винагорода і виграш: Відстежувати функцію винагороди R при навчанні з підкріпленням потрібно постійно. Вона критично важлива при налаштуванні алгоритму, його оптимізації, а також у разі припинення навчання. Вона залежить від поточного стану світу, тільки що зроблених дій і наступного стану світу.
- Стратегії: стратегія - це правило, відповідно до якого агент обирає наступну дію. Набір стратегій також іменується «мозком» агента.

1.3. Методи оцінки точності передбачень

Помилка прогнозу – величина відхилення прогнозу від дійсного стану об'єкта. Якщо говорити про прогноз продажів, то це показник відхилення фактичних продажів від прогнозу.

Точність прогнозування є поняття прямо протилежне помилці прогнозування. Якщо помилка прогнозування велика, то точність мала і навпаки, якщо помилка прогнозування мала, то точність велика. По суті справи оцінка помилки прогнозу MAPE є зворотна величина для точності прогнозування - залежність тут проста.

Точність прогнозу в % = 100% - MAPE, зустрічається ще назва цього показника Forecast Accuracy. Практично немає матеріалів про прогнозування, в яких наведені оцінки саме точності прогнозу, хоча з точки зору здорового маркетингу коректніше говорити саме про високу точність. У рекламних статтях завжди буде написано про високу точність. Показник точності прогнозу виражається у відсотках:

- Якщо точність прогнозу дорівнює 100%, то обрана модель описує фактичні значення на 100%, тобто дуже точно. Потрібно відразу обмовитися, що такого показника ніколи не буде, основна властивість прогнозу в тому, що він завжди помилковий.
- Якщо 0% або негативне число, то зовсім не описує, і даній моделі довіряти не варто.

Вибрати відповідну модель прогнозу можна за допомогою розрахунку показника точності прогнозу. Модель прогнозу, у якій показник точності прогнозу буде ближче до 100%, з більшою ймовірністю зробить більш точний прогноз. Таку модель можна назвати оптимальною для обраного часового ряду. Говорячи про високу точність, говориться про низьку помилку прогнозу і в цій області непорозуміння немає. Не має значення, що саме будете відслідковуватися, але важливо, щоб порівнювались моделі прогнозування або

цільові показники по одному показнику - помилка прогнозу або точність прогнозування.

Коефіцієнт детермінації

Коефіцієнт детермінації (R^2) - це частка дисперсії залежної змінної, що пояснюється розглянутою моделлю залежності, тобто пояснюючими змінними. Більш точно - це одиниця мінус частка непоясненої дисперсії (дисперсії випадкової помилки моделі, або умовної за факторами дисперсії залежної змінної) в дисперсії залежної змінної. Його розглядають як універсальну міру залежності однієї випадкової величини від безлічі інших. В окремому випадку лінійної залежності R^2 є квадратом так званого множинного коефіцієнта кореляції між залежною змінною і пояснюючими змінними. Зокрема, для моделі парної лінійної регресії коефіцієнт детермінації дорівнює квадрату звичайного коефіцієнта кореляції між y і x. Коефіцієнт детермінації R^2 — статистичний показник, що використовується в статистичних моделях як міра залежності варіації залежної змінної від варіації незалежних змінних[16]. Коефіцієнт детермінації R^2 розраховується за наступною формулою[17]:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}},$$

де SS_{res} - сума квадратів лишків регресії, SS_{tot} - загальна сума квадратів.

Суми квадратів розраховуються за наступними формулами:

$$SS_{res} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$

$$SS_{tot} = \sum_{i=1}^{n} (y_i - \bar{y})^2,$$

де y_i , $\hat{y}_i$ – фактичне та розрахункове значення досліджуваної змінної, $\bar{y}$ - середнє арифметичне значення досліджуваної функції.

Коефіцієнт детермінації для моделі з константою приймає значення від 0 до 1. Чим ближче значення коефіцієнта до 1, тим сильніше залежність. При оцінці регресійних моделей це інтерпретується як відповідність моделі даних. Для прийнятних моделей передбачається, що коефіцієнт детермінації повинен бути хоча б не менше 50% (в цьому випадку коефіцієнт множинної кореляції перевищує по модулю 70%). Моделі з коефіцієнтом детермінації вище 80% можна визнати досить хорошими (коефіцієнт кореляції перевищує 90%). Значення коефіцієнта детермінації 1 означає функціональну залежність між змінними.

При відсутності статистичного зв'язку між пояснюючою змінною і факторами, статистика nR^2 для лінійної регресії має асимптотичний розподіл $\chi^2(k-1)$, де $k-1$- кількість факторів моделі. У разі лінійної регресії з нормально розподіленими випадковими помилками статистика

$$F = \frac{R^2 / (k-1)}{(1-R^2) / (n-k)}$$

має точний (для вибірок будь-якого обсягу) розподіл Фішера $F(k-1, n-k)$. Інформація про розподіл цих величин дозволяє перевірити статистичну значущість регресійної моделі виходячи із значення коефіцієнта детермінації. Фактично в цих тестах перевіряється гіпотеза про рівність істинного коефіцієнта детермінації нулю.

У загальному випадку коефіцієнт детермінації може бути і негативним, це говорить про крайню неадекватності моделі: просте середнє наближає краще.

Основна проблема застосування (вибіркового) R^2 полягає в тому, що його значення збільшується (не зменшується) від додавання в модель нових змінних, навіть якщо ці змінні ніякого відношення до прогнозованої змінної не мають! Тому порівняння моделей з різною кількістю чинників за допомогою коефіцієнта детермінації - некоректно. Для цих цілей можна використовувати альтернативні показники.

Середня абсолютна помилка

Середня абсолютна помилка (mean absolute error - MAE) - це міра різниці між двома безперервними змінними. Припустимо, *X* і *Y* - змінні парних спостережень, які виражають одне і те ж явище. Приклади *Y* порівняно з *X* включають порівняння прогнозованого з спостережуваним, подальший час проти початкового часу та одну техніку вимірювання проти альтернативної методики вимірювання. Розглянемо графік розсіяння n точок, де точка i має координати (x_i, y_i). Середня абсолютна помилка (MAE) - середня вертикальна відстань між кожною точкою та ідентичною лінією. MAE - це також середня горизонтальна відстань між кожною точкою та ідентичною лінією.

Середня абсолютна похибка визначається за формулою:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}.$$

Середня абсолютна помилка - це середнє значення абсолютних помилок $e_i = |y_i - x_i|$, де y_i - передбачення, а x_i - справжнє значення. Зауважте, що альтернативні склади можуть включати відносні частоти як вагові фактори. Середня абсолютна помилка використовує ту саму шкалу, що і вимірювані дані. Це відомо як міра точності, що залежить від масштабу, і тому його не можна використовувати для порівняння між серіями, використовуючи різні шкали. Середня абсолютна похибка - це звичайний показник помилки прогнозу в аналізі часових рядів [18], який іноді використовується в змішуванні з більш стандартним визначенням середнього абсолютного відхилення. Та ж плутанина існує і загальніше.

Середня абсолютна процентна похибка

Середня абсолютна процентна похибка (mean absolute percentage error - MAPE), також відома як середнє абсолютне відхилення у відсотках, є показником точності прогнозування методу прогнозування в статистиці, наприклад, в оцінці тренду, також використовується як функція втрат для

проблем регресії в машинному навчанні. Зазвичай це виражає точність у відсотках і визначається формулою:

$$MAPE = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|,$$

де A_t - це фактичне значення, а F_t - значення прогнозу. Різниця між A_t і F_t знову ділиться на фактичне значення A_t. Абсолютне значення в цьому розрахунку підсумовується для кожного прогнозованого моменту часу і ділиться на кількість встановлених точок n. Множення на 100% робить відсоткову помилку.

Проблеми можуть виникнути при обчисленні значення MAPE за допомогою серії невеликих знаменників. Може виникнути проблема сингулярності форми "один поділити на нуль" або створення дуже великих змін абсолютної відсоткової похибки, спричиненої невеликим відхиленням помилки.

Як альтернатива, кожне фактичне значення (A_t) рядів у вихідній формулі може бути замінено середнім значенням усіх фактичних значень ($\bar{A}_t$) цього ряду. Ця альтернатива досі використовується для вимірювання продуктивності моделей, які прогнозують наявні ціни на електроенергію. [19]

Це те саме, що ділити суму абсолютних різниць на суму фактичних значень, і іноді її називають WAPE (зважена абсолютна процентна похибка).

Хоча концепція MAPE звучить дуже просто та переконливо, вона має основні недоліки в практичному застосуванні [20], і існує багато досліджень щодо недоліків та оманливих результатів MAPE. [21, 22]

- MAPE не можна використовувати, якщо є нульові значення (що іноді трапляється, наприклад, у даних про попит), оскільки було б ділення на нуль.
- Для занадто низьких прогнозів процентна помилка не може перевищувати 100%, але для занадто точних прогнозів немає верхньої межі процентної помилки.
- MAPE накладає більш серйозний штраф за негативні помилки, $A_t < F_t$, ніж за позитивні помилки. Як наслідок, коли MAPE використовується для

порівняння точності методів прогнозування, це необ'єктивна оцінка, оскільки MAPE буде систематично вибирати метод, прогнози якого занадто низькі. Цю маловідому, але серйозну проблему можна подолати за допомогою вимірювання точності на основі логарифму відношення точності (відношення передбачуваного до фактичного значення), заданого $\log\left(\frac{predicted}{actual}\right)$. Такий підхід призводить до вищих статистичних властивостей і призводить до прогнозів, які можна інтерпретувати через геометричне середнє значення.

Середньоквадратична помилка

У статистиці середньоквадратична помилка (mean square error - MSE) або середньоквадратичне відхилення (MSD) оцінювача вимірює середнє значення квадратів помилок - тобто середньоквадратична різниця між оціненими значеннями та фактичними значеннями. MSE - це функція ризику, що відповідає очікуваному значенню втрат у квадраті. Те, що MSE майже завжди є суто позитивним (а не нульовим), відбувається через випадковість або через те, що оцінювач не враховує інформацію, яка могла б дати більш точну оцінку. [24]

MSE - це показник якості оцінювача - він завжди негативний, а чим більше значення наближені до нуля, тим кращі.

MSE - це другий момент помилки, і, таким чином, включає як дисперсію оцінювача (наскільки широко розповсюджені оцінки від однієї вибірки даних до іншої), так і його зміщення. Для неупередженого оцінювача, MSE є дисперсією оцінки. Як і дисперсія, MSE має ті самі одиниці виміру, що і квадрат оцінюваної кількості. Аналогічно стандартному відхиленню, приймаючи квадратний корінь MSE, виходить помилка середньоквадратичного кореня або середньоквадратичне відхилення (RMSE або RMSD), яка має ті самі одиниці, що і оцінювана кількість; для неупередженого оцінювача RMSE - квадратний корінь дисперсії, відомий як стандартна помилка.

MSE оцінює якість предиктора (тобто функція, яка відображає довільні входи до вибірки значень якоїсь випадкової величини), або оцінювача (тобто, математична функція, яка відображає вибірку даних для оцінки параметру сукупності з яких відбираються дані). Визначення MSE відрізняється залежно від того, описується предиктор або оцінювач.

Якщо вектор n прогнозів, сформований із зразка n точок даних по всіх змінних, а Y - вектор спостережуваних значень змінної, що прогнозується, з $\hat{Y}_i$ передбачуваними значеннями (наприклад, з розміру найменших квадратів), тоді внутрішній зразок MSE предиктора обчислюється як:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 .$$

Тобто MSE - середнє значення $\frac{1}{n}\sum_{i=1}^{n}$ квадратів помилок $\left(Y_i - \hat{Y}_i\right)^2$. Це легко обчислювана кількість для певного зразка (отже, залежить від вибірки).

MSE також можна обчислити на q даних, які не використовувались при оцінці моделі, або тому, що вони не були використані для цієї мети, або тому, що ці дані були нещодавно отримані. У цьому процесі, який відомий як перехресне підтвердження, MSE часто називають середньою помилкою прогнозування в квадраті, і обчислюють як:

$$MSPE = \frac{1}{n}\sum_{i=n+1}^{n+q}\left(Y_i - \hat{Y}_i\right)^2 .$$

MSE оцінювача $\hat{\theta}$ щодо невідомого параметра θ визначається як

$$MSE\left(\hat{\theta}\right) = E_{\hat{\theta}}\left[\left(\hat{\theta} - \theta\right)^2\right].$$

Це визначення залежить від невідомого параметра, але MSE є апріорною властивістю оцінювача. MSE може бути функцією невідомих параметрів; в цьому випадку будь-який оцінювач MSE, заснований на оцінках цих параметрів, буде функцією даних і, таким чином, випадковою змінною. Якщо оцінювач $\hat{\theta}$ отриманий як вибіркова статистика і використовується для оцінки

деякого параметра сукупності, то очікування щодо розподілу вибірки вибіркової статистики.

MSE можна записати як суму дисперсії оцінювача та квадратичне зміщення оцінювача, забезпечуючи ефективний спосіб обчислення MSE і маючи на увазі, що у випадку неупереджених оцінок MSE та дисперсія є рівнозначними.

$$MSE\left(\hat{\theta}\right) = Var_{\hat{\theta}}\left(\hat{\theta}\right) + Bias\left(\hat{\theta},\theta\right)^2.$$

1.4. Перенавчання та недонавчання в методах машинного навчання

Будь-яка модель створюється для того, щоб вирішити деяку задачу. Передбачити значення, класифікувати об'єкт і т.п. Для навчання є набір даних, в якому наведено точні та розмічені дані. В результаті роботи навчання ціль отримати модель, яка буде правильно працювати на будь-яких даних, з певною точністю. Але буває, що модель прекрасно працює на навчальній вибірці, а на тестовій жахливо. Тут з'являються проблеми перенавчання або недонавчання.

Недонавчання - небажане явище, яке виникає при вирішенні завдань навчання по прецедентах, коли алгоритм навчання не забезпечує досить малої величини середньої помилки на навчальній вибірці. Недонавчання виникає при використанні недостатньо складних моделей, коли навіть на навчальних даних ми не можемо досягти малої помилки моделі. Причин тому може бути багато:

- занадто рання зупинка;
- неправильно підібраний алгоритм навчання;
- неадекватна функція помилки;
- неправильно налаштовані параметри.

Перенавчання (overtraining, overfitting) - небажане явище, яке виникає при вирішенні завдань навчання по прецедентах, коли ймовірність помилки

навченого алгоритму на об'єктах тестової вибірки виявляється істотно вище, ніж середня помилка на навчальній вибірці. Перенавчання виникає при використанні надмірно складних моделей.

На рис. 1.6 наведено приклад сигналу з шумом, який близький до лінійного, що апроксимується лінійною функцією і поліномом. Хоча поліном гарантує ідеальний збіг, лінійна апроксимація краще узагальнює закономірність і буде давати кращі передбачення.

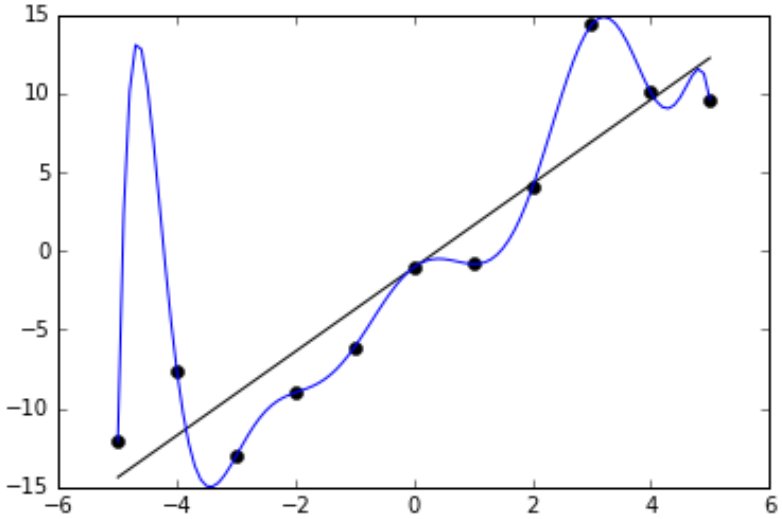

Рис. 1.6. Сигнал з шумом, апроксимований лінійною функцією(чорна лінія) і поліномом (синя лінія)

Причини для перенавчання приблизно такі ж, за винятком зупинки, тут вона навпаки дуже пізня.

Потенціал перенавчання залежить не лише від кількостей параметрів та даних, але й від відповідності структури моделі формі даних, та величини похибки моделі в порівнянні з очікуваним рівнем шуму або похибки в даних.

Для того, щоб модель була ефективною, потрібно знайти той момент, коли модель вже навчена, але ще не перенавчилась (рис. 1.7). В основному використовується одночасна перевірка помилки на навчальній і тестовій вибірці, обидві вони з часом повинні зменшуватися, але якщо тестова раптово

починає збільшуватися, то треба звернути на це увагу і зупинити процес навчання. Тому що, в більшості випадків це означає початок перенавчання.

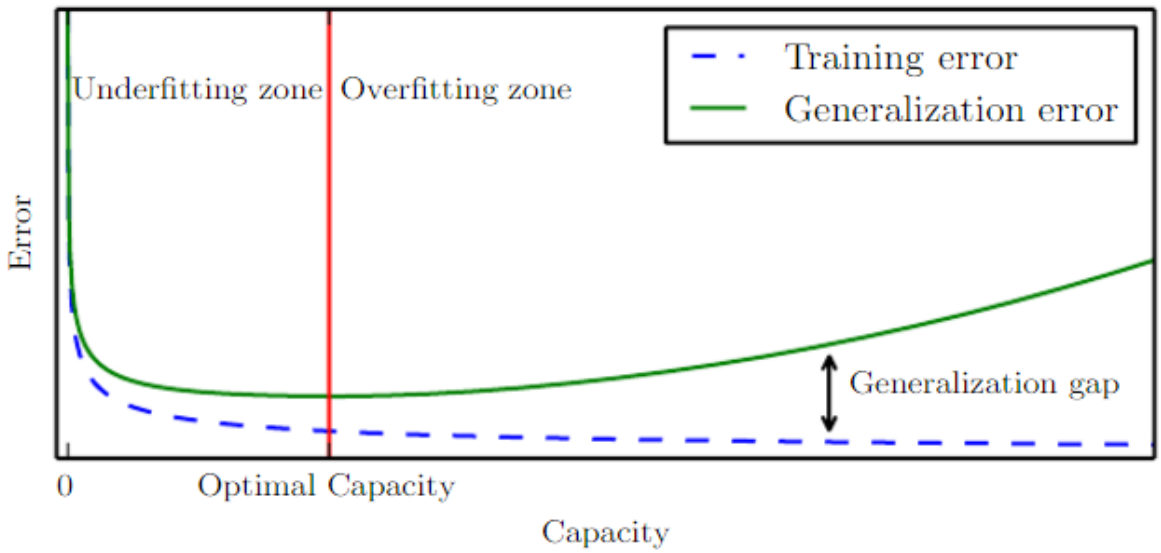

Рис. 1.7. Графік залежності помилки від точності для моделей машинного навчання

Емпіричним ризиком називається середня помилка алгоритму на навчальній вибірці. Метод мінімізації емпіричного ризику (empirical risk minimization, ERM) найбільш часто застосовується для побудови алгоритмів навчання. Він полягає в тому, щоб в рамках заданої моделі вибрати алгоритм, який має мінімальне значення середньої помилки на заданій навчальній вибірці.

З перенавчанням методу ERM пов'язано два твердження, які на перший погляд можуть здатися парадоксальними.

1. Мінімізація емпіричного ризику не гарантує, що ймовірність помилки на тестових даних буде мала [26]. Прикладом може слугувати абсурдний алгоритм навчання, який мінімізує емпіричний ризик до нуля, але при цьому абсолютно не здатний навчатися. Алгоритм полягає в наступному. Отримавши навчальну вибірку, він запам'ятовує її і будує функцію, яка порівнює пропонований об'єкт з навчальними об'єктами. Якщо пропонований об'єкт в точності збігається з одним з навчальних, то ця функція видає для нього вивчену правильну відповідь. Інакше видається довільна відповідь

(наприклад, випадковий або завжди одна і та ж). Емпіричний ризик алгоритму дорівнює нулю, однак він не відновлює залежність і не володіє ніякою здатністю до узагальнення. Отже, для успішного навчання необхідно не тільки запам'ятовувати, але й узагальнювати.

2. Перенавчання з'являється саме внаслідок мінімізації емпіричного ризику. [26] Нехай задано кінцева множина з D алгоритмів, які допускають помилки незалежно і з однаковою ймовірністю. Число помилок будь-якого з цих алгоритмів на заданій навчальній вибірці підкоряється тому ж біноміальному розподілу. Мінімум емпіричного ризику - це випадкова величина, що дорівнює мінімуму з D незалежних однаково розподілених біноміальних випадкових величин. Її очікуване значення зменшується з ростом D. Відповідно, з ростом D збільшується перенавчання - різниця ймовірності помилки і частоти помилок на навчанні.

В даному модельному прикладі легко побудувати довірчий інтервал перенавчання, так як функція розподілу мінімуму відома. Однак в реальній ситуації алгоритми мають різні ймовірності помилок, не є незалежними, а множина алгоритмів, з якої вибирається найкращий, може бути нескінченною. З цих причин висновок кількісних оцінок перенавчання є складним завданням, яким займається теорія обчислювального навчання. До сих пір залишається відкритою проблема сильно завищених верхніх оцінок імовірності перенавчання.

3. Перенавчання пов'язано з надлишковою складністю використовуваної моделі. Завжди існує оптимальне значення складності моделі, при якому перенавчання мінімально[26].

На рис. 1.8 показано приклад перенавчання. Зелена розділова лінія показує перенавчену модель, а чорна лінія - врегульовану модель. Хоча зелена лінія краще відповідає зразкам, за якими проходило навчання, класифікація по зеленій лінії дуже залежить від конкретних даних, і швидше за все нові дані будуть погано відповідати класифікації по зеленій лінії і краще - класифікації по чорній лінії.

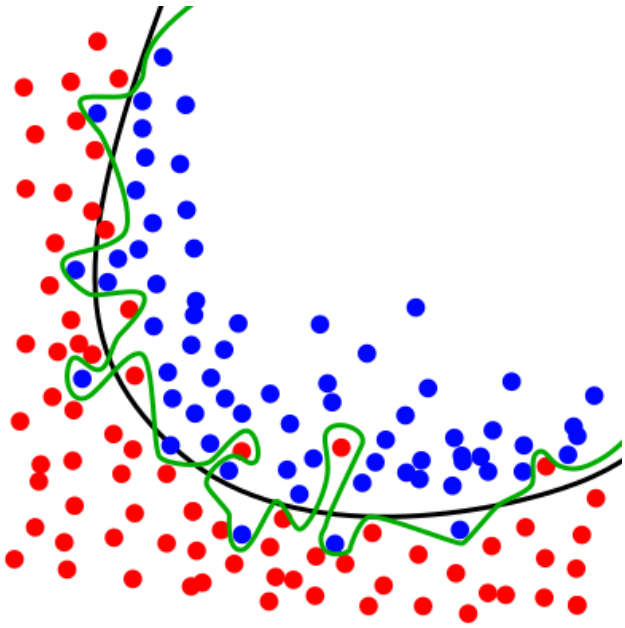

Рис. 1.8.Приклад перенавчання моделі (зелена лінія) та врегульованої моделі (чорна лінія)

Способи боротьби з перенавчанням залежать від методу моделювання і способу побудови моделі. Наприклад, якщо будується дерево прийняття рішень, то можна обрізати деякі його гілки в процесі побудови. Методи запобігання перенавчання:

- Перехресна перевірка - метод оцінки аналітичної моделі і її поведінки на незалежних даних. При оцінці моделі наявні дані розбиваються на k частин. Потім на k-1 частинах даних проводиться навчання моделі, а частина даних, що залишилася використовується для тестування. Процедура повторюється k разів; в результаті кожен з k елементів даних використовується для тестування. В результаті виходить оцінка ефективності обраної моделі з найбільш рівномірним використанням наявних даних;
- Регуляризація. В статистиці, машинному навчанні, теорії обернених задач - метод додавання деяких додаткових обмежень до умови з метою вирішити некоректно поставленого завдання або запобігти перенавчанню. Ця інформація часто має вигляд штрафу за складність моделі. Наприклад, це можуть бути обмеження гладкості результуючої функції або обмеження по нормі векторного простору;
- Рання зупинка;

- Вербалізація нейронних мереж - мінімізований опис роботи синтезованої і вже навченої нейронної мережі у вигляді декількох взаємозалежних алгебраїчних або логічних функцій;
- Апріорна ймовірність - розподіл ймовірностей, яке виражає припущення про p до обліку експериментальних даних. Наприклад, якщо p - частка виборців, готових голосувати за певного кандидата, то апріорним розподілом буде припущення про p до врахування результатів опитувань або виборів;
- Байєсова порівняння моделей - це метод вибору моделі на основі факторів Байєса. Розглянуті моделі є статистичними моделями. Метою фактора Байєса є кількісна оцінка підтримки моделі над іншою, незалежно від того, чи правильні ці моделі.

Такі методи можуть вказати, коли подальше навчання більше не веде до поліпшення оцінок параметрів. В основі цих методів лежить явне обмеження на складність моделей, або перевірка здатності моделі до узагальнення шляхом оцінки її ефективності на безлічі даних, які не використовувалися для навчання і вважаються наближенням до реальних даних, до яких модель буде застосовуватися.

Висновки до першого розділу

При порівнянні типів баз даних було вирішено використовувати нереляційні бази даних через необхідність зберігати велику кількість інформації з датчиків. Також NoSQL корисні коли навантаження на систему може зростати з часом, і система потребує масштабування на декілька машин, а також через їх швидкодію, що є позитивними характеристиками для машинного навчання. Було наведено класифікацію завдань машинного навчання за типом навчання: з вчителем, без вчителя, з частковим залученням вчителя та з підкріпленням. В подальшій роботі було використанно навчання з вчителем, так як цей тип навчання накраще відповідає умовам поставленного завдання.

2. МОДЕЛІ МАШИННОГО НАВЧАННЯ

Існує таке поняття, як «No Free Lunch» теорема. Її суть полягає в тому, що немає такого алгоритму, який був би кращим вибором для кожного завдання, що особливо стосується навчання з учителем.

Наприклад, не можна сказати, що нейронні мережі завжди працюють краще, ніж дерева рішень, і навпаки. На ефективність алгоритмів впливає безліч факторів на кшталт розміру і структури набору даних.

З цієї причини доводиться пробувати багато різних алгоритмів, перевіряючи ефективність кожного на тестовому наборі даних, і потім вибирати кращий варіант. Потрібно вибирати серед алгоритмів, відповідних поставленій задачі.

Алгоритми машинного навчання можна описати як навчання цільової функції f, яка найкращим чином співвідносить вхідні змінні X і вихідну змінну Y: $Y = f(X)$. Точно не відомо, що з себе представляє функція f. Якби це було відомо, то використовували б її безпосередньо, а не намагалися навчити за допомогою різних алгоритмів.

Найбільш поширеним завданням в машинному навчанні є передбачення значень Y для нових значень X. Це називається прогностичним моделюванням, і мета - зробити якомога більше точне передбачення.

2.1. Лінійна регресія

Лінійна регресія - один з найбільш відомих і зрозумілих алгоритмів в статистиці і машинному навчанні. Прогностичне моделювання в першу чергу стосується мінімізації помилки моделі, іншими словами, як можна більш точного прогнозування.

Лінійну регресію можна представити у вигляді рівняння, яке описує пряму (рис. 2.1), найбільш точно ніколи взаємозв'язок між вхідними змінними

X і вихідними змінними Y. Для складання цього рівняння потрібно знайти певні коефіцієнти B для вхідних змінних.

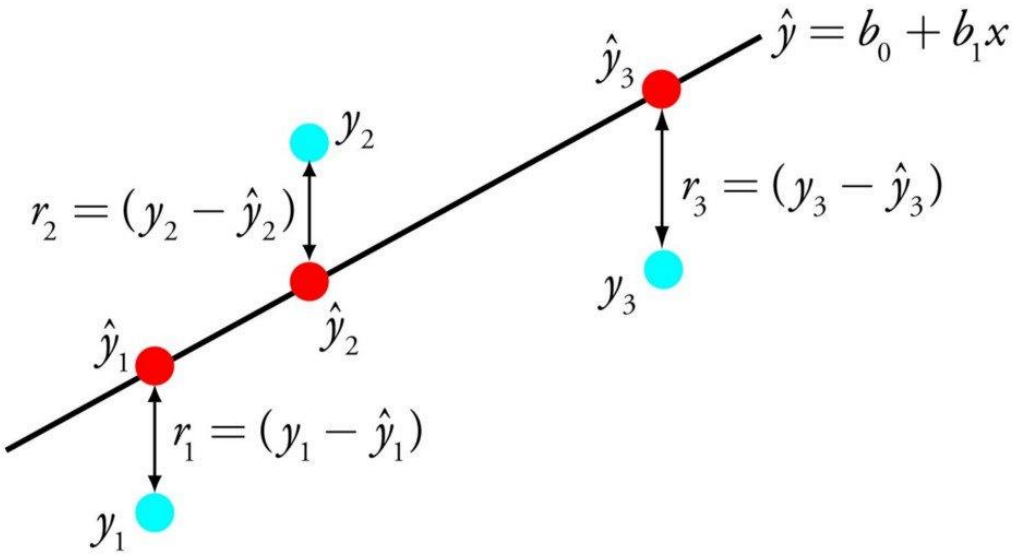

Рис. 2.1. Приклад опису даних за допомогою лінійної регресії

Наприклад: $Y = B_0 + B_1 \cdot X$.

Знаючи *X*, необхідно знайти *Y*, і ціль лінійної регресії полягає в пошуку значень коефіцієнтів B_0 і B_1.

Для оцінки регресійній моделі використовуються різні методи на кшталт лінійної алгебри або методу найменших квадратів.

Лінійна регресія існує вже понад 200 років, і за цей час її встигли ретельно вивчити. Пара практичних правил: прибрати схожі (корелюючі) змінні і позбутися від шуму в даних, якщо це можливо. Лінійна регресія - швидкий і простий алгоритм.

2.2. Логістична регресія

Логістична регресія - ще один алгоритм, який прийшов в машинне навчання з статистики. Її добре використовувати для завдань бінарної класифікації (це завдання, в яких на виході ми отримуємо один з двох класів).

Логістична регресія схожа на лінійну тим, що в ній теж потрібно знайти значення коефіцієнтів для вхідних змінних. Різниця полягає в тому, що вихідне значення перетворюється за допомогою нелінійної або логістичної функції.

Логістична функція (рис. 2.2) виглядає як велика буква *S* і перетворює будь-яке значення в число в межах від 0 до 1. Це дуже корисно, тому що ми можемо застосувати правило до виходу логістичної функції для прив'язки до 0 і 1 (наприклад, якщо результат функції менше 0.5, то на виході отримуємо 1) і передбачення класу.

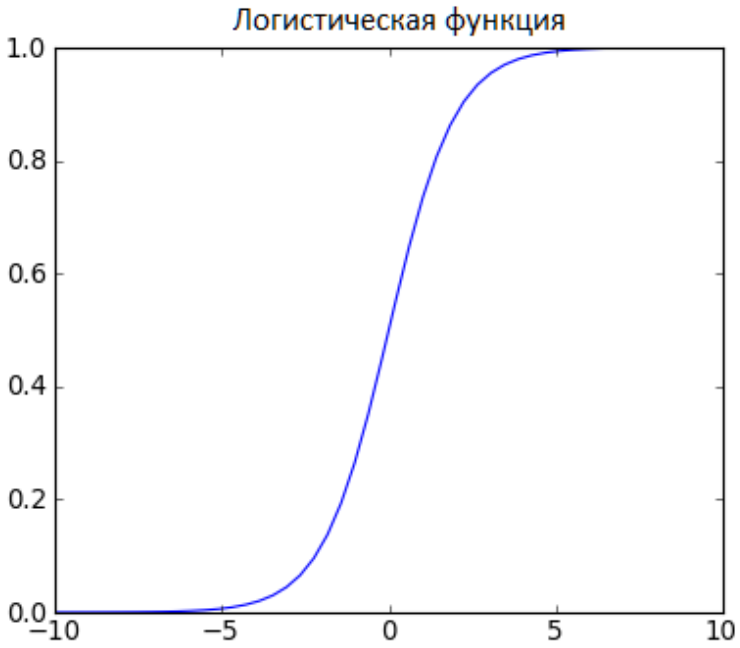

Рис. 2.2. Приклад логічтичної функції

Завдяки тому, як навчається модель, передбачення логістичної регресії можна використовувати для відображення ймовірності приналежності зразка до класу 0 або 1. Це корисно в тих випадках, коли потрібно мати більше обґрунтувань для прогнозування.

Як і у випадку з лінійної регресією, логістична регресія виконує своє завдання краще, якщо прибрати зайві і схожі змінні. Модель логістичної регресії швидко навчається і добре підходить для задач бінарної класифікації.

2.3. Лінійний дискримінантний аналіз (LDA)

Логістична регресія використовується, коли потрібно віднести зразок до одного з двох класів. Якщо класів більше, ніж два, то краще використовувати алгоритм LDA (Linear discriminant analysis).

Представлення LDA досить просте (рис. 2.3). Воно складається зі статистичних властивостей даних, розрахованих для кожного класу. Для кожної вхідної змінної це включає:

- середнє значення для кожного класу;
- дисперсію, розраховану по всіх класах.

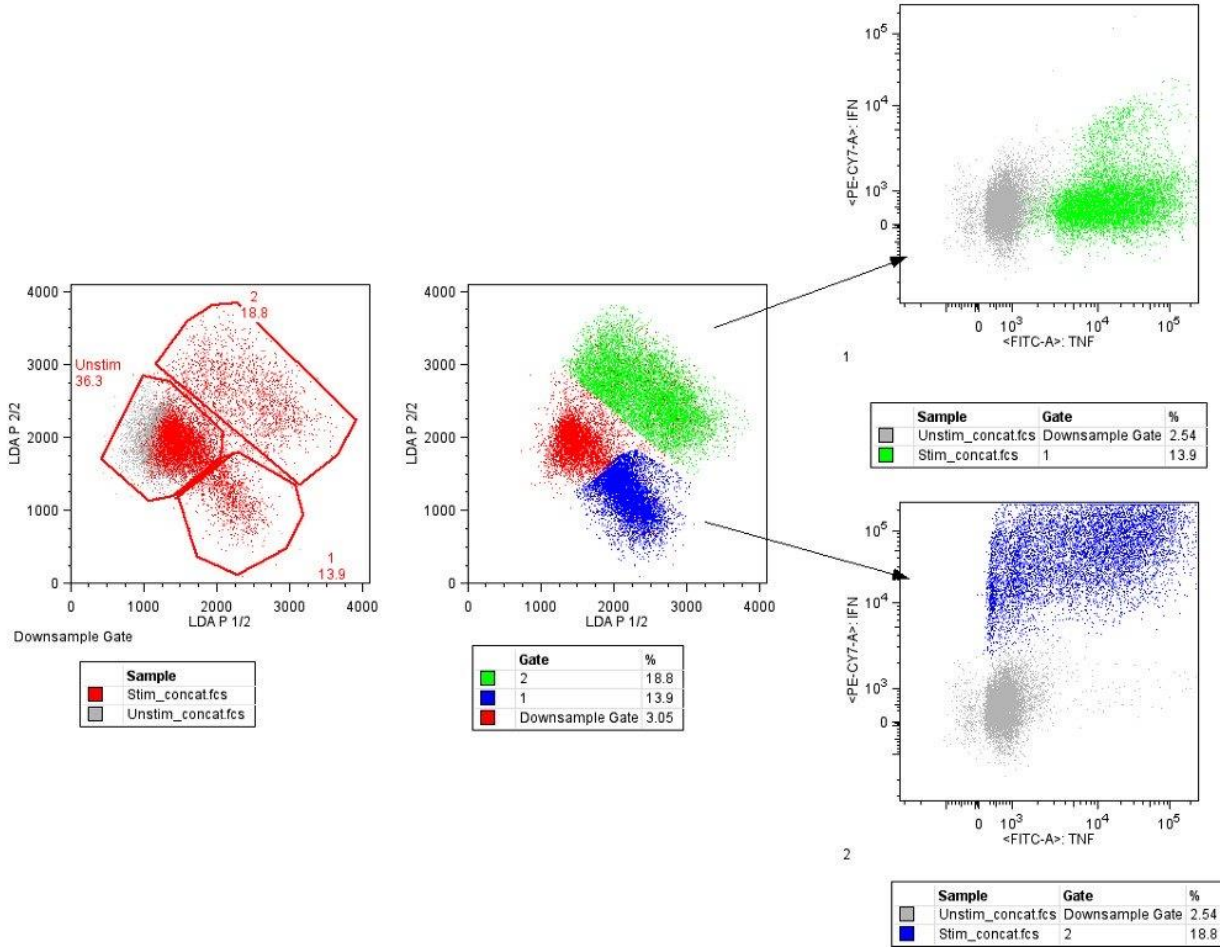

Рис. 2.3. Приклад алгоритму лінійного дискримінантного аналізу

Передбачення виробляються шляхом обчислення дискрімінантного значення для кожного класу і вибору класу з найбільшим значенням.

Передбачається, що дані мають нормальний розподіл, тому перед початком роботи рекомендується видалити з даних аномальні значення. Це простий і ефективний алгоритм для задач класифікації.

2.4. Дерева прийняття рішень

Дерева прийняття рішень (decision trees) зазвичай використовуються для вирішення задач класифікації даних або, інакше кажучи, для завдання апроксимації заданої булевої функції. Ситуація, в якій варто застосовувати дерева прийняття рішень, зазвичай виглядає так: є багато випадків, кожен з яких описується деяким кінцевим набором дискретних атрибутів, і в кожному з випадків дано значення деякої (невідомої) булевої функції, що залежить від цих атрибутів. Завдання полягає в тому, щоб створити досить економічну конструкцію, яка б описувала цю функцію і дозволяла класифікувати нові, що надходять ззовні, дані (рис. 2.4).

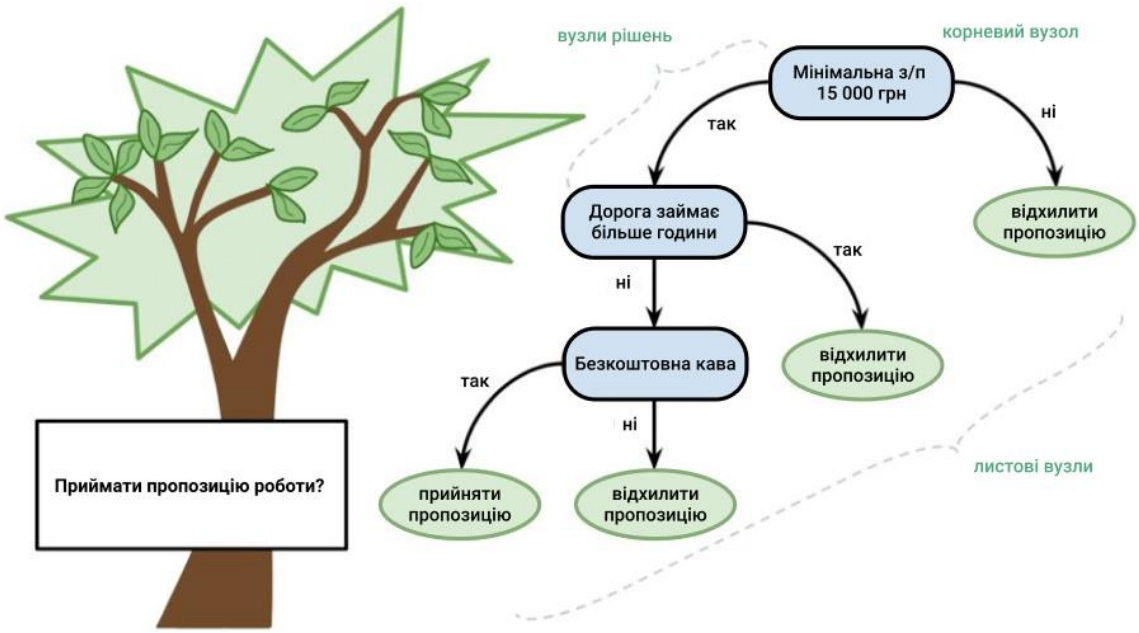

Рис. 2.4. Алгоритм дерева прийняття рішень

Листові вузли - це вихідна змінна, яка використовується для передбачення. Передбачення виробляються шляхом проходу по дереву до листового вузла і виведення значення класу на цьому вузлі.

Дерева швидко навчаються і роблять прогнози. Крім того, вони точні для широкого кола завдань і не вимагають особливої підготовки даних.

Дерева рішень бувають двох основних типів[29]:

- Дерево для класифікації, коли передбачається клас, до якого належать дані;
- Дерево для регресії, коли результат передбачень можна розглядати як дійсне число (наприклад, ціна на будинок, або тривалість перебування пацієнта в лікарні).

Серед інших методів, метод дерева прийняття рішень має кілька переваг:

- Простий в розумінні та інтерпретації. Люди здатні інтерпретувати результати моделі дерева прийняття рішень після короткого пояснення
- Не потребує підготовки даних. Інші техніки вимагають нормалізації даних, додавання фіктивних змінних, а також видалення пропущених даних.
- Здатний працювати як з категоріальним, так і з інтервальними змінними. Інші методи працюють лише з тими даними, де присутній лише один тип змінних. Наприклад, метод відносин може бути застосований тільки на номінальних змінних, а метод нейронних мереж тільки на змінних, виміряних за інтервального шкалою.
- Використовує модель «білого ящика». Якщо певна ситуація спостерігається в моделі, то її можна пояснити за допомогою булевої логіки. Прикладом «чорного ящика» може бути штучна нейронна мережа, так як результати даної моделі важко піддаються поясненню.
- Дозволяє оцінити модель за допомогою статистичних тестів. Це дає можливість оцінити надійність моделі.
- Є надійним методом. Метод добре працює навіть в тому випадку, якщо були порушені початкові припущення, включені в модель.
- Дозволяє працювати з великим об'ємом інформації без спеціальних підготовчих процедур. Даний метод не вимагає спеціального обладнання для роботи з великими базами даних.

Недоліки методу:

- Проблема отримання оптимального дерева рішень є NP-повною з точки зору деяких аспектів оптимальності навіть для простих завдань [30] [31]. Таким чином, практичне застосування алгоритму дерев рішень засноване на евристичних алгоритмах, таких як алгоритм «жадібності», де єдино оптимальне рішення вибирається локально в кожному вузлі. Такі алгоритми не можуть забезпечити оптимальність всього дерева в цілому.
- В процесі побудови дерева рішень можуть створюватися занадто складні конструкції, які недостатньо повно представляють дані. Дана проблема називається перенавчанням [32]. Для того, щоб її уникнути, необхідно використовувати метод «регулювання глибини дерева».
- Існують концепти, які складно зрозуміти з моделі, так як модель описує їх складним шляхом. Дане явище може бути викликано проблемами XOR, парності або мультиплексарності. У цьому випадку ми маємо справу з непомірно великими деревами. Існує кілька підходів вирішення даної проблеми, наприклад, спроба змінити репрезентацію концепту в моделі (складання нових суджень) [33], або використання алгоритмів, які більш повно описують і репрезентують концепт (наприклад, метод статистичних відносин, індуктивна логіка програмування).
- Для даних, які включають категоріальні змінні з великим набором рівнів, більша інформаційна вага присвоюється тим атрибутам, які мають більшу кількість рівнів. [34]

Деякі методи дозволяють побудувати більш одного дерева рішень (ансамблі дерев рішень):

- початкове завантаження, або бегінг (англ. bagging - це мета-алгоритм композиційного навчання машин, призначений для поліпшення стабільності і точності алгоритмів машинного навчання, які використовуються в статистичній класифікації і регресії) над деревами рішень, найбільш ранній підхід. Будує кілька дерев рішень, неодноразово інтерполюючи дані з заміною,

і в якості консенсусного рішення видає результат голосування дерев (їх середній прогноз); [35]

- класифікатор «Випадковий ліс» заснований на бегінгу, проте на додаток до нього випадковим чином вибирається підмножина ознак в кожному з вузлів, з метою зробити дерева більш незалежними;
- підсилювання (англ. boosting) дерев може бути використане для таких задач як регресія, так і класифікація. [36]
- «Обертання лісу» - дерева, в яких кожне дерево рішень аналізується першим застосуванням методу головних компонентів на випадкові підмножини вхідних функцій. [37]

2.5. Наївний Байєсівський класифікатор

Наївний Байес - простий, але ефективний алгоритм.

Модель складається з двох типів ймовірностей, які розраховуються за допомогою тренувальних даних:

- Імовірність кожного класу.
- Умовна ймовірність для кожного класу при кожному значенні *x*.

Після розрахунку імовірнісної моделі її можна використовувати для передбачення з новими даними за допомогою теореми Байеса. Якщо суттєві дані, то, припускаючи нормальний розподіл, розрахувати ці ймовірності не складає особливої складності.

Наївний Байес називається наївним, тому що алгоритм передбачає, що кожна вхідна змінна незалежна. Це сильне припущення, яке не відповідає реальним даним. Проте даний алгоритм вельми ефективний для цілого ряду складних завдань на зразок класифікації спаму або розпізнавання рукописних цифр.

2.6. K-найближчих сусідів (KNN)

K-найближчих сусідів - дуже простий і дуже ефективний алгоритм. Модель KNN (K-nearest neighbors) представлена всім набором тренувальних даних.

Передбачення для нової точки робиться шляхом пошуку K найближчих сусідів в наборі даних і підсумовування вихідної змінної для цих K екземплярів (рис. 2.5).

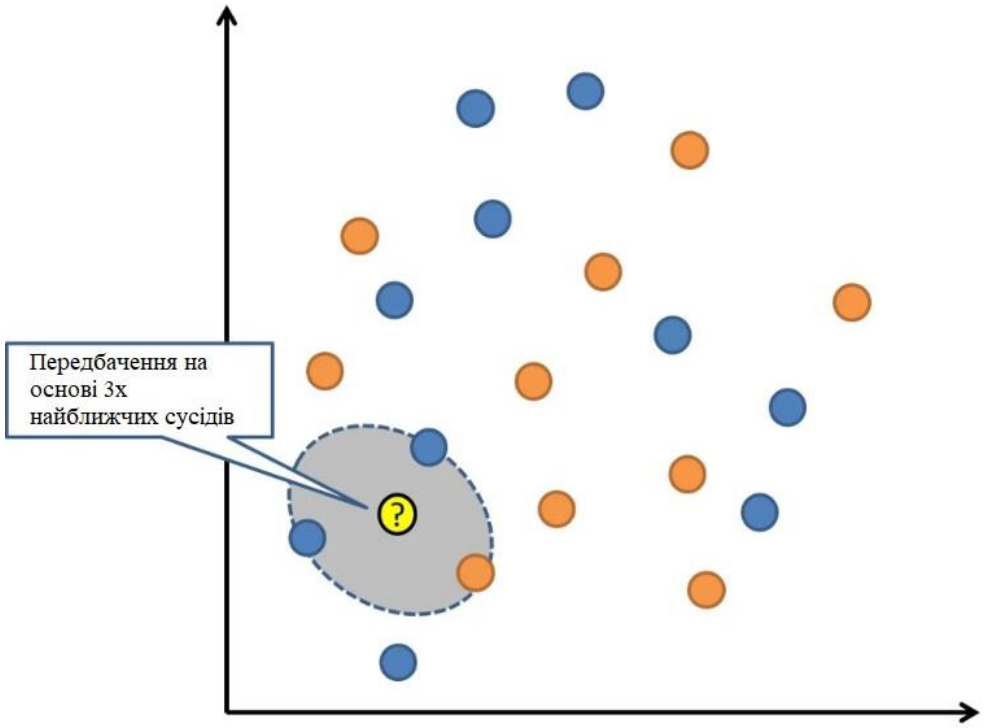

Рис. 2.5. Алгоритм K-найближчих сусідів

Питання лише в тому, як визначити схожість між екземплярами даних. Якщо всі ознаки мають один і той же масштаб (наприклад, сантиметри), то найпростіший спосіб полягає у використанні евклідової відстані - числа, яке можна розрахувати на основі відмінностей з кожної вхідної змінної.

KNN може вимагати багато пам'яті для зберігання всіх даних, але зате швидко зробить прогноз. Також навчальні дані можна оновлювати, щоб передбачення залишалися точними з плином часу.

Ідея найближчих сусідів може погано працювати з багатовимірними даними (безліч вхідних змінних), що негативно позначиться на ефективності алгоритму при вирішенні задачі. Це називається прокляттям розмірності. Іншими словами, варто використовувати лише найбільш важливі для передбачення змінні.

2.7. Мережі векторного квантування (LVQ)

Недолік KNN полягає в тому, що потрібно зберігати весь тренувальний набір даних. Якщо KNN добре себе показав, то є сенс спробувати алгоритм LVQ (Learning vector quantization), який позбавлений цього недоліку.

LVQ являє собою набір кодових векторів. Вони вибираються на початку випадковим чином і протягом певної кількості ітерацій адаптуються так, щоб найкращим чином узагальнити весь набір даних. Після навчання ці вектори можуть використовуватися для передбачення так само, як це робиться в KNN. Алгоритм шукає найближчого сусіда (найбільш підходящий кодовий вектор) шляхом обчислення відстані між кожним кодовим вектором і новим екземпляром даних. Потім для найбільш підходящого вектора в якості передбачення повертається клас або число в разі регресії (рис. 2.6).

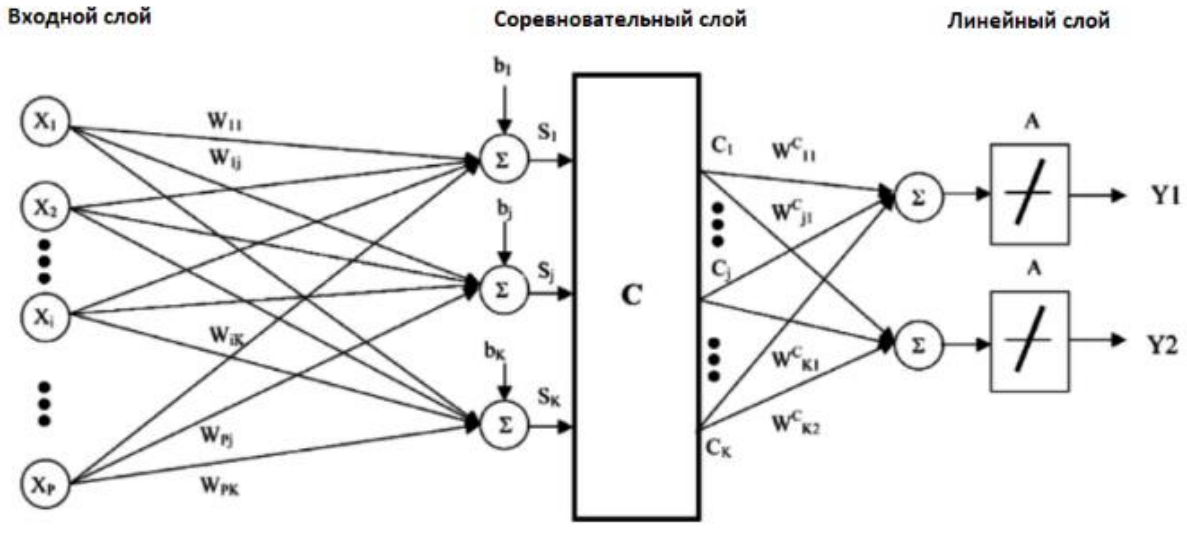

Рис. 2.6. Алгоритм мережі векторного квантування

Кращого результату можна досягти, якщо всі дані будуть знаходитися в одному діапазоні, наприклад від 0 до 1.

2.8. Метод опорних векторів

Гіперплощина - це лінія, що розділяє простір вхідних змінних. У методі опорних векторів гіперплощина вибирається так, щоб найкращим чином розділяти точки в площині вхідних змінних по їх класу: 0 або 1. В двовимірній площині це можна уявити як лінію, яка повністю поділяє точки всіх класів (рис. 2.7). Під час навчання алгоритм шукає коефіцієнти, які допомагають краще розділяти класи гіперплощиною.

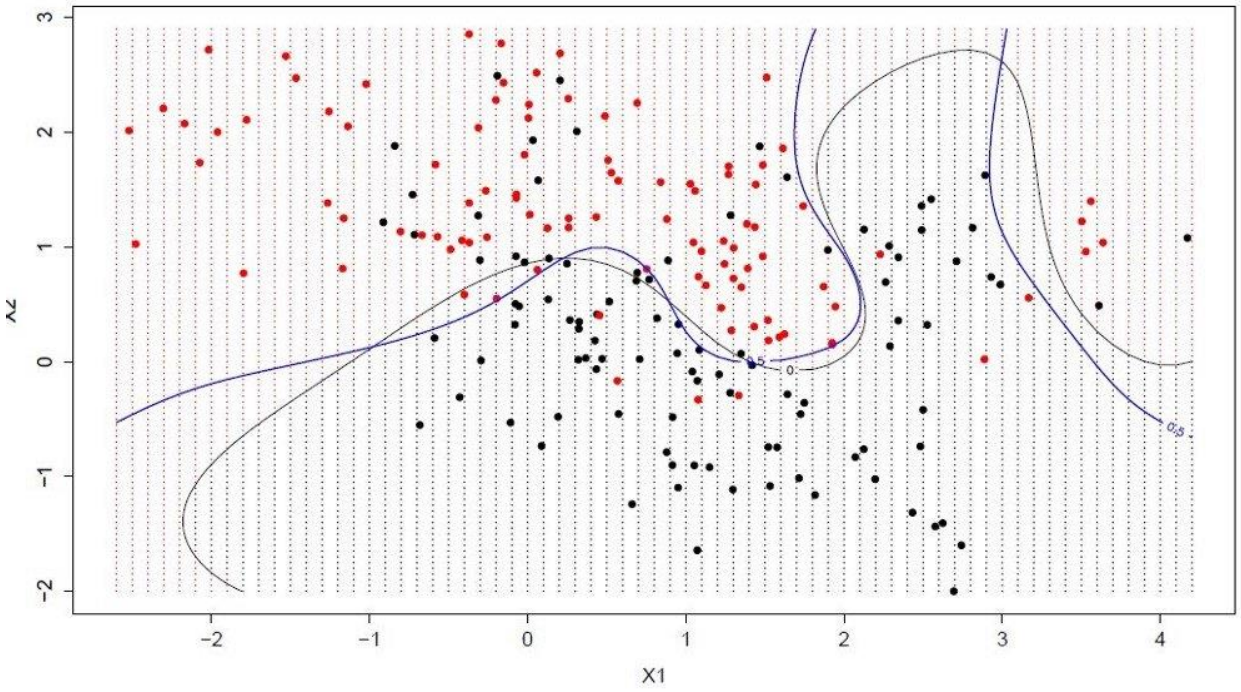

Рис. 2.7. Розділення даних на класи методом опорних векторів

Відстань між гіперплощиною і найближчими точками даних називається різницею. Краща або оптимальна гіперплощина, що розділяє два класи, - це лінія з найбільшою різницею. Тільки ці точки мають значення при визначенні гіперплощини і при побудові класифікатора. Ці точки називаються опорними

векторами. Для визначення значень коефіцієнтів, максимізують різницю, використовуються спеціальні алгоритми оптимізації.

2.9. Випадковий ліс

Випадковий ліс складається з комітету дерев рішень, які також називаються деревами класифікації або регресійними деревами і вирішують однойменні завдання. Вони застосовуються в статистиці, аналізі даних і машинному навчанні. Кожне окреме дерево - досить проста модель, яка має гілки, вузли і листя. У вузлах записані атрибути, від значень яких залежить цільова функція. Далі по гілках в листя потрапляють значення цільової функції. У процесі класифікації нового випадку потрібно спуститися по дереву через гілки до листа, пройшовши через усі значення атрибутів по логічному принципу "ЯКЩО-ТО". Залежно від цих умов, цільової змінної буде присвоєно те чи інше значення або клас (цільова змінна потрапить в конкретний лист). Мета побудови дерева рішень - створення моделі, яка передбачає значення цільової змінної в залежності від декількох змінних на вході.[39]

Випадковий ліс будується шляхом простого голосування дерев рішень за алгоритмом Bagging (Беггінг). Bagging - це штучне слово, утворене від англійського словосполучення bootstrap aggregating - це мета-алгоритм композиційного навчання машин, призначений для поліпшення стабільності і точності алгоритмів машинного навчання, які використовуються в статистичній класифікації і регресії.

Bootstrap в статистиці - це такий спосіб формування вибірки, коли вибирається рівно стільки ж об'єктів, скільки їх початково і було. Але об'єкти ці вибираються з повтореннями. Іншими словами, обраний випадковий об'єкт повертається назад і може бути обраний повторно. При цьому число об'єктів, яке буде вибрано, складе приблизно 63% від вихідної вибірки, а частка об'єктів, що залишилися (приблизно 37%), жодного разу не потраплять в

навчальну вибірку. З цієї сформованої вибірки навчаються базові алгоритми. Це теж відбувається випадковим чином: беруться випадкові підмножини (семпли) заданої довжини і навчаються на обраній випадковій підмножині ознак (атрибутів). Решта 37% вибірки використовуються для перевірки узагальнюючої здатності побудованої моделі.

Потім всі навчені дерева об'єднуються в композицію за допомогою простого голосування, з використанням усередненої помилки для всіх семплів. В результаті застосування bootstrap aggregating зменшується середній квадрат помилки, знижується дисперсія класифікатора. На різних вибірках помилка буде відрізнятися не так сильно. В результаті модель буде менше перенавчатися.[40] Ефективність беггінга полягає в тому, що базові алгоритми (дерева рішень) навчаються за різними випадковими підвибірками і їх результати можуть сильно відрізнятися, але їх помилки взаємно компенсуються при голосуванні.

Можна сказати, що Випадковий ліс - це спеціальний випадок бегінга, коли в якості базового сімейства використовуються дерева рішень. При цьому, на відміну від звичайного способу побудови дерев рішень, не використовується усічення дерева. Метод налаштований на те, щоб можна було побудувати композицію якомога швидше по великим вибірках даних. Кожне дерево будується специфічним чином. Ознака (атрибут) для побудови вузла дерева вибирається не з загального числа ознак, а з їх випадкової підмножини. Якщо ми будуємо регресійну модель, то число ознак n / 3. У разі класифікації це $\sqrt{n}$. Все це є емпіричними рекомендаціями і називається декореляцією (усунення лінійних залежностей): в різні дерева потрапляють різні набори ознак, і дерева навчаються на різних вибірках. Схема функціонування алгоритму наведена на рис. 2.8.

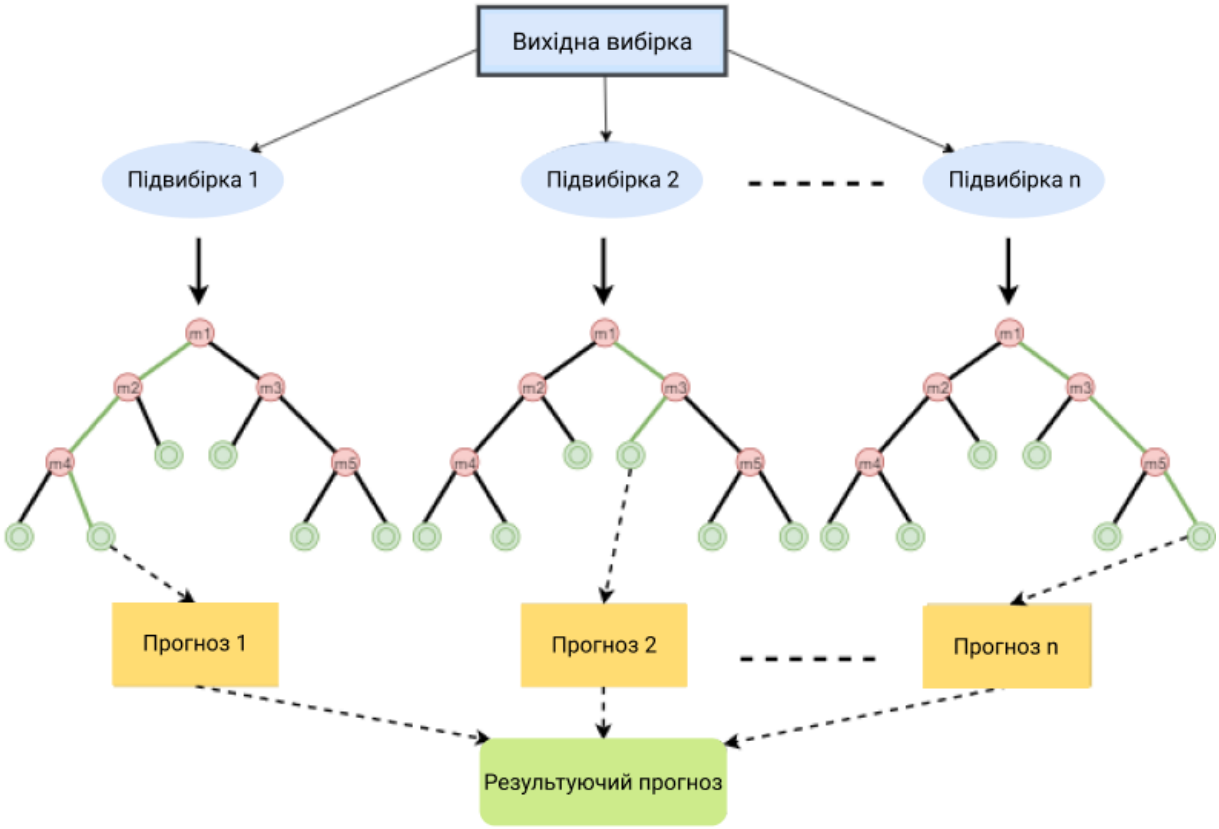

Рис. 2.8. Схема функціонування алгоритму Випадковий ліс

Алгоритм випадкового лісу виявився надзвичайно ефективним, здатним вирішувати практичні завдання. Він дає високу точність навчання при великій кількості випадків, внесених в процес побудови моделі[40]. Перевага перед іншими моделями машинного навчання - оцінка викидів даних для частини множини, що не потрапила в навчальну вибірку. Тому для дерев рішень не обов'язково проводити крос-валідацію або тестування на окремій вибірці. Досить обмежитися оцінкою викидів даних для подальшого покращення моделі: підбору кількості вирішальних дерев і регулюючої складової.

Переваги алгоритму:

- висока швидкість навчання;
- неітеративне навчання - алгоритм завершується за фіксоване число операцій;
- масштабованість (здатність обробляти великі обсяги даних);

- висока якість одержуваних моделей (прирівняне до нейронних мереж і ансамблів нейронних мереж);
- відсутність чутливості до викидів в даних через випадковий вібір підмножин;
- мала кількість параметрів, що налаштовуються;
- відсутність чутливості до масштабування (і взагалі до будь-яких монотонних перетворень) значень ознак, завдяки вибору випадкових підпросторів;
- не вимагає ретельного налаштування параметрів, добре працює з початковими налаштуваннями. За допомогою покращення налаштувань параметрів можна досягти приросту від 0.5 до 3% точності в залежності від завдання і даних;
- добре працює з пропущеними даними - зберігає високу точність, навіть якщо більша частина даних пропущена;
- внутрішня оцінка здатності моделі до узагальнення;
- можливість роботи з сирими даними, без попередньої обробки.

Недоліки алгоритму:

- побудована модель займає велику кількість пам'яті. Якщо ми будуємо комітет з K дерев на основі навчальної множини розміром N, то вимоги до пам'яті складуть O (K · N). Але обсяги оперативної пам'яті у сучасних ПК досить великий, так що це не самий серйозний недолік;
- навчена модель працює трохи повільніше ніж інші алгоритми (якщо в модель входить 100 дерев, необхідно пройтися по всіх, щоб отримати результат). Але на сучасних швидких машинах і це не настільки помітно;
- алгоритм працює гірше багатьох лінійних методів, коли у вибірці дуже багато розріджених ознак (тексти) або коли об'єкти, що класифікуються, свідомо можуть бути розділені лінійно;
- для даних, що включають категоріальні змінні з різною кількістю рівнів, Випадковий ліс упереджений на користь ознак з великою кількістю рівнів.

Дерево буде сильніше підлаштовуватися саме під такі ознаки, оскільки на них можна отримати більш високе значення функціоналу, що оптимізується;

- як і дерева рішень, алгоритм абсолютно не здатний до екстраполяції (але це можна вважати і плюсом, тому що не буде екстремальних значень в разі потрапляння викиду).

2.10. Підсилення

Підсилення або бустінг - це сімейство ансамблевих алгоритмів, суть яких полягає в створенні сильного класифікатора на основі декількох слабких. Для цього спочатку створюється одна модель, потім інша модель, яка намагається виправити помилки в першій (рис. 2.9). Моделі додаються до тих пір, поки тренувальні дані не будуть ідеально передбачати або поки не буде перевищено максимальну кількість моделей.

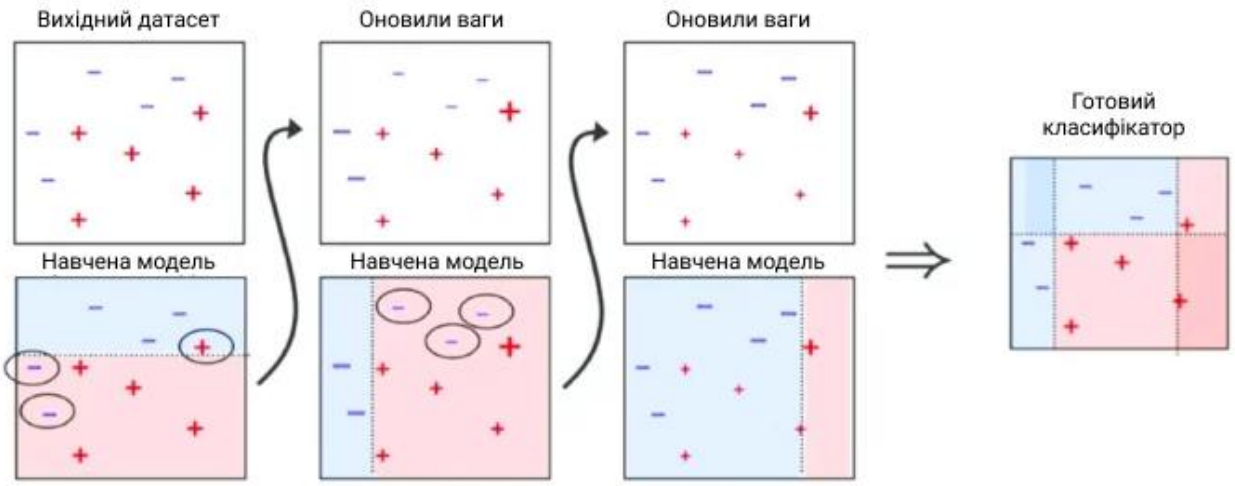

Рис. 2.9. Схема алгоритму підсилення

AdaBoost був першим дійсно успішним алгоритмом підсилення, розробленим для бінарної класифікації. Сучасні методи на кшталт стохастичного градієнтного підсилення ґрунтуються на AdaBoost.

AdaBoost використовують разом з короткими деревами рішень. Після створення першого дерева перевіряється його ефективність на кожному

тренувальному об'єкті, щоб зрозуміти, скільки уваги має приділити наступне дерево всім об'єктам. Тим даними, які складно передбачити, дається більша вага, а тим, які легко передбачити, - менша. Моделі створюються послідовно одна за одною, і кожна з них оновлює ваги для наступного дерева. Після побудови всіх дерев робляться прогнози для нових даних, і ефективність кожного дерева залежить від того, наскільки точним воно було на тренувальних даних.

Так як в цьому алгоритмі велика увага приділяється виправленню помилок моделей, важливо, щоб в даних були відсутні аномалії.

Висновки до другого розділу

В цьому розділі було розглянуто та наведено принцип роботи 10 моделей машинного навчання. Для подальшої роботи було відібрано три моделі: лінійну, випадковий ліс та k-найближчих сусідів. Лінійна регресія була обрана як один з найбільш відомих і зрозумілих алгоритмів в статистиці і машинному навчанні. Модель Випадковий ліс характеризується високою точністю навчання при великій кількості випадків, внесених в процес побудови моделі. Перевага перед іншими моделями машинного навчання - оцінка викидів даних для частини множини, що не потрапила в навчальну вибірку. Також перевагами цієї моделі машинного навчання є висока швидкість навчання, масштабованість, здатність працювати з пропущеними даними, невисока чутливість до викидів, відсутність чутливості до масштабування. Метод k-найближчих сусідів був обраний через високу швидкість передбачень, але потребує попереднього відбору данних.

3. АНАЛІЗ І ПЕРЕДБАЧЕННЯ ВИКОРИСТАНОЇ ТА ЗГЕНЕРОВАНОЇ ЕЛЕКТРОЕНЕРГІЇ З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ

3.1. Підготовка, аналіз та відбір даних для прогнозування

В якості даних було обрано датасет розумного будинку, який обладнаний сонячними батареями для генерації власної електроенергії, яка частково покриває потреби будинку. В датасеті присутні колонки:

- Час («time») – дата та години в які було зроблено вимірювання.
- Використана електроенергія («use [kW]») – сумарне значення використаної електроенергії
- Згенерована електроенергія («gen [kW]») – сумарне значення згенерованої електроенергії
- Погодні умови:
  - температура («temperature»);
  - вологість («humidity»);
  - видимість («visibility») ;
  - тиск («pressure») ;
  - швидкість вітру («windSpeed») ;
  - хмарний покрив («cloudCover») ;
  - напрям вітру («windBearing») ;
  - як відчувається температура людиною («apparentTemperature») ;
  - підсумок («summary») – загальний опис погоди (вітряно, мряка, дощ, сніг, туман та інше) ;
  - інтенсивність опадів («precipIntensity») ;
  - точка роси («dewPoint») ;
  - імовірність опадів («precipProbability») ;
- енергія, що споживається конкретними елементами будинку;
  - посудомийна машина («Dishwasher [kW]») ;
  - піч («Furnace 1 [kW]», «Furnace 2 [kW]») ;

- домашній офіс («Home office [kW]») ;
- холодильник («Fridge [kW]») ;
- винний погреб («Wine cellar [kW]») ;
- дверцята гаражів («Garage door [kW») ;
- амбар («Barn [kW]»);
- кухня («Kitchen 12 [kW]», «Kitchen 14 [kW]», «Kitchen 38 [kW]») ;
- мікрохвильова піч («Microwave [kW]»);
- вітальня («Living room [kW]»).

Період фіксації даних з 1 січня 2016 року по 16 грудня 2016 року. Часовий інтервал між записами – 1 хвилина.

В обраному датасеті присутні окремо виміри спожитої енергії по конкретним елементам будинку, а також їх сумарне значення (рис. 3.1). Так як завданням дисертації є визначення всієї спожитої енергії, то для прогнозування використовувалась ознака «use [kW]».

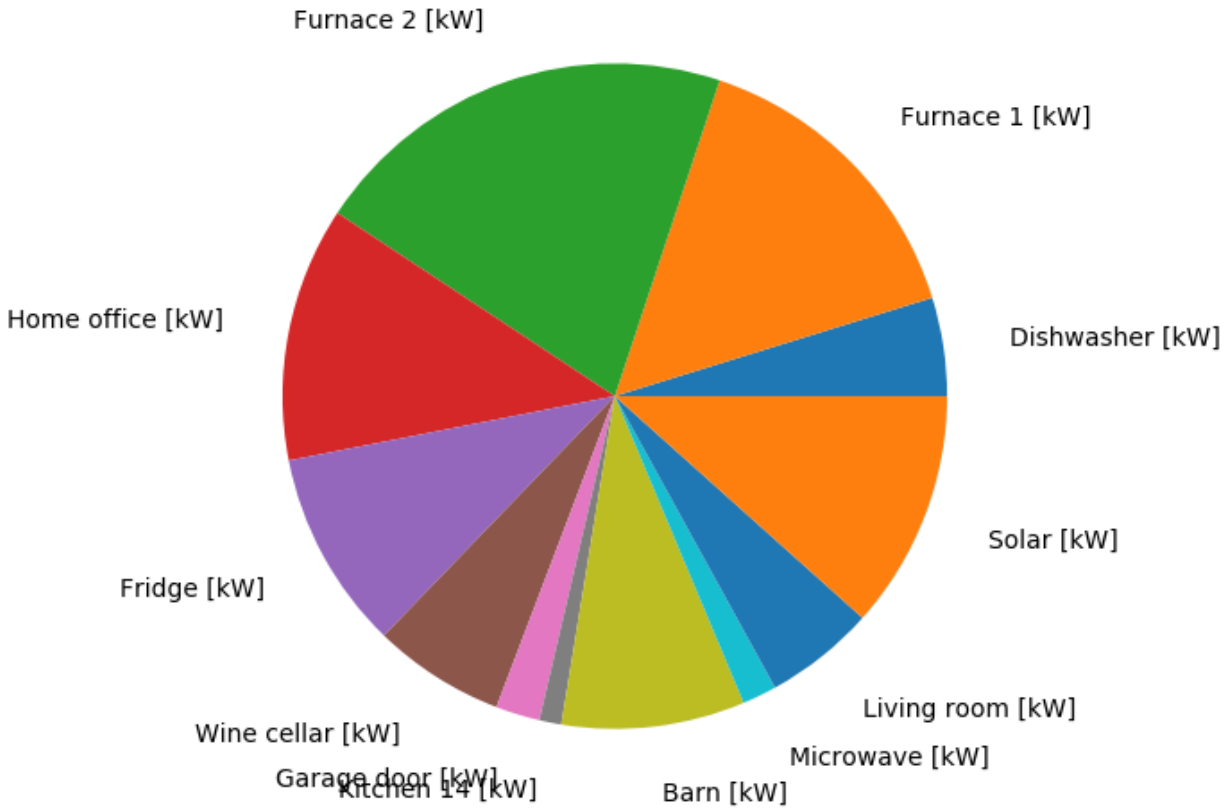

Рис. 0.1 Діаграма використання енергії окремими об’єктами будинку протягом року

Реальні виміри з датчиків не можуть бути ідеальними. Причиною цьому можуть слугувати: збої в роботі датчиків, відсутність з'єднання або живлення та інше. Саме тому перед аналізом даних і подальшим навчанням моделей машинного навчання необхідно провести обробку отриманих даних.

В першу чергу було досліджено кількість пропущених та нульових значень в отриманих даних, а також їх відсоток від загальної кількості даних. Результати висвітленні в табл. 3.1.

Таблиця 0.1

Назва	**Нульові значення**	**Пропущені значення**	**Кіл-ть нульових і пропущених значень**	**% Нульових і пропущених значень**
precipProbability	416607	1	416608	82,7
precipIntensity	416607	1	416608	82,7
Kitchen 38 [kW]	247710	1	247711	49,2
Dishwasher [kW]	182720	1	182721	36,3
cloudCover	68236	59	68295	13,6
Kitchen 14 [kW]	50310	1	50311	10
windBearing	1787	1	1788	0,4
Kitchen 12 [kW]	367	1	368	0,1
windSpeed	230	1	231	0
Living room [kW]	144	1	145	0
Microwave [kW]	140	1	141	0
gen [kW]	64	1	65	0
Barn [kW]	12	1	13	0
use [kW]	1	1	2	0
House overall [kW]	1	1	2	0
apparentTemperature	0	1	1	0
dewPoint	0	1	1	0
Furnace 1 [kW]	0	1	1	0
Furnace 2 [kW]	0	1	1	0
pressure	0	1	1	0
Garage door [kW]	0	1	1	0
visibility	0	1	1	0
humidity	0	1	1	0
Wine cellar [kW]	0	1	1	0
Home office [kW]	0	1	1	0
Fridge [kW]	0	1	1	0
temperature	0	1	1	0

Як видно з табл. 3.1, в даних присутні показники з кількістю нульових і пропущених даних більше 50% - інтенсивність опадів («precipIntensity») та імовірність опадів («precipProbability»). Але проаналізувавши ці два параметри робимо висновок, що велика кількість нульових значень в них спричинена їх природою, тому видаляти їх немає необхідості.

Наступний крок – позбутися викидів. Викид - це спостереження, яке знаходиться на ненормальній відстані від інших значень у випадковій вибірці з сукупності. Вони можуть бути пов'язані з помилками, помилками в одиницях виміру або бути коректними, але надто екстремальними значеннями. У певному сенсі це визначення залишає аналітику (або процесу консенсусу) рішення, що буде вважатися ненормальним. Перш ніж ненормальні спостереження можна буде виділити, необхідно охарактеризувати нормальні спостереження.

Для характеристики набору даних необхідні дві дії [15]:

1. Вивчення загальної форми графічних даних на предмет важливих особливостей, включаючи симетрію і відхилення від припущень;
2. Вивчення даних для незвичайних спостережень, які далекі від маси даних. Ці точки часто називають викидами. Два графічних методи для визначення викидів, діаграми розсіювання і коробчаті діаграми, а також аналітична процедура для виявлення викидів, коли розподіл нормальне (тест Граббса).

Діаграма розмаху представляє собою корисне графічне відображення для опису поведінки даних в середині, а також на кінцях розподілів. Діаграма розмаху використовує медіану, нижній і верхній квартилі (визначені як 25 і 75 відсотків). Якщо нижній квартиль - Q_1, а верхній квартиль - Q_3, то різниця $(Q_3 - Q_1)$ називається міжквартильним діапазоном або IQ.

Для визначення аномальних даних використовують такі терміни [15]:

- нижня внутрішня межа: $Q_1 - 1{,}5 \cdot IQ$;
- верхня внутрішня межа: $Q_3 + 1{,}5 \cdot IQ$;
- нижня зовнішня межа: $Q_1 - 3 \cdot IQ$;

- верхня зовнішня межа: $Q_3 - 3 \cdot IQ$;

Точки, що лежать в межах $Q_1 - 3 \cdot IQ \le x < Q_1 - 1{,}5 \cdot IQ$ або $Q_1 + 1{,}5 \cdot IQ < x \le Q_3 + 3 \cdot IQ$ називаються помірними викидами, а $Q_1 - 3 \cdot IQ > x > Q_3 + 3 \cdot IQ$ називаються екстримальними викидами.

В данній роботі було видалено екстримальні значення, результат зображено на рис. 3.2.

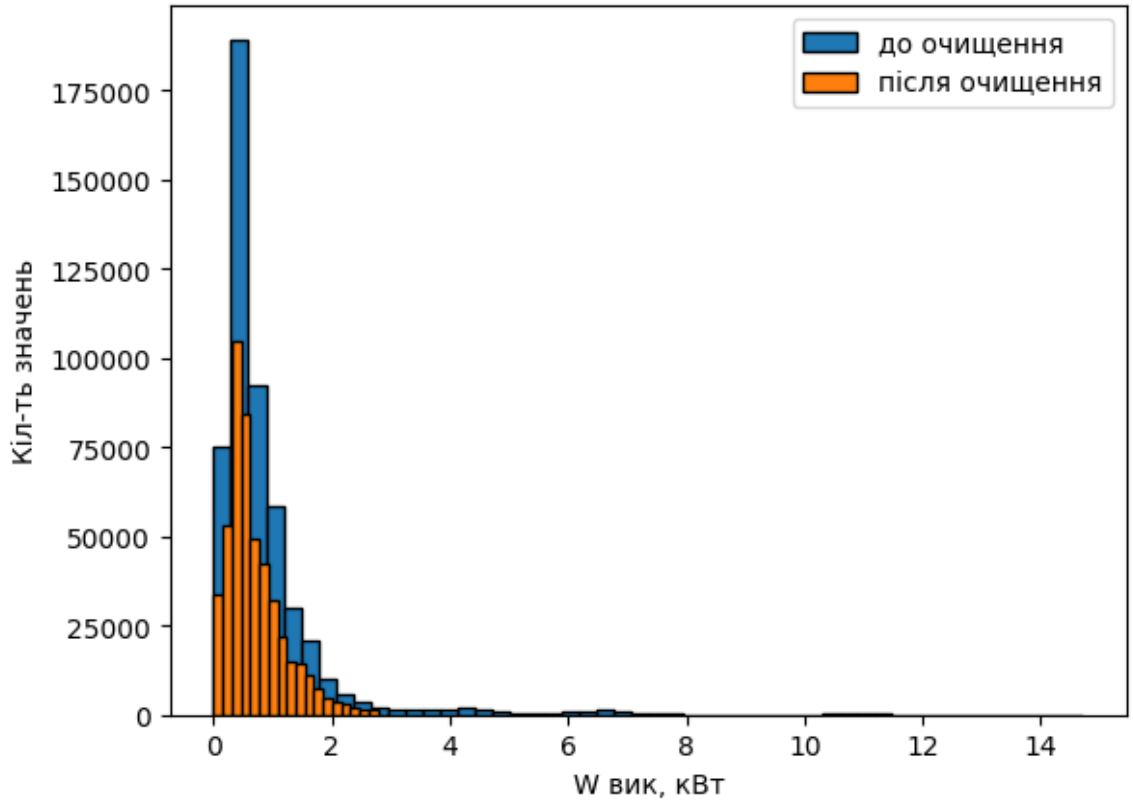

Рис. 0.2.Розподіл значень використаної електроенергії до та після очищення

Для того, щоб виконати повний аналіз необхідно забезпечити числові значення у всіх полях датасету, позбутися категоріальних змінних. Категоріальні дані – це змінні, які містять значення міток, а не числові значення. Кількість можливих значень часто обмежується фіксованим набором.

Так як в датасеті є категоріальна змінна - хмарний покрив («summary») , необхідно замінити текстові поля - такі як: «хмарно», «вітряно», «дощ»,

«легкий дощ», «сніг», «туман», «прохолодно» та інші - на числові коди. Деякі алгоритми можуть працювати безпосередньо з категорійними даними. Наприклад, дерево рішень можна побудувати безпосередньо з категоричних даних без необхідної трансформації даних (це залежить від конкретної реалізації). Але багато алгоритмів машинного навчання не можуть безпосередньо працювати над даними міток. Вони вимагають, щоб усі вхідні та вихідні змінні були числовими. Загалом, це здебільшого обмеження ефективної реалізації алгоритмів машинного навчання, а не жорсткі обмеження в самих алгоритмах. Це означає, що категоричні дані повинні бути перетворені в числову форму. Якщо категоріальна змінна є вихідною змінною, то прогнози моделі можна перетворити назад у категорійну форму, щоб представити їх або використовувати їх у іншій програмі.

Для категоричних змінних, де немає порядкового співвідношення, цілочисельного кодування недостатньо. Фактично, використання цього кодування та надання можливості моделі припускати природне впорядкування між категоріями може призвести до низької продуктивності або несподіваних результатів (прогнози на півдорозі між категоріями). У цьому випадку до цілого представлення може бути застосоване бінарне кодування – кожному унікальному значенню присвоюється нова двійкова змінна. Перевага такого кодування над цілочисельним кодуванням в тому, що можливо проаналізувати, сукупну ознаку двох змінних, як нову величину, так і окрему складову кожної з них. Саме тому набору даних в стовпці «summary» було присвоєно номери в степені двійки від 1 до 14 (за кількістю наявних неповторюваних ознак).

Вибір ознак полягає в тому, що нам необхідно обрати ознаки найбільш підходящі для тренування моделі. Багато з наявних ознак для нашої моделі надлишкові, тому що деякі з них сильно корелюють. Наприклад, залежність «temperature» від «apparentTemperature» (рис. 3.3) має коефіцієнт кореляції 0.993, що буде негативно впливати на модель при її навчанні.

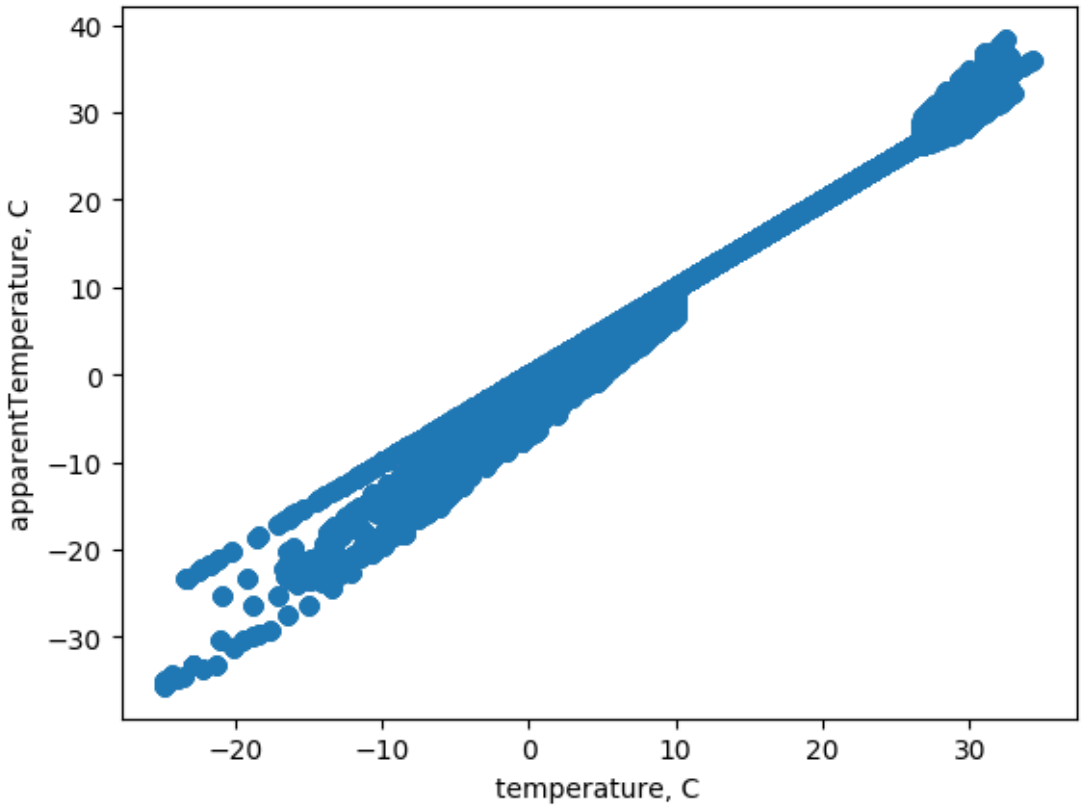

Рис. 0.3. Графік залежності між реальною температурою і тим, як відчувається людиною

Для того щоб чисельно оцінити ступінь впливу ознак на значення спожитої та згенерованої електроенергії використано коефіцієнт кореляції Пірсона. Це міра ступеня і позитивності лінійних зв'язків між двома змінними[43].

При наявності двох вибірок $x^m = (x_1, \ldots, x_m), y^m = (y_1, \ldots, y_m)$, коефіцієнт кореляції Пірсона розраховується за формулою[43]:

$$r_{xy} = \frac{\sum_{i=1}^{m} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{m} (x_i - \bar{x})^2 (y_i - \bar{y})^2}}$$

де $\bar{x}, \bar{y}$ – середнє значення вибірок.

Коефіцієнт кореляції Пірсона знаходиться в межах [-1,1]:

- $|r_{xy}| = 1 \Rightarrow x, y$ лінійно залежні,
- $r_{xy} = 0 \Rightarrow x, y$ лінійно незалежні.

Використовуючи бібліотеку Pandas, розрахуємо величину кореляції. Результати наведено на рис. 3.5.

Ознаки, які сильно корельовані, називають колінеарними, і досить залишити одну з таких ознак (за умови, що ознаки корельовані між собою, а не з цільовою ознакою), щоб допомогти алгоритму краще узагальнювати і отримувати більш інтерпретовані результати на виході.

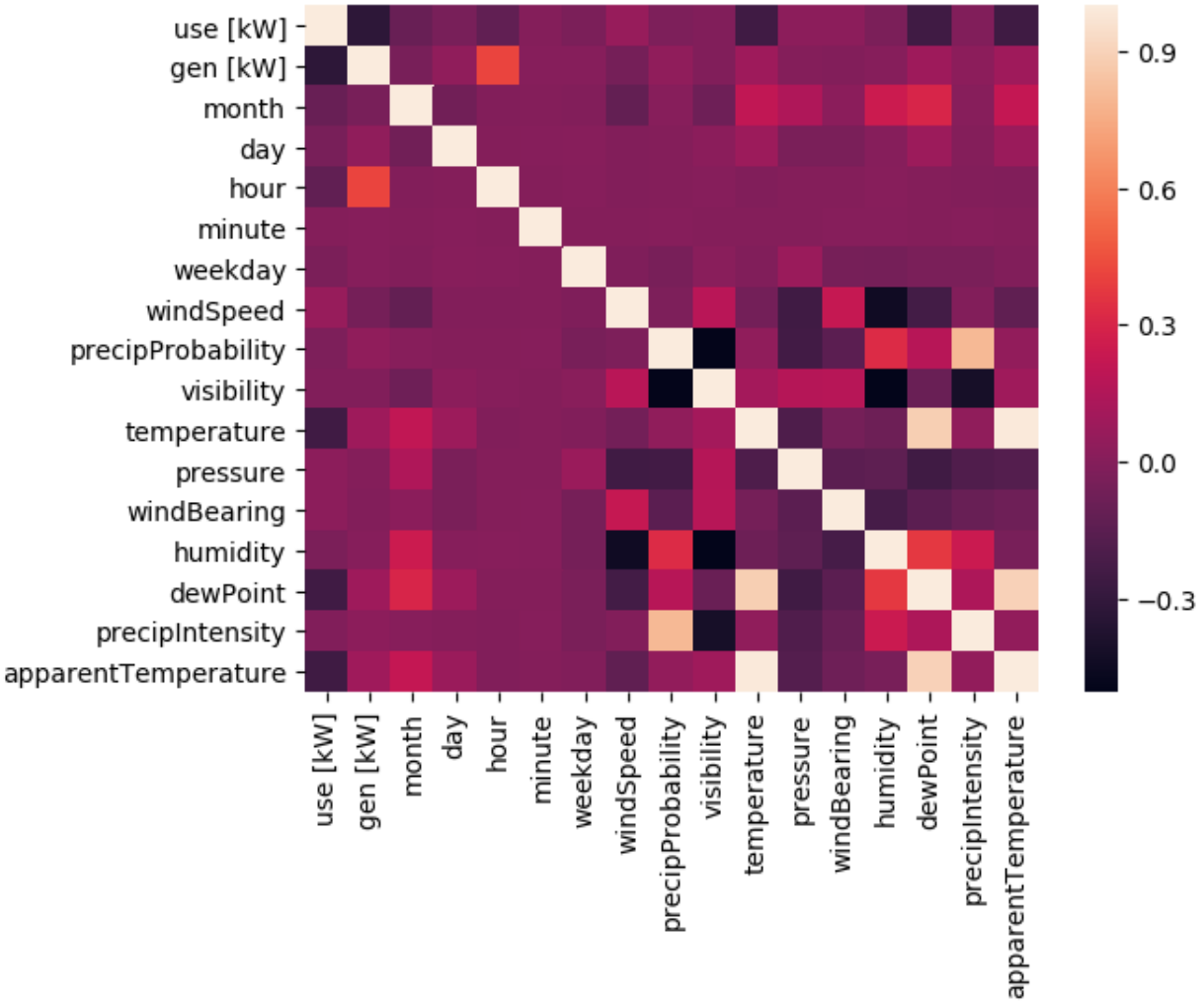

Рис. 0.4. Графічне зображення матриці кореляції Пірсона

Саме тому з вибірки прибираємо три параметри: значення температури, яка відчувається людиною («apparentTemperature»), точку роси («dewPoint») і імовірність опадів («precipProbability»).

3.2. Порівняння та вибір найкращого методу машинного навчання для поставленої задачі

Для початку передбачень і оцінки точності, потрібно розділити вибірку на навчальну і перевірочну (тестову).

Навчальна вибірка це набір який подається на вхід моделі в процесі навчання разом з відповідями, з метою навчити модель бачити зв'язок між цими ознаками і правильною відповіддю.

Тестова вибірка використовується для перевірки моделі. Модель не отримує цільову ознаку на вхід і, більш того, повинна передбачити її величину використовуючи значення інших ознак. Ці передбачення потім порівнюються з реальними відповідями.

Останній крок перед початком навчання моделі - визначення критерію, за яким можна зрозуміти, чи є хоч якийсь сенс від обраного алгоритму. Наприклад, можна порівняти результат роботи моделі зі спробою просто вгадати цільову ознаку, нічим особливим не керуючись. Якщо обраний алгоритм працює гірше, ніж неусвідомлений перебір можливих значень цільової ознаки, тоді варто спробувати інший підхід, можливо, навіть не пов'язаний з машинним навчанням.

Для задач регресії таким можливим критерієм може виступати підстановка медіанних значень цільової ознаки. Хоча це досить низький поріг для більшості алгоритмів. Як метрику для оцінки точності було вирішено використовувати коефіцієнт детермінації R^2 — статистичний показник, що використовується в статистичних моделях як міра залежності варіації залежної змінної від варіації незалежних змінних[8]. Коефіцієнт детермінації R^2 розраховується за наступною формулою[9]:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}},$$

де SS_{res} - сума квадратів лишків регресії, SS_{tot} - загальна сума квадратів.

Суми квадратів розраховуються за наступними формулами:

$$SS_{res} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$

$$SS_{tot} = \sum_{i=1}^{n} (y_i - \bar{y})^2,$$

де y_i , $\hat{y}_i$ – фактичне та розрахункове значення досліджуваної змінної, $\bar{y}$ - середнє арифметичне значення досліджуваної функції.

Для навчання і тестування було розділено початкову базу даних: 70% - навчальна вибірка, 30% - тестова. Для порівняння було обрано три моделі машинного навчання з бібліотеки scikit-learn: лінійна, випадковий ліс, k-найближчих сусідів.

Як видно з рис. 3.6 найбільшу точність для передбачень використаної та згенерованої електроенергії показала модель Випадковий ліс.

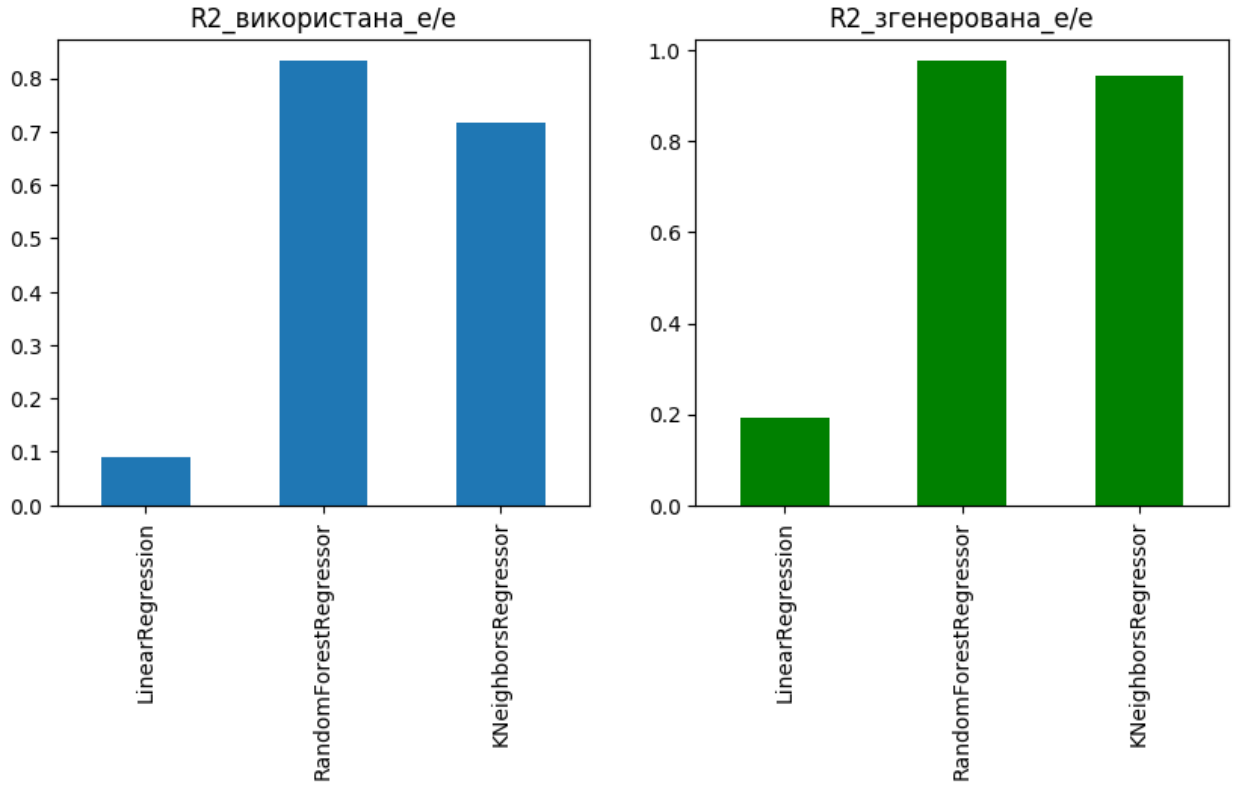

Рис. 0.5. Оцінка точності згідно коефіцієнта детермінації

3.3. Аналіз результатів та пошук ваги факторів, які впливають на кількість спожитої та згенерованої електроенергії

Для передбачень кількості використаної та згенерованої електроенергії, було обрано метод Випадковий ліс. В табл. 3.2 наведено вклад найбільш вагомих ознак, а результати прогнозування для кількості використаної електроенергії наведено на рис. 3.7.

Таблиця 0.2

Ознака	Вклад, %
Місяць ("month")	7,47
День ("day")	4,35
Години ("hour")	16,6
Хвилини ("minute")	33,89
День тижня ("weekday")	3,33
Швидкість вітру («windSpeed»)	5,01
Інтенсивність опадів («precipIntensity")	1,03
Видимість («visibility»)	2,36
Температура («temperature»)	9,09
Тиск («pressure»)	7,06
Напрям вітру («windBearing»)	5,18
Вологість («humidity»)	4,64

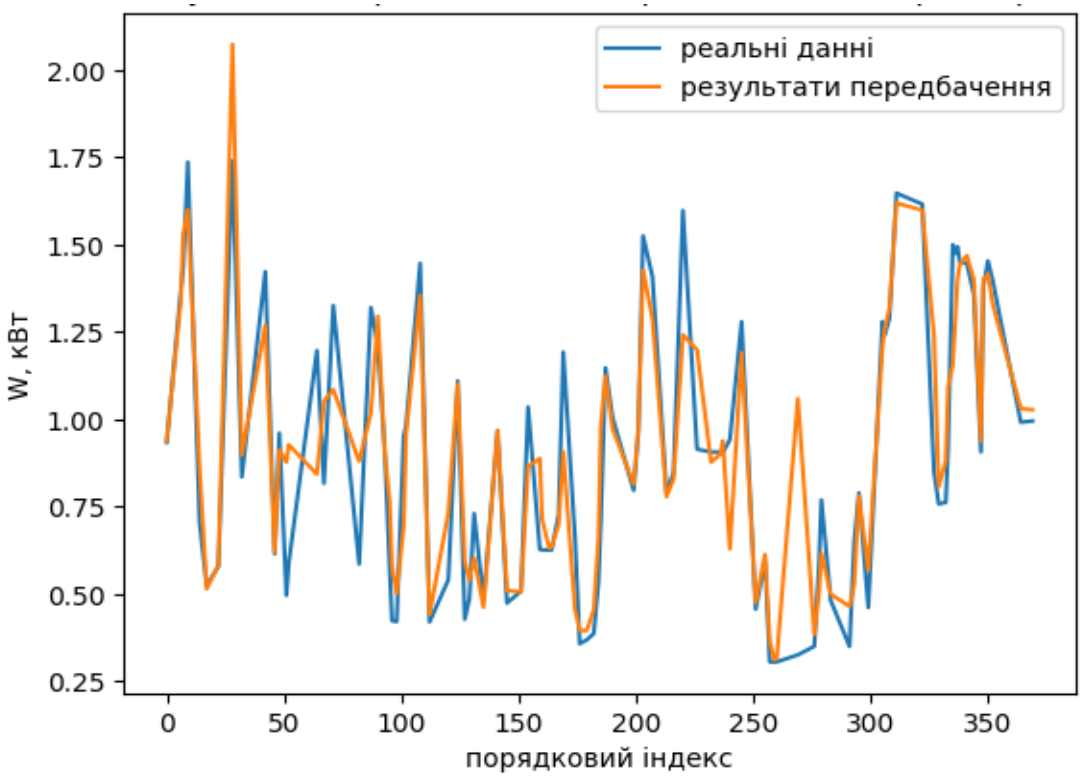

Рис. 0.6. Результати передбачень використаної електроенергії

В табл. 3.3 наведено вклад найбільш вагомих ознак, а результати прогнозування для кількості використаної електроенергії наведено на рис. 3.7.

Таблиця 0.3

Ознака	Вклад, %
Місяць ("month")	4,8
День ("day")	4,9
Години ("hour")	55,6
Хвилини ("minute")	5,2
День тижня ("weekday")	2,6
Швидкість вітру («windSpeed»)	3,8
Інтенсивність опадів («precipIntensity")	1
Видимість («visibility»)	1,8
Температура («temperature»)	6
Тиск («pressure»)	5,7
Напрям вітру («windBearing»)	4
Вологість («humidity»)	4

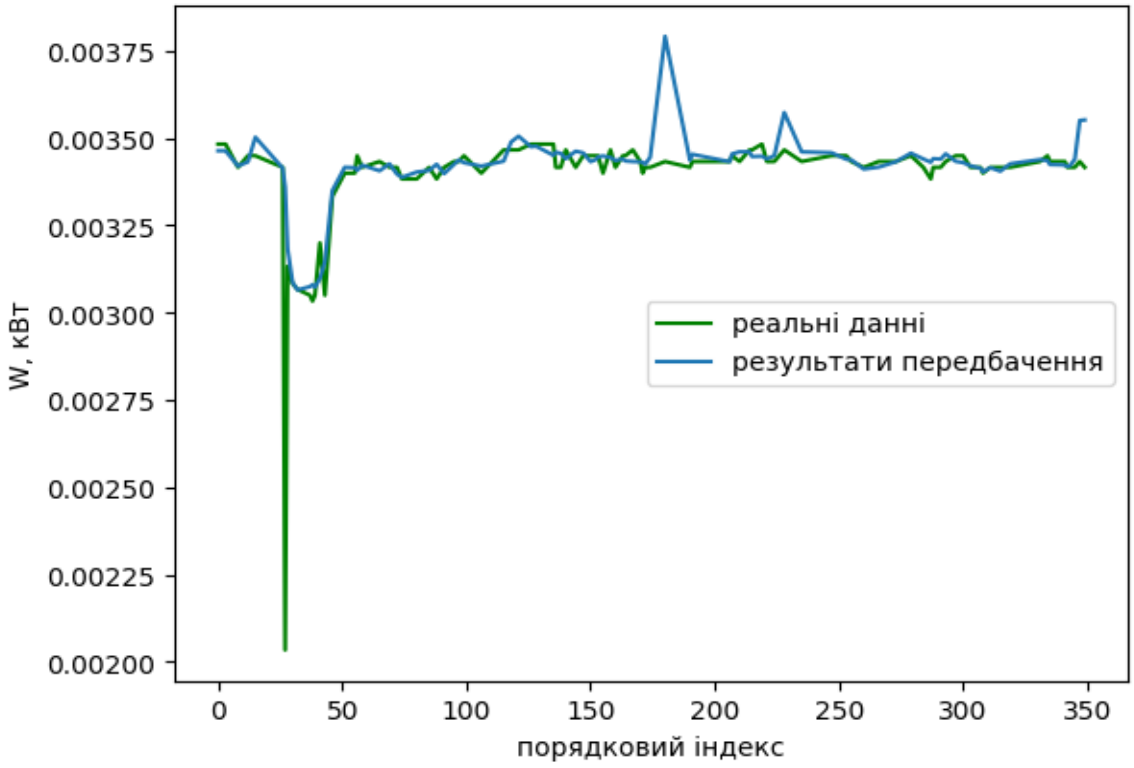

Рис. 0.7. Результати передбачень згенерованої електроенергії

З отриманих результатів можна зробити висновок, що інтенсивність опадів та видимість мають низький вплив на модель. Це може бути

спричинено малою кількістю даних і покращено в майбутньому зі збільшенням даних.

3.4. Розрахунок вартісного критерія і вигоди від прогнозування даних

Собівартість 1 кВт*год. електроенергії, генерованої енергоустановкою на базі альтернативних джерел енергії можна обчислити за допомогою наступної формули[44]:

$$C_{уст} = \frac{K(t) + C(t)}{W_{ген}(t)},$$

де Wген – сумарна згенерована потужність енергоустановкою за деякий термін t; К(t) – затрати на генерацію енергії протягом часу t;

Затрати на генерацію енергії протягом часу t рівні:

$$K(t) = \frac{K_{уст} + K_{пр} + K_{мон}}{Тексп} \cdot t,$$

де $К_{уст}$ – вартість комплекту обладнання; $К_{пр}$– вартість проектних робіт, визначення місця встановлення на місцевості; $К_{мон}$ – вартість будівельних та монтажних робіт, вартість встановлення, $Т_{експ}$ – період експлуатації

На рис. 3.9 – 3.11 наведено коливання ціни в залежності від періоду її прорахунку: година, день, місяць.

Виходячи з отриманих графіків було вирішено використовувати розрахунок ціни з періодом день.

Використовуючи методи машинного навчання для прогнозування кількості використаної та згенерованої електроенергії можна досягти економії витрат споживачів за рахунок закупівлі електроенергії в нічний час за зниженим тарифом.

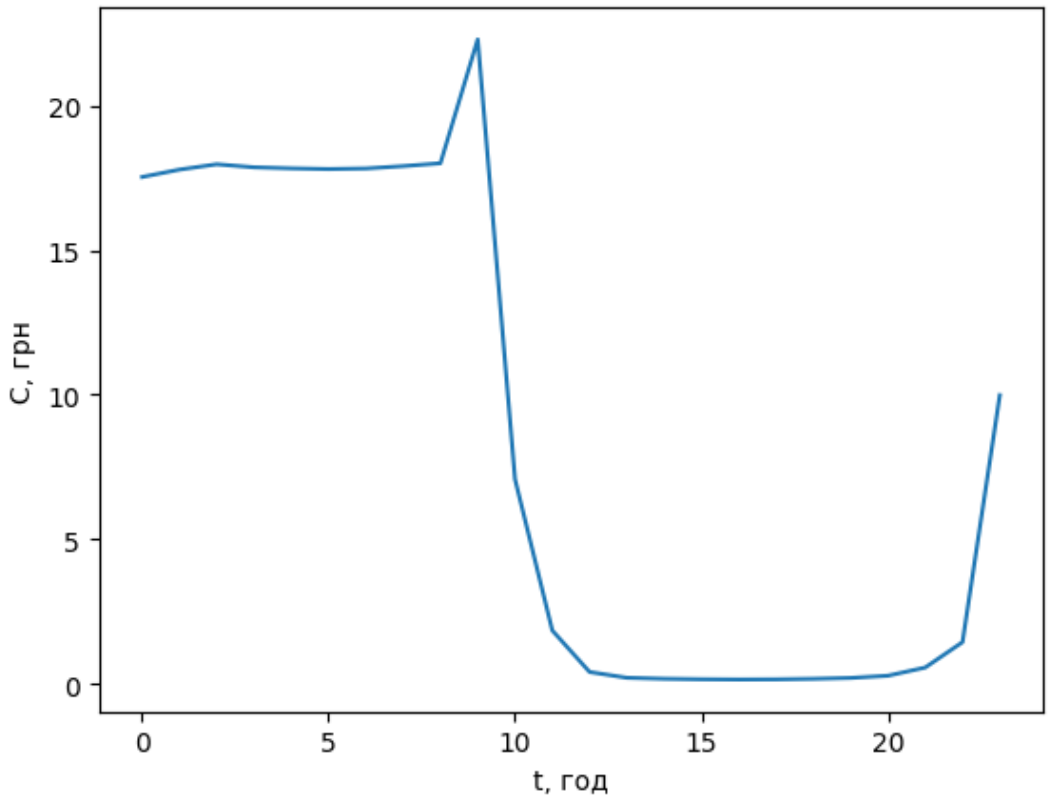

Рис. 0.8. Графік зміни ціни погодинно

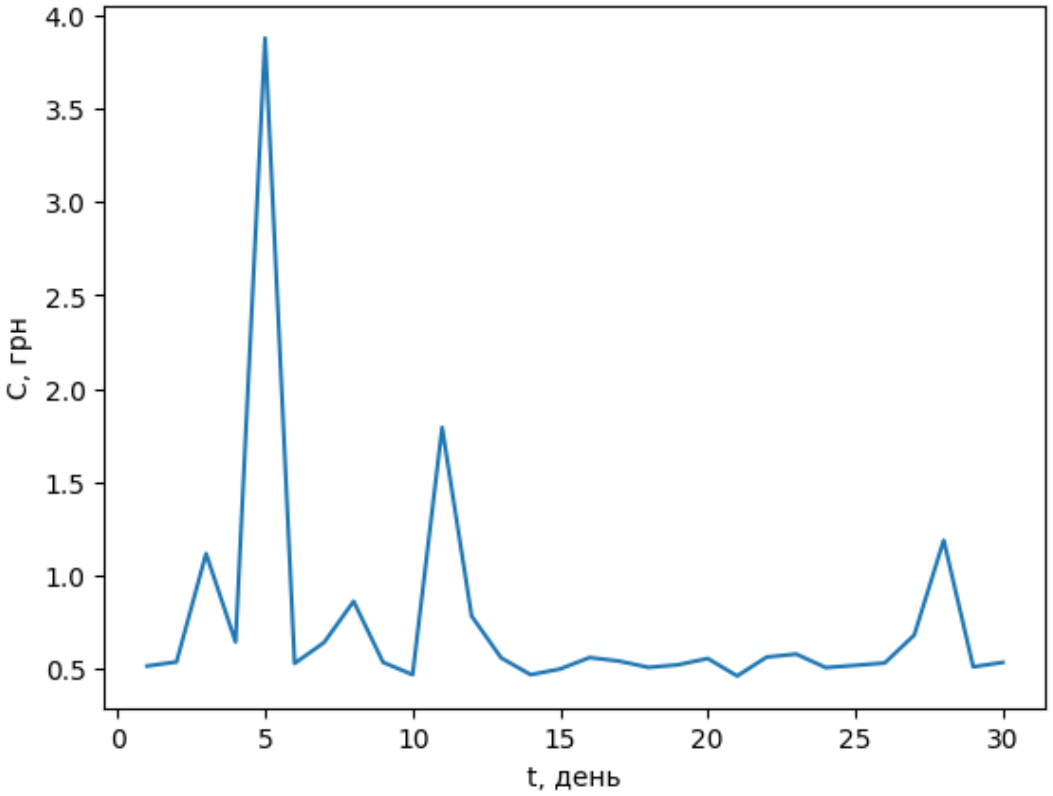

Рис. 0.9 Графік зміни ціни щоденно

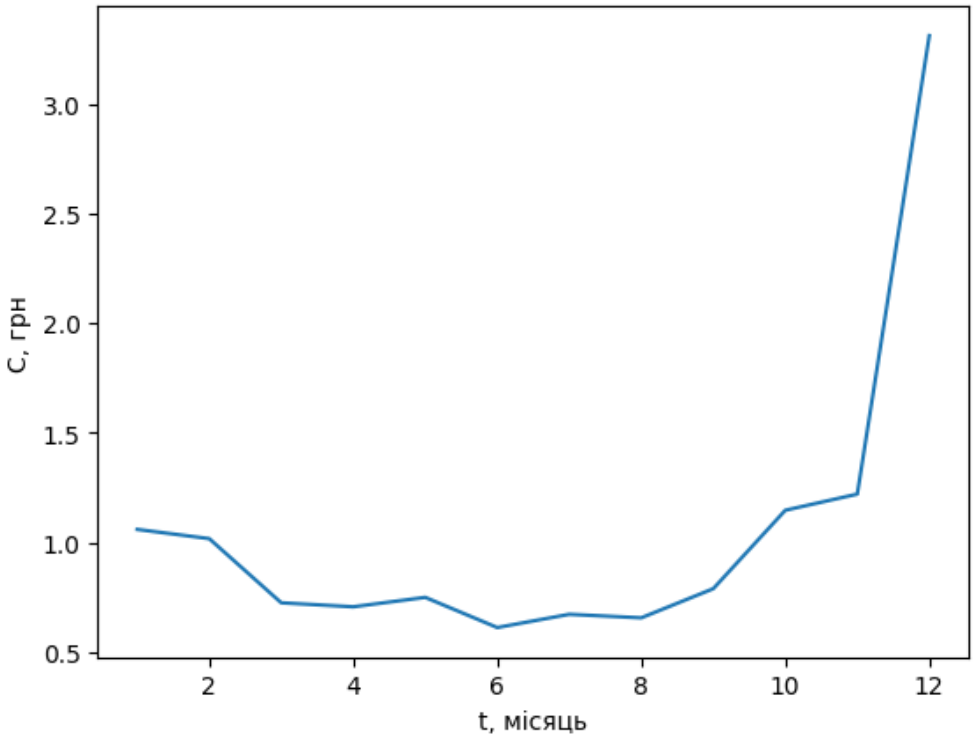

Рис. 0.10. Графік зміни ціни щомісяця

Для цього необхідно прогнозувати витрати та генерацію електроенергії на один день вперед, а далі запасати невистачаючу електроенергію з мережі у нічний час. Це дозволить не тільки економити витрати споживачів (рис. 3.12), але і врівноважити навантаження на міську систему.

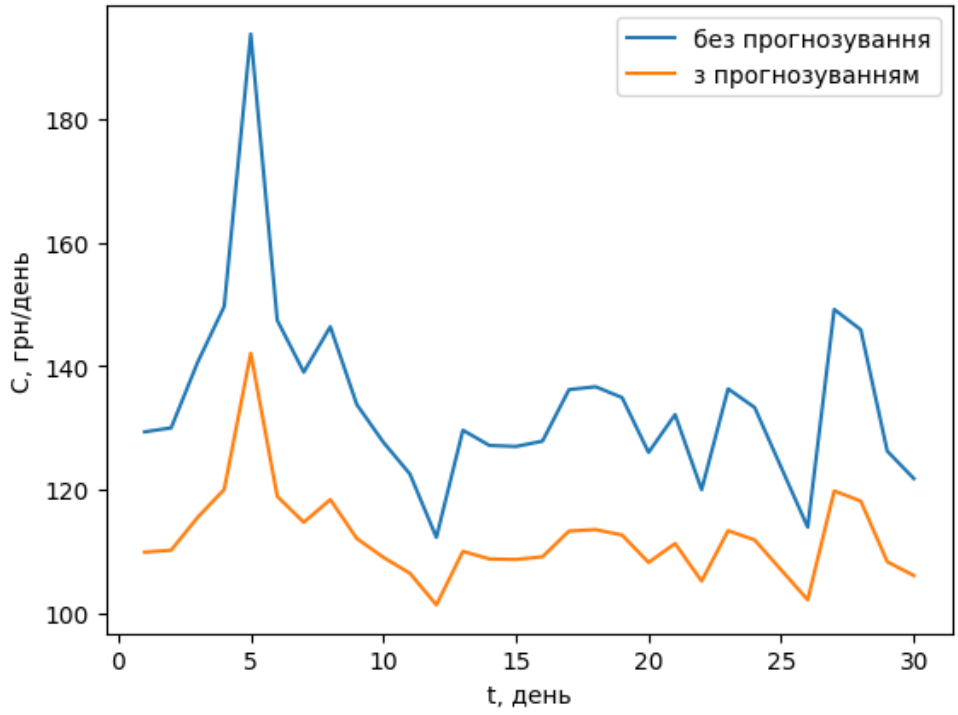

Рис. 0.11. Графік ціни електроенергії на день без та з використанням передбачень

Висновки до третього розділу

В якості даних було обрано датасет розумного будинку, який обладнаний сонячними батареями для генерації власної електроенергії, яка частково покриває потреби будинку. Набір даних, що використовувався в даній статті, містить відомості про використання та генерацію електроенергії, а також погодні показники за 11 місяців з періодом фіксації даних 1 хвилина. Оброблення даних ґрунтувалось на статистичних методах обробки інформації, визначенні кількості пропущених даних, лінійних залежностях між ознаками, сумісності типів даних. Для прогнозування було обрано три моделі машинного навчання: лінійну, випадковий ліс та k-найближчих сусідів. Для оцінки точності передбачень було використано коефіцієнт детермінації.

Серед протестованих моделей машинного навчання найвища точність була у моделі Випадковий ліс (84% для використаної електроенергії, та 95% для згенерованої електроенергії). Також використовуючи описані методи обробки даних стало можливим покращити точність прогнозу (на 25% для використаної електроенергії та на 2% для згенерованої електроенергії). Було виокремленно вклад найбільших ознак, які впливали на передбачення моделі Випадковий ліс.

Також було прораховано собівартість згенерованої електроенергії за 1 кВт*год та наведено вигоду для користувачів систем MicroGrid при використанні прогнозування для завчасної закупівлі електроенергії в нічний час (за зниженими тарифами).

4. РОЗРОБКА СТАРТАП-ПРОЕКТУ

Стартап як форма малого ризикового (венчурного) підприємництва впродовж останнього десятиліття набула широкого розповсюдження у світі через зниження бар'єрів входу в ринок (із появою Інтернету як інструменту комунікацій та збуту стало простіше знаходити споживачів та інвесторів, займатись пошуком ресурсів, перетинати кордони між ринками різних країн), і вважається однією із наріжних складових інноваційної економіки, оскільки за рахунок мобільності, гнучкості та великої кількості стартап-проектів загальна маса інноваційних ідей зростає.

Проте створення та ринкове впровадження стартап-проектів відзначається підвищеною мірою ризику, ринково успішними стає лише невелика частка, що за різними оцінками складає від 10% до 20%. Ідея стартап-проекту, взята окремо, не вартує майже нічого: головним завданням керівника проекту на початковому етапі його існування є перетворення ідеї проекту у працюючу бізнес-модель, що починається із формування концепції товару (послуги) для визначеної клієнтської групи за наявних ринкових умов.

Розробка та введення стартап-проекту на ринок передбачає собою багато здійснених кроків, де визначають ринкові графік, перспективи проекту, принципи організації виробництва аналіз ризиків та фінансовий аналіз і заходи з просування пропозиції для інвесторів.

Етапи розроблення стартап-проекту:

1. Маркетинговий аналіз стартап-проекту

В межах цього етапу:

- розробляється опис самої ідеї проекту та визначаються загальні напрями використання потенційного товару чи послуги, а також їх відмінність від конкурентів;
- аналізуються ринкові можливості щодо його реалізації;
- на базі аналізу ринкового середовища розробляється стратегія ринкового впровадження потенційного товару в межах проекту.

2. Організація стартап-проекту

В межах цього етапу:

- складається календарний план-графік реалізації стартап-проекту;
- розраховується потреба в основних засобах та нематеріальних активах;
- визначається плановий обсяг виробництва потенційного товару, на основі чого формулюється потреба у матеріальних ресурсах та персоналі;
- розраховуються загальні початкові витрати на запуск проекту та планові загальногосподарські витрати, необхідні для реалізації проекту.

3. Фінансово-економічний аналіз та оцінка ризиків проекту

В межах цього етапу:

- визначається обсяг інвестиційних витрат;
- розраховуються основні фінансово-економічні показники проекту (обсяг виробництва продукції, собівартість виробництва, ціна реалізації, податкове навантаження та чистий прибуток) та визначаються показники інвестиційної привабливості проекту (запас фінансової міцності, рентабельність продажів та інвестицій, період окупності проекту);
- визначається рівень ризикованості проекту, визначаються основні ризики проекту та шляхи їх запобігання (реагування на ризики).

4. Заходи з комерціалізації проекту

Цей етап спрямовано на пошук інвесторів та просування інвестиційної пропозиції (оферти). Він передбачає:

- визначення цільової групи інвесторів та опису їх ділових інтересів;
- складання інвест-пропозиції (оферти): стислої характеристики проекту для попереднього ознайомлення інвестора із проектом;
- планування заходів з просування оферти: визначення комунікаційних каналів та площадок та планування системи заходів з просування в межах обраних каналів;
- планування ресурсів для реалізації заходів з просування оферти.

Означені етапи, реалізовані послідовно та вчасно – створюють передумови для успішного ринкового старту. Проте фахівці зі створення та розвитку стартап-проектів окремо відзначають, що відсутність маркетингових знань та умінь, що уможливлюють розробку ринково затребуваного проекту із вихідної ідеї, є основною причиною високого рівня банкрутств стартап-компаній, і ця проблема може бути вирішена за рахунок навчання винахідників.

4.1. Опис ідеї проекту

Опис ідеї стартап-проекту наведено в табл. 4.1, визначення сильних, слабких та нейтральних характеристик ідеї проекту в табл. 4.2.

Таблиця 4.1. Опис ідеї стартап - проекту

Зміст ідеї	**Напрямки застосування**	**Вигоди для користувача**
Створення програмного забезпечення для передбачень кількості використаної та згенерованої електроенергії	Енергетична галузь	• Зменшення вартості електроенергії за кВт*год; • Врівноваження навантаження на систему загального електропостачання.

Таблиця 4.2.
Визначення сильних, слабких та нейтральних характеристик ідеї проекту

Техніко-економічні характеристики ідеї	*(потенційні) товари/концепції конкурентів*			*W слабка сторона*	*N нейтральна сторона*	*S сильна сторона*
	Мій проект	*Конкурент 1*	*Конкурент 2*			
Середня ціна за 1 кВт*год, грн	**0,84**	1,68	2,6			+
Екологічність	так	так	ні			+
Можливість контролювати кіл-ть виробленої електроенергії	**ні**	ні	так	-		

4.2. Технологічний аудит ідеї проекту

Технологічна здійсненність ідеї проекту наведена в табл. 4.3.

Таблиця 4.3.

Технологічна здійсненність ідеї проекту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
Прогнозувати кількість використаної та згенерованої електроенергії на один день вперед. Недостачу електроенергії закупати в нічний час, за зниженим тарифом і зберігати до використання на акумуляторних батареях	Розробка, дослідження, програмування;	Наявна	Доступна

4.3. Аналіз ринкових можливостей запуску стартап-проекту

Попередню характеристика потенційного ринку стартап проекту наведено в табл. 4.4, а характеристика потенційних клієнтів в табл. 4.5.

Таблиця 4.4.

Попередня характеристика потенційного ринку стартап проекту

Показники стану ринку (найменування)	*Характеристика*
Кількість головних гравців, од	5
Загальний обсяг продаж, грн/ум.од	-
Динаміка ринку (якісна оцінка)	Зростає
Наявність обмежень для входу (вказати характер обмежень)	Необхідність дорогих початкових інвестицій
Специфічні вимоги до стандартизації та сертифікації	Немає
Середня норма рентабельності в галузі (або по ринку), %	33.55

Таблиця 4.5.

Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Необхідність переходу на відновлювальні джерела енергії, та бажання створите незалежну,	1) Компанії, що займаються реалізацією систем електроживлення з використанням	Кількість користувачів	- до продукції: Ефективність Надійність - до компанії-постачальника: Професіоналізм
децентралізовану систему електропостачання	відновлюваних джерел енергії 2) Користувачі систем Micro Grid		Чесність Порядність Технічна підтримка

Фактори загроз, їх зміст і можлива реакція компанії наведено в табл. 4.6.

Таблиця 4.6.

Фактори загроз

Фактор	*Зміст загрози*	*Можлива реакція компанії*
Конкуренція	Можливість появи нових гравців на ринку, точність яких буде кращою за ту ж вартість	Модифікація алгоритмів системи для аналізу вхідних даних та передбачень систем.
Збої в роботі датчиків, або алгоритму роботи	Наявність вірусів, або сбої в налаштуваннях	Оновлення програмного забезпечення

В табл. 4.7 наведено фактори можливостей, їх зміст та можливу реакцію компанії.

Таблиця 4.7.

Фактори можливостей

Фактор	Зміст можливості	Можлива реакція компанії
Якість діагностики системи	Функціонально добре написане програмне забезпечення	Залучення нових клієнтів за допомогою маркетингу
Збільшення користувачів	Можливість торгувати надлишковою електроенергією з такими самими користувачами	Залучення нових клієнтів за допомогою маркетингу

В табл.4.8 наведено аналіз конкуренції в галузі за М.Портером.

Таблиця 4.8.

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: чиста	В кого дешевше - в того купують	Покращення точності передбачень
2. За рівнем конкурентної боротьби: світова	Належить до вузького ринку збуту;	Розширення функціоналу та орієнтації користувачів
3. За галузевою ознакою: міжгалузева	Може бути використана в декількох галузях, які працюють з постачанням електроенергії	Розширення функціоналу та галузей застосування, покращення точності передбачень
4. Конкуренція за видами товарів: товарно-видова	Відрізняється методом передбачення кількості використаної та згенерованої електроенергії	Покращення ефективності перетворювача
5. За характером конкурентних переваг: цінова та нецінова	Чим дешевше – тим привабливіше; Чим краще – тим рентабельніше;	Покращення цінової політики та якості товару
6. За інтенсивністю: не марочна	Не жорстка конкуренція	Не агресивні форми піару

В табл. 4.9 наведено фактори конкурентоспроможності та обґрунтування їх значущості.

Таблиця 4.9.

Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
Ціна	Серед схожих по характеристикам систем обиратимуть ту, яка дешевше
Точність	Серед схожих по ціні системах обиратимуть ту, яка буде мати більшу точність
Відомість	При рівності двох перших факторів обиратимуть більш відомий товар

В табл. 4.10 наведено SWOT- аналіз стартап-проекту.

Таблиця 4.10.

SWOT- аналіз стартап-проекту

Сильні сторони: Менша ціна Зменшення нвантаження на систему	**Слабкі сторони:** Немасовість Індивідуальне налаштування алгоритму
Можливості: Можливість переходу на децентралізоване електропостачання Можливість створення свого віртуального ринку	**Загрози:** Поява більш ефективних технологій прорахунку Збої в системі

4.4. Розробка маркетингової програми стартап-проекту

Для розроблення маркетингової програми стартап-проекту перш за все необхідно визначити базову стратегію конкурентної поведінки. Результати наведені в табл. 4.11.

Таблиця 4.11.

Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
З точки зору принципів побудови – ні З точки зору обраного алгоритму керування – так	Шукати нових	Так – загальні принципи побудови топології силової частини	Стратегія заняття конкурентної ніші

В табл. 4.12 наведено визначення ключових переваг концепції потенційного товару, тобто, вигоду, яку пропонує дана система для споживачів.

Таблиця 4.12.

Визначення ключових переваг концепції потенційного товару

Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
Підвищення точності передбачень кількості використаної та згенерованої електроенергії	Нижча ціна на електроенергію для споживачів та конкуренто-спроможність	Застосування покращеного алгоритму обробки даних перед прогнозуванням, що дозволяє підвищити точність передбачень.

В табл.4.13 наведено рівень цін на товари замінники та аналоги, а також верхня та нижня межа ціни на систему.

Таблиця 4.13.

Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
100-115% від ціни власного продукту	100-115% від ціни власного продукту	від 80000 грн	65000-120000грн

Формування системи збуту наведено в табл. 4.14.

Таблиця 4.14.

Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Оптова закупівля продукту	Збут та налаштування товару	Усі можливі канали збуту (глибока)	Власна

Висновки до четвертого розділу

В результаті проведеного маркетингового аналізу перспектив реалізації запропонованих науково-технічних рішень та пропозицій, оцінювання можливостей їх ринкового впровадження можна стверджувати, що розроблений алгоритм обробки даних для подальших передбачень системи є рентабільним проектом та має можливість ринкової комерціалізації. Зростання попиту на аналогічні товари додає масовості придбання подібних товарів, але створює жорсткі конкурентні умови виходу на ринок.

Проект має високі перспективи впровадження з огляду на сучасний стан ринку, що потребує більш ефективних рішень. Перешкодами входження на ринок може бути необхідність високих початкових вкладень, потреба у великій кількості кваліфікованих кадрів та коштовної апаратури. Подальша імплементація проекту є доцільною та рентабельною.

ВИСНОВКИ

В даній роботі було вирішено актуальну науково-практичну задачу підвищення точності передбачень кількості використаної та згенерованої електроенергії методами машинного навчання.

1. Для зберігання даних було обрано NoSQL бази даних, так як їх краще використовувати при необхідності зберігати велику кількість інформації з датчиків і подальшої їх обробки; NoSQL корисні коли навантаження на систему може зростати з часом, і система потребує масштабування на декілька машин, а також через їх швидкодію, що є позитивними характеристиками для машинного навчання;

2. Було протестовано три моделі машиного навчання: лінійна, випадковий ліс та k-найближчих сусідів. Серед протестованих моделей машинного навчання найвища точність була у моделі Випадковий ліс (84% для використаної електроенергії, та 95% для згенерованої електроенергії). Також перевагами цієї моделі машинного навчання є висока швидкість навчання, масштабованість, здатність працювати з пропущеними даними, невисока чутливість до викидів, відсутність чутливості до масштабування.

3. Використовуючи описані методи обробки даних стало можливим покращити точність прогнозу (на 25% для використаної електроенергії та на 2% для згенерованої електроенергії);

4. Було встановлено, що прогноз використаної електроенергії найбільше залежить від даних про: час доби (хвилини та години) і температури вуличного повітря. Прогноз згенерованої електроенергії найбільше залежить від часу доби (години), температури та тиску.

5. Більша точність може бути досягнута з використанням додаткових методів попередньої обробки даних, а також за рахунок збільшення проміжку спостережень.

6. Було продемонстровано, що використання прогнозування для завчасної закупівлі електроенергії в нічний час (за зниженими тарифами) допомагає зменшити ціну для споживачів систем MicroGrid, а також зменшити навантаження на загальну систему електропостачання в пікові години.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. D. Alex, C. Timothy, "A regression based approach to short term load forecasting", IEEE transaction on power systems, vol. 5, no. 4, pp. 1535-1550, 1990.

2. Al-Shareef, A.J., E.A. Muhammad, E. Al-Judaibi, "One Hour Ahead Load Forecasting Using Artificial Neural Network for the Western Area of Saudi Arabia", International Journal of Electrical Systems Science and Engineering, vol. 37, pp. 219-224, 2008.

3. Eugene A. Feinberg, Dora Genethliou, "Applied mathematics for power systems Load Forecasting", Gross and Galiana 1987. “Short-Term Load Forecasting” proceedings of the IEEE, vol. 75, no. 12, pp. 1558-1570.

4. Ibrahim Moghram, Saifure Rahman, "Analysis Evaluation of five short term load forecasting techniques", IEEE transaction on power systems, vol. 4, no. 4, pp. 1484-1491, 1989.

5. Швець М.Ю., Заруба Д.С., Хохлов Ю.В Порівняння SQL та NoSQL баз даних. Вчені записки Таврійського національного університету імені В.І. Вернадського. Серія: Технічні науки. 2018. Том 29(68) № 6, частина 2, с. 21-25.

6. R. Agrawal, A. Ailamaki, P. A. Bernstein, E. A. Brewer, M. J. Carey, S. Chaudhuri, A. Doan, D. Florescu, M. J. Franklin, H. Garcia-Molina, J. Gehrke, L. Gruenwald, L. M. Haas, A. Y. Halevy, J. M. Hellerstein, Y. E. Ioannidis, H. F. Korth, D. Kossmann, S. Madden, R. Magoulas, B. C. Ooi, T. OReilly, R. Ramakrishnan, S. Sarawagi, M. Stonebraker, A. S. Szalay, and G. Weikum, “The claremont report on database research,” 2008, 11 pages.

7. SQL – Transactions. https://www.tutorialspoint.com/sql/sql-transactions.htm [Електронний ресурс]

8. Richards J. Advantages of NoSQL Databases. https://bigdata-madesimple.com/advantages-of-nosql-databases-what-you-need-to-know/ [Електронний ресурс]

9. The Limitations of NoSQL Database Storage. http://www.channelfutures.com/cloud-services/limitations-nosql-database-storage-why-nosqls-not-perfect [Електронний ресурс]

10. K. Chodorow and M. Dirolf, MongoDB: The Definitive Guide. O'Reilly Media, 2010, 409 pages.

11. Wodehouse C. SQL vs. NoSQL Databases: What's the Difference? / Carey Wodehouse.www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/ [Електронний ресурс]

12. Nicholas Png, Training Machine Learning Models with MongoDB https://www.mongodb.com/blog/post/training-machine-learning-models-with-mongodb [Електронний ресурс]

13. Обучение нейросети с учителем, без учителя, с подкреплением — в чем отличие? URL: https://neurohive.io/ru/osnovy-data-science/obuchenie-s-uchitelem-bez-uchitelja-s-podkrepleniem/(дата звернення: 15.10.2019).

14. Николенко С., Кадурин А., Архангельская Е., Глубокое обучение – Питер, 2017. – 480с.

15. NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/ [Електронний ресурс]

16. Бахрушин В. Е. Методы оценивания характеристик нелинейных статистических связей // Системные технологии. — 2011. — № 2(73). — С. 9—14.

17. Glantz, Stanton A.; Slinker, B. K., Primer of Applied Regression and Analysis of Variance. McGraw-Hill, 1990

18. Hyndman, R. and Koehler A. (2005). "Another look at measures of forecast accuracy"

19. V. Mynsbrugge (2010). "Bidding Strategies Using Price Based Unit Commitment in a Deregulated Power Market", K.U.Leuven

20. Tofallis (2015). "A Better Measure of Relative Prediction Accuracy for Model Selection and Model Estimation", Journal of the Operational Research Society, 66(8):1352-1362.

21. Hyndman, R.., A. Koehler (2006). "Another look at measures of forecast accuracy." International Journal of Forecasting, 22(4):679-688 doi:10.1016/j.ijforecast.2006.03.001.

22. K. Sungil, K. Heeyoung (2016). "A new metric of absolute percentage error for intermittent demand forecasts." International Journal of Forecasting, 32(3):669-679 doi:10.1016/j.ijforecast.2015.12.003.

23. Бринк Х., Ричардс Д., Феверолф М., Машинное обучение – Питер, 2017. – 336с.

24. Lehmann E., Casella G. (1998). Theory of Point Estimation (2nd ed.). New York: Springer. ISBN 978-0-387-98502-2. MR 1639875

25. Jonsson, T.; Pinson, P.; Nielsen, H.A.; Madsen, H.; Nielsen, T.S. (2013). "Forecasting Electricity Spot Prices Accounting for Wind Power Predictions". IEEE Transactions on Sustainable Energy. 4 (1): 210–218. doi:10.1109/TSTE.2012.2212731

26. Переобучение www.machinelearning.ru/wiki/index.php?title=Переобучение [Електронний ресурс]

27. Batlle, Carlos; Barquin, J. (2005). "A strategic production costing model for electricity market price analysis". IEEE Transactions on Power Systems. 20 (1): 67–74. doi:10.1109/TPWRS.2004.831266. ISSN 0885-8950.

28. С.А. Айвазян, В.М. Бухштабер, И.С. Енюков, Л.Д. Мешалкин; под ред. С.А. Айвазяна. - М.: Финансы и статистика, 1989. - 607с., ил.

29. Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8.

30. Hyafil L., Rivest RL (1976). «Constructing Optimal Binary Decision Trees is NP-complete». Information Processing Letters 5 (1): 15-17. DOI:10.1016/0020-0190(76)90095-8.

31. Murthy S. (1998). Automatic construction of decision trees from data: A multidisciplinary survey. Data Mining and Knowledge Discovery

32. Principles of Data Mining. 2007. DOI:10.1007/978-1-84628-766-4. ISBN 978-1-84628-765-7

33. H. Tamás, Y. Akihiro, eds. (2003). Inductive Logic Programming. Lecture Notes in Computer Science. 2835. DOI:10.1007/b13700. ISBN 978-3-540-20144-1.

34. E Deng,H, Runger, G., Tuv, E. (2011). «Bias of importance measures for multi-valued attributes and solutions». Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN). pp. 293—300.

35. H Breiman, L. (1996). Bagging Predictors. «Machine Learning, 24»: pp. 123—140.

36. Г Friedman, J. H. (1999). Stochastic gradient boosting. Stanford University.

37. Ш Hastie, T., Tibshirani, R., Friedman, J. H. (2001). The elements of statistical learning : Data mining, inference, and prediction. New York: Springer Verlag.

38. Антонов А.В. Системный анализ. — М.: Высшая школа, 2004. — С. 454

39. М. Дмитриевский, Random Decision Forest в обучении с подкреплением, 2018 https://www.mql5.com/ru/articles/3856 [Електронний ресурс]

40. Сава С. Основы машинного обучения https://sashasava.gitbooks.io/guide-to-deep-learning/osnovi_mashinnogo_obucheniya%D1%91.html [Електронний ресурс]

41. Петергеря Ю.С. Електро-вартісні моделі генераторів і навантажень для керування електроспоживанням локального об’єкта / Ю.С. Петергеря / Електроніка та зв’язок. – 2006. – № 2. – С. 33–39

42. Ledin, S.V. (2012), “Koncepcija “elektrojenergija — tovar” kak katalizator razvitija Smart Grid”, Avtomatizacija v promyshlennosti, Vol. 4, p. 4.

43. Коэффициент корреляции Пирсона http://www.machinelearning.ru/wiki/index.php?title=Коэффициент_корреляции_Пирсона [Електронний ресурс]

44. В.Я. Жуйков, Ю.С. Ямненко, І.Ю. Бойко, Л.Є. Клепач Статична та динамічна тарифікація електроенергії автономних Micro Grid. Вісник ЖДТУ. Серія: Технічні науки. №3,2016, с. 66-75

45. Петергеря Ю.С. Принципы эффективного интеллектуального управления потреблением энергии в локальных объектах / Ю.С. Петергеря, В.Я. Жуйков // Технічна електродинаміка. Тематичний випуск «Проблеми сучасної електротехніки». – 2002. – Ч. 1. – С. 90–96.

46. Жуйков В.Я. Керування споживанням електроенергії в локальному об’єкті з використанням вартісних електротехнічних моделей / В.Я. Жуйков, Ю.С. Петергеря, Р.В. Садрицький // Технічна електродинаміка. – Тематичний випуск «Силова електроніка та енергоефективність». – 2003. – Ч. 4. – С. 49–53.

47. Ушаков Д.Р. Оцінка вартісних характеристик джерел електричної енергії для систем гарантованого електроживлення / Д.Р. Ушаков // VI Міжнародна науково-технічна конференція молодих вчених «Електроніка-2013» / Збірник статей. – К., 2013. – С. 319–322.

48. Бажинов, А.Н. Деревья принятия решений в задаче отбора значимых факторов для прогнозирования объемов электропотребления в металлургическом производстве Текст. / А.Н. Бажинов, Е.В. Ершов. Вестник Череповецкого гос. ун-та.-2011- № 4 Т.3- С. 9-11.

49. Бажинов, А.Н. Прогноз потребления электроэнергии как средство повышения эффективности металлургического производства Текст. / А.Н. Бажинов, Е.В. Ершов. Металлург 2011, № 11 - С. 34—37.

50. J.C. Mourao, A.E. Ruano, "Application of Computation Intelligence Techniques for Energy Load and Price Forecast in some States of USA", Intelligent Signal Processing 2007. WISP 2007. IEEE International Symposium on, pp. 1-6, Oct. 2007.

ABSTRACT

Actuality of theme. Today, the development and dissemination of MicroGrid technology is rapidly evolving, the main task of which is to ensure energy efficiency by using alternative sources of electricity as the main elements of the power grid. It is therefore necessary to ensure mutually beneficial conditions for the using and generation of renewable electricity.

Balance of production and consumption of electricity is the basis of technological stability of the power system, its violation affects the quality of electricity (degradation of frequency and voltage in the network), which reduces the efficiency of the equipment. Short-term load forecasting is mainly aimed at forecasting system load with a time lag of one hour to seven days, which is necessary for adequate planning and operation of power systems. Load forecasting has also become an important component of energy brokerage systems. This makes it possible to manage the cost of purchasing electricity by regulating the loading of equipment, translating, for example, the main volumes of electricity generation into hours and zones of the wholesale energy market with the lowest price.

Association of work with scientific programs, plans and topics. The dissertation was prepared in accordance with the research plan of the Department of Industrial Electronics of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

The purpose of the study is to improve the results of predictions of machine learning models through preliminary data processing, to determine the weights, criteria and factors that affect the use and generation of electricity.

To achieve this goal, the following tasks were solved:

- analytical review of relational and non-relational databases;

- classification of machine learning methods;

- selection of the most important criteria and factors that influence the supply and demand in electricity;

- forecasting using and generating of electricity;

- comparison of accuracy for different machine learning models.

The object of the study is machine learning methods, where, based on the data obtained, the system analyzes and predicts future behavior.

The subject of the study is the dependence and demand for electricity at different weights, such as: weather, time of year, time of day.

The scientific novelty of the obtained results is to determine the degree of dependence between the amount using and generating of electricity by temperature, humidity, time of year, month, day, hour and the contribution of each of them. The accuracy of predictions by machine learning methods with statistical methods of information processing is improved.

The practical value of the results obtained is to save the money of MicroGrid users and to make recommendations for:

1. Cleaning and formatting of data
2. Preliminary data analysis
3. Choosing the most useful features and creating new more representative ones
4. Model checks on the test sample
5. Interpretation of results

Personal contribution of the applicant. The dissertation is a synthesis of the results of theoretical and experimental researches carried out by the author independently. In the work published with the co-authors, the dissertation includes analytical review of relational and non-relational databases, analysis of advantages and disadvantages of SQL and NoSQL databases, testing of performance of management systems SQL and NoSQL databases, analysis of data for predictions, prediction of amount of used and generated electricity by methods of machine learning, the separation of weight factors in training machine learning model Random forest.

Examination of the dissertation results. The main theoretical principles and results of the master's study were presented in the report at the scientific and

technical conference III Ukrainian scientific and technical conference of students, graduate students and scientists "Modern technologies of cinema and audiovisual systems", Kyiv, December 9-10, 2019.

Publications. The main content of the dissertation is reflected in 2 scientific works, 1 of which is published and 1 is in the edition in scientific professional editions according to the list of VAK of Ukraine:

- Shvets M.Y., Zaruba D.S., Khokhlov Y.V. Comparison of SQL and NoSQL databases. Notes of the VI Vernadsky Taurida National University. Series: Technical Sciences. 2018. Volume 29 (68) No. 6, Part 2, p. 21-25.
- Shvets M.Y., Khokhlov Y.V., Zaruba D.S. Machine learning for a power consumption and generation prediction. Microsystems, Electronics and Acoustics (ed)

Structure and scope of the thesis. The dissertation consists of an introduction, four sections, conclusions, a list of used sources of 50 titles and 1 appendix. The total volume of the dissertation is 94 pages, including 74 pages of the main text, 28 figures and 17 tables.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split

# Fill in the line below: Read the file into a variable my_data
my_filepath = 'HomeC.csv'
# my_data = pd.read_csv(my_filepath, parse_dates=True)
my_data = pd.read_csv(my_filepath, sep = ',', parse_dates = ['time'])
my_data['cloudCover'] = pd.to_numeric(my_data['cloudCover'], errors='coerce')
home_dat = my_data.select_dtypes(exclude=['object'])
# you can convert a time from unix epoch timestamp to normal stamp using
# import time
# print( ' start ' , time.strftime('%Y-%m-%d %H:%S', time.localtime(1451624400)))
print(home_dat)

time_index = pd.date_range('2016-01-01 05:00', periods=503910,  freq='min')
# time_index = pd.DatetimeIndex(time_index)
# time = pd.to_datetime(time_index)
# home_dat = home_dat.set_index(time_index)
# home_dat['use [kW]'] = data['Appliances'] + data['lights']
home_dat['day'] = time_index.day #useless
home_dat['month'] = time_index.month
# home_dat['year'] = time_index.year
home_dat['weekday'] = time_index.weekday + 1
home_dat['hour'] = time_index.hour
home_dat['minute'] = time_index.minute
home_dat['total_time'] = home_dat['minute'] + home_dat['hour'] * 60
home_dat['h+w'] = home_dat['weekday'] * 24 + home_dat['hour']

plt.figure()
plt.plot(time_index, home_dat['use [kW]'], label = 'використана е/е')
plt.plot(time_index, home_dat['gen [kW]'], label = 'згенерована е/е', c = 'g')
plt.legend(loc='upper right')
plt.xlabel('дата'); plt.ylabel('W, кВт');
plt.title('Графік використанної та згенерованної електроенергії за весь період');
plt.savefig("input.png", bbox_inches='tight')

plt.figure()
plt.plot(time_index, home_dat['gen [kW]'], c = 'g')
plt.xlabel('дата'); plt.ylabel('W, кВт');
plt.title('Графік згенерованної електроенергії');
plt.savefig("input_gen.png", bbox_inches='tight')

plt.figure()
plt.hist(home_dat['use [kW]'].dropna(), bins = 50, edgecolor = 'k');
plt.xlabel('W вик, кВт'); plt.ylabel('Кіл-ть значень');
plt.title('Energy');
plt.savefig("useW.png", bbox_inches='tight')

plt.figure()
plt.hist(home_dat['gen [kW]'].dropna(), bins = 50, edgecolor = 'k');
plt.xlabel('W ген, кВт'); plt.ylabel('Кіл-ть значень');
plt.title('Energy');
plt.savefig("genW.png", bbox_inches='tight')
```

```
# Function to calculate missing values by column
def missing_zero_values_table(df):
    zero_val = (df == 0.00).astype(int).sum(axis=0)
    mis_val = df.isnull().sum()
    mis_val_percent = 100 * df.isnull().sum() / len(df)
    mz_table = pd.concat([zero_val, mis_val, mis_val_percent], axis=1)
    mz_table = mz_table.rename(
        columns={0: 'Нульові значення', 1: 'Пропущені значення', 2: '% від загальної кіл-ті'})
    mz_table['Назва'] = df.columns
    mz_table['Кіл-ть нульових і пропущенних значень'] = mz_table['Нульові значення'] + mz_table['Пропущені
значення']
    mz_table['% Нульових і пропущенних значень'] = 100 * mz_table['Кіл-ть нульових і пропущенних значень']
/ len(df)
    mz_table['Тип даних'] = df.dtypes
    mz_table = mz_table[
        mz_table.iloc[:, 1] != 0].sort_values(
        '% Нульових і пропущенних значень', ascending=False).round(1)
    print("Your selected dataframe has " + str(df.shape[1]) + " columns and " + str(df.shape[0]) + " Rows.\n"
        "There are " + str(mz_table.shape[0]) + " columns that have missing values.")
    mz_table.to_excel('missing_and_zero_values.xlsx', freeze_panes=(1,0), index = False)
    return mz_table

print(missing_zero_values_table(home_dat))

# Removing Outliers
# Calculate first and third quartile for use Energy
first_quartile = home_dat['use [kW]'].describe()['25%']
third_quartile = home_dat['use [kW]'].describe()['75%']

# Interquartile range
iqr = third_quartile - first_quartile

home_data = home_dat[(home_dat['use [kW]'] > (first_quartile - 3 * iqr)) & (home_dat['use [kW]'] < (third_quartile
+ 3 * iqr))]

plt.figure()
plt.hist(home_data['use [kW]'].dropna(), bins = 50, edgecolor = 'k');
plt.xlabel('W вик, кВт'); plt.ylabel('Кіл-ть значень');
plt.title('Energy');
plt.savefig("useW_u.png", bbox_inches='tight')

plt.figure()
plt.hist(home_data['gen [kW]'].dropna(), bins = 50, edgecolor = 'k');
plt.xlabel('W ген, кВт'); plt.ylabel('Кіл-ть значень');
plt.title('Energy');
plt.savefig("genW_u.png", bbox_inches='tight')

# print(np.any(np.isnan(home_dat)))
# print(np.all(np.isfinite(home_dat)))

# home_dat['date'] = time_index
# home_dat = home_dat[['use [kW]', 'gen [kW]', 'month', 'day', 'hour', 'minute', 'weekday', 'total_time', 'h+w',
'windSpeed', 'precipProbability', 'visibility', 'apparentTemperature', 'temperature', 'pressure', 'windBearing',
'humidity', 'dewPoint', 'precipIntensity']]
# home_dat = home_data[['use [kW]', 'gen [kW]', 'month', 'day', 'hour', 'minute', 'weekday', 'windSpeed',
'precipProbability', 'visibility', 'temperature', 'pressure', 'windBearing', 'humidity', 'precipIntensity']]
home_dat = home_data[['use [kW]', 'gen [kW]', 'month', 'day', 'hour', 'minute', 'weekday',
'windSpeed','precipIntensity', 'visibility', 'temperature', 'pressure', 'windBearing', 'humidity']]
plt.figure()
correlations_data = home_dat.corr()
a = sns.heatmap(correlations_data).plot
plt.savefig("Кореляционная_матрица.png", bbox_inches='tight')
```

```
print(correlations_data.head(50).sort_values(by = 'use [kW]'))

#Отделим от нашей выборки прогнозные значения:
trg = home_dat[['use [kW]', 'gen [kW]']]
trn = home_dat.drop(['use [kW]', 'gen [kW]'], axis=1)

#поместим все наши модели в один список для удобства дальнейшего анализа:
models = [
                    LinearRegression(), # метод наименьших квадратов
      RandomForestRegressor(n_estimators=100, max_features ='sqrt'), # случайный лес
      KNeighborsRegressor(n_neighbors=6), # метод ближайших соседей
      #don`t work SVR(kernel='linear'), # метод опорных векторов с линейным ядром
      # LogisticRegression() # логистическая регрессия
      ]

# разобьем наши исходные данные на 2 подвыборки: тестовую и обучающую
Xtrn, Xtest, Ytrn, Ytest = train_test_split(trn, trg, test_size=0.3)

#создаем временные структуры
TestModels = pd.DataFrame()
tmp = {}
# для каждой модели из списка
for model in models:
   #получаем имя модели
   m = str(model)
   tmp['Model'] = m[:m.index('(')]
   # print('Start fit for model - ')
   #для каждого столбцам результирующего набора
   for i in range(Ytrn.shape[1]):
      #обучаем модель
      model.fit(Xtrn, Ytrn.iloc[:,i])
      #вычисляем коэффициент детерминации
      tmp['R2_Y%s'%str(i+1)] = r2_score(Ytest.iloc[:,i], model.predict(Xtest))
          # print(i)
   #записываем данные и итоговый DataFrame
   TestModels = TestModels.append([tmp])
#делаем индекс по названию модели
TestModels.set_index('Model', inplace=True)
print("рез")
print(TestModels)
plt.figure()
fig, axes = plt.subplots(ncols=2, figsize=(10,4))
TestModels.R2_Y1.plot(ax=axes[0], kind='bar', title='R2_використана_е/е')
TestModels.R2_Y2.plot(ax=axes[1], kind='bar', color='green', title='R2_згенерована_е/е')
# plt.title('Результати точності передбачень');
plt.savefig("u_Погрешность.png", bbox_inches='tight')

model = models[1]
print('Start training...')
model.fit(Xtrn, Ytrn)
print(r2_score(Ytest, model.predict(Xtest)))
print(model.feature_importances_)
# use_f = pd.DataFrame(data = model.feature_importances_, columns = 'Use')
# model.fit(Xtrn, Ytrn.iloc[:,1])
# print(model.feature_importances_)
# gen_f = pd.DataFrame(data = model.feature_importances_, columns = 'Gen')
# feature_table = pd.concat([use_f, gen_f], axis=1)
# feature_table = feature_table.rename(
#     columns={0: 'Use', 1: 'Gen'})
# feature_table['Назва'] = trn.columns
# feature_table.to_excel('features.xlsx', freeze_panes=(1,0), index = False)

res_w = pd.DataFrame(index = Ytest.index, data = model.predict(Xtest), columns = ['use_p [kW]', 'gen_p [kW]'])
```

```
plt.figure()
plt.plot(Ytest['use [kW]'].sort_index().head(100), label = 'реальні данні')
plt.plot(res_w['use_p [kW]'].sort_index().head(100), label = 'результати передбачення')
plt.ylabel('W, кВт'); plt.xlabel('порядковий індекс');
plt.legend(loc='upper right')
# plt.title('Результати передбачень використанної електроенергії');
plt.savefig("use_Res_u.png", bbox_inches='tight')
plt.figure()
plt.plot(Ytest['gen [kW]'].sort_index().head(100), label = 'реальні данні', color='green')
plt.plot(res_w['gen_p [kW]'].sort_index().head(100), label = 'результати передбачення')
plt.ylabel('W, кВт'); plt.xlabel('порядковий індекс');
plt.legend(loc='center right')
# plt.title('Результати передбачень згенерованної електроенергії');
plt.savefig("gen_Res_u.png", bbox_inches='tight')
```