

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,  
Факультет комп'ютерних наук і технологій  
(повне найменування інституту, факультету)

Кафедра програмних засобів.  
(повне найменування кафедри )

## Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ  
ТЕХНІЧНОГО ДІАГНОСТУВАННЯ ДЕТАЛЕЙ АВТОМОБІЛІВ. RESEARCH  
AND SOFTWARE IMPLEMENTATION OF METHODS FOR TECHNICAL  
DIAGNOSIS OF CAR COMPONENTS

Виконав: студент(ка) 2 курсу, групи КНТ-210м  
Спеціальності 122 Комп'ютерні науки  
(Код і найменування спеціальності)

Освітня програма (спеціалізація):

Системи штучного інтелекту

Ярмачек М. В.

(прізвище та ініціали)

Керівник Льовкін В.М.

(прізвище та ініціали)

Рецензент Скрупський С.Ю.

(прізвище та ініціали)

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  
**Національний університет «Запорізька політехніка»**  
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти магістр

Спеціальність 122 Комп'ютерні науки  
 (код і найменування)

Освітня програма (спеціалізація) Системи штучного інтелекту  
 (назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“      ”                      20   року

**З А В Д А Н Н Я**

**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)**

Ярмачека Максима Вадимовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів технічного діагностування деталей автомобілів. Research and Software Implementation of Methods for Technical Diagnosis of Car Components

керівник проєкту (роботи) Льовкін Валерій Миколайович,  
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “5” листопада 2021 року № 435

2. Строк подання студентом проєкту (роботи) 5 грудень 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи 3. Розробка архітектури програми. 4. Основні рішення щодо реалізації компонентів системи. 5. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)       

Слайди презентації

## 6. Консультанти розділів проєкту (роботи)

| Розділ              | Прізвище, ініціали та посада консультанта | Підпис, дата   |                           |
|---------------------|-------------------------------------------|----------------|---------------------------|
|                     |                                           | завдання видав | прийняв виконане завдання |
| 1-5 Основна частина | Льовкін В.М., к.т.н., доцент              |                |                           |
|                     |                                           |                |                           |
|                     |                                           |                |                           |
| Нормоконтроль       | Камінська Ж.К., асистент                  |                |                           |

7. Дата видачі завдання 10 вересня 2021 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                                  | Строк виконання етапів проєкту (роботи) | Примітка     |
|-------|---------------------------------------------------------------------------|-----------------------------------------|--------------|
| 1     | Постановка завдання роботи.                                               | 1 тиждень                               | Завдання, ТЗ |
| 2     | Аналіз предметної області.                                                | 2–3 тижні                               | Розділ 1     |
| 3     | Розробка та удосконалення методів, моделей й алгоритмів вирішення задачі. | 4–7 тижні                               | Розділ 2     |
| 4     | Розробка архітектури програми.                                            | 8 тиждень                               | Розділ 3     |
| 5     | Розробка програми.                                                        | 8–10 тижні                              | Розділ 4     |
| 6     | Експлуатація та експериментальне дослідження програмного забезпечення.    | 11 тиждень                              | Розділ 5     |
| 7     | Оформлення пояснювальної записки та документів до неї.                    | 12-13 тиждень                           | Додатки      |
| 8     | Нормоконтроль та рецензування.                                            | 14–15 тижні                             |              |
| 9     | Захист роботи.                                                            | 16 тиждень                              |              |

Студент (ка) \_\_\_\_\_ Ярмачек М. В.  
( підпис ) (прізвище та ініціали)

Керівник проєкту (роботи) \_\_\_\_\_ Льовкін В. М.  
( підпис ) (прізвище та ініціали)

## РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 142 с., 6 табл., 63 рис., 3 дод., 59 джерел.

РЕКУРЕНТНА НЕЙРОННА МЕРЕЖА, КОМП'ЮТЕРНА ПРОГРАМА, ВЕБЗАСТОСУНОК, АВТОМОБІЛЬ, ДІАГНОСТИКА, КОРОБКА ПЕРЕДАЧ, DJANGO, LSTM.

Об'єкт дослідження – процес технічного діагностування деталей автомобілів.

Предмет дослідження – нейронні мережі як метод технічного діагностування деталей автомобілів.

Мета роботи – створення програмного вебзастосунку технічного діагностування деталей автомобілів для спрощення та підвищення ефективності моніторингу стану автомобіля.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мови програмування Python, HTML, CSS та jQuery, CMS Django, система управління базою даних – PostgreSQL, Apple MacBook Pro 13'' з процесором M1, що базується на архітектурі ARM під управлінням операційної системи macOS Big Sur.

Результати. Розроблено метод технічного діагностування деталей автомобілів, що характеризується використанням моделі класифікації на основі нейронної мережі архітектури LSTM з визначенням спеціальної попередньої обробки даних, що дозволяє зменшити витрати на виявлення несправностей у коробці передач, включаючи часові та фінансові витрати.

Висновки. Розроблено вебзастосунок для моніторингу стану автомобіля та виявлення несправностей у коробці передач, який покращує та автоматизує процес догляду за автомобілем.

Галузь використання – спостереження за станом автомобіля власником або службою автомобільного сервісу.

## **ABSTRACT**

Explanatory note to the qualifying work of the bachelor: 142 pages, 6 tables, 63 figures, 3 appendixes, 59 sources.

RECURRENT NEURAL NETWORK, COMPUTER PROGRAM, WEB-APPLICATION, CAR, DIAGNOSTICS, TRANSMISSION, DJANGO, LSTM.

The object of research – the process of technical diagnostics of car parts.

The subject of research – neural networks as a method of technical diagnosis of car parts.

The purpose of the work – to create a software web application for technical diagnostics of car parts to simplify and increase the efficiency of vehicle condition monitoring.

Materials, methods and technical means: structural and object-oriented programming, Python programming languages, HTML, CSS and jQuery, CMS Django, database management system – PostgreSQL, Apple MacBook Pro 13" with M1 processor based on architecture ARM running the macOS Big Sur operating system.

Results. The method of technical diagnostics of car parts has been developed, characterized using a classification model based on the neural network architecture LSTM with the definition of special pre-processing, which reduces the cost of detecting faults in gear box, including time and money.

Conclusions. The web application has been developed to monitor the condition of the car and detect faults in the transmission, which improves and automates the process of car care.

Field of use – monitoring the condition of the car by the owner or service of the car service.

## ЗМІСТ

|                                                                                             |    |
|---------------------------------------------------------------------------------------------|----|
|                                                                                             | С. |
| Перелік скорочень та умовних позначок .....                                                 | 9  |
| Вступ.....                                                                                  | 10 |
| 1 Аналіз проблеми.....                                                                      | 12 |
| 1.1 Типи та принципи технічного діагностування автомобілів .....                            | 12 |
| 1.2. Огляд програмних засобів, що вирішують завдання діагностики<br>автомобіля.....         | 13 |
| 1.2.1 Огляд AVL DiTEST DIAGNOSE FAMILIE.....                                                | 13 |
| 1.2.2 Огляд AutoEnginuity's ScanTool.....                                                   | 16 |
| 1.2.3 Огляд OBD Auto Doctor.....                                                            | 17 |
| 1.2.4 Огляд EOBD Facile Source.....                                                         | 19 |
| 1.2.5 Огляд PCMScan .....                                                                   | 20 |
| 1.3 Загальні висновки з виконаного огляду засобів технічної діагностики<br>автомобілів..... | 22 |
| 1.4 Основні терміни проблемної області. ....                                                | 22 |
| 1.5 Огляд принципів діагностування стану деталей.....                                       | 24 |
| 1.6 Висновки за розділом 1 .....                                                            | 24 |
| 2 Матеріали і методи.....                                                                   | 25 |
| 2.1 Методи класифікації для технічної діагностики.....                                      | 25 |
| 2.1.1 Огляд методу класифікації Decision tree .....                                         | 25 |
| 2.1.2 Огляд алгоритму класифікації Deep Learning .....                                      | 27 |
| 2.1.3 Огляд алгоритму класифікації Support vector machines .....                            | 28 |
| 2.1.4 Огляд методу класифікації Naïve Bayes .....                                           | 29 |
| 2.1.5 Огляд методу класифікації Linear Discriminant Analysis .....                          | 31 |
| 2.1.6 Огляд методу класифікації K-Nearest Neighbors .....                                   | 32 |
| 2.1.7 Огляд методу класифікації Self-organizing maps .....                                  | 34 |
| 2.1.8 Огляд методу класифікації Growing neural gas .....                                    | 35 |
| 2.1.9 Огляд методу класифікації Random forest.....                                          | 37 |
| 2.2 Методи класифікації засновані на DL.....                                                | 38 |

|                                                                       |    |
|-----------------------------------------------------------------------|----|
| 2.2.1 Огляд RNN .....                                                 | 38 |
| 2.2.2 Огляд CNN .....                                                 | 41 |
| 2.2.3 Огляд LSTM.....                                                 | 44 |
| 2.3 Порівняння розглянутих методів для технічного діагностування..... | 49 |
| 2.4 Використані матеріали у роботі.....                               | 50 |
| 2.5 Запропонований метод діагностики стану КП.....                    | 53 |
| 2.6 Висновки за розділом 2 .....                                      | 57 |
| 3 Проєктування програмного забезпечення .....                         | 59 |
| 3.1 Вибір засобів реалізації програмного забезпечення.....            | 59 |
| 3.1.1 Вибір мови програмування .....                                  | 59 |
| 3.1.2 Вибір бібліотек Python для класифікації та прогнозування .....  | 61 |
| 3.1.3 Вибір засобів Python для попередньої обробки даних .....        | 62 |
| 3.1.4 Вибір фреймворку Python для створення вебзастосунку.....        | 63 |
| 3.1.5 Вибір СУБД .....                                                | 64 |
| 3.1.6 Вибір редактору коду.....                                       | 65 |
| 3.2 Функціональні вимоги до програмного забезпечення .....            | 66 |
| 3.3 Проєктування бази даних .....                                     | 67 |
| 3.4 Структура програмного забезпечення .....                          | 69 |
| 3.5 Висновки за розділом 3 .....                                      | 74 |
| 4 Основні рішення щодо реалізації компонентів системи.....            | 75 |
| 4.1 Алгоритм функціонування програмного забезпечення .....            | 75 |
| 4.2 Опис модулів реалізованої системи .....                           | 76 |
| 4.2.1 Опис модулю для роботи з БД.....                                | 76 |
| 4.2.2 Опис модулю для роботи з формами .....                          | 76 |
| 4.2.3 Опис модулю для роботи з view .....                             | 77 |
| 4.2.4 Опис модулю діагностування стану КП .....                       | 78 |
| 4.2.5 Опис модулю URL маршрутизації .....                             | 80 |
| 4.2.6 Опис модулю HTML шаблонів.....                                  | 81 |
| 4.3 Висновки за розділом 4 .....                                      | 83 |
| 5 Експлуатація та тестування програми .....                           | 84 |

|                                                            |     |
|------------------------------------------------------------|-----|
| 5.1 Призначення програми .....                             | 84  |
| 5.2 Умови виконання програми .....                         | 84  |
| 5.3 Виконання програми .....                               | 85  |
| 5.4 Експериментальне дослідження розробленого методу ..... | 92  |
| 5.5 Висновки за розділом 5 .....                           | 94  |
| Висновки .....                                             | 95  |
| Перелік джерел посилань .....                              | 97  |
| ДОДАТОК А Технічне завдання .....                          | 103 |
| ДОДАТОК Б Текст програми .....                             | 107 |
| ДОДАТОК В Слайди презентації .....                         | 133 |

**ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК**

|      |                                    |
|------|------------------------------------|
| ANN  | – Artificial Neural Networks;      |
| CMS  | – Content Management System;       |
| CNN  | – Convolutional Neural Networks;   |
| DL   | – Deep Learning;                   |
| DT   | – Decision tree;                   |
| GNG  | – Growing neural gas;              |
| KNN  | – K-Nearest Neighbors;             |
| LDA  | – Linear Discriminant Analysis;    |
| LSTM | – Long Short Term Memory Networks; |
| MVC  | – Model-View-Controller;           |
| NB   | – Naïve Bayes;                     |
| RNN  | – Recurrent Neural Networks;       |
| SOM  | – Self-organizing maps;            |
| SVM  | – Support vector machines.         |
| БД   | – база даних;                      |
| КП   | – коробка передач;                 |
| МН   | – машинне навчання;                |
| НМ   | – нейронна мережа;                 |
| ПЗ   | – програмне забезпечення;          |
| СУБД | – система управління базою даних;  |
| ТД   | – технічна діагностика;            |

## ВСТУП

Сьогодні світ навколо нас заповнений автомобілями, майже кожна родина має хоча б один автомобіль. Деякі сучасні бізнеси взагалі неможливі без автомобілів, наприклад галузь пасажироперевезення. Проте зараз наявна проблема відстеження стану автомобілів і в родинах, і в бізнесі. Здається, що сьогодні немає єдиної системи для моніторингу стану транспорту. Тому сьогодні зазвичай дуже складно змінити автомобільного техніка бо лише він знає повну історію ремонтів автомобіля, що у деяких випадках може допомогти швидко полагодити несправність.

Є також є інша проблема пов'язана з виявлення неполадок у автомобілі. На сьогоднішній день для того, щоб зрозуміти чи потрібен автомобілю ремонт зазвичай треба повністю або частково розібрати ту частину в, якій може бути несправність. Хоча треба зауважити, що сучасні виробники автомобілів намагаються виправити цю проблему, додаючи дуже багато датчиків у кожний транспортний засіб та вбудовуючи системи моніторингу та діагностування кожної підсистеми автомобіля, наприклад датчики тиску в шинах чи датчики встановленні на амортизаторах.

Проте жодна з цих компаній не зацікавлена в тому щоб їх засоби були універсальними і, як вже було сказано вище вам потрібен спеціальний сертифікований технік, який має доступ до програмного забезпечення компанії виробника автомобіля.

Тому розробка зручного застосунку для запису стану вашого автомобіля та збереження його історії ремонтів з можливістю швидкого виявлення несправності є актуальним запитом у суспільстві.

Таким чином метою цієї дипломної кваліфікаційної роботи є створення вебзастосунку для моніторингу стану автомобіля та виявлення неполадок у коробці передач за допомогою методів технічного діагностування, а саме за допомогою рекурентної нейронної мережі (НМ).

Об'єкт дослідження – процес технічного діагностування деталей автомобілів.

Предмет дослідження – нейронні мережі як метод технічного діагностування деталей автомобілів.

Мета роботи – створення програмного вебзастосунку технічного діагностування деталей автомобілів для спрощення та підвищення ефективності моніторингу стану автомобіля.

## **1 АНАЛІЗ ПРОБЛЕМИ**

### **1.1 Типи та принципи технічного діагностування автомобілів**

Технічна діагностика (ТД) – галузь науково-технічних знань, сутність яких складають теорію, методи та засоби пошуку та виявлення дефектів технічних об'єктів [1].

Сьогодні існує багато різних методів та принципів технічного діагностування будь-якого виробу. У цій роботі ми сфокусуємось саме на проблемах пов'язаних з діагностування неполадок в автомобілі, а саме у коробці передач (КП).

ТД надзвичайно корисна у системах активної та пасивної безпеки транспортних засобів, бо зазвичай дуже складно виявити проблему у на перший погляд правильно працюючому пристрої [2]. Проілюструємо це на предметі дослідження цієї роботи. Не кожна людина взагалі здатна почути, якісь дивні звуки в коробці передач, а при тому, що зараз виробники намагаються зробити так, щоб в салоні автомобіля взагалі не було чути ніяких зайвих звуків то виявити, якісь недоліки самостійно наразі майже неможливо. Але якщо відмовить саме коробка передач, яка є критичним елементом автомобіля, то водій ні в якому разі не зможе рухатися далі. ТД в свою чергу надає можливість виявити проблему до того, як вона стане критичною та призведе до поломки. Наприклад, можна зрозуміти, що шестерні, які використовуються в КП, вже мають декілька зламаних зубів, що може призвести до несправності та зробити обслуговування КП завчасним.

Тому сьогодні на ринку представлено багато різних систем діагностики автомобілів, які зазвичай є комплексними та базуються на вже встановлених датчиках в автомобілі. Також вони вимагають спеціального обладнання, що може зчитувати дані з комп'ютера автомобіля.

## **1.2. Огляд програмних засобів, що вирішують завдання діагностики автомобіля**

На ринку представлені системи ТД від звичайних пристроїв, що можуть розпізнати товщину фарби на кузові, тобто чи була автівка перекрашена чи ні, до великих комплексних систем, що можуть зчитувати дані напряду з автомобіля та показувати, чи система контролю автомобіля вважає, що з підсистемою все гаразд. Проте майже у всіх них немає самостійного модулю діагностування та вони не можуть видавати в якості результату вердикт, чи підсистема працює правильно чи ні.

У цій роботі ми розглянемо такі програмні засоби, як:

- AVL DiTEST DIAGNOSE FAMILIE [3, 4];
- AutoEnginuity's ScanTool [5];
- OBD Auto Doctor [6];
- EOBD Facile Source [7];
- PCMScan [8].

### **1.2.1 Огляд AVL DiTEST DIAGNOSE FAMILIE**

Програмне забезпечення AVL DiTEST DIAGNOSE FAMILIE представляє собою цілу лінійку різноманітних програмних та апаратних засобів для діагностики стану автомобіля, представляє засоби і для звичайних користувачів, і для сервісних центрів, наприклад, набір AVL DiTEST MDS DRIVE 188 позиціонується, як набір для звичайних користувачів. У набір входить кейс для переноски, ручний апарат для зчитування – VCI 1000, набір кабелів та ручний персональний комп'ютер (у вигляді планшету) з вже встановленим програмним забезпеченням виробника для діагностики [3].

Також виробник представляє рішення для професійних сервісних центрів у вигляді набору AVL DiTEST MDS 650, куди входить потужний стаціонарний персональний комп'ютер, набір різноманітних сканерів та систем діагностики.

Виробник надає великі можливості персоналізації саме цього набору, тому немає єдиного стандартного варіанту, що входить в цей набір, що в свою чергу надає велику гнучкість цій системі та можливість сформувати саме той варіант, який потрібен під конкретну задачу [3]. Наприклад, можна взяти компоненти, необхідні лише для діагностики великих вантажних автомобілів, та не використовувати жодних для діагностики легкових.

Як зауважено на сайті виробника, у службі технічного обслуговування або під час оцінки пошкодження всі системи, вбудовані в автомобіль, і їх пам'ять можна прочитати протягом декількох хвилин, щоб отримати швидкий огляд можливих технічних (проблемних) змін в автомобілі [4]. Щоденні роботи з технічного обслуговування та ремонту, необхідні в майстернях для легкових та комерційних транспортних засобів, можуть бути виконані надійно завдяки повній базі даних автомобілів, рішення також надає професійну інформаційну систему для отримання додаткової інформації про виробника [4].

На рис. 1.1 зображено вигляд повного мінімального набору діагностики.



Рисунок 1.1 – Набір AVL DiTEST MDS 650

На рис.1.2 зображено вигляд програмного забезпечення, що повинно використовуватись разом із системою.

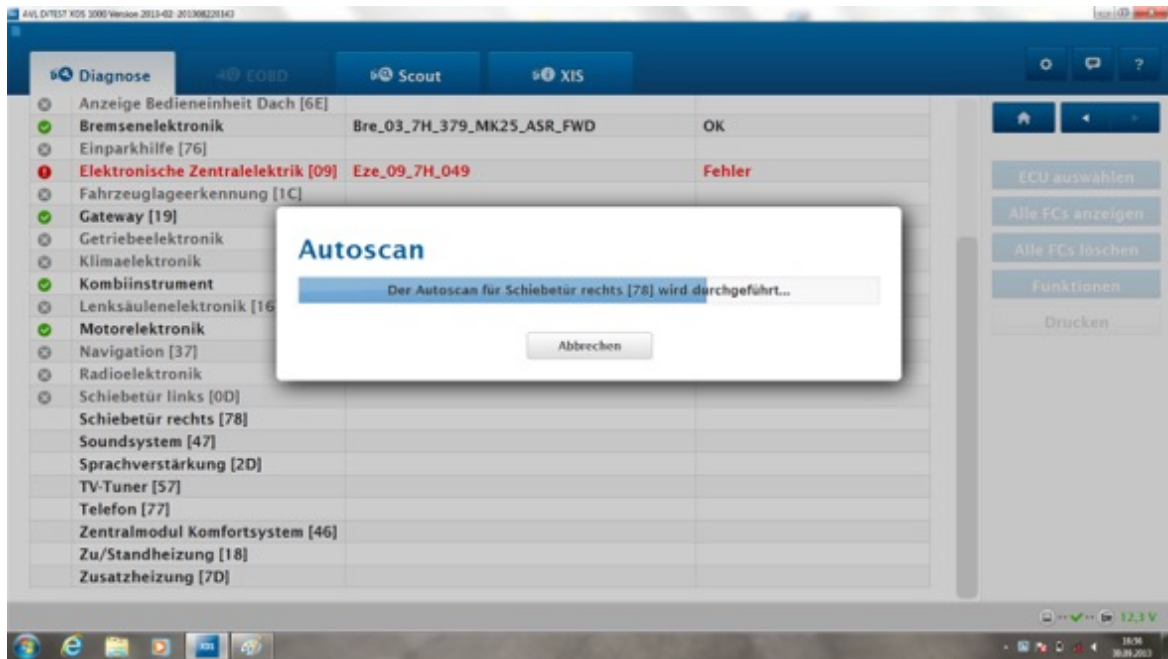


Рисунок 1.2 – Процес повного сканування автомобіля

З найважливіших систем діагностики компанія AVL виділяє такі:

- зчитування та видалення кодів несправності;
- зчитування даних автомобіля в режимі реального часу;
- тести систем автомобіля;
- параметризація систем;
- кодування;
- скидання налаштування служб до стану «за замовченням»;
- первинне налаштування систем [4].

Хоча рішення є безумовно комплексним та виконує майже всі необхідні задачі, воно не позбавлене недоліків та дуже сильно залежно від людського фактору. Людина без базових знань програмування та автомобільної механіки навряд зможе розібратися з усіма параметрами, що видає програмне забезпечення. Система не може працювати без експерта, який в свою чергу буде обробляти дані отримані від програмного забезпечення (ПЗ). Також одним з великих недоліків ПЗ є повна відсутність будь-яких рішень з боку системи, тобто система сама нічого не діагностує, а лише бере отримані від автомобілю дані та

вводить їх на екран, інколи структуруючи їх у вигляді графіку, якщо це можливо для обраного типу даних.

### 1.2.2 Огляд AutoEnginuity's ScanTool

Giotto ScanTool від AutoEnginuity – це потужна діагностична платформа для комп'ютера на базі Windows, також застосунок пропонує професійні варіанти обробки даних машини для 58 автовиробників [5]. Завдяки розширеним додаткам користувач можете отримати доступ до систем ABS, подушок безпеки, проводки, комбінації приладів, трансмісії та десятків інших контролерів кузова та шасі, глибина охоплення складається з даних в реальному часі, двонапрямого керування, адаптивного скидання та навчання, системних тестів, а також пошуку та очищення кодів несправностей [5].

Застосунок також має апаратну та програмну частину. Апаратна частина також постачається у кейсі, та має у комплекті прилад OBD-II для взаємодії з інтерфейсом автомобіля, набір кабелів, накопичувач USB 2.0 з встановленим ПЗ виробника для діагностика автомобіля.

Серед переваг розробник виділяє такі можливості:

- автоматичне розшифрування VIN-коду;
- зчитування та очищення кодів несправностей з усього автомобіля за один крок;
- друк оперативних даних або звітів про DTC/Викиди за один крок;
- швидке встановлення за допомогою справжнього накопичувача USB Plug and Play (драйвери пристроїв не потрібні) [5].

AutoEnginuity's ScanTool має широкі можливості діагностування у реальному часі, приклад яких зображено на рис. 1.3.

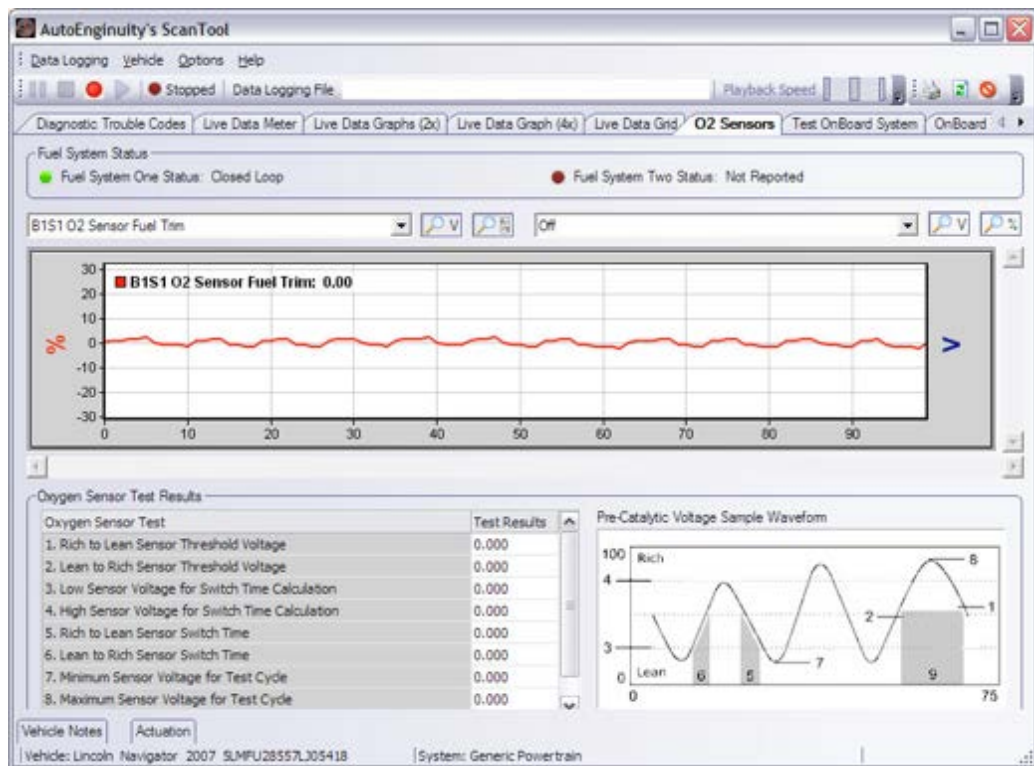


Рисунок 1.3 – Вид екрану перегляду даних датчика кисню та результати тесту у реальному часі

Великою перевагою саме цієї системи є абсолютна портативність та можливість збереження усіх тестових даних для подальшого діагностування. Також слід відмітити, що застосунок потребує дуже мало обчислювальних потужностей комп'ютера та всього 100 Мб оперативної пам'яті.

Проте недоліком є те, що стан автомобіль можливо відстежити лише тут та зараз, і за допомогою застосунку неможливо перевірити, які роботи проводились з автомобілем до моменту перевірки, також треба відмітити, що це ПЗ працює лише з операційною системою Windows.

### 1.2.3 Огляд OBD Auto Doctor

OBD Auto Doctor — це передове програмне забезпечення OBD-2 для Windows, Mac і Linux, ПЗ дає змогу зв'язуватися з бортовою системою діагностики вашого автомобіля та перетворити комп'ютер у високофункціональний автомобільний сканер [6].

Цей додаток є лише ПЗ, тобто без апаратної частини. Користувачу самому треба мати комп'ютер та спеціалізовані коннектори для підключення комп'ютера до автомобіля.

Основною відмінністю цього додатка є мультиплатформовість та досить сучасний дизайн клієнту, також серед переваг можна відмітити досить низьку вартість продукту, він розповсюджується за підпискою з фіксованою вартістю на рік. Мала вартість продукту досягається шляхом не постачання апаратних частин, необхідних для з'єднання автомобіля та комп'ютера. Проте, як зауважено на сайті розробника, якщо автомобіль обладнаний вбудованою системою Bluetooth то вони не потрібні [6].

У якості прикладу роботи з додатком можна навести відмінну від інших ПЗ сучасну та гнучку систему моніторингу датчиків у реальному часі, яку зображено на рис. 1.4.

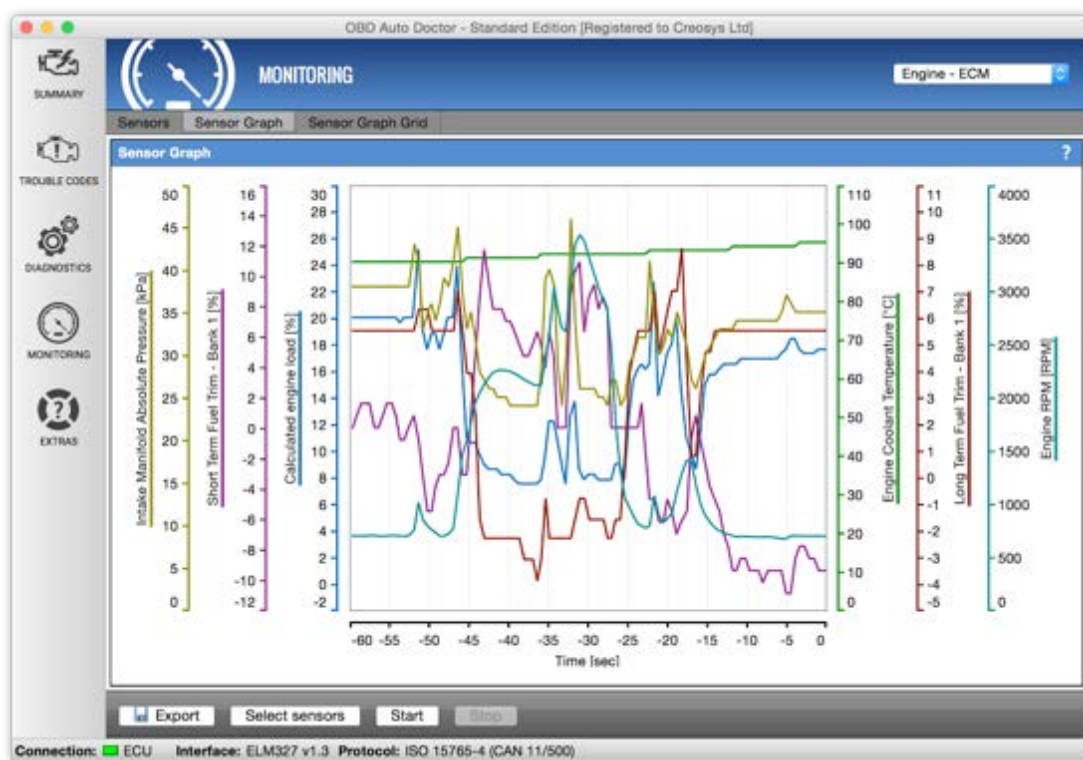


Рисунок 1.4 – Вид екрану перегляду даних датчика кисню та результати тесту у реальному часі

У всіх інших аспектах цей застосунок майже повністю аналогічний попереднім додаткам, та може впроваджувати лише ті тести, які були внесені виробником в пам'ять автомобіля, що перевіряється.

#### **1.2.4 Огляд EOBD Facile Source**

EOBD Facile Source постачається роздільно з апаратної частиною, але для його правильної роботи треба, щоб були обидві частини. Тобто треба купити апаратну частину «Klavkart OBD2 scanner», а потім докупити програмну частину. Проте слід також відмітити, що розробник надає можливість користуватися безкоштовною версією з дуже урізаною функціональністю та не платити за програмну частину [7].

Серед переваг саме цього продукту можна відмітити те, що ПЗ може працювати не лише на комп'ютерах, а ще й на смартфонах під управлінням iOS або Android [7]. У роботі зі смартфонами є свої обмеження наприклад робота під управлінням iOS можлива лише під час WiFi підключення, що зазвичай призводить до унеможливлення сканування датчиків у реальному часі, під час руху автомобіля. Також великим недоліком є несумісність з більш старими портами в автомобілі, а саме з портом OBD та старими інтерфейсами порта другої версії.

Безкоштовна версія додатку не дозволяє дуже багато речей, наприклад, унеможливорює проведення тестів виробника та надає лише один графік даних в реальному часі. Тобто для повноцінного діагностування та роботи з автомобілем за допомогою цього застосунку все ж таки прийдеться купувати повну версію. Проте додаток розповсюджується за системою купив один раз та працюй, але буде працювати лише на одному обраному пристрої з можливістю зміни через 2 роки [7].

Вид програми на смартфоні під управлінням операційної системи iOS зображено на рис. 1.5.



Рисунок 1.5 – Меню діагностики у застосунку EOBDFacile Source

Не дивлячись на вагомі переваги цей застосунок страждає від тих самих проблем, що і його аналоги та фактично виступає засобом вивантаження даних з автомобіля.

### 1.2.5 Огляд PCMScan

PCMSCAN — це повнофункціональний загальний сканер OBD-II та діагностичний інструмент, який підтримує широкий спектр апаратних інтерфейсів OBD2, застосунок дозволяє переглядати, складати діаграми, реєструвати та відтворювати діагностичні дані в режимі реального часу через порт діагностичних даних OBD2 автомобіля, також дозволяє переглядати діагностичні коди несправностей транспортних засобів, дані стоп-кадри та іншу інформацію про автомобіль [8].

Вид застосунку зображено на рис. 1.6.



Рисунок 1.6 – Відтворення записаних даних автомобіля у застосунку PCMScan

Серед найважливіших переваг застосунку є майже повна сумісність з усіма автомобілями, що мають OBD порт, також слід зауважити, що це ПЗ має можливість відтворювати залоговані дані у псевдореальному часі для діагностування після проведення тестів. Серед переваг також варто відмітити занадто низькі системні вимоги.

Серед недоліків треба відзначити те, що застосунок працює лише під операційною системою Windows, та має занадто старий дизайн. Також можна зауважити, що додаток не позбавлений проблем його аналогів, хоча і має вбудовані елементи діагностування, проте також покладається на системні тести виробника автомобіля.

### **1.3 Загальні висновки з виконаного огляду засобів технічної діагностики автомобілів**

Можна побачити, що розглянуті вище програмні засоби хоча і позиціонують себе, як засоби технічного діагностування, насправді такими не являються бо лише видають результати тестів створених виробником автомобіля. Для подальшої діагностики вони, зазвичай, надають змогу записати зчитані дані у файл типу .csv та передати його на подальше діагностування, що у свою чергу, зазвичай, передбачає наявність людини-експерта, яка розбирається у конкретному аспекті діагностика автомобіля.

Наразі лише декілька автомобільних концернів впровадили, або лише починають впроваджувати технічне діагностування стану автомобіля у реальному часі за допомогою нейронних мереж, а не за допомогою звичайних тестів та перевірки відхилення від норми.

У якості прикладу такої компанії можна привести компанію Tesla, яка за рахунок повної діджеталізації електромобіля збирає багато даних про усі його підсистеми та надає змогу користувачу автомобіля не виходячи з нього отримати повні дані використання автомобіля та дані про стан кожної з підсистем автомобіля [9]. Також надається можливість майже повного налаштування підсистем згідно з вимогами користувача, автомобіль сам попереджує про можливі майбутні поломки та запрошує володаря відвідати сервісний центр [9].

### **1.4 Основні терміни проблемної області.**

Діагностика – галузь знань, що досліджує технічний стан об'єктів діагностування і прояв технічних станів, розробляючи методи їх визначення, а також принципи побудови і організацію використання систем діагностування [15].

Об'єкт технічного діагностування – технічний об'єкт, стан якого визначають без його розбирання, або з частковим підрозбиранням [15].

Технічний стан об'єкту діагностування – сукупність схильних до зміни в процесі виробництва або експлуатації властивостей об'єкту, що характеризується в певний момент часу ознаками, встановленими технічною документацією на цей об'єкт [15].

Видами технічного стану є, наприклад, справне, працездатне, несправне, непрацездатне і так далі. До чинників, під впливом яких змінюється технічний стан об'єкту, можна віднести дії кліматичних умов, старіння протягом часу експлуатації і т. д. Про зміну технічного стану об'єкту судять по значеннях діагностичних (контрольованих) параметрів, що дозволяють визначити технічний стан об'єкту без його розбирання [15].

Контроль технічного стану – перевірка відповідності значень параметрів об'єкту вимогам технічної документації і визначення на цій основі одного із заданих видів технічного стану в даний момент часу [15].

Параметр технічного стану – фізична величина, що характеризує працездатність або справність об'єкту діагностування, що змінюється в процесі роботи [15].

Діагностичний параметр – параметр об'єкту діагностування, використовуваний в установленому порядку для визначення технічного стану об'єкту діагностування [15].

Алгоритм діагностування (контролю) встановлює склад і порядок проведення елементарних перевірок об'єкту і правила аналізу їх результатів. Елементарна перевірка визначається робочою або тестовою дією, що поступає або подається на об'єкт, а також складом ознак і параметрів, що створюють відповідь об'єкту на відповідну дію [15].

Діагностична модель – формалізований опис об'єкту, необхідний для вирішення завдань діагностування [15].

### 1.5 Огляд принципів діагностування стану деталей

Як зауважено у науковій статті [10], задача діагностика зводиться до визначення стану об'єкту діагностування. Що в свою чергу зводиться до вирішення задачі класифікації. Якщо звертатись до одного з пунктів мети цієї роботи, а саме до задачі класифікації стану коробки передач, то потрібно вирішити завдання, чи коробка повністю працездатна, чи в ній наявні зламані зубці на шестерні.

Ця задача в свою чергу може бути вирішена багатьма різними алгоритмами, наприклад, K-Nearest Neighbors(KNN) або алгоритм Random forest, також є можливість використовувати нейронні мережі глибинного навчання та інші методи.

### 1.6 Висновки за розділом 1

У першому розділі було проведено аналіз предметної області роботи, розглянуто програми аналоги для технічної діагностики автомобілів, були зроблені висновки про функціональність та працездатність розглянутих ПЗ.

Для досягнення поставленої в роботі мети було визначено наступні завдання:

- визначення вимог до програмного забезпечення;
- вибір мови і засобів розробки;
- проектування програмного забезпечення;
- реалізація програмного забезпечення;
- експериментальне дослідження програмного забезпечення.

## 2 МАТЕРІАЛИ І МЕТОДИ

### 2.1 Методи класифікації для технічної діагностики

Як вже було зауважено в першому розділі, методів класифікації є велика кількість, проте не кожен з них підходить задля вирішення проблеми цієї роботи. У цьому розділі буде розглянуто найпопулярніші методи класифікації даних та буде проведено опис первинних матеріалів, використаних у роботі.

Серед методів класифікації буде розглянуто наступні:

- дерево рішень (Decision tree);
- нейронні мережі (Artificial neural networks);
- опорні векторні матриці (Support vector machines);
- наївний Байєс (Naïve Bayes);
- лінійний дискримінантний аналіз (Linear discriminant analysis);
- К-найближчих сусідів (K-Nearest Neighbours);
- карти, що самоорганізуються (Self-organizing maps);
- нервовий газ, що розширюється (Growing neural gas);
- випадкові ліси (Random forest).

#### 2.1.1 Огляд методу класифікації Decision tree

Decision tree (DT) – це метод класифікації з вчителем, який створює модель, що передбачає значення цільової змінної залежно від вхідних параметрів [11]. Різні дослідники з різних галузей розглядали проблему розширення дерева рішень з доступних даних таких, як машинне навчання, розпізнавання образів і так далі [18]. DT – це послідовна модель, яка ефективно та злагоджено об'єднує серію базових тестів, де числова ознака порівнюється з пороговим значенням у кожному тесті [19].

DT – це модель класифікації, яка зазвичай використовується в аналізі та класифікації даних [20]. Кожен вузол представляє ознаки в категорії, що підлягає класифікації, і кожна гілка визначає значення, яке в теорії може прийняти

вузол [21, 22]. Завдяки простому аналізу та точності у багатьох формах даних, дерева рішень знайшли багато полів реалізації [23].

Схематичний приклад роботи дерева зображено на рис. 2.1.

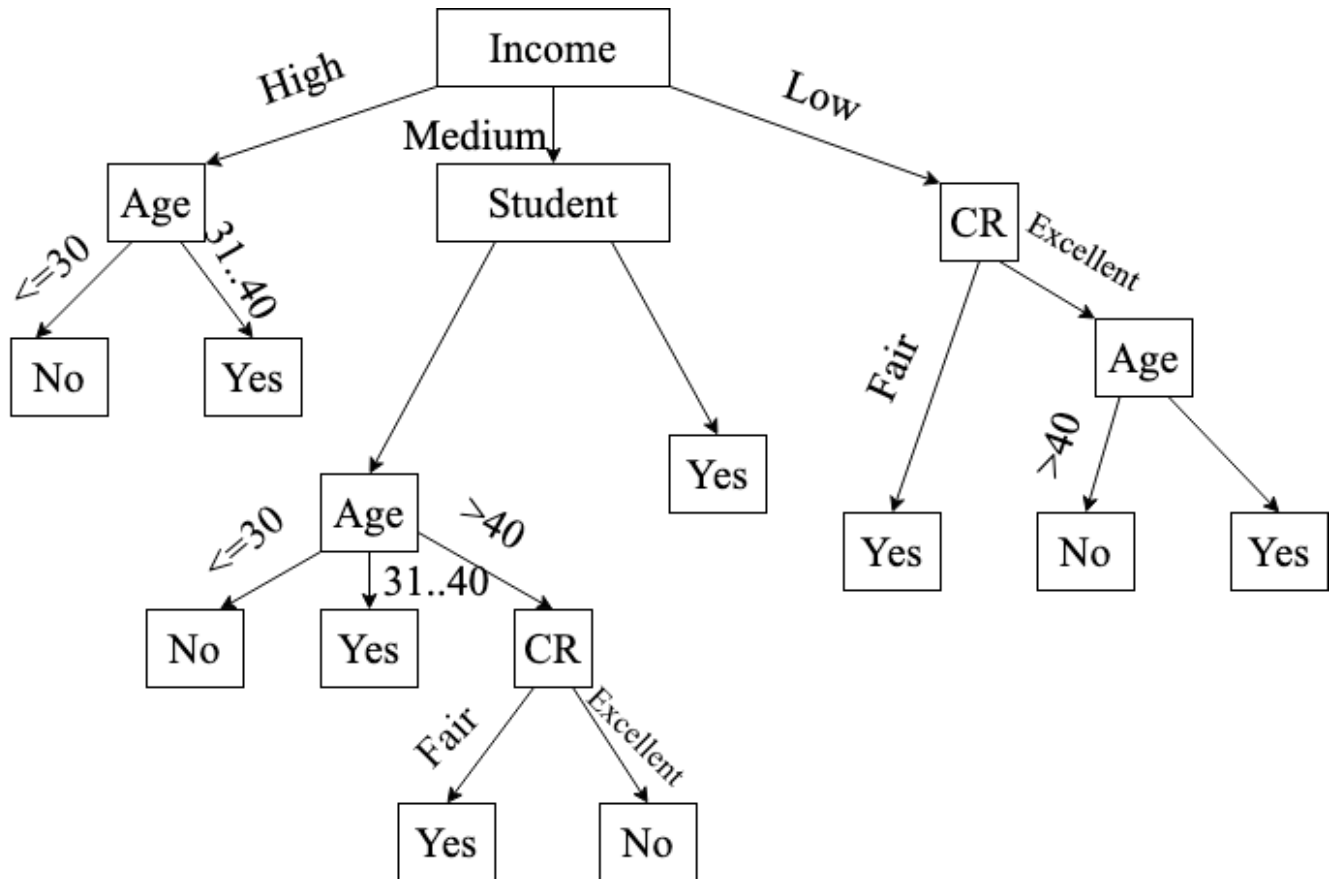


Рисунок 2.1 – Приклад роботи алгоритму DT

Переваги:

- потребує набагато менше часу на підготовку даних (не потребує нормалізації та масштабування);
- відсутні значення в даних не мають значного впливу на процес побудови дерева рішень;
- модель дуже інтуїтивна та легка для перевірки людиною експертом [24].

Недоліки:

- маленька зміна в даних може викликати великі зміни в структурі дерева;

- для деяких задач потребується набагато більше обчислювальних потужностей, якщо порівнювати з іншими алгоