Вінницький національний технічний університет
Факультет комп'ютерних систем і автоматики
Кафедра системного аналізу та інформаційних технологій

ІНФОРМАЦІЙНА ТЕХНОЛОГЯ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ ЕЛЕКТРООБЛАДНАННЯ НА ОСНОВІ ІНТЕРНЕТУ РЕЧЕЙ

Пояснювальна записка до магістерської кваліфікаційної роботи

Виконав: студент 2 курсу, групи 2ІСТ-19м
спеціальності 126 – «Інформаційні системи та технології»
Науменко Д.С.

Керівник: д.т.н., проф. Мокін О. Б. _____

Рецензент: к.т.н., доц. Бойко О. Р. _____

Вінниця ВНТУ – 2020 року

Вінницький національний технічний університет
Факультет комп'ютерних систем і автоматики
Кафедра системного аналізу та інформаційних технологій

Освітньо-кваліфікаційний рівень магістр
Спеціальність 126 - Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н.,проф. В. Б. Мокін
"__" _____ 2020 р.

ЗАВДАННЯ
на магістерську кваліфікаційну роботу студенту
Науменку Денису Сергійовичу

1. Тема роботи: «Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей»,
керівник роботи: Мокін О. Б., д.т.н., проф,
затверджені наказом закладу вищої освіти від "__"_____2020 року №__
2. Строк подання студентом роботи_____
3. Вихідні дані до роботи:
   - датасет Appliances Energy Prediction (Kaggle), який містить дані сенсорів будівлі з низьким енергоспоживанням, а також дані зовнішнього середовища, отримані через інтернет речей;
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
   - обґрунтування доцільності створення інформаційної технології прогнозування енергоспоживання електрообладнання;
   - розроблення інформаційної технології прогнозування енергоспоживання електрообладнання;
   - вдосконалення моделей прогнозування.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
   - графік енергоспоживання приладів;
   - графік енергоспоживання будніх днів;
   - графік енергоспоживання вихідних днів;
   - графіки функцій вихідних даних;
   - таблиця кореляції даних;
   - діаграма порівняння методів прогнозування;
   - діаграма важливості функцій;
   - результат вдосконалення моделей;

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Руда Л.П., к.е.н., доц. каф. ЕПВМ		

7. Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МКР	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області		
2	Розробка технології прогнозування		
3	Вдосконалення технологій прогнозування		
4	Економічна частина		
5	Оформлення матеріалів до захисту МКР		

Студент _____ Науменко Д.С.

Керівник роботи _____ Мокін О. Б.

Рецензент _____ Бойко О.Р.

РЕФЕРАТ

Магістерська кваліфікаційна робота: 89 стор., 4 табл., 36 рис., 33 джерела.

Об’єкт досліджень – процес споживання електричної енергії електрообладнанням будівлі з низьким енергоспоживанням.

Мета роботи – покращення енергоефективності будівлі з низьким енергоспоживанням за рахунок прогнозування потреб у енергії електрообладнання будівлі та вибору оптимального режиму роботи цього електрообладнання з використанням даних інтернету речей.

Проведено аналіз існуючих технологій прогнозування енергоспоживання. Здійснено проектування модулів для реалізації інформаційної технології. Визначено оптимальні методи для реалізації.

Прогнозні припущення про розвиток об’єкта дослідження – розробка інформаційної технології , яка буде прогнозувати краще за аналоги.

Галузь застосування – приватні особи, або компанії, що спеціалізуються на продажу електроенергії.

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ ЕЛЕКТРООБЛАДНАННЯ НА ОСНОВІ ІНТЕРНЕТУ РЕЧЕЙ, ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, ЕНЕРГОСПОЖИВАННЯ, ІНТЕРНЕТ РЕЧЕЙ.

ABSTRACT

Master's qualification work: 89 pages, 4 tables, 36 figures, 33 sources.

The object of research - is the process of electricity consumption by electrical equipment of a building with low energy consumption.

The purpose of the work is to improve the energy efficiency of a building with low energy consumption by forecasting the energy needs of the building's electrical equipment and choosing the optimal mode of operation of this electrical equipment using the Internet of Things data.

The analysis of existing technologies of energy consumption forecasting is carried out. The design of modules for the implementation of information technology. The optimal methods for implementation are determined.

Predictive assumptions about the development of the object of study - the development of information technology that will predict better than analogues.

Scope - individuals or companies specializing in the sale of electricity.

INFORMATION TECHNOLOGY OF ENERGY CONSUMPTION FORECASTING OF ELECTRICAL EQUIPMENT ON THE BASIS OF THE INTERNET OF THINGS, INFORMATION TECHNOLOGY, ENERGY CONSUMPTION, INTERNET OF THINGS.

ЗМІСТ

ВСТУП

У сучасному світі, що швидко зростає та розвивається, проблема ефективного використання енергії стає все більш актуальною. Оптимізація витрат є одним з ключових викликів сучасного світу. Одним з напрямків такої оптимізації є створення інформаційних технологій прогнозування витрат енергії, які дозволяють краще розуміти потреби у енергії та ефективніше підбирати джерела її генерації. Це і зумовлює актуальність обраного напрямку дослідження, а впровадження та розвиток смарт-технологій та інтернету речей дозволяють перевести інформаційні технології такого типу на новий рівень аналізу та прогнозування.

Об'єктом дослідження є процес споживання електричної енергії електрообладнанням будівлі з низьким енергоспоживанням.

Предметом дослідження є інформаційна технологія прогнозування енергоспоживання електрообладнання будівлі з низьким енергоспоживанням.

Метою дослідження є покращення енергоефективності будівлі з низьким енергоспоживанням за рахунок прогнозування потреб у енергії електрообладнання будівлі та вибору оптимального режиму роботи цього електрообладнання з використанням даних інтернету речей.

Наукова новизна одержаних результатів. Основні результати, які були отримані в процесі вирішення поставлених завдань та становлять наукову новизну дослідження, полягають у вдосконалені інформаційної технології прогнозування енергоспоживання електрообладнання будівлі, що дозволяє знизити енергоспоживання за рахунок технології інтернету речей.

Практичне значення одержаних результатів полягає у можливості користувача даної інформаційної технології:

– прогнозувати показники енергоспоживання та ефективно розподіляти роботу електрообладнання будівлі з низьким енергоспоживанням;

– підбирати кращі тарифні плани серед тих, що пропонують електроенергетичні компанії;

– оцінити рівень енергоефективності будівлі з урахуванням показників зовнішнього середовища, отриманого за допомогою інтернету речей.

Достовірність теоретичних положень кваліфікаційної роботи обґрунтовується всебічним ознайомленням з предметною областю для коректного розуміння значення та розробки технології, аналізом та поясненням доцільності застосування тих чи інших моделей та методів, а також програмно-технічних засобів.

Апробація результатів роботи. Результати роботи були апробовані на всеукраїнській науково-практичній інтернет-конференції студентів аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи».

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано: 1 тези на всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» [1].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ ЕЛЕКТРООБЛАДНАННЯ

1.1 Аналіз предметної області

У цій дипломній роботі метою дослідження є вдосконалення існуючої системи прогнозування витрат електроенергії, на основі інформації зібраної різноманітними датчиками.

Зібрана інформація уже була присутня в датасеті проекту який я вдосконалюю [2].

Данна технологія поліпшить керування фінансовою складовою користувача. Цю технологію можна використовувати в багатьох сферах діяльності.

Контроль енергоспоживання є невід’ємною складовою в сучасному світі, так як, споживачів електроенергії стає дедалі більше, то і попит на використання даної технології буде рости з кожним днем.

Задачею даної технології є прогноз споживання електроенергії, в суворий період пандемії, коли більшість людей повинно залишатись вдома, затрати електроенергії збільшились, і керування ними на основі прогнозів, оптимізує фінансові витрати.

Також ця технологія прогнозування допоможе керувати витратами підприємствам, які на пряму пов’язані з електроспоживанням, для оптимізації робочого процесу, та на базі прогнозів збільшувати, або зменшувати, відповідно, потужності підприємства тим самим налагоджувати робочій процес максимально сприятливо для підприємства.

За допомогою прогнозування, можна зменшити витрати сировини для виготовлення електроенергії, цим самим зберегти кошти користувачам, цим самим оптимізувати заробіток.

Сучасний прогрес неможливий без автоматизації різних процесів, особливо в сфері прогнозування. Сектор електроенергетики є ключовим компонентом всієї енергетичної системи в цілому, оскільки електроенергія становить вагому частку у витратах практично всіх галузей економіки. У зв'язку з цим, недолік електроенергії в країні є обмежувачем економічного зростання [3].

Прогнозні оцінки динаміки електроспоживання в швидко розвиваються, що входять до Європейського союзу (куди входить і Україна), набувають особливої актуальності, так як є необхідним елементом планів по довгостроковому розвитку енергетичних систем національних економік.

Реалізація подібних можливостей може бути здійснена за рахунок розробки модельних комплексів, що включають в себе окремі моделі, які розглядають процес електроспоживання з використанням різних підходів. Актуальність розробки подібних комплексів моделей неодноразово підкреслювалася провідними вітчизняними вченими [4-6].

В даний час Україна є одним з лідерів по споживанню енергоресурсів, і цей факт висуває на перший план необхідність розробки стратегії національної енергетичної безпеки [3, 4].

Відповідно до низки міжнародних угод, підписаних Україною, Україна зобов'язана зменшити енергоємність власної економіки та негативний вплив на клімат, стимулюючи тим самим споживачів енергії, включаючи промислові підприємства, до підвищення енергоефективності. Водночас, з одного боку, укладена угода («Заключна угода») між Україною та Європейським Союзом, Європейським співтовариством з атомної енергії та його державами-членами містить вимоги щодо прийняття багатьох механізмів стимулювання енергоефективності на законодавчому рівні. І захист клімату [7].

Через сезонні фактори та низьку енергоефективність ціни на електроенергію зросли.

Голова парламентського Комітету з питань енергетики, житлово-комунального господарства та комунальних послуг України, заявив.

"Електроенергія є товаром, і її ціна залежить від попиту та пропозиції. Навесні та влітку, коли з'явився коронавірус, попит різко впав. Восени та взимку ціни дійсно зростають.

Другий фактор - ефективне використання наявних ресурсів. Ми використовуємо державний "Енергоатом" неефективно, тоді як інші покоління (приватні) хочуть заробити більше. Вони мають певний обсяг ринкової сили і безпідставно піднімають ціни [8].

Штучний інтелект та машинне навчання можуть прискорити ділову роботу, зробити її більш ефективною та потужною, та вивести на новий рівень. Використовуючи ці інструменти, менеджери можуть вирішувати багатовимірні проблеми, що вимагають нетрадиційних методів. За допомогою роботизованої автоматизації процесів можна оптимізувати витрати. Ці інструменти автоматизують завдання, завдяки використанню комп’ютерного зору, обробки природної мови та розпізнавання мови, що робить можливими нові бізнес-моделі [9].

Машинне навчання базується на ідеї, що системи аналізу можуть навчитися розпізнавати закономірності та приймати рішення з мінімальним втручанням людини. У машинному навчанні є чотири ключові завдання:

- Регресія - числове значення символу передбачення, наприклад прогнозування майбутніх продажів на основі відомих даних про продажі в минулому;
- Класифікація-прогнозування, до якої відомої категорії належить об’єкт, наприклад, передбачити, чи буде позичальник повертати позику на основі даних про те, як позичальник погашав позику в минулому;
- Кластеризація-поділ великої кількості об’єктів-класів, в яких об’єкти схожі між собою, наприклад сегментація ринку, поділяє всіх

споживачів на класи, так що споживачі всередині класу схожі між собою, а в різних класах-різні;

- Зменшення розмірності - зменшення великої кількості об’єктів на менші об’єкти (зазвичай 2-3) для полегшення подальшої візуалізації (наприклад, стиснення даних); Аномальний пошук-пошук рідкісних та аномальних об'єктів, які суттєво відрізняються від оптових товарів, наприклад, пошук шахрайських операцій [10].

Машинне навчання поділяється на кілька вузьких областей: нейронні мережі, обробка природних мов (NLP) та глибоке навчання.

Машинне навчання швидко впроваджується у всіх галузях, і існує величезний попит на кваліфікованих фахівців. Очікується, що до 2022 року ринок машинного навчання зросте до 8,81 мільярда доларів США. ML використовується для аналізу даних, аналізу даних та розпізнавання образів. Для кінцевих користувачів машинне навчання має покращувати результати веб-пошуку, керувати рекламою в режимі реального часу, виявляти вторгнення в мережу та робити багато інших корисних справ [11].

Дана розробка базується на вже існуючому ноутбуці, тому це полегшує розробку технології прогнозування енергоспоживання [12].

Розробником ноутбука був Mayur Sand – людина яка займається аналітикою та штучним інтелектом [13].

На його профілі в середовищі Kaggle є чотири відкритих ноутбука з назвами: Google Playstore Analysis, Hourly Energy Prediction та Graduate Admissions – Regression [14].

1.2 Огляд існуючих програмних продуктів

На платформі Каггл існує багато різноманітних ноутбуків по передбаченнях, також є і по передбаченню затрат електроенергії, з різноманітними можливостями вирішення проблеми, ця платформа допомагає

розробникам переглянути уже існуючі рішення та на основі них зробити певні висновки, і покращити свою розробку.

Було проведено моніторинг ноутбуків аналогів:

– Appliances_energy_prediction_LSTM_ARIMA_FBPROPHET;
– Predicting Energy Usage With LSTM RNN;
– Energy prediction using KNN regression.

Ноутбук Appliances_energy_prediction_LSTM_ARIMA_FBPROPHET – використовує модель авторегресії ARIMA – це модель, яка є розширенням моделі ARMA для нестандартних числових рядів, ціль використання даної моделі створення прийнятного розрізу деякого порядку від вихідного тимчасового ряду.

Також використовується пакет fbprophet створену компанією facebook, який базується на адаптивній моделі нелінійних тенденцій для прогнозування даних часових рядів, на основі врахування сезонності та святкових ефектів.

Цей пакет стійкий до пропущених даних, а також до змін у тренді, та зазвичай працює добре, так заявляє facebook. Але як показує графік зображений на рисунку 1.1. Прогноз не такий точний як хотілось би бачити.

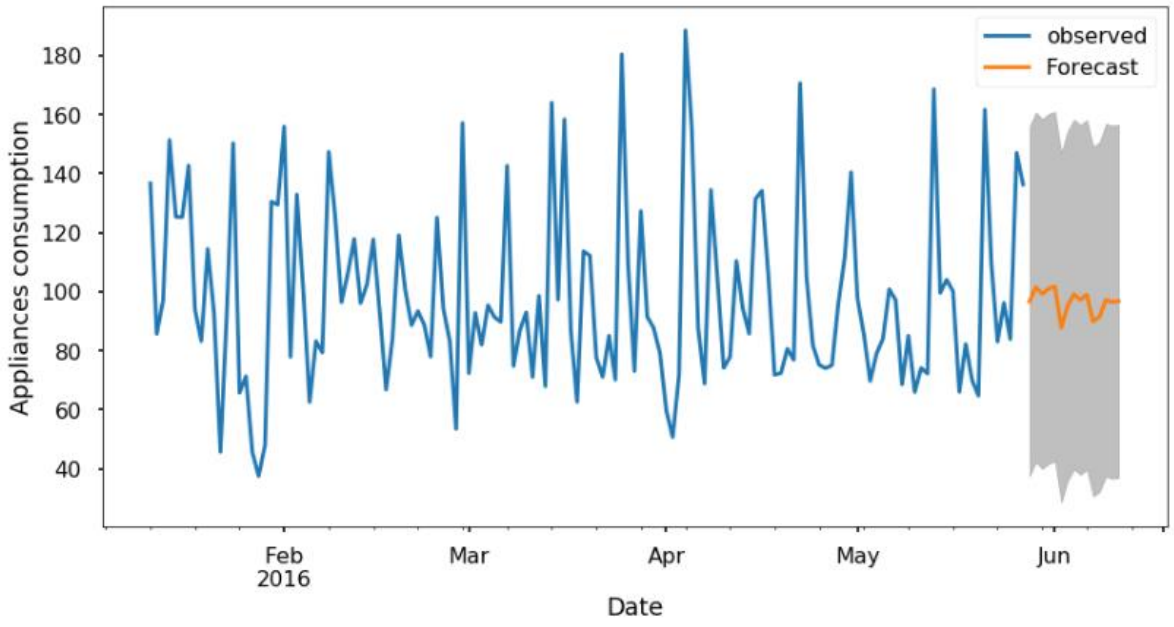


Рисунок 1.1 – Графік прогнозу енергоспоживання

Перевагою ноутбуку є використання моделі ARIMA, яка допомагає усунути не стаціонарності.

Недоліком ноутбуку є занадто мале передбачення, всього 32,8%.

Ноутбук Predicting Energy Usage With LSTM RNN – використовує довгу короткочасну пам’ять, яка використовує архітектуру рекурентних нейронних мереж, яка була запропонована ще в 1997 році, її превілегія в тому, що вона може проводити обчислення будь чого, що може обчислювати звичайний комп’ютер, якщо обчислення має підходящу матрицю числових коефіцієнтів.

Вона чудово підходить тоді, коли існують часові затримки невизначеної тривалості. Але вона більш за все підходить до розпізнавання несегментованого та неперервного рукописного тексту.

Також використовується рекурентні нейронні мережі, які з’єднаннями між вузлами створюють орієнтований у часі граф.

Переваги ноутбуку – це використання RNN, яка може проявити динамічну поведінку в часі, та використання довгої короткочасної пам’яті, яка сприяє обрахункам.

Недоліком ноутбуку є те, що передбачення не використовується.

Ноутбук Energy prediction using KNN regression – цей ноутбук використовує метод k-найближчих сусідів, це непараметричний метод, який усереднює спостереження в тому ж сусідстві, між незалежними змінними.

Хоч цей метод і є досить привабливим, але на великій кількості незалежних вхідних даних він стає не практичним.

Результат передбачення методу зображений на рисунку 1.2.

	Train_RMSE_score	Train_R^2_score	Test_RMSE_score	Test_R^2_score
reg_model_6_1464	57.28	0.70	68.32	0.55
reg_model_7_1185	56.95	0.71	68.32	0.55
reg_model_6_313	57.05	0.70	68.33	0.55
reg_model_7_3906	57.27	0.70	68.35	0.55
reg_model_7_3220	56.22	0.71	68.42	0.55
reg_model_7_4870	56.87	0.71	68.42	0.55
reg_model_7_6059	56.85	0.71	68.45	0.55
reg_model_8_3854	56.86	0.71	68.45	0.55
reg_model_8_4770	56.83	0.71	68.47	0.55
reg_model_6_4934	56.83	0.71	68.50	0.55

Рисунок 1.2 – Результат передбачення методу k-найближчих сусідів

Перевага ноутбуку – це використання цікавого методу найближчих сусідів.

Недоліком ноутбуку є те, що вхідних даних забагато для цього методу.

1.3 Висновки

У розділі було розглянуто проблему з енергоспоживанням у світі, та проаналізовано доцільність вдосконалення інформаційної технології.

Проведено порівняння існуючих систем, результатом порівняння було вирішено вдосконалити існуючу технологію, а також прогнозування енергоспоживання точніше ніж аналоги.

2 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ ЕЛЕКТРООБЛАДНАННЯ

2.1 Огляд та вибір елементів для розробки технології прогнозування енергоспоживання

Для виконання МКР було використано бібліотеки Python.

NumPy – це основний пакет, необхідний для наукових обчислень з Python. Цей пакет містить:

– потужний N-вимірний об'єкт масиву;
– складні (мовні) функції;
– основні функції лінійної алгебри;
– основні перетворення Фур'є;
– складні можливості випадкових чисел;
– інструменти для інтеграції коду Fortran;
– інструменти для інтеграції коду C / C ++.

Окрім очевидного наукового використання, NumPy також може використовуватися як ефективний багатовимірний контейнер загальних даних. Можна визначити довільні типи даних. Це дозволяє NumPy легко та швидко інтегруватися з широким розмаїттям баз даних.

NumPy є наступником двох попередніх наукових бібліотек Python: Numeric та Numarray [15].

NumPy можна розглядати як вільну альтернативу MATLAB. Мова програмування MATLAB зовні нагадує NumPy: обидва вони інтерпретуються, обидва дозволяють виконувати операції над масивами (матрицями), а не над скалярами. Перевага MATLAB в наявності великої кількості пакетів («тулбоксів»), наприклад, Simulink (англ.). Для NumPy теж існують подібні «пакети», наприклад, бібліотека SciPy надає більше MATLAB-подібної

функціональності, бібліотека Matplotlib дозволяє створювати графіки в стилі MATLAB. І MATLAB, і NumPy для вирішення основних задач лінійної алгебри використовують код, заснований на коді бібліотеки LAPACK [16].

Оскільки Python — інтерпретована мова, математичні алгоритми, часто працюють в ньому набагато повільніше ніж у компільованих мовах, таких як C або навіть Java. NumPy намагається вирішити цю проблему для великої кількості обчислювальних алгоритмів забезпечуючи підтримку багатовимірних масивів і безліч функцій і операторів для роботи з ними. Таким чином будь-який алгоритм, який може бути виражений в основному як послідовність операцій над масивами і матрицями, працює так само швидко, як еквівалентний код, написаний на C [16].

Pandas — це високорівнева Python бібліотека для аналізу даних. Чому я її називаю високорівневою, тому що побудована вона поверх більш низькорівневої бібліотеки NumPy (написана на Ci), що є великим плюсом в продуктивності. В екосистемі Python , pandas є найбільш просунутою, що швидко розвивалася, бібліотекою для обробки і аналізу даних.

Щоб ефективно працювати з pandas, необхідно освоїти найголовніші структури даних бібліотеки: DataFrame і Series. Без розуміння, що вони з себе представляють, неможливо в подальшому проводити якісний аналіз.

Структура / об'єкт Series вдає із себе об'єкт, схожий на одновимірний масив (пітонівський список, наприклад), але відмітною його рисою є наявність асоційованих міток, т.зв. індексів, уздовж кожного елемента зі списку. Така особливість перетворює його в асоціативний масив або словник в Python.

Об'єкт DataFrame найкраще уявляти собі у вигляді звичайної таблиці і це правильно, адже DataFrame є табличній структурою даних. У будь-якій таблиці завжди присутні рядки і стовпці. Стовпцями в об'єкті DataFrame виступають об'єкти Series, рядки яких є їхніми безпосередніми елементами [17].

Seaborn - це бібліотека для створення статистичної графіки на Python. Він побудований на вершині matplotlib і тісно інтегрується із структурами даних pandas.

Seaborn допомагає вам досліджувати та розуміти ваші дані. Його функції побудови графіків працюють на кадрах даних та масивах, що містять цілі набори даних, і внутрішньо виконують необхідне семантичне відображення та статистичне агрегування для отримання інформативних графіків.

Декларативний API, орієнтований на набір даних, дозволяє зосередитись на тому, що означають різні елементи ваших графіків, а не на деталях того, як їх намалювати.

За лаштунками Сіборн використовує matplotlib, щоб намалювати свої сюжети. Для інтерактивної роботи рекомендується використовувати інтерфейс Jupyter / IPython в режимі matplotlib , інакше вам доведеться використовувати, matplotlib.pyplot.show() коли ви хочете побачити сюжет.

Не існує універсально найкращого способу візуалізації даних. На різні запитання найкраще відповідати різними сюжетами. Seaborn полегшує перемикання між різними візуальними поданнями за допомогою послідовного API, орієнтованого на набір даних [19].

Scikit-learn (також відома як sklearn або scikits.learn) — це безкоштовна програмна бібліотека машинного навчання для мови програмування Python, яка надає функціональність для створення та тренування різноманітних алгоритмів класифікації, регресії та кластеризації, таких як лінійна регресія, random forest, градієнтний бустинг, і працює у зв'язці з бібліотеками NumPy та SciPy. Scikit-learn є однією з найбільш популярних бібліотек машинного навчання.

Перша версія бібліотеки була написана Девідом Корнапе влітку 2007 року в рамках програми Google Summer of Code. Пізніше цього ж року Метью Брюхер почав працювати над нею, як частиною своєї дипломної роботи.

2010 року дослідники з INRIA продовжили розробку бібліотеки і 1 лютого 2010 року випустили перший офіційний реліз. Відтоді з'явилося декілька нових релізів бібліотеки, а навколо неї утворилася процвітаюча спільнота з розробників, що продовжують підтримувати та удосконалювати проект [20].

Бібліотека plotlyPython - це інтерактивна бібліотека для побудови графіків із відкритим кодом, яка підтримує понад 40 унікальних типів діаграм, що охоплюють широкий спектр статистичних, фінансових, географічних, наукових та тривимірних випадків використання.

Побудований поверх бібліотеки Plotly JavaScript (plotly.js), plotly дозволяє користувачам Python створювати чудові інтерактивні візуалізації на основі Інтернету, які можна відображати в блокнотах Jupyter, зберігати у самостійних файлах HTML або служити частиною чистих веб-додатків, побудованих на Python за допомогою Dash. Бібліотеку plotlyPython іноді називають "plotly.py", щоб відрізнити її від бібліотеки JavaScript.

Завдяки глибокій інтеграції з утилітою експортування зображень orca , plotly також забезпечує чудову підтримку немережевих контекстів, включаючи редактори робочих столів (наприклад, QtConsole, Spyder, PyCharm) та статичну публікацію документів (наприклад, експорт блокнотів у PDF із високоякісними векторними зображеннями).

Plotly.py підтримує експорт статичних зображень, використовуючи або kaleido пакет (підтримується з plotlyверсії 4.9), або утиліту командного рядка orca (застарілу plotly версію з версії 4.9).

Деякі функції plotly.py покладаються на досить великі файли географічної форми. Одним з таких прикладів є окружна фабрика фігур хороплетів. Ці файли фігур розповсюджуються як окремий plotly-geo пакет. Цей пакет можна встановити за допомогою системи керування пакунками pip [21].

Kaggle — платформа для змагань з аналітики та передбачувального моделювання, в рамках якого статистики та добувачі даних конкурують у створенні найкращих моделей для прогнозування та опису даних, запропонованих компаніями або користувачами. Цей краудсорсинговий підхід ґрунтується на тому, що є безліч стратегій, які можуть бути застосовані до будь-якого завдання з передбачувального моделювання, і наперед не відомо, яка методика або аналітичний підхід буде найбільш ефективним.

8 березня 2017 року компанія Google оголосила, що вони придбали Kaggle, який приєднається до команди Google Cloud і продовжить бути окремою торговою маркою.

Як влаштовані змагання на Kaggle:

– Організатор змагань готує дані та описує проблему. Kaggle пропонує консультації з цього приводу, також допомагає побудувати каркас конкурсу, анонімізувати дані та інтегрувати модель переможця в діяльність організатора;

– Учасники конкурсу експериментують з різними технологіями та конкурують один з одним, щоб створити кращі моделі. Робота доступна загалу через сценарії Kaggle з метою досягнення кращого результату та для обміну ідеями. Результати роботи надсилаються за допомогою скриптів або завантажуються вручну. Для більшості змагань результат роботи оцінюється відразу (точність прогнозу порівнюється з прихованим файлом, який містить правильний розв'язок) та відображається на публічній дошці результатів;

– Після закінчення змагань виплачуються призові гроші в обмін на «всесвітню, безстрокову, безвідмовну ліцензію та безоплатну ліцензію на використання роботи переможця», тобто розроблений алгоритм, програмне забезпечення та відповідну інтелектуальну власність, яка є «невід'ємною, якщо інше не зазначено» [22];

Для серйозних програмістів з університетів і стронг сеньйорів це можливість поборотися за призовий фонд і місце в світовому рейтингу.

Для початківців, це можливість отримати досвід, схожий з реальним, і прокачати себе в різних сферах.

Адже найважливіше питання, яке частенько стоїть перед такого роду фахівцями – де знайти реальні завдання? Як зрозуміти що краще в якій ситуації і де протестувати гіпотези?

Коротка інструкція як почати:

- Вибрати змагання з поміткою "Getting Started";
- Прочитати опис завдання;
- Дослідити метрику;
- Дослідити дані (EDA);
- Зробити бейзлайн рішення (підглядаючи у Kernels);
- Вигадувати і пробувати різні гіпотези для поліпшення результату (ідеї можна пошукати в Kernels та Discussion).

На мій погляд, у Kaggle є одна незаперечна перевага в контексті вивчення чогось нового в порівнянні з курсами. Як правило, в курсах дуже абстрактні завдання і не завжди зрозуміти навіщо це вчити, і як саме в подальшому, використовувати на практиці.

У змаганнях такої проблеми немає. Скопіював алгоритм, хочеш поліпшити - доведеться розібратися як він працює і за що відповідають гіперпараметри. Хочеш зробити нові ознаки(features)? Необхідно читати про доменну сферу і детально дивитися ті дані, які є [23].

2.2 Аналіз і нормалізація вхідних даних

Список атрибутів з нашого датасету:

- Дата: Рік, місяць, день, година, хвилина та секунда;
- Appliances: Споживання енергії у Wh;
- lights: Енерговикористання світильників у будинку в Wh;
- T1: Температура в кухонній зоні, в Цельсіях;
- RH_1: Вологість кухонних приміщень,%;
- T2: Температура в зоні вітальні, в Цельсіях;
- RH_2: Вологість в площі вітальні, у%;
- T3: Температура в площі пральні;
- RH_3: Вологість в площі пральні, в%;
- T4: Температура в офісному приміщенні, в Цельсіях;
- RH_4: Вологість в офісному приміщенні , в%;
- T5: Температура у ванній кімнаті, в Цельсіях;
- RH_5: Вологість у ванній кімнаті, у%;
- T6: Температура поза будівлею (північна сторона), в Цельсіях;
- RH_6: Вологість біля будівлі (північна сторона), у%;
- T7: Температура в прасуванні кімната, у Цельсіях;
- RH_7: Вологість у прасувальній кімнаті, у%;
- T8: Температура в кімнаті для підлітків 2, у Цельсіях;
- RH_8: Вологість у кімнаті для підлітків 2, у%;
- T9: Температура в кімнаті для батьків, в Цельсіях;
- RH_9: Вологість в кімнаті для батьків, у%;
- T_out: Температура зовні (від метеостанції Chievres);
- Press_mm_hg: Тиск Цельсія (від метеостанції в Chievres), мм мм рт.ст;

– RH_out: Вологість біля будівлі у%;
– Windspeed: Швидкість вітру (від метеостанції в Chievres) в м / с;
– Visibility: Видимість (від метеостанції Chievres), в км;
– Tdewpoint: (від метеостанції Chievres), ° C;
– rv1: Випадкова змінна 1;
– rv2: Випадкова змінна 2.

Для початку роботи необхідно проаналізувати дані, які відомі.

Інформація, яку ми отримали, це зібрані показники з датчиків вказаних вище. Підключення файлу з зібраними даними та код виводу таблиці на екран зображений на рисунку 2.1.

```
data = pd.read_csv("../input/KAG_energydata_complete.csv")

data.head()
```

Рисунок 2.1 – Підключення та вивід інформації

Функція “head()” виводить інформацію на екран у вигляді таблички з 5 елементами і зображений на рисунку 2.2.

	T1	T2	T3	T4	T5	T6	T7
count	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000
mean	21.685153	20.343487	22.268005	20.857724	19.589105	7.923834	20.264300
std	1.605537	2.199037	1.999986	2.040012	1.842916	6.083047	2.105079
min	16.790000	16.100000	17.200000	15.100000	15.335000	-6.065000	15.390000
25%	20.745000	18.790000	20.790000	19.533333	18.290000	3.663333	18.700000
50%	21.600000	20.000000	22.100000	20.666667	19.390000	7.300000	20.028571
75%	22.600000	21.533333	23.290000	22.100000	20.633333	11.293333	21.600000
max	26.260000	29.856667	29.200000	26.200000	25.745000	28.290000	25.963333

Рисунок 2.2 – Таблиця з вхідними даними.

Під час розвідувального аналізу даних було виявлено параметр з великою кількістю нульових значень (фрагмент таблиці наведено на рисунку 2.3).

oint	RH_out	Press_mm_hg	Windspeed	Visibility	lights	rv1	rv2
.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000
053	79.744066	755.561311	4.057009	38.345054	3.809202	25.014452	25.014452
370	14.952250	7.398129	2.449080	11.785900	7.940816	14.539819	14.539819
0000	24.500000	729.366667	0.000000	1.000000	0.000000	0.006033	0.006033
333	70.000000	750.983333	2.000000	29.000000	0.000000	12.469764	12.469764
000	83.833333	756.100000	3.666667	40.000000	0.000000	24.936900	24.936900
667	91.666667	760.966667	5.500000	40.000000	0.000000	37.736202	37.736202
0000	100.000000	772.300000	14.000000	66.000000	60.000000	49.993173	49.993173

Рисунок 2.3 – Фрагмент таблиці з нулями.

Тому було вирішено перевірити цей стовпець на кількість значень які рівні нулю. Після перевірки виявилось що з 14801 рядка було знайдено понад 11438 рядків зі значенням 0, тому цей стовпець не зіграє ролі в прогнозі і було вирішено його видалити з таблиці з вхідними даними, таким чином з цього випливає, що потрібно проаналізувати всі вхідні дані і з’ясувати чи всі колонки доцільно використовувати для реалізації технології.

Також аналізуючи дані з таблиці було видно коливання показників різноманітних датчиків, такі як датчики температури поза домом, також датчики в приміщенні, особливо виділяється датчик вологості в ванній кімнаті, також датчики техніки показують, що споживання приладів маленьке і випадків перенавантаження мережі буде мінімум.

За допомогою методу drop було видалено стовбець з освітленням та проіндексовано таблицю зображено на рисунку 2.4.

```
_ = feature_vars.drop(['lights'], axis=1 , inplace= True) ;
```

Рисунок 2.4 – Видалення та пере індексування таблиці

Для візуалізації використовувались бібліотека «plotly» . За допомогою неї було виведено графік споживання зображений на рисунку 2.5.

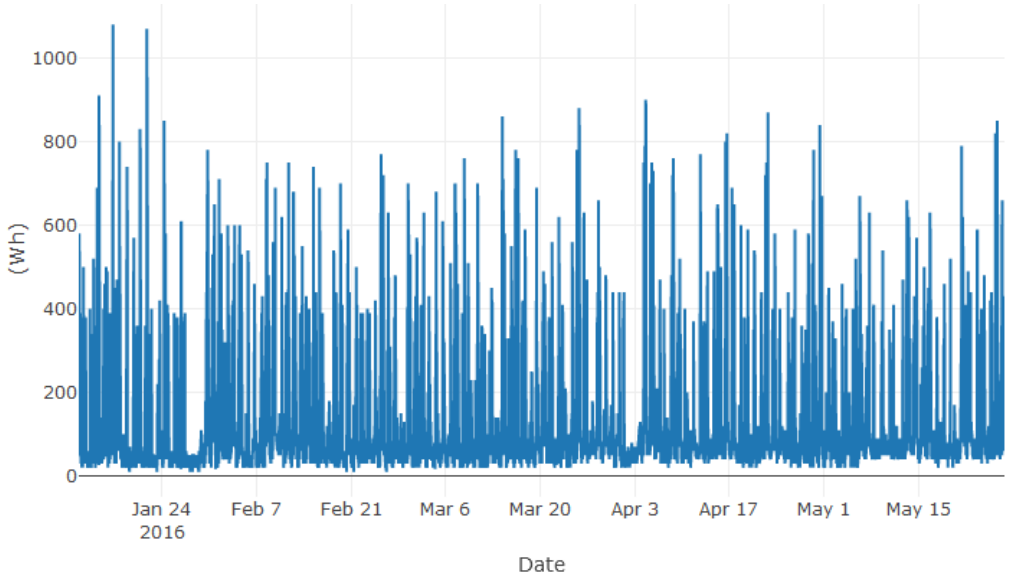


Рисунок 2.5 – Графік споживання приладів

Після цих дій для оцінки часових рядів було додано позначки будніх та вихідних днів, щоб перевірити чи впливатиме це на вірність прогнозування.

При розподілі на будні та вихідні отримали такі результати 14263 та 5472 значень будніх та вихідних відповідно, та виведено новий графік для візуалізації змін, якщо відображати тільки будні дні, та тільки вихідні. Таким чином отримали графіки зображені на рисунках 2.6 та 2.7 відповідно.

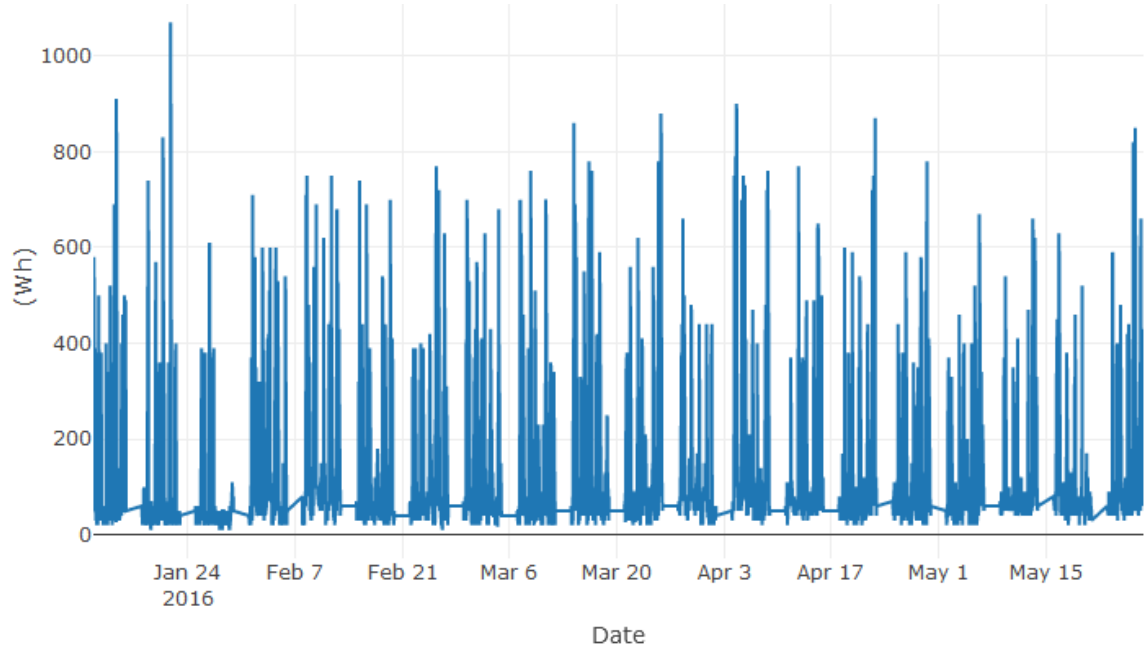


Рисунок 2.6 – Графік споживання в будні дні

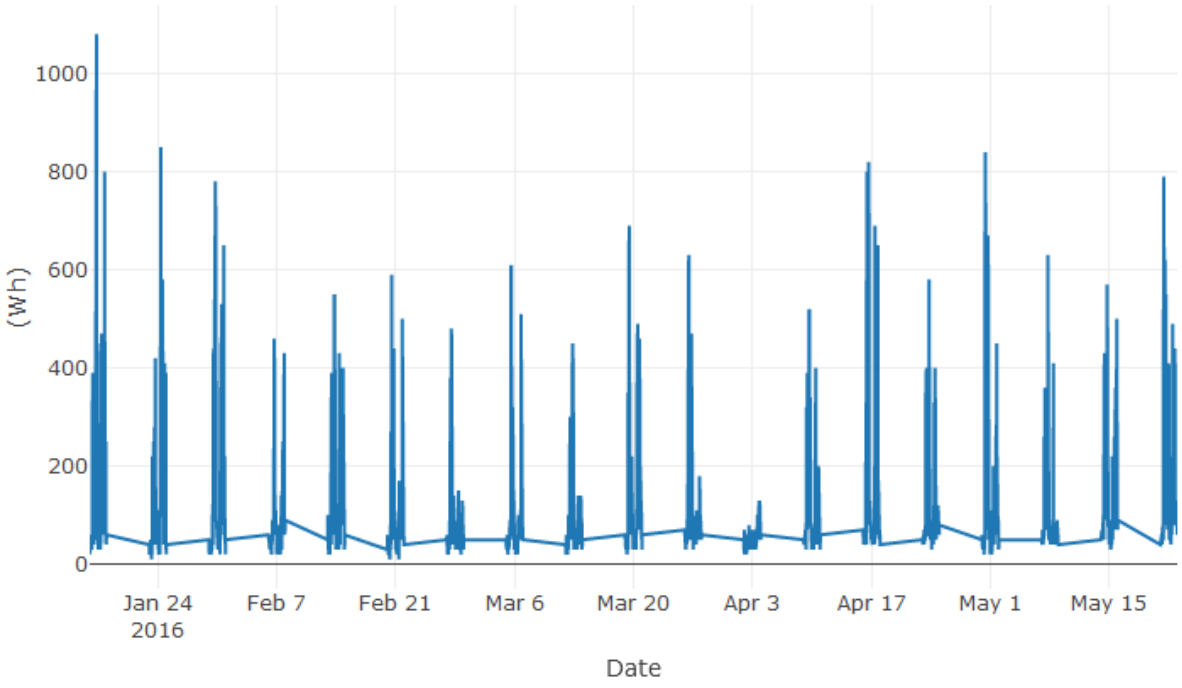


Рисунок 2.7 – Графік споживання в вихідні дні

За для кращого розуміння вхідних даних було прийнято рішення вивести гістограму всіх функцій, щоб зрозуміти розподіл.

Було побудовано графіки 26 функцій за допомогою бібліотеки matplotlib.

Фрагмент коду зображено на рисунку 2.8.

```
feature_vars.hist(bins = 20 , figsize= (12,16)) ;
```

Рисунок 2.8 – Код виводу функцій

Результатом даного коду було побудовано 26 графіків. Зображених на рисунку 2.9.

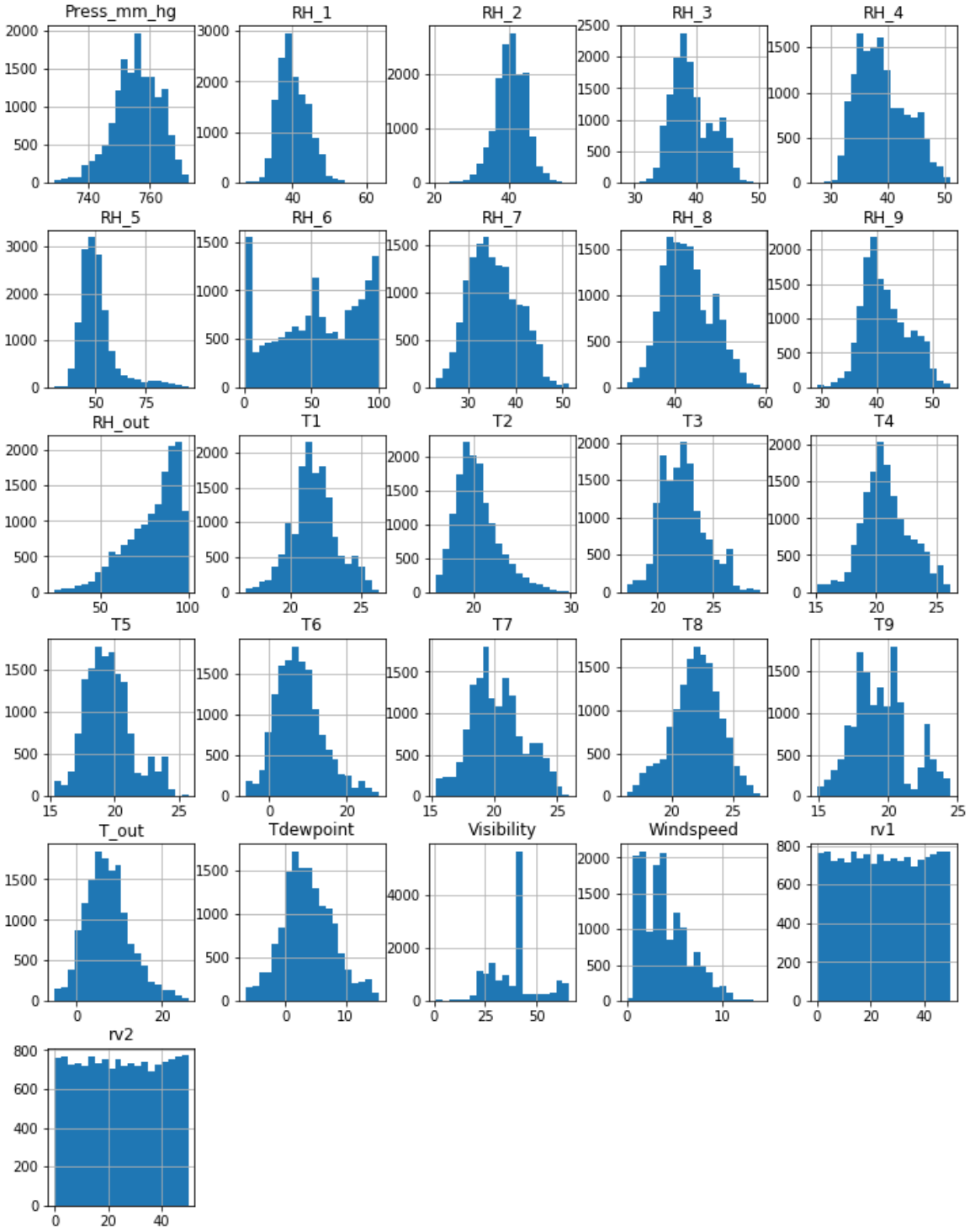


Рисунок 2.9 – Графіки функцій вхідних даних.

Visibilty – Цей стовпець негативно перекошений.

Windspeed – Цей стовпець похило перекошений.

Температурні стовпчики - Температура всередині будинку коливається в межах від 14,89 градусів до 29,85 градусів, температура зовні зовнішня (T6)

коливається від -6,06 до 28,29 градусів. Причиною такої різниці є те, що датчики тримаються поза будинком.

Колони з вологістю - Вологість в будинку коливається від 20,60% до 63,36% за винятком RH_5 (Ванна кімната) та RH_6 (Зовнішні будинку), яка коливається відповідно від 29,82% до 96,32% та 1% до 99,9%.

Appliances – 75% споживання приладу менше 100 Вт. При максимальному споживанні 1080 Вт в цьому стовпці будуть люди, що переживають люди, і є невелика кількість випадків, коли споживання дуже велике.

Сфокусуємось на функціях RH_6 , RH_out , Visibility , Windspeed через нерегулярний розподіл та побудуємо, відповідно, до цих функцій графіки за допомогою методу distplot

Фрагмент коду для візуалізації зображено на рисунку 2.10.

```
f, ax = plt.subplots(2,2,figsize=(12,8))
vis1 = sns.distplot(feature_vars["RH_6"],bins=10, ax= ax[0][0])
vis2 = sns.distplot(feature_vars["RH_out"],bins=10, ax=ax[0][1])
vis3 = sns.distplot(feature_vars["Visibility"],bins=10, ax=ax[1][0])
vis4 = sns.distplot(feature_vars["Windspeed"],bins=10, ax=ax[1][1])
```

Рисунок 2.10 – Код візуалізації

Після виконання коду було побудовано 4 графіки: RH_6, RH_out, Visibility та Windspeed. На яких побудовано одночасно два графіки для кращої візуалізації. Графіки вологості навколо будівлі, видимості та швидкості вітру зображені на рисунку 2.11

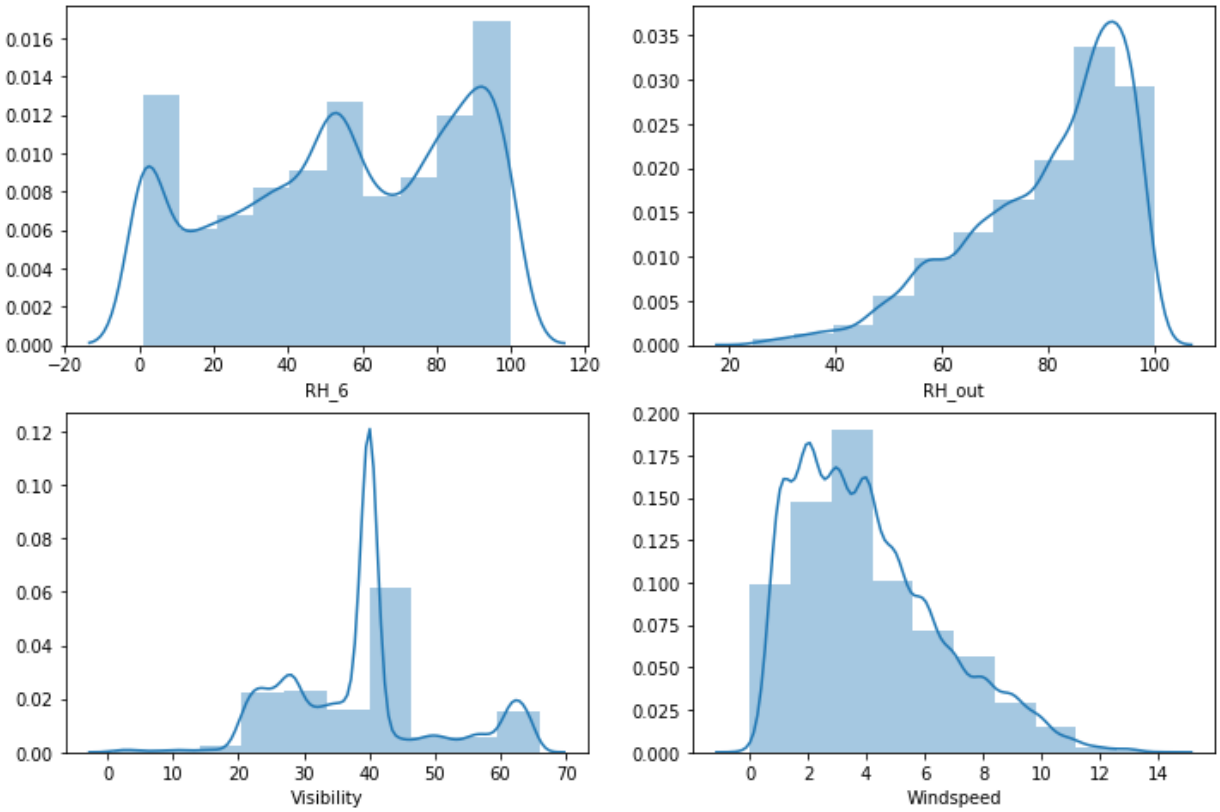


Рисунок 2.11 – Графіки вологості, видимості та швидкості вітру

Але вхідних даних занадто багато, і це може погіршити прогноз в подальшій розробці, тому для того, аби переконатись в тому, що доцільно використовувати саме всю інформацію, було вирішено проаналізувати всі графіки

Після аналізу всіх графіків, було вирішено, на основі даних з погоди, температури, споживання приладів та випадкового стовбця, провести кореляцію.

За допомогою бібліотеки Mathplotlib, було виведено діаграму кореляції. Фрагмент коду, та сама діаграма зображені на рисунках 2.12, 2.13 – відповідно.

```
train_corr = train[col_temp + col_hum + col_weather +col_target+col_randoms]
corr = train_corr.corr()
# Mask the repeated values
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

f, ax = plt.subplots(figsize=(16, 14))
#Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, annot=True, fmt=".2f" , mask=mask,)
    #Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
    #Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
    #show plot
plt.show()
```

Рисунок 2.12 – Код виводу діаграми

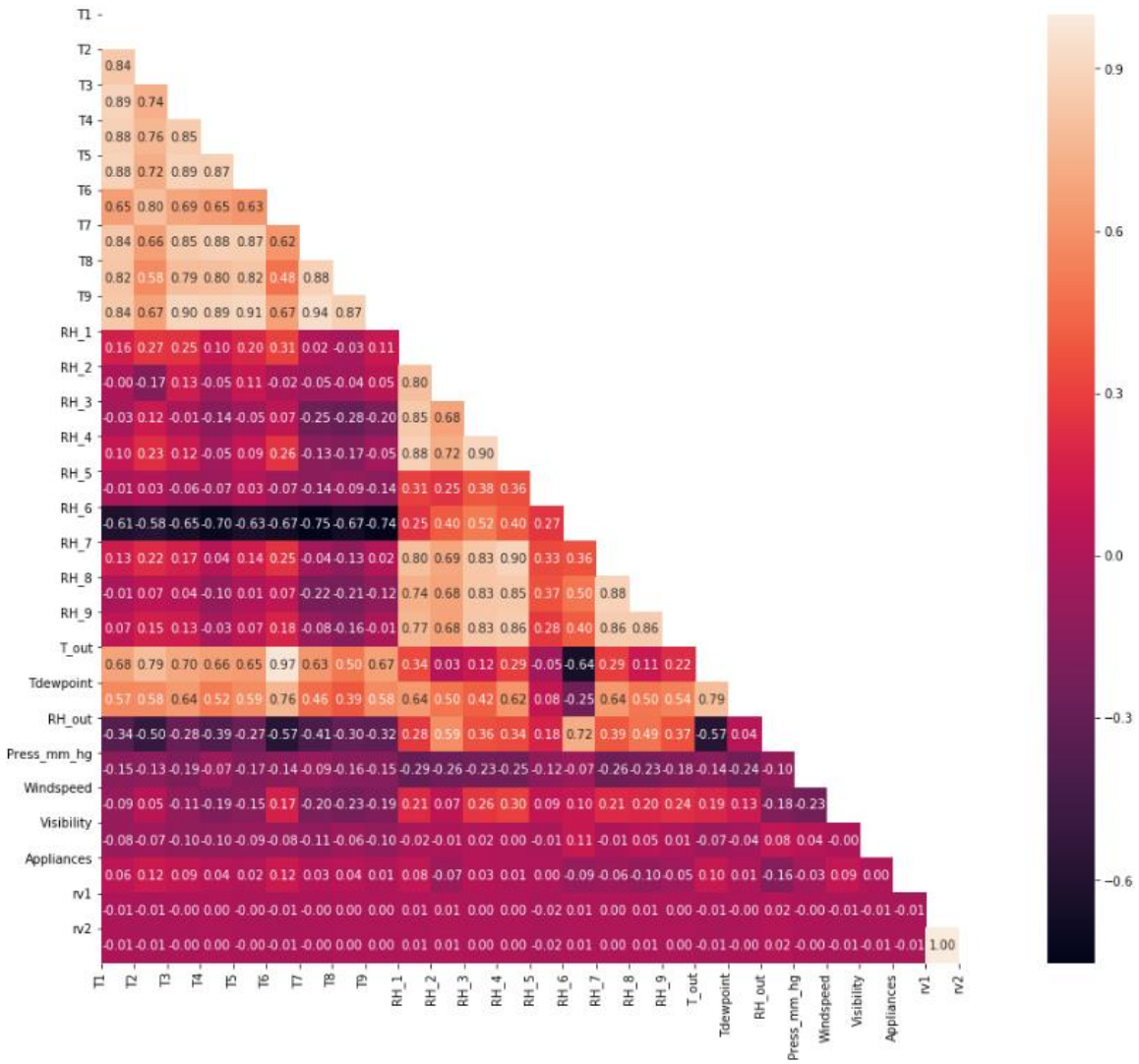


Рисунок 2.13 – Діаграма кореляції

Діаграма не виявилась саме такою інформативною, як потрібно, тому за для кращого розуміння кореляції було вирішено вивести ці значення текстом, а щоб не виводити всі, нам буде достатньо 40 з найбільшими показниками, тому потрібно створити функцію, яка перетворить трикутну кореляцію в масив з даними, а потім виведе нам перші 40 впорядкованих від більшого до меншого. Результат зображений на рисунку 2.14.

```
Top Absolute Correlations
rv1    rv2      1.000000
T6     T_out    0.974683
T7     T9       0.943921
T5     T9       0.910009
T3     T9       0.900209
RH_3   RH_4     0.899889
RH_4   RH_7     0.895120
T1     T3       0.892292
T4     T9       0.888264
T3     T5       0.887545
RH_7   RH_8     0.884849
T1     T5       0.884692
T7     T8       0.881513
RH_1   RH_4     0.880504
T1     T4       0.876682
T4     T7       0.876677
       T5       0.871733
T5     T7       0.869655
T8     T9       0.868251
RH_7   RH_9     0.860155
RH_4   RH_9     0.858454
RH_8   RH_9     0.857476
T3     T4       0.851790
RH_4   RH_8     0.848829
T3     T7       0.846021
RH_1   RH_3     0.845592
T1     T9       0.842862
       T7       0.837079
       T2       0.835729
RH_3   RH_9     0.834419
       RH_7     0.833924
```

Рисунок 2.14 – Впорядковані значення кореляції

Усі змінні температури від T1-T9 і T_out мають позитивну кореляцію з цільовими приладами. Для внутрішніх температур кореляції є високими, як очікувалося, оскільки вентиляція працює від HRV і мінімізує перепади температур повітря між приміщеннями. Чотири колони мають високий ступінь кореляції з T9 - T3, T5, T7, T8, також T6 & T_Out має високу кореляцію (обидві температури зовні). Отже, T6 та T9 можна вилучити з навчального набору, оскільки надана ними інформація може надаватися в інших полях.

Погодні атрибути – Visibility, Tdewpoint, Press_mm_hg мають низькі значення кореляції. А випадкові змінні не грають ніякої ролі.

Тому рядки T6, T9, та випадкові змінні буде вилучено.

Код вилучення T6, T9, Visibility і випадкові змінних, та переіндексації всіх атрибутів зображено на рисунку 2.15.

```
train_X.drop(["rv1","rv2","Visibility","T6","T9"],axis=1 , inplace=True)
test_X.drop(["rv1","rv2","Visibility","T6","T9"], axis=1, inplace=True)
```

Рисунок 2.15 – Вилучення елементів з низьким рівнем кореляції та переіндексація масиву даних

Для отримання відсортованого переліку функцій у порядку важливості, використаємо функцію argsort() яка повертає індекси, сортувальні елементи вихідного масиву [24]. Фрагмент коду з використанням методу argsort зображено на рисунку 2.16.

```
feature_indices = np.argsort(grid_search.best_estimator_.feature_importances_)
```

Рисунок 2.16 – Використання методу argsort

Тепер створимо діаграму для отримання візуалізації результатів. Зображено фрагмент коду на рисунку 2.17.

```
importances = grid_search.best_estimator_.feature_importances_
indices = np.argsort(importances)[::-1]
names = [train_X.columns[i] for i in indices]
# Створюєм таблицю
plt.figure(figsize=(10,10))

# Заголовок таблиці
plt.title("Feature Importance")

# Додаєм стовбці
plt.bar(range(train_X.shape[1]), importances[indices])

# Додаєм імена стовбців по осі x
plt.xticks(range(train_X.shape[1]), names, rotation=90)

# Виводимо діаграму
plt.show()
```

Рисунок 2.17 – Створення та виведення діаграми на екран

На рисунку зображеному вище, видно як ми задаєм параметри для створення діаграми, через метод figure, створено заголовок таблиці, стовбці та імена стовбців. Також методом show було виведено діаграму на екран. Діаграма зображена на рисунку 2.19.

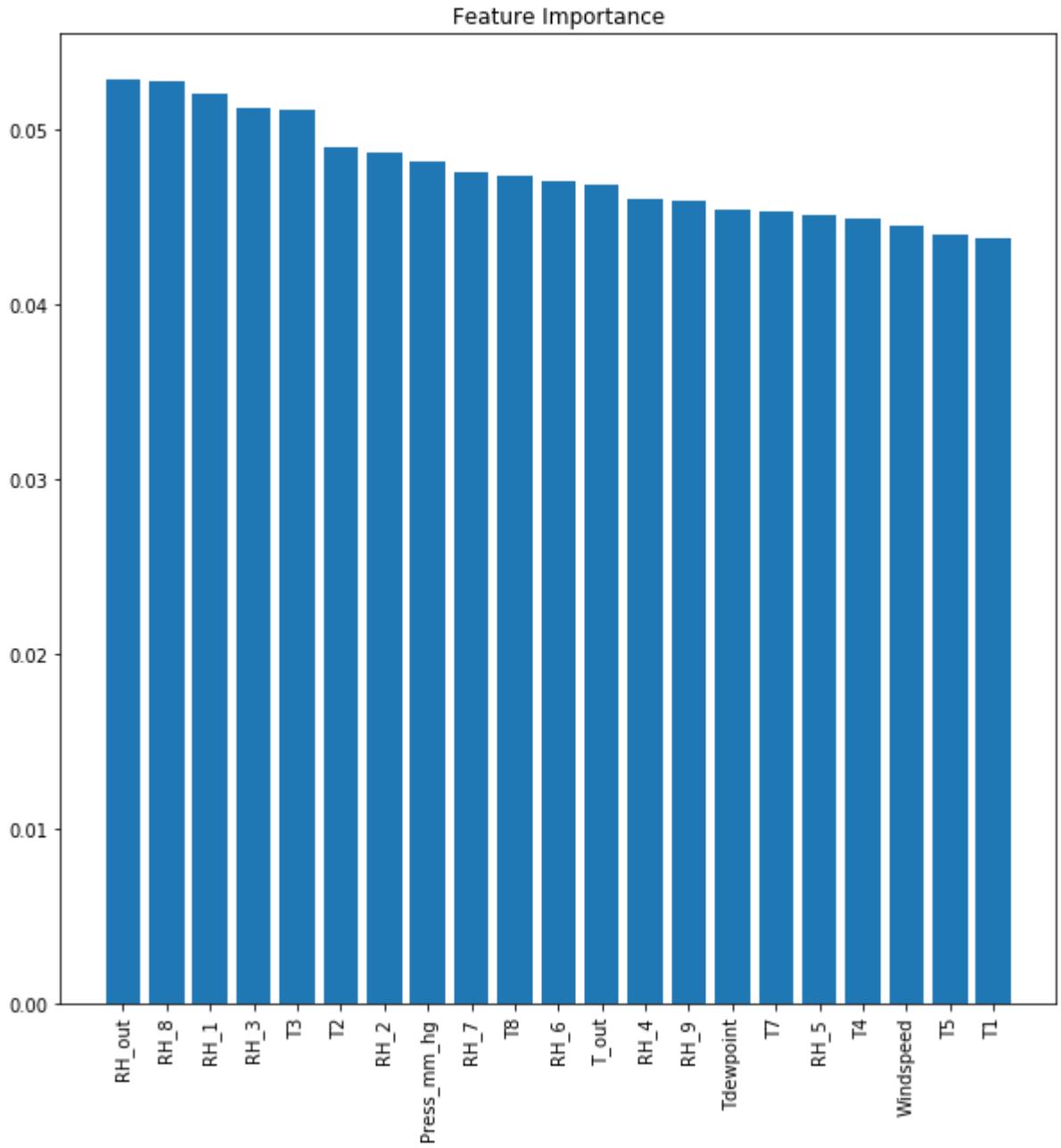


Рисунок 2.19 – Діаграма важливості даних

З побаченого на рисунку, можна визначити, що останні 3 стовбці найменше впливають на майбутній прогноз. Тому вони не будуть приймати участь у прогнозі.

Так як температура в ванній кімнаті та у кухонній зоні, майже не впливають на майбутній прогноз, також і швидкість вітру.

На базі отриманих результатів вилучимо атрибут, який будемо прогнозувати та проведемо порівняння методів. За допомогою бібліотеки sklearn імпортуємо методи прогнозування та конвертуємо результати в Дата фрейм. Таблиця результатів зображена на рисунку 2.17.

	Name	Test_R2_Score	Test_RMSE_Score	Train_R2_Score	Train_Time
0	Lasso:	0.000000	1.000000	0.000000	0.048688
1	Ridge:	0.121391	0.937341	0.137553	0.042787
2	KNeighborsRegressor:	0.485560	0.717245	0.681464	0.030380
3	SVR:	0.209934	0.888857	0.235724	15.629574
4	RandomForest	0.525681	0.688708	0.913691	3.217018
5	ExtraTreeRegressor :	0.577183	0.650244	1.000000	0.822365
6	GradientBoostingClassifier:	0.232897	0.875844	0.333526	1.765342
7	XGBRegressor:	0.226322	0.879590	0.322174	1.813901
8	MLPRegressor:	0.243178	0.869955	0.298567	3.191503

Рисунок 2.17 – Таблиця результатів прогнозування різних методів

Як ми бачимо лідерами з обраних методів прогнозування є ExtraTreeRegressor, RandomForest, KNeighborsRegressor, так як їх точність прогнозування є найвищою з обраних, а середньоквадратична похибка моделей є найменшою. Тому покращувати будемо саме ці моделі.

2.3 Висновки

У процесі вдосконалення інформаційної технологію прогнозування енергоспоживання електрообладнання будівлі з низьким енергоспоживанням на основі інтернету речей, було обрано моделі, які доцільно використовувати для поставленої задачі. Також було проведено: аналіз вхідних даних; проведено огляд всіх моделей; обрано моделі для вдосконалення.

3 ВДОСКОНАЛЕННЯ МОДЕЛЕЙ ПРОГНОЗУВАННЯ

3.1 Вдосконалення моделі RandomForest

Всі дерева комітетів будуються незалежно одне від одного згідно з наступним процесом:

Створення випадкової підвибірки з величиною повторення n із навчальної вибірки. (Тому деякі приклади будуть вводити цей приклад кілька разів, і приблизно N / 3 приклади не будуть включені взагалі).

Ми створюємо дерево рішень, яке класифікує приклади цієї під вибірки. Під час створення наступного вузла дерева ми будемо вибирати функції на основі функцій, які слід розділити, а не з усіх функцій M, і Він вибирає лише з m функцій, вибраних випадковим чином. Найкращий вибір цих m функцій можна зробити різними способами. Оригінальний код Бреймана використовує тест Джині, який також використовується в алгоритмі дерева рішень CART. Деякі реалізації цього алгоритму використовують критерій приросту інформації замість.

Дерево не будується доти, поки субвибірки повністю не вичерпаються і не будуть придатними для процедури різання (на відміну від дерев прийняття рішень на основі алгоритмів, таких як CART або C4.5).

Класифікація об'єктів проводиться шляхом голосування: кожному дереву комітету присвоюється об'єкт, який класифікується в категорії і виграє категорію дерева з найбільшою кількістю голосів.

Обирається оптимальна кількість дерев, щоб мінімізувати помилку класифікатора в тестовій вибірці. Якщо ні, то оцінка помилок, що випадають із сумки, буде мінімізована: частка комітетів неправильно класифікувала приклади навчальної вибірки, виключаючи звуки дерев у їхніх навчальних під вибірках [25].

Те, що ми щойно описали, - це критерії створення випадкового лісу. Однак у випадкових лісах використовуються дерева рішень з однією або

кількома глибинами. Термін випадковий походить від того, що ми випадковим чином обираємо навчальний набір, і оскільки у нас є колекція дерев, ми природно називаємо його лісом, отже, випадковим лісом. Який вузол у дереві обирає випадкову підмножину функції. Для кожної з цих вибраних функцій алгоритм шукає найкращу точку різання для визначення розділу даної функції. Потім для створення кореневого вузла буде використана функція із випадково вибраного підмножини, яка дасть найчистіший спліт. Дерево росте на глибину, а потім той самий процес повторюється для всіх інших вузлів у дереві, поки не буде досягнута бажана глибина дерева. Нарешті, слід зазначити, що кожне дерево будується окремо з використанням різних навантажувальних стрічок, що змінить дерево. Отже, кожне дерево допускає різні помилки, і в поєднанні можна створити сильний класифікатор [26].

Проведемо тюнінг параметрів випадкових лісів, для цього імпортуємо GridSearchCV з бібліотеки sklearn.

GridSearchCV – це дуже потужний інструмент для автоматичного вибору параметрів моделей машинного навчання. GridSearchCV використовує простий пошук, щоб знайти найвищі параметри: він створює модель для кожної можливої комбінації параметрів. Важливо зазначити, що цей метод може зайняти багато часу.

Перший крок - це написати параметри, які ми хочемо врахувати, і з цих параметрів вибрати найкращі. Перший крок зображений на рисунку 3.1.

```
param_grid = [{
              'n_estimators' : [50,100,150],
              'max_features': ["sqrt"],
              'max_depth': [3,4,5,6,7,8,9,10,11,12],
            }]
```

Рисунок 3.1 – Параметри які ми хочемо перевірити

Тепер ми можемо створити наш об’єкт GridSearchCV і вмістити дані:

– Обрано кількість дерев у випадковому лісі, максимальна кількість рівнів у дереві;

– Кількість функцій, які слід враховувати при пошуку найкращого розділення;

– Максимальна глибина дерева. Якщо немає, то вузли розширюються, поки всі листки не стануть чистими або поки всі листки не містять менше ніж min_samples_split sample;

– Кількість завдань, які потрібно виконувати паралельно. fit, predict, decision_path і apply все розпаралелювання над деревами. Noneозначає 1, якщо не в joblib.parallel_backend контексті. -1 означає використання всіх процесорів;

– Визначає стратегію розділення перехресних перевірок. Можливі входи для cv.

Введення всіх цих дій зображено на рисунку 3.2

```
from sklearn.model_selection import GridSearchCV
param_grid = [{
              'n_estimators' : [50,100,150],
              'max_features': ["sqrt"],
              'max_depth': [3,4,5,6,7,8,9,10,11,12],
            }]
rfr = RandomForestRegressor(random_state=40)
# Застосуєм моделі пошуку сітки
grid_search_rfr = GridSearchCV(estimator = rfr, param_grid = param
_grid, cv = 5, n_jobs = -1, scoring='r2',  verbose=2)
grid_search_rfr.fit(train_X, train_y)
```

Рисунок 3.2 – Введення параметрів

На рисунку вище, зображено введення параметрів та підбір моделі пошуку. Встановлення 5 складок для кожного з 30 кандидатів, загалом 150

підходів. Це оптимальний набір параметрів для визначеної функції, що найкраще відповідає заданому набору спостережень.

Все проходить в 2 етапи, ці етапи зображені на рисунку 3.3.

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  33 tasks      | elapsed:   17.7s
[Parallel(n_jobs=-1)]: Done 150 out of 150 | elapsed:  2.0min finished
```

Рисунок 3.3 – Етапи виконання методу fit

Перший етап виконувався в продовж 17.7 секунд та закінчив 33 задачі;

Другий етап виконувався 2 хвилини та завершив обчислення виконавши всі 150 задач;

Тепер потрібно дізнатись які параметри були найкращими для обрахунків, саме це і зображено на рисунку 3.4.

```
grid_search_rfr.best_params_
```

```
{'max_depth': 12, 'max_features': 'sqrt', 'n_estimators': 150}
```

Рисунок 3.4 – Кращі параметри моделі

Виведемо найкращий оцінювач, який був обраний пошуком, тобто оцінювач, який дав найвищий бал на відсутніх даних. Зображений на рисунку 3.5.

```
grid_search_rfr.best_estimator_
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=12,
           max_features='sqrt', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=150, n_jobs=None,
           oob_score=False, random_state=40, verbose=0, warm_start=False)
```

Рисунок 3.5 – Дані з найкращого оцінювача

Як бачимо кількість функцій які слід застосовувати при пошуку найкращого розділення має параметр sqrt, найкраща кількість дерев у лісі це 150, Значення bootstrap дорівнює True, це свідчить що для побудови кожного дерева використовуються зразки bootstrap. Мінімальна кількість зразків, необхідних для розділення внутрішнього вузла дорівнює 2. Максимальної глибини дерева було вказано від 3 до 12 та найкращим оцінювачем глибина дерева є 12 [27].

Виведемо результати прогнозування та помножимо їх на 100 для зручності читання. Фрагмент коду та результат зображено на рисунку 3.6 – 3.7 відповідно.

```
print('Training set rfr R2 Score - ', grid_search_rfr.best_estimato
r_.score(train_X,train_y) * 100)
print('Testing set rfr R2 Score - ', grid_search_rfr.best_estimator
_.score(test_X,test_y) * 100)
print('Testing set rfr RMSE Score - ', np.sqrt(mean_squared_error(t
est_y, grid_search_rfr.best_estimator_.predict(test_X))) * 100)
```

Рисунок 3.6 – Вивід результатів прогнозу

```
Training set rfr R2 Score -  67.73102771588601
Testing set rfr R2 Score -  41.48186453266495
Testing set rfr RMSE Score -  76.49714731108281
```

Рисунок 3.7 – результати прогнозу

Досягнуто зменшення розбіжності в тестових результатах та тренувальних, але це вплинуло на самі результати прогнозу. І погіршило їх на 11% в тестовому режимі і на 24% на тренувальному. Це свідчить тому що ця модель не підходить для обчислення. Тому що при подальшому збільшенні кількості дерев та збільшення їх глибини призводить до оверфітингу. Тому ця модель не влаштовує нас.

3.2 Вдосконалення моделі ExtraTreesRegressor

Розберемось як саме працює ця модель.

Класифікатор Extra Trees працює подібно до випадкового лісу. Однак існують відмінності в продуктивності.

А саме:

– дерева рішень демонструють велику дисперсію;

– випадкові ліси - середню дисперсію;

– зайві дерева - низьку дисперсію.

Кожен пень рішення буде побудований за такими критеріями:

– Всі дані, наявні в навчальному наборі, використовуються для побудови кожного пня.

– Щоб сформувати кореневий вузол або будь-який вузол, найкращий розподіл визначається шляхом пошуку в підмножині випадково вибраних об’єктів розміром sqrt (кількість об’єктів). Розбиття кожної обраної функції вибирається випадковим чином.

– Максимальна глибина куксу рішення – одна [26].

Проведемо тюнінг параметрів за для покращення прогнозу.

Тюнінг параметрів зображений на рисунку 3.8.

```
from sklearn.model_selection import GridSearchCV
param_grid = [{
              'n_estimators' : [300,350,400,450],
              'max_features': ["auto", "sqrt", "log2"]
            }]
reg = ExtraTreesRegressor(random_state=90)
# Застосуєм моделі пошуку сітки
grid_search = GridSearchCV(estimator = reg, param_grid = param_grid, cv = 5, n_jobs = -1 , scoring='r2' , verbose=2)
grid_search.fit(train_X, train_y)
```

Рисунок 3.8 – Тюнінг параметрів прогнозування

З наведеного вище малюнку було проведено тюнінг параметрів прогнозування.

n_estimators – вказує на кількість дерев у лісі цей параметр було задано за для збільшення можливих дерев, хоч це і збільшить час прогнозування.

max_features – вказує на функції, які слід враховувати для пошуку найкращих рішень, за замовчуванням стоїть авто [24].

За допомогою методу GridSearchCV- було передано такі параметри:

– Першим параметром estimator, передається досліджувана функція яка записана в змінній reg;

– Наступним передаємо param_grid, а саме кількість дерев, та функції, які слід враховувати, а саме: auto, sqrt та log2;

– Потім ми передаєм параметр, який вказує методу, що потрібно зробити 5-ти кратну перехресну перевірку, цей метод потрібно вказати тому що за замовчуванням, таку перевірку використовувати не обов'язково;

– Параметр n_jobs – вказує на кількість завдань, які потрібно виконувати паралельно, переданий нами параметр -1 означає що потрібно використовувати всі процесори;

– Scoring – що оцінюємо, передаєм параметр r2 [28];

– Verbose – встановлення рівня багатослівності, якщо вважати що в більшості оцінювачів всього 2-3 рівня деталізації, тому я зупинився на 2 [29].

Метод fit – це тип оптимізації, який знаходить оптимальний набір параметрів для визначеної функції, що найкраще відповідає заданому набору спостережень.

Функція відображення, яку також називають базовою, може мати будь-яку форму, яка вам подобається, включаючи пряму лінію (лінійна регресія), криву лінію (поліноміальна регресія) та багато іншого. Це забезпечує гнучкість та контроль для визначення форми кривої, де використовується процес оптимізації для пошуку конкретних оптимальних параметрів функції [16].

Результати методу зображенні на рисунку 3.9.

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  33 tasks      | elapsed:  4.0min
[Parallel(n_jobs=-1)]: Done  60 out of  60 | elapsed:  5.6min finished

GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=ExtraTreesRegressor(bootstrap=False, criterion='mse', max_depth=None,
          max_features='auto', max_leaf_nodes=None,
          min_impurity_decrease=0.0, min_impurity_split=None,
          min_samples_leaf=1, min_samples_split=2,
          min_weight_fraction_leaf=0.0, n_estimators='warn', n_jobs=None,
          oob_score=False, random_state=90, verbose=0, warm_start=False),
       fit_params=None, iid='warn', n_jobs=-1,
       param_grid=[{'n_estimators': [300, 350, 400, 450], 'max_features': ['auto', 'sqrt', '
log2']}],
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring='r2', verbose=2)
```

Рисунок 3.9 – Результати методу fit

Тепер виведемо результати нашого покращеного прогнозу, за допомогою виведення кращого оцінювача, результати тренування, тестових

даних та середньоквадратичну помилку моделі, для зручності всі результати було помножено на 100. Фрагмент коду та результати прогнозування зображені на рисунку 3.10 – 3.11 відповідно.

```
print('Training set R2 Score - ', grid_search.best_estimator_.score
(train_X,train_y) * 100)
print('Testing set R2 Score - ', grid_search.best_estimator_.score
(test_X,test_y) * 100)
print('Testing set RMSE Score - ', np.sqrt(mean_squared_error(test_
y, grid_search.best_estimator_.predict(test_X))) * 100)
```

Рисунок 3.10 – Код виведення результатів прогнозування

```
Training set R2 Score -  100.0
Testing set R2 Score -  68.48834662363605
Testing set RMSE Score -  60.42487350120308
```

Рисунок 3.11 – Результати прогнозування

Як бачимо з зображення 3.11 покращення технології сприяло кращому прогнозу понад 11%, та зменшення середньоквадратичної похибки майже на 5%. Але в моделі занадто великий overfitting, тобто модель занадто тісно або точно відповідає певному набору даних, і, отже, може не вписати додаткові дані або надійно передбачити майбутні спостереження. Іншими словами, модель пам’ятає величезну кількість прикладів, замість того, щоб навчитися помічати особливості.

Найбільш очевидним наслідком переобладнання є низька продуктивність набору даних перевірки. Інші негативні наслідки включають:

– Функція, яка переобладнана, швидше за все, вимагатиме більше інформації про кожен елемент у наборі даних перевірки, ніж оптимальна функція; збір цих додаткових непотрібних даних може бути дорогим або схильним до помилок, особливо якщо кожен окремий фрагмент інформації

повинен бути зібраний шляхом людського спостереження та введення даних вручну;

– Більш складна, переобладнана функція, швидше за все, буде менш портативною, ніж проста. В одному крайньому випадку, лінійна регресія з однією змінною настільки портативна, що за необхідності її можна зробити навіть вручну. Інша крайність - це моделі, які можна відтворити, лише точно копіюючи всю установку оригінального моделіста, ускладнюючи повторне використання або наукове відтворення [30].

3.3 Вдосконалення KNeighborsRegressor

Алгоритм k-найближчих сусідів (KNN) - це простий керований алгоритм машинного навчання, який може бути використаний для вирішення задач класифікації та регресії. Це легко впровадити та зрозуміти, але має основний недолік – значно зменшуватися, оскільки обсяг цих даних, що використовуються, зростає.

KNN працює, знаходячи відстань між запитом та всіма прикладами в даних, вибираючи вказану кількість прикладів (K), найближчу до запиту, потім голосує за найчастішу мітку (у випадку класифікації) або усереднює мітки (у випадок регресії).

Алгоритм KNN може бути використаний як для задач класифікації, так і для регресії. Алгоритм KNN використовує " схожість функцій " для прогнозування значень будь-яких нових точок даних. Це означає, що новій точці присвоюється значення на основі того, наскільки вона схожа на точки в навчальному наборі.

Перший крок повинні обчислити відстань між новою точкою і кожної навчальної точкою. Існують різні методи обчислення цієї відстані, серед яких найвідоміші методи - Евклідіан, Манхеттен (для безперервної) та відстань Хеммінга (для категоричної).

– Евклідова відстань: Евклідова відстань обчислюється як квадратний корінь із суми квадратних різниць між новою точкою (x) та існуючою точкою (y). Обчислюється за формулою:

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}. \quad (3.1)$$

– Відстань до Манхеттена: Це відстань між реальними векторами з використанням суми їх абсолютної різниці. Обчислюється за формулою:

$$\sum_{i=1}^{k}|x_i - y_i|. \quad (3.2)$$

– Відстань Хеммінга: використовується для категоріальних змінних. Якщо значення (x) і значення (y) однакові, відстань D буде дорівнює 0. В іншому випадку D = 1. Формула зображена на рисунку 3.12.

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$
$$x = y \Rightarrow D = 0$$
$$x \neq y \Rightarrow D = 1$$

Рисунок 3.12 – Формула відстані Хеммінга

Як тільки буде виміряна відстань нового спостереження від точок у нашому навчальному наборі, наступним кроком є вибір найближчих точок. Кількість балів, які слід врахувати, визначається значенням k.

При дуже низькому значенні k (припустимо, k = 1) модель переставляє дані тренувань, що призводить до високого рівня помилок у наборі

перевірки. З іншого боку, при високому значенні k модель погано працює як на поїзді, так і на наборі перевірки [25].

Щоб вибрати K, який підходить для наших даних, ми запускаємо алгоритм KNN кілька разів з різними значеннями K і вибираємо K, який зменшує кількість помилок, з якими ми стикаємось, зберігаючи здатність алгоритму точно робити прогнози, коли йому надано дані, яких він не бачив раніше.

Ось кілька речей, про які слід пам’ятати:

– Коли ми зменшуємо значення K до 1, наші прогнози стають менш стабільними. Подумайте хвилинку, уявіть K = 1, і ми маємо точку запиту, оточену кількома червоними та одним зеленим, але зелений - це найближчий сусід. Розумно, ми вважаємо, що точка запиту, швидше за все, червона, але оскільки K = 1, KNN неправильно передбачає, що точка запиту зелена;

– І навпаки, коли ми збільшуємо значення K, наші прогнози стають стабільнішими завдяки голосуванню більшості / усередненню, і, отже, частіше роблять більш точні прогнози (до певної точки). Зрештою, ми починаємо спостерігати все більшу кількість помилок. Саме в цей момент ми знаємо, що ми занадто сильно відсунули значення K;

– У випадках, коли ми беремо більшість голосів (наприклад, вибираємо режим у класифікаційній задачі) серед ярликів, ми зазвичай робимо K непарним числом, щоб мати тайбрек.

Імпортуємо GridSearchCV з бібліотеки sclearn.

Grid Search є ефективним методом для коригування параметрів під контролем навчання та покращення результатів узагальнення моделі. За допомогою Grid Search можна побудувати всі можливі комбінації параметрів, що цікавлять, і знайти найкращі.

Scikit-learn забезпечує клас GridSeaechCV . Очевидно, що спочатку нам потрібно вказати параметри, які ми хочемо шукати, а потім GridSearchCV виконає всі необхідні підгонки моделі [31].

Імпорт та налаштування моделі зображені на рисунку 3.13.

```
from sklearn.model_selection import GridSearchCV
params = {'n_neighbors':[2,3,4,5,6,7,8,9]}

knn = neighbors.KNeighborsRegressor()

model = GridSearchCV(knn, params, cv=5)
model.fit(train_X,train_y)
model.best_params_
```

Рисунок 3.13 – Внесення параметрів в модель

В змінну params внесені параметри для n_neighbors від 2 до 9, аби знайти найкращий. Потім передано в GridSearchCV змінну з класифікатором, змінну параметрів та передано параметр cv зі значенням 5, для 5 кратної перехресної перевірки.

Виведемо на екран найкращі параметри. Які зображені на рисунку 3.14.

```
{'n_neighbors': 2}
```

Рисунок 3.14 – Найкращі параметри

Виведемо найкращий оцінювач, який зображений на рисунку 3.15.

```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=None, n_neighbors=2, p=2,
          weights='uniform')
```

Рисунок 3.15 – Найкращий оцінювач

Вибір алгоритму пошуку сусідів контролюється за допомогою ключового слова 'algorithm', яке повинно бути одним із: 'auto', 'ball_tree', 'kd_tree', 'brute'. Так як ми його не вказували був обраний auto.

Наступним параметром є leaf_size. Як велике, так і мале leaf_size може призвести до неоптимальних витрат на запит. При leaf_size наближенні до 1, накладні витрати, що беруть участь у обходженні вузлів, можуть значно сповільнити час запитів. Для leaf_size наближення розміру навчального набору запити стають по суті грубою формою. Хорошим компромісом між ними є значення параметра за замовчуванням. leaf_size = 30.

Weights значення за замовчуванням , присвоює однакові ваги кожному сусіду. присвоює ваги, пропорційні зворотному відстані від точки запиту. Як варіант, для обчислення ваг може бути надана користувацька функція відстані. weights = 'uniform' weights = 'distance' [32].

Виведемо результати прогнозування, зображені на рисунку 3.16 – 3.17 відповідно.

```
print('Training set knr R2 Score - ', model.best_estimator_.score(t
rain_X,train_y) * 100)
print('Testing set knr R2 Score - ', model.best_estimator_.score(te
st_X,test_y) * 100)
print('Testing set knr RMSE Score - ', np.sqrt(mean_squared_error(t
est_y, model.best_estimator_.predict(test_X))) * 100)
```

Рисунок 3.16 – Вивід результатів

```
Training set knr R2 Score -  73.23026930811155
Testing set knr R2 Score -  57.52448943875144
Testing set knr RMSE Score -  65.17323880339887
```

Рисунок 3.17 – Результати прогнозування

З результатів прогнозу видно, що отримані результати покращились:

– Тренувальні результати збільшились на 5%;
– Тестові результати збільшились на 9%;
– Середньоквадратична похибка зменшилась на 5%.

На підставі цих результатів можна сказати, що метод KNN показав себе краще ніж інші методи, так як він не перенавчився занадто і результати тесту та тренування не мають такої великої різниці, як у інших.

Основний недолік KNN –

стає значно повільнішим із збільшенням обсягу даних - робить його непрактичним вибором в середовищах, де прогнозування потрібно робити швидко. Більше того, існують більш швидкі алгоритми, які можуть дати точніші результати класифікації та регресії [32].

3.4 Висновки

В даному розділі було проведено вдосконалення моделей прогнозування таких як:

– Extra Trees;
– Random Forest;
– K-Nearest Neighbors.

З отриманих результатів можна зазначити, що хоч ліси на тестовому наборі показали кращі результати основним недоліком їх було те, що вони перенавчились на тренувальному наборі таким чином вони чудово прогнозують тренувальні набори, але тестові дані сильно різняться результатами, чого не можна сказати про k-найближчих сусідів, цей метод є лідером на цьому наборі даних.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання економічного потенціалу розробки

Метою технічного аудиту є оцінка комерційного потенціалу розробок в результаті науково-технічної діяльності.

Об'єктом дослідження кваліфікаційного іспиту магістра є "Інтернет інформаційних технологій прогнозування споживання енергії електричним обладнанням"

Три незалежні експерти взяли участь у технічному огляді. Кожен фахівець повинен бути ознайомлений із запропонованою розробкою та заповнити форму, яка визначає рекомендовані критерії для оцінки комерційного потенціалу розробки та можливі пункти оцінки. Після цього розраховується середнє арифметичне балів та визначається рівень комерційного потенціалу нового проекту розвитку.

Оцінювання комерційного потенціалу розробки здійснюється за критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
8	У технічній та комерційній реалізації цієї ідеї немає експертів	Потрібно найняти експертів або витратити багато грошей та часу на підготовку наявнрих експертів	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потреби фінансових ресурсів незначні. Немає джерела коштів	Потрібно багато фінансів Ресурси. Є джерела фінансування	Потреби незначного фінансового ресурсу. Є джерела фінансування	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Матеріали, необхідні як у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі І вже давно використовується у виробництві
111	Термін реалізації ідеї більший за 10 років	Термін виконання ідей більше 5 років. Тривалий термін окупності 10 років	Термін виконання ідеї від 3 до 5 років. Тривалий термін окупності 5 років	Термін втілення ідеї менше ніж 3 роки. Термін окупності від 3 до 5 років	Кінцевий термін реалізації ідеї Менше 3 років. Термін окупності менше 3 років
112	Необхідно розробити нормативні документи та отримати ліцензії на масове виробництво та реалізацію продукції	Потрібно отримати ліцензію на масове виробництво та реалізацію продукції, що вимагає великих грошей та часу	Процес отримання ліцензії на виробництво та продаж товару майже не вимагає грошей і часу	Потрібно лише інформувати відповідні департаменти про виробництво та реалізацію продукції	Нормативні обмеження щодо виробництва та реалізації продукції відсутні

Результати оцінювання комерційного потенціалу наведено в (табл. 4.2).

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Експерти		
	Експерт 1	Експерт 2	Експерт 3
	Бали, виставлені експертами:		
1	4	3	3
2	3	2	2
3	4	3	3
4	4	4	3
5	3	2	3
6	2	2	4
7	2	4	3
8	3	2	2
9	1	1	3
10	4	4	4
11	4	3	3
12	4	4	4
Сума балів	$СБ_1$=37	$СБ_2$=34	$СБ_3$=38
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{37+34+38}{3} = 36{,}3$		

Отже, з даних, отриманих у таблиці 4.2, видно, що середнє арифметичне цих точок дорівнює 36,3, тобто рівень комерційного потенціалу нового проекту розвитку вище середнього рівня.

4.2 Прогнозування витрат на виконання науково-дослідної роботи

Прогнозування вартості дослідницьких, конструкторських та проектних робіт може включати такі етапи:

– Етап 1: Розрахунок вартості, безпосередньо пов'язаний з виконавцем цієї частини роботи.

– Етап 2: Обчисліть загальну вартість цієї роботи.

– Етап 3: передбачити загальну вартість впровадження та впровадження результатів цієї роботи.

1-й етап. Розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи, можна здійснити за такими статтями та формулами:

1) Основна заробітна плата кожного із розробників (дослідників) $З_о$, якщо вони працюють в наукових установах бюджетної сфери:

$$З_о = \frac{М}{Т_р} \cdot t \text{ [грн]}, \qquad (4.1)$$

де М – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р = 21$ день;

t – число робочих днів роботи розробника (дослідника).

Заробітна плата розробника:

$$З_р = \frac{13500}{21} \cdot 60 = 37142{,}86 \text{ (грн)}.$$

Заробітна плата наукового керівника проекту:

$$З_{НК} = \frac{8500}{21} \cdot 19 = 7690{,}48 \text{ (грн)}.$$

Витрати на оплату праці, основна заробітна плата:

$$З_о = З_Р + З_{НК} = 37142{,}86 + 7690{,}48 = 44833{,}34 \text{ (грн)}.$$

Розрахунки основної заробітної плати спеціаліста наведено в (табл. 4.3).

Таблиця 4.3 – Розрахунки основної заробітної плати спеціаліста

Найменування посади виконавця	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Програміст	13000	619,05	60	37142,86
Науковий керівник	8500	404,76	19	7690,48
Всього:				44833,34

2) Додаткова заробітна плата $З_Д$ всіх розробників та робітників, які брали участь у виконанні даної роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_Д = (0{,}1...0{,}12) \cdot З_О, \quad (4.2)$$

Додаткова заробітна плата усіх робітників та розробників:

$$З_Д = 0{,}12 \cdot 44833{,}34 = 5380 \text{ (грн)}.$$

3) Нарахування на заробітну плату $Н_{ЗП}$ розробників та робітників, які брали участь у виконанні даної роботи, розраховуються за формулою:

$$Н_{ЗП} = (З_О + З_Р + З_Д) \cdot \frac{\beta}{100}, \quad (4.3)$$

де $З_о$ – основна заробітна плата розробників, грн;

$З_д$ – додаткова заробітна плата розробників, грн;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, %.

В 2020 році ставка єдиного внеску на загальнообов'язкове державне соціальне страхування встановлено 22%.

Нарахування на заробітну плату:

$$Н_{ЗП} = (44833{,}34 + 5380) \cdot \frac{22}{100} = 11046{,}93 \text{ (грн)}.$$

4) Амортизаційні відрахування обладнання можна розрахувати за формулою:

$$А = \frac{Ц * Н_а}{100} * \frac{Т}{12} \text{ [грн]}, \quad (4.4)$$

де Ц – загальна балансова вартість обладнання, Ц = 12100 грн;

$Н_а$ – річна норма амортизаційних відрахувань. Для нашого випадку можна прийняти, що $Н_а$ = 20%;

Т – термін використання обладнання, Т = 4 міс.

Амортизаційні відрахування для персонального комп'ютера становить:

$$А = \frac{12100 \cdot 20}{100} \cdot \frac{4}{12} = 753{,}33 \text{ (грн)}.$$

5) Витрати на матеріали М, що були використанні під час виконання роботи, розраховуються по кожному виду матеріалів за формулою:

$$М = \sum_1^n H_i * Ц_і * К_і - \sum_1^n В_і * Ц_в \text{ [грн]}, \quad (4.5)$$

де $Н_і$ – витрати матеріалу *і*-го найменування;

$Ц_і$ – вартість матеріалу *і*-го найменування, грн/кг;

$К_і$ – коефіцієнт транспортних витрат, $К_і$ = 1,12;

$В_і$ – маса відходів матеріалу *і*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *і*-го найменування, грн/кг;

n – кількість видів матеріалів.

Вартість матеріалів, що були використані для розробки наведено в (табл. 4.4).

Таблиця 4.4 – Вартість матеріалів, що були використані для розробки ПЗ

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	Шт.	700	1	700
Зошит	Шт.	45	2	90
Олівець	Шт.	12	2	24
Всього				814

Загальна вартість витрат становить:

$$М = 846{,}4 \cdot 1{,}12 = 947{,}97 \text{ (грн)}.$$

6) Витрати на силову електроенергію $В_е$, розраховується за формулою:

$$В_е = В * П * Ф * К_П \text{ [грн]}, \quad (4.6)$$

де В – вартість 1кВт-год. електроенергії, В = 2,69 грн/кВт;

П – установлена потужність ноутбука, П = 0,1 кВт;

Ф – фактична кількість годин роботи ноутбука, $Ф = 263$ дні;

Кп – коефіцієнт використання потужності, $К_п$ = 0,72.

Витрати на силову енергію становлять:

$$В_е = 2{,}69 \cdot 0{,}1 \cdot 263 \cdot 0{,}72 = 51{,}11 \text{ (грн)}.$$

7) Інші витрати $І_в$ можна прийняти як 200% від суми основної заробітної плати розробників, тобто:

$$В_{ін} = 2 \cdot З_о, \quad (4.7)$$

Інші витрати становлять:

$$В_{ін} = 2 \cdot 44833{,}34 = 89666{,}68 \text{ (грн)}.$$

8) Сума усіх витрат становить:

$$В_{ін} = 44833{,}34 + 5380 + 11046{,}93 + 753{,}33 + 947{,}97 + 51{,}11 + 89666{,}68 = 152679{,}36 \text{ (грн)}.$$

2-й етап. Розрахунок загальних витрат на виконання роботи. Загальна вартість роботи визначається за $В_{заг}$ формулою:

$$В_{заг} = \frac{В}{\alpha}, \quad (4.8)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{152679,36}{1} = 152679,36 \text{ (грн)}.$$

3-й етап. Прогнозування загальних витрат на виконання та впровадження результатів. Прогнозування загальних витрат ЗВ на виконання та впровадження результатів здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta}, \quad (4.9)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи, на стадії розробки промислового зразка, $\beta \approx 0,7$.

$$ЗВ = \frac{152679,36}{0,7} = 218113,37 \text{(грн)}.$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Впровадження наукової роботи та впровадження результатів буде здійснено протягом одного року. Очікується, що протягом трьох років після розробки та впровадження, розробка та впровадження досягнуть позитивних результатів. Одним з головних позитивних результатів є збільшення прибутку.

Збільшення чистого прибутку забезпечить більше коштів для компанії (організації), тим самим покращуючи фінансові показники.

Збільшення чистого прибутку підприємства (організації) $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_{1}^{n} (\Delta П_Я \cdot N + П_Я \cdot \Delta N)_i, \quad (4.10)$$

де $\Delta П_Я$ – Покращено основні показники якості шляхом впровадження результатів розвитку цього року;

N – Визначає основні кількісні показники діяльності компанії в даному році до впровадження результатів наукових розробок;

ΔN – Удосконалення основних кількісних показників підприємства від впровадження результатів розвитку;

$П_Я$ – Після впровадження результатів наукових розробок визначає основні якісні показники корпоративної діяльності у певному році;

n – Кількість років, протягом яких, проект розвитку повинен досягти позитивних результатів.

В результаті впровадження результатів наукової розробки витрати на заробітну плату робітникам, зменшиться на 1200 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 1200 грн), а кількість достовірних матеріалів збільшиться: протягом першого року – на 350 матеріалів, протягом другого року – на 200 матеріалів, протягом третього року – на 100 матеріалів.

Кількість опублікованих матеріалів до впровадження наукової розробки складала 700 матеріалів, а прибуток, що отримало підприємство (організація) на один матеріал до впровадження наукової розробки – 400 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Збільшення чистого продукту $\Delta П_i$ протягом першого року складатиме:

$$\Delta П_1 = 1200 \cdot 700 + (400 + 1200) \cdot 350 = 1400000 \text{ (грн)}.$$

Протягом другого року:

$$\Delta П_2 = 1200 \cdot 700 + (400 + 1200) \cdot (350 + 200) = 1720000 \text{ (грн)}.$$

Протягом третього року:

$$\Delta П_3 = 1200 \cdot 700 + (400 + 1200) \cdot (350 + 200 + 100) = 1880000 \text{ (грн)}.$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

У цьому розділі необхідно кількісно визначити вигоди та прибутки, отримані впровадженням результатів роботи.

Розрахунок ефективності вкладених інвестицій передбачає проведення таких робіт:

1-й крок. Розрахуємо теперішню вартість інвестицій PV, що вкладаються в наукову розробку. Такою вартістю ми можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, розраховану раніше, тобто будемо вважати, що ЗВ = PV =218113,37(грн).

2-й крок. Розрахуємо очікуване збільшення прибутку $\Delta П_i$, що його отримає підприємство (організація) від впровадження результатів наукової

розробки, для кожного із років, починаючи з першого року впровадження. Таке збільшення прибутку також було розраховане нами раніше та становить:

$\Delta П_1 = 1400000$ (грн), $\Delta П_2 = 1720000$ (грн), $\Delta П_3 = 1880000$ (грн).

3-й крок. Для спрощення подальших розрахунків необхідно побудувати вісь часу, на яку наносять всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів.

Якщо загальні витрати ЗВ на виконання та впровадження результатів НДДКР (або теперішня вартість інвестицій PV) дорівнюють 218113,37 грн, а результати вкладених у наукову розробку інвестицій почнуть виявлятися вже вкінці другого року впровадження. То ці результати виявляться у тому, що у першому році підприємство отримає збільшення чистого прибутку на 1400000 грн. відносно базового року, у другому році – збільшення чистого прибутку на 1720000 грн (відносно базового року), у третьому році – збільшення чистого прибутку на 1880000 грн (відносно базового року).

Тоді рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати такий вигляд, наведений на рисунку 4.1.

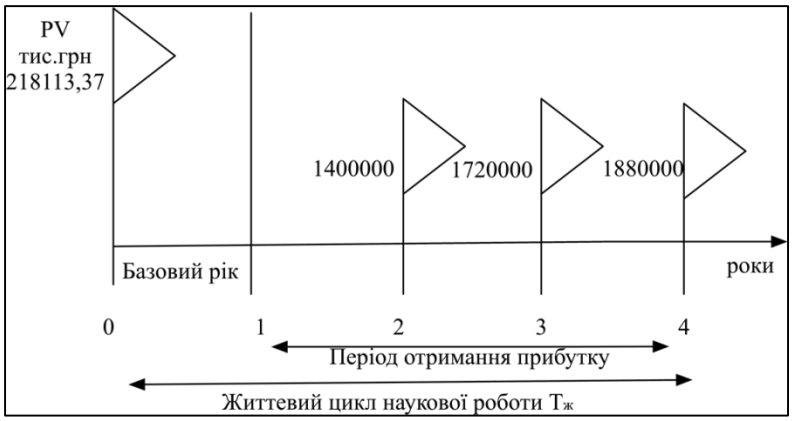


Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4-й крок. Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$. Якщо $E_{абс} \le 0$ то результати від проведення наукових досліджень та їх

впровадження буде збитковим і вкладати гроші в проведення цих досліджень ніхто не буде.

Для цього показника скористаємося формулою:

$$E_{абс} = (ПП - PV), \quad (4.11)$$

де ПП – вартість усього чистого прибутку, отриманого підприємством (організацією) від впровадження досягнень наукового розвитку, грн;

PV – теперішня вартість інвестицій PV = ЗВ = 218113,37 грн.

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.12)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки „0”.

$$ПП = \frac{1400000}{(1+0{,}1)^2} + \frac{1720000}{(1+0{,}1)^3} + \frac{1880000}{(1+0{,}1)^4}$$

$$= 1157024{,}79 + 1292261{,}46 + 1284065{,}3 = 3733351{,}55(грн),$$

$$E_{абс} = (3733351{,}55 - 218113{,}37) = 3951464{,}92\ (грн).$$

Оскільки $Е_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

5-й крок. Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$. Для цього використаємо формулу:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1, \quad (4.13)$$

де $Е_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV –теперішня вартість інвестицій PV = ЗВ, грн;

$Т_ж$ – життєвий цикл наукової розробки, роки.

Далі, розрахована величина $Е_в$ порівнюється з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau = d + f, \quad (4.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; d = 0,16;

f – показник, що характеризує ризикованість вкладень; f = 0,05.

Якщо величина $Е_в > \tau_{мін}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки. В іншому випадку фінансування наукової розробки здійснюватися не буде.

Спочатку спрогнозуємо величину $\tau_{мін}$. Припустимо, що за даних умов:

$$\tau_{мін} = 0,16 + 0,05 = 0,21, \quad (4.15)$$

Тоді відносна (річна) інвестиційна ефективність дослідження та впровадження його результатів буде:

$$E_в = \sqrt[4]{1 + \frac{3951464,92}{218113,37}} - 1 = \sqrt[4]{19,12} - 1 = 1,09 \text{ або } 109\%, \qquad (4.16)$$

Оскільки $E_в = 109\% > \tau_{мін} = 21\%$, то інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

6-й крок. Розраховують термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ОК}$ можна розрахувати за формулою:

$$T_{ОК} = \frac{1}{E_в}, \qquad (4.17)$$

Якщо $T_{ОК}$ < 3-х років, то фінансування даної наукової розробки в принципі є доцільним. В інших випадках потрібні додаткові розрахунки та обґрунтування.

Термін окупності становить:

$$T_{ОК} = \frac{1}{1,09} = 0,91, \qquad (4.18)$$

$T_{ОК}$ < 3-х років, що свідчить про доцільність фінансування даної наукової розробки.

4.5 Висновки

Рівень комерційного потенціал розробки інформаційної технології прогнозування енергоспоживання електрообладнання на основі інтернету речей є вище середнього. Показники ефективності показують, що даний метод є доцільним і буде цікавий для інвестора. Термін окупності розробленого проекту менше 3-х років, що підтверджує доцільність вкладання коштів в дану розробку.

ВИСНОВКИ

В результаті виконання магістерської кваліфікаційної роботи було вдосконалено інформаційну технологію прогнозування енергоспоживання електрообладнання будівлі з низьким енергоспоживанням.

В першому розділі було проаналізовано доцільність вдосконалення технології, було обґрунтовано залежність економічного зростання в країні від витрат електроенергії. Розглянуто важливість прогнозування енергії в країні за для зменшення її витрат. Переглянуті програми аналоги, для аналізу переваг та недоліків.

В другому розділі було проведено вагому кількість роботи, були переглянуті всі наявні вхідні дані, після виявлення багатьох значень рівних нулю було перевірено всі вхідні дані і ті які не мали вагомого значення на прогноз було видалено з вибірки, потім було проведено нормалізацію даних, порівняння з іншими методами прогнозування, обрано найкращі.

В третьому розділі було оглянуто обрані методи та проведено покращення трьох методів прогнозування та порівняно між собою, найкращим методом виявився метод k-найближчих сусідів його прогноз вдалось покращити на 9%.

В четвертому розділі було проведено оцінювання потенціалу роботи, було виявлено що розробка має рівень потенціалу вище середнього так як середньоарифметична сума балів 36,3. Проведено прогнозування витрат на розробку, також прогнозування збільшення чистого прибутку та розрахунок ефективності вкладених інвестицій та часу окупності. Було виявлено, що термін окупності проекту менше 3 років та підтверджує доцільність вкладання коштів.

За результатами магістерської кваліфікаційної роботи опубліковано тези доповіді на всеукраїнській науково-практичній інтернет-конференції студентів аспірантів та молодих науковців «Молодь в науці: дослідження,

проблеми, перспективи». Таким чином, завдання магістерської кваліфікаційної роботи було виконано в повному обсязі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Науменко Д.С. інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей/ Д. С. Науменко,// Всеукраїнська науково-практична Інтернет-конференція студентів, аспірантів та молодих науковців" Молодь в науці: дослідження, проблеми, перспективи (МН-2021)", Вінниця.: 01 – 14 травня 2021. – Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/viewFile/10999/9178

2. Датасет Каггл [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/loveall/appliances-energy-prediction/.

3. Селищев А. С. економіка в XXI столітті / А. С. Селищев, Н. А. Селищев. – СПб. : Пітер, 2004. – 240 с.

4. Анісімов А. Н. Економетричний аналіз розвитку енергетики / А. Н. Анісімов. – Наука, 1991. – 104 с.

5. Методи і моделі прогнозних взаємозв'язків енергетики та економіки / Ю. Д. Кононов, Е. В. Гальперова, Д. Ю. Кононов та ін. – Наука, 2009. – 178 с.

6. Нефтегазовая промисловість індустріально розвинених капіталістичних і країн, що розвиваються (1976 – 1985 рр.) : Довідник / Під. ред. М. С. Моделевского. – М. : Недра, 1988. – 174 с.

7. Новинний портал [Електронний ресурс] – Режим доступу до ресурсу: https://ecolog-ua.com/news/shcho-cogodni-stymulyuye-ukrayinski-promyslovi-pidpryyemstva-do-energoefektyvnosti/.

8. Вплив енергетики на економіку [Електронний ресурс] – Режим доступу до ресурсу: https://www.ukrinform.ua/rubric-economy/.

9. Softserve [Електронний ресурс] – Режим доступу до ресурсу: https://www.softserveinc.com/uk-ua/services/artificial-intelligence-machine-learning/.

10. Машинне навчання [Електронний ресурс] – Режим доступу до ресурсу: https://newdaycrypto.com/uk/machine-learning-methods-types-tasks-and-examples/.

11. Технологічні тренди 2020 [Електронний ресурс] – Режим доступу до ресурсу: https://www.makeitinua.com/posts-ua/tehnologichni-trendi-2020-roku-na-yaki-it-profesiyi-popit-zroste-shche-bilshe/.

12. Ноутбук Каггл [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/msand1984/appliance-energy-prediction /.

13. Профіль Каггл [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/msand1984/.

14. Каггл [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/msand1984/notebooks/.

15. NumPy [Електронний ресурс] – Режим доступу до ресурсу: https://numpy.org/doc/stable/about.html/.

16. NumPy [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/NumPy/.

17. Pandas [Електронний ресурс] – Режим доступу до ресурсу: https://khashtamov.com/ru/pandas-introduction/.

18. Seaborn [Електронний ресурс] – Режим доступу до ресурсу: https://seaborn.pydata.org/introduction.html/.

19. Scikit-learn [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Scikit-learn/.

20. Plotly [Електронний ресурс] – Режим доступу до ресурсу: https://plotly.com/python/getting-started/.

21. Kaggle [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Kaggle/.

22. Senior.ua [Електронний ресурс] – Режим доступу до ресурсу: https://senior.ua/articles/scho-take-kaggle-ta-chi-varto-vitrachati-na-nogo-chas/.

23. Pyprog [Електронний ресурс] – Режим доступу до ресурсу: https://pyprog.pro/sort/argsort.html/.

24. RandomForest [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Random_forest/.

25. Regressors [Електронний ресурс] – Режим доступу до ресурсу: https://towardsdatascience.com/an-intuitive-explanation-of-random-forest-and-extra-trees-classifiers-8507ac21d54b /.

26. RandomForestClassifier [Електронний ресурс] – Режим доступу до ресурсу: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

27. GridSearchCV [Електронний ресурс] – Режим доступу до ресурсу: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html/.

28. Stackoverflow [Електронний ресурс] – Режим доступу до ресурсу: https://stackoverflow.com/questions/29995249/verbose-argument-in-scikit-learn/.

29. Overfitting [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Overfitting /.

30. GridSearch [Електронний ресурс] – Режим доступу до ресурсу: https://towardsdatascience.com/machine-learning-gridsearchcv-randomizedsearchcv-d36b89231b10.

31. Найближчі сусіди [Електронний ресурс] – Режим доступу до ресурсу: https://scikit-learn.org/stable/modules/neighbors.html#nearest-neighbor-algorithms.

32. machine-learning [Електронний ресурс] – Режим доступу до ресурсу: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761/.

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет комп'ютерних систем і автоматики

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. В.Б. Мокін
(підпис)
"__" _____ 2020 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу

«СТВОРЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ ЕЛЕКТРООБЛАДНАННЯ НА ОСНОВІ ІНТЕРНЕТУ РЕЧЕЙ»

08–53.МКР.005.02.000.ТЗ

Керівник магістерської кваліфікаційної роботи
д.т.н., проф.
_____ О. Б. Мокін
(підпис)
"__" _____ 2020 р.

Розробив студент гр. 2ІСТ-19м
_____ Д.С. Науменко
(підпис)
"__" _____ 2020 р.

Вінниця 2020

1. Підстава для проведення робіт

Підставою для виконання роботи є наказ № __ по ВНТУ від «__» _____ 2020 р., та індивідуальне завдання на МКР, затверджене протоколом № __ засідання кафедри САІТ від «__» _____ 2020 р.

2. Джерела розробки:

- Kaggle [Електронний ресурс]. – Режим доступу до ресурсу: https://www.kaggle.com/
- Python [Електронний ресурс]. – Режим доступу до ресурсу: https://www.python.org/

3. Мета і призначення роботи

Розробка інформаційної технології прогнозування енергоспоживання.

4. Вихідні дані для проведення робіт:

- датасет Appliances Energy Prediction (Kaggle), який містить дані сенсорів будівлі з низьким енергоспоживанням, а також дані зовнішнього середовища, отримані через інтернет речей;

5. Методи дослідження:

– прогнозування за допомогою ExtraTreesRegressor;

6. Етапи роботи і терміни їх виконання

a) Аналіз предметної області ..... __. – __
b) Вибір інформаційної технології ..... __ – __
c) Вдосконалення інформаційної технології..... __ – __

7. Очікувані результати та порядок реалізації

Покращення прогнозу обраного методу.

8. Вимоги до розробленої документації

Пояснювальна записка оформлена у відповідності до вимог «Методичних вказівок до виконання та оформлення магістерських кваліфікаційних робіт для студентів спеціальності 126 – «Інформаційні системи та технології» денної форми навчання».

9. Порядок приймання роботи

Публічний захист……__.__.2020 р.
Початок розробки «___» _____2020 р.
Граничні терміни виконання МКР «___» _____2020 р.

Розробив студент групи 2ІСТ-19м _____ Науменко Д.С.

Додаток Б

Лістинг програми

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing, model_selection, metrics
import warnings
warnings.filterwarnings("ignore")

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will
list the files in the input directory

import os
print(os.listdir("../input"))
data = pd.read_csv("../input/KAG_energydata_complete.csv")
data.head()
data.info()
from sklearn.model_selection import train_test_split

# 75% of the data is usedfor the training of the models and the rest is used
for testing
train, test = train_test_split(data,test_size=0.25,random_state=40)
col_temp = ["T1","T2","T3","T4","T5","T6","T7","T8","T9"]

col_hum = ["RH_1","RH_2","RH_3","RH_4","RH_5","RH_6","RH_7","RH_8","RH_9"]

col_weather = ["T_out", "Tdewpoint","RH_out","Press_mm_hg",
               "Windspeed","Visibility"]
col_light = ["lights"]

col_randoms = ["rv1", "rv2"]

col_target = ["Appliances"]
feature_vars = train[col_temp + col_hum + col_weather + col_light +
col_randoms ]
```

```
target_vars = train[col_target]
feature_vars.describe()
import plotly.plotly as py
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
import plotly.graph_objs as go

# To understand the timeseries variation of the applaince energy consumption
visData = go.Scatter( x= data.date  ,  mode = "lines", y = data.Appliances )
layout = go.Layout(title = 'Appliance energy consumption measurement' ,
xaxis=dict(title='Date'), yaxis=dict(title='(Wh)'))
fig = go.Figure(data=[visData],layout=layout)

iplot(fig)
data['WEEKDAY'] = ((pd.to_datetime(data['date']).dt.dayofweek)// 5 ==
1).astype(float)
# There are 5472 weekend recordings
data['WEEKDAY'].value_counts()
temp_weekday =  data[data['WEEKDAY'] == 0]
# To understand the timeseries variation of the applaince energy consumption
visData = go.Scatter( x= temp_weekday.date  ,  mode = "lines", y =
temp_weekday.Appliances )
layout = go.Layout(title = 'Appliance energy consumption measurement on
weekdays' , xaxis=dict(title='Date'), yaxis=dict(title='(Wh)'))
fig = go.Figure(data=[visData],layout=layout)

iplot(fig)
temp_weekend =  data[data['WEEKDAY'] == 1]

# To understand the timeseries variation of the applaince energy consumption
visData = go.Scatter( x= temp_weekend.date  ,  mode = "lines", y =
temp_weekend.Appliances )
layout = go.Layout(title = 'Appliance energy consumption measurement on
weekend' , xaxis=dict(title='Date'), yaxis=dict(title='(Wh)'))
fig = go.Figure(data=[visData],layout=layout)

iplot(fig)
f, ax = plt.subplots(2,2,figsize=(12,8))
vis1 = sns.distplot(feature_vars["RH_6"],bins=10, ax= ax[0][0])
vis2 = sns.distplot(feature_vars["RH_out"],bins=10, ax=ax[0][1])
vis3 = sns.distplot(feature_vars["Visibility"],bins=10, ax=ax[1][0])
```

```
vis4 = sns.distplot(feature_vars["Windspeed"],bins=10, ax=ax[1][1])
f = plt.figure(figsize=(12,5))
plt.xlabel('Appliance consumption in Wh')
plt.ylabel('Frequency')
sns.distplot(target_vars , bins=10 ) ;
print('Percentage of the appliance consumption is less than 200 Wh')
print(((target_vars[target_vars <= 200].count()) / (len(target_vars)))*100 )
train_corr = train[col_temp + col_hum + col_weather +col_target+col_randoms]
corr = train_corr.corr()
# Mask the repeated values
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

f, ax = plt.subplots(figsize=(16, 14))
#Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, annot=True, fmt=".2f" , mask=mask,)
    #Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
    #Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
    #show plot
plt.show()
def get_redundant_pairs(df):
    '''Get diagonal and lower triangular pairs of correlation matrix'''
    pairs_to_drop = set()
    cols = df.columns
    for i in range(0, df.shape[1]):
        for j in range(0, i+1):
            pairs_to_drop.add((cols[i], cols[j]))
    return pairs_to_drop

# Function to get top correlations

def get_top_abs_correlations(df, n=5):
    au_corr = df.corr().abs().unstack()
    labels_to_drop = get_redundant_pairs(df)
    au_corr =
au_corr.drop(labels=labels_to_drop).sort_values(ascending=False)
    return au_corr[0:n]

print("Top Absolute Correlations")
```

```
print(get_top_abs_correlations(train_corr, 40))
train_X = train[feature_vars.columns]
train_y = train[target_vars.columns]
#Split testing dataset into independent and dependent varibales
test_X = test[feature_vars.columns]
test_y = test[target_vars.columns]
# Due to conlusion made above below columns are removed
train_X.drop(["rv1","rv2","Visibility","T6","T9"],axis=1 , inplace=True)
# Due to conlusion made above below columns are removed
test_X.drop(["rv1","rv2","Visibility","T6","T9"], axis=1, inplace=True)
train_X.columns
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()

train = train[list(train_X.columns.values) + col_target ]

test = test[list(test_X.columns.values) + col_target ]

# Create dummy test and training set to hold scaled values

sc_train = pd.DataFrame(columns=train.columns , index=train.index)

sc_train[sc_train.columns] = sc.fit_transform(train)

sc_test= pd.DataFrame(columns=test.columns , index=test.index)

sc_test[sc_test.columns] = sc.fit_transform(test)
sc_train.head()
train_X =  sc_train.drop(['Appliances'] , axis=1)
train_y = sc_train['Appliances']

test_X =  sc_test.drop(['Appliances'] , axis=1)
test_y = sc_test['Appliances']
from sklearn.linear_model import Ridge, Lasso
from sklearn.ensemble import RandomForestRegressor,
GradientBoostingRegressor, ExtraTreesRegressor
from sklearn.neural_network import MLPRegressor
import xgboost as xgb
from sklearn import neighbors
from sklearn.svm import SVR
models = [
```

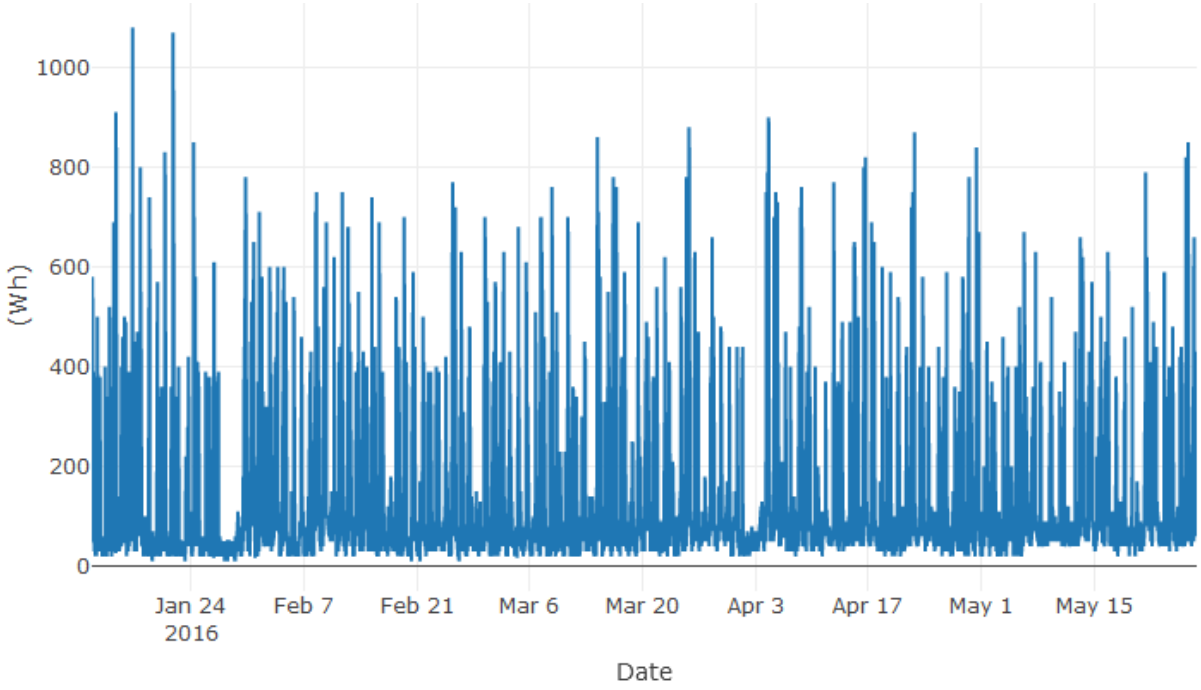
```
            ['Lasso: ', Lasso()],
            ['Ridge: ', Ridge()],
            ['KNeighborsRegressor: ',  neighbors.KNeighborsRegressor()],
            ['SVR:' , SVR(kernel='rbf')],
            ['RandomForest ',RandomForestRegressor()],
            ['ExtraTreeRegressor :',ExtraTreesRegressor()],
            ['GradientBoostingClassifier: ', GradientBoostingRegressor()] ,
            ['XGBRegressor: ', xgb.XGBRegressor()] ,
            ['MLPRegressor: ', MLPRegressor(  activation='relu',
solver='adam',learning_rate='adaptive',max_iter=1000,learning_rate_init=0.01,
alpha=0.01)]
         ]
# Run all the proposed models and update the information in a list model_data
import time
from math import sqrt
from sklearn.metrics import mean_squared_error

model_data = []
for name,curr_model in models :
    curr_model_data = {}
    curr_model.random_state = 78
    curr_model_data["Name"] = name
    start = time.time()
    curr_model.fit(train_X,train_y)
    end = time.time()
    curr_model_data["Train_Time"] = end - start
    curr_model_data["Train_R2_Score"] =
metrics.r2_score(train_y,curr_model.predict(train_X))
    curr_model_data["Test_R2_Score"] =
metrics.r2_score(test_y,curr_model.predict(test_X))
    curr_model_data["Test_RMSE_Score"] =
sqrt(mean_squared_error(test_y,curr_model.predict(test_X)))
    model_data.append(curr_model_data)
model_data
from sklearn.model_selection import GridSearchCV
param_grid = [{
              'n_estimators' : [300,350,400,450],
              'max_features': ["auto", "sqrt", "log2"]
            }]
reg = ExtraTreesRegressor(random_state=90)
# Застосуєм моделі пошуку сітки
```

```
grid_search = GridSearchCV(estimator = reg, param_grid = param_grid, cv = 5,
n_jobs = -1 , scoring='r2' , verbose=2)
grid_search.fit(train_X, train_y)
grid_search.best_params_
grid_search.best_estimator_
grid_search.best_estimator_.score(train_X,train_y) * 100
grid_search.best_estimator_.score(test_X,test_y) * 100
np.sqrt(mean_squared_error(test_y,
grid_search.best_estimator_.predict(test_X))) * 100
feature_indices =
np.argsort(grid_search.best_estimator_.feature_importances_)
importances = grid_search.best_estimator_.feature_importances_
indices = np.argsort(importances)[::-1]
names = [train_X.columns[i] for i in indices]
# Створюєм таблицю
plt.figure(figsize=(10,10))

# Заголовок таблиці
plt.title("Feature Importance")

# Додаєм стовбці
plt.bar(range(train_X.shape[1]), importances[indices])

# Додаєм імена стовбців по осі x
plt.xticks(range(train_X.shape[1]), names, rotation=90)

# Виводимо діаграму
plt.show()
```

Додаток В

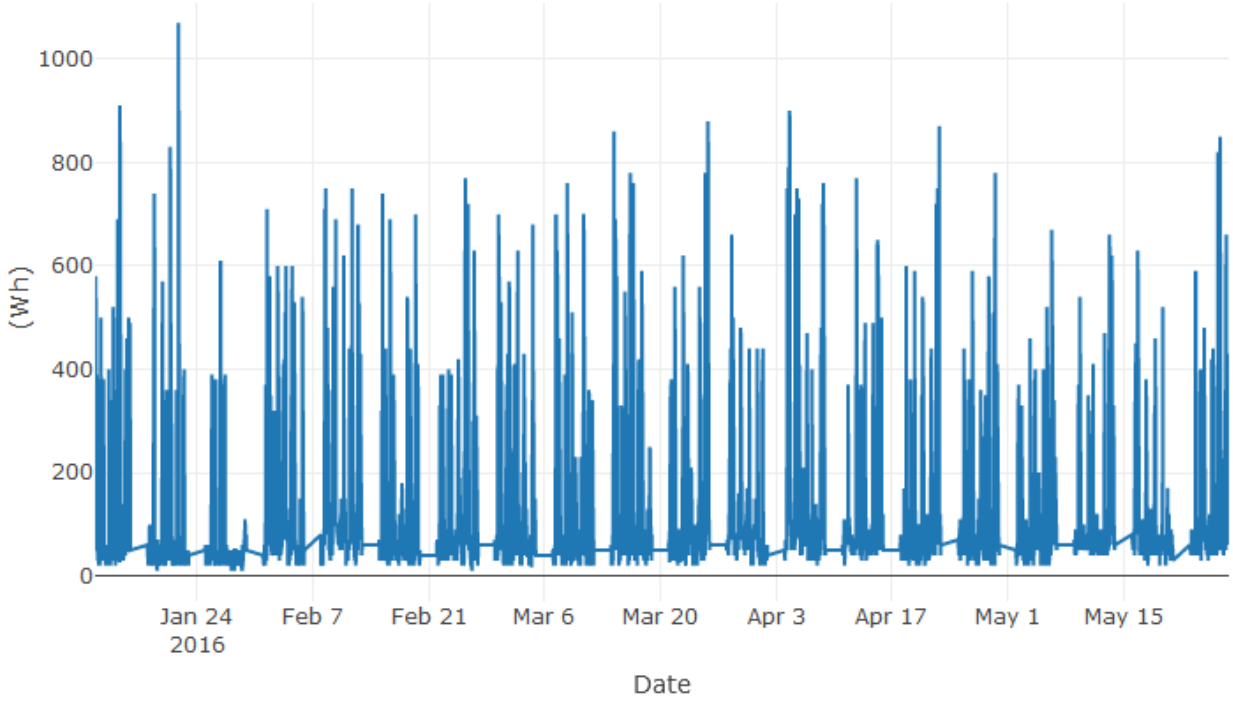
Графічна частина

Графік енергоспоживання приладів



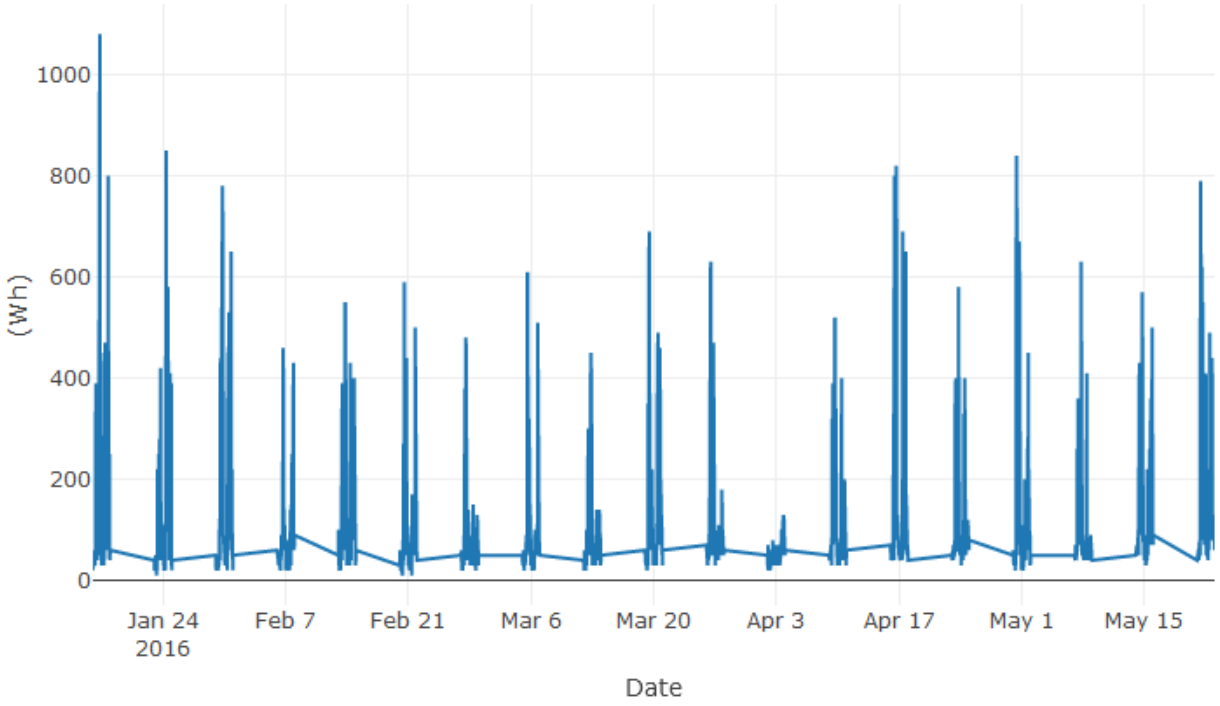
					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (1)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Графік енергоспоживання приладів	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Графік енергоспоживання будніх днів



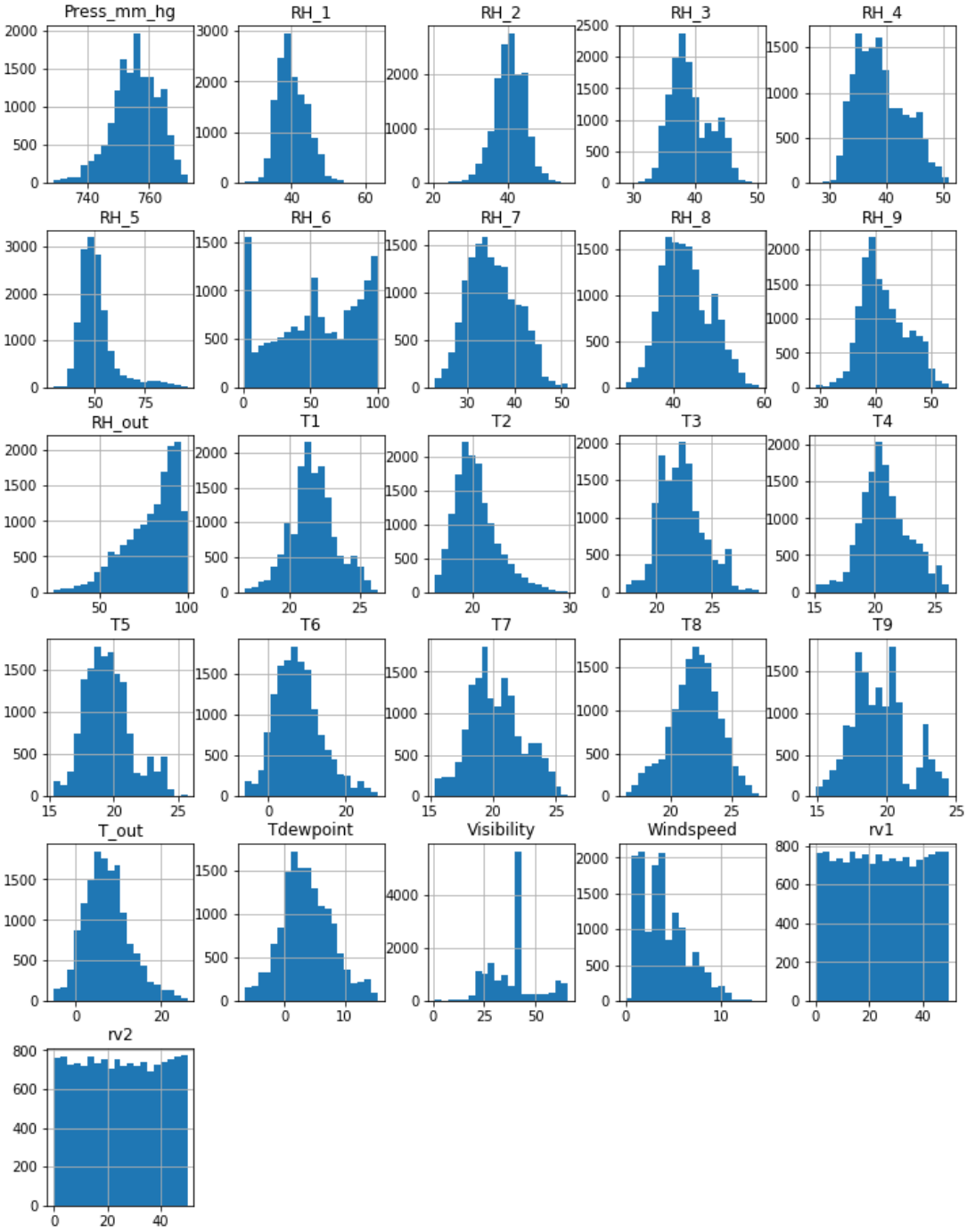
					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (2)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Графік енергоспоживання будніх днів	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Графік енергоспоживання вихідних днів



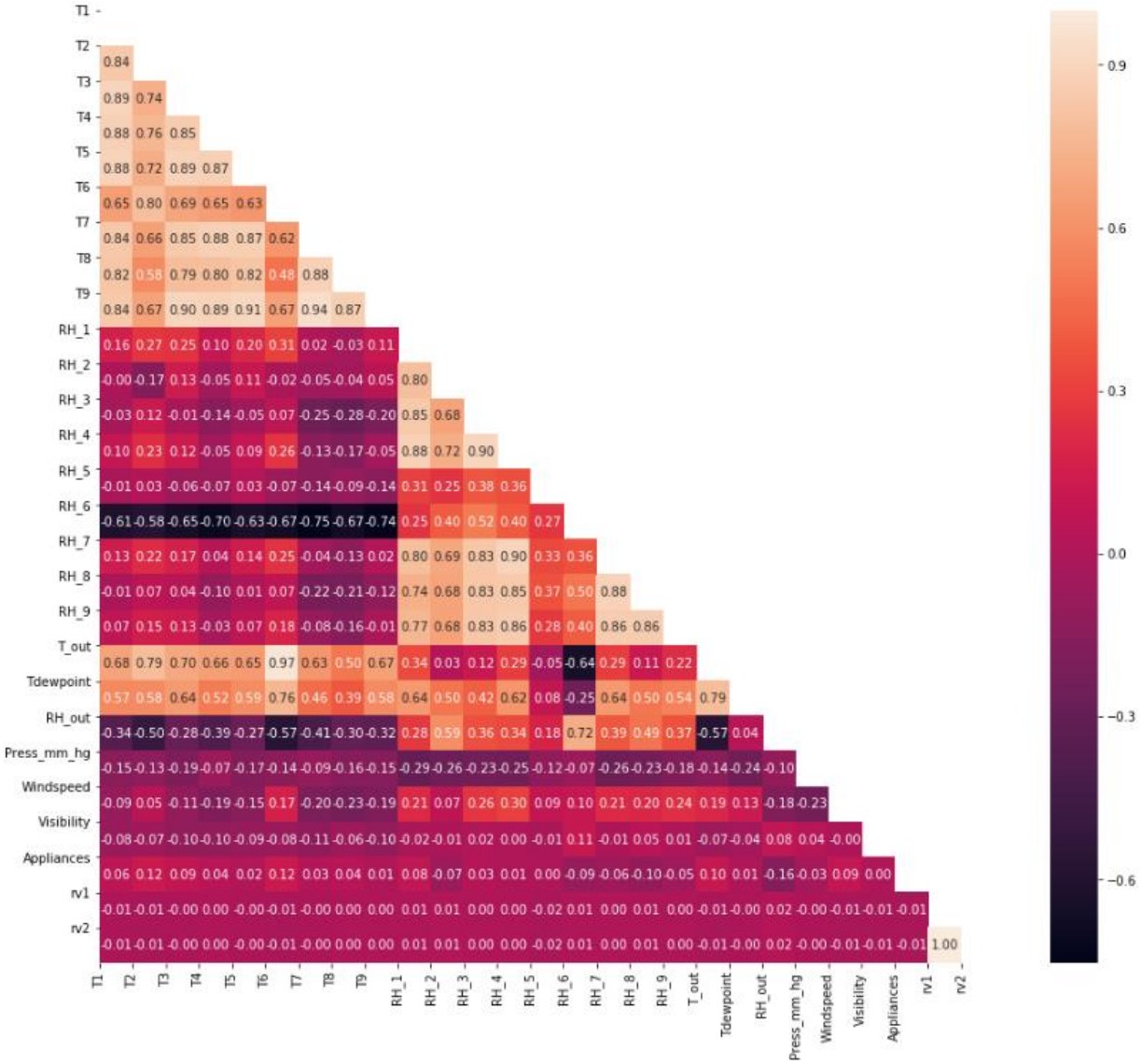
					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (3)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Графік енергоспоживання вихідних днів	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Графіки функцій вихідних даних



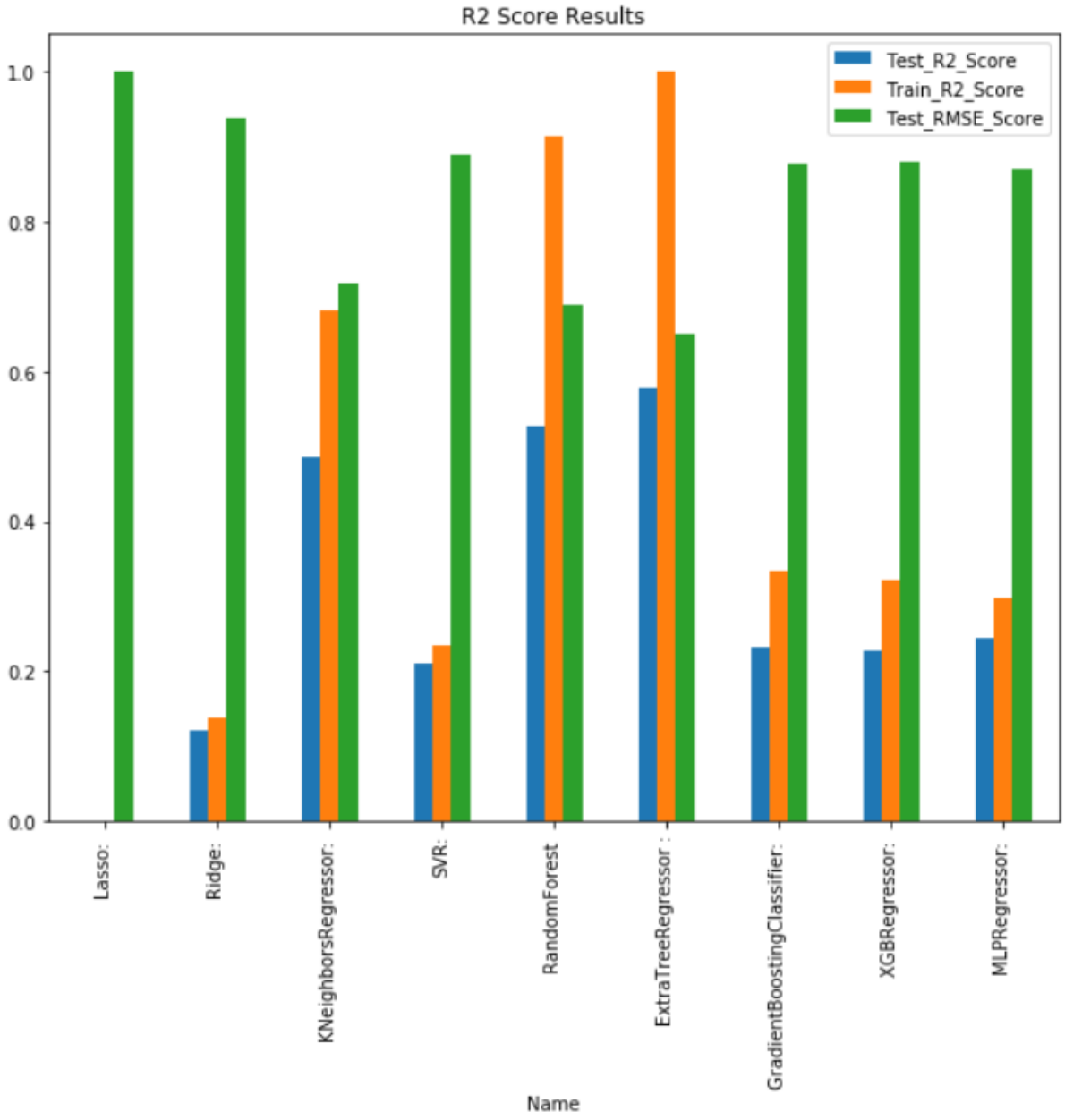
					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (4)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Графіки функцій вихідних даних	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Таблиця кореляції даних



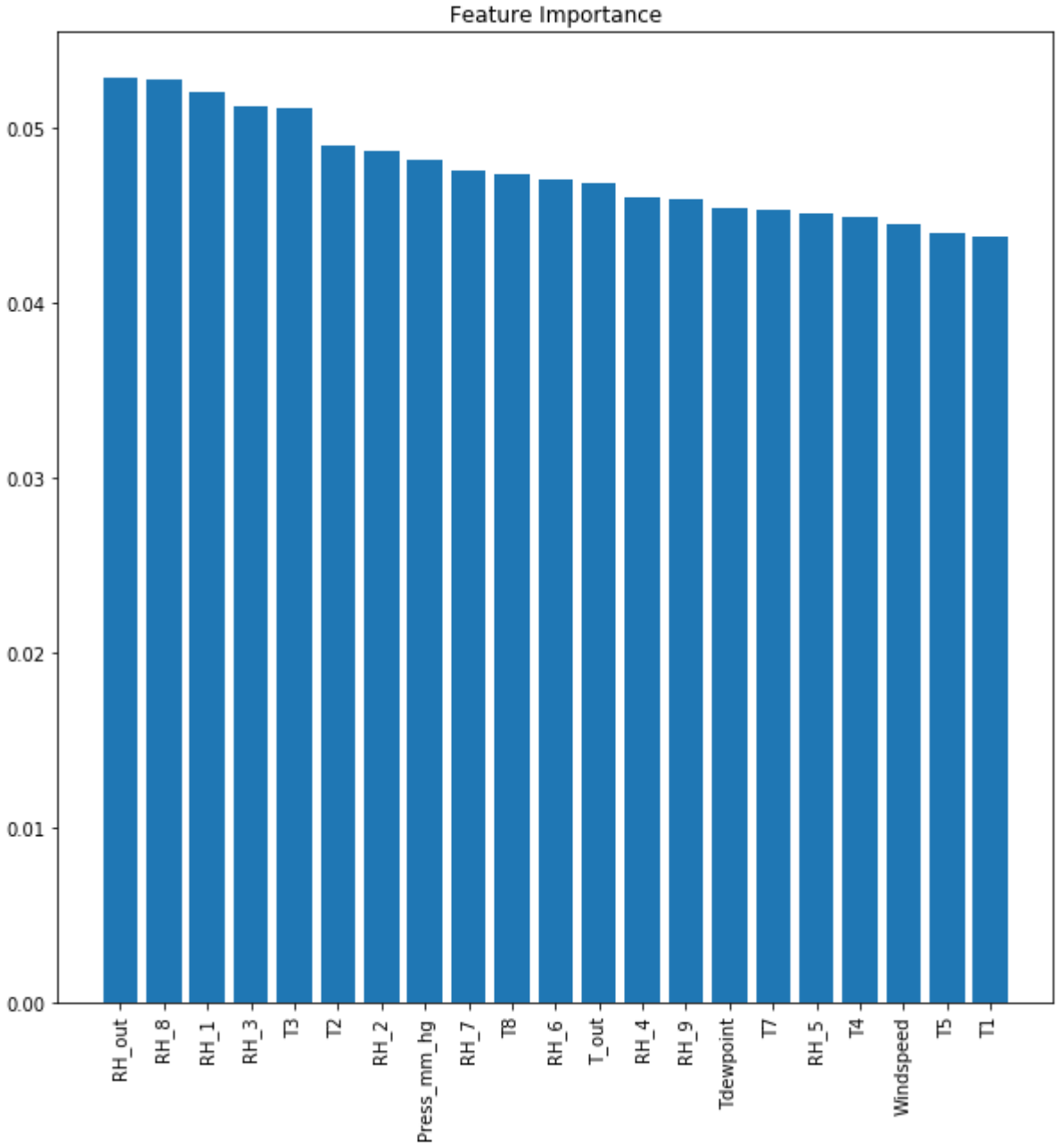
					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (5)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Таблиця кореляції даних	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Діаграма порівняння методів прогнозування



					08–53.МКР.005.02.00.ПЛ		
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (6)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Діаграма порівняння методів прогнозування	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Діаграма важливості функцій



					08–53.МКР.005.02.00.ПЛ			
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (7)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Діаграма важливості функцій	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						

Результат вдосконалення моделей

```
Training set R2 Score -  100.0
Testing set R2 Score -  68.48834662363605
Testing set RMSE Score -  60.42487350120308
```

Результат вдосконалення моделі ExtraTrees

```
Training set rfr R2 Score -  67.73102771588601
Testing set rfr R2 Score -  41.48186453266495
Testing set rfr RMSE Score -  76.49714731108281
```

Результат вдосконалення моделі RandomForest

```
Training set knr R2 Score -  73.23026930811155
Testing set knr R2 Score -  57.52448943875144
Testing set knr RMSE Score -  65.17323880339887
```

Результат вдосконалення моделі KNN

					08–53.МКР.005.02.00.ПЛ			
					Інформаційна технологія прогнозування енергоспоживання електрообладнання на основі інтернету речей	Літ.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата				1 : 1
Розробив		Науменко Д.С.				Аркуш (8)	Аркушів (8)	
Перевірив		Мокін О.Б.						
Рецензент		Бойко О. Р.						
					Результат вдосконалення моделей	2ІСТ-19м		
Н. Контр.		Жуков С.О.						
Зав. каф.		Мокін В.Б.						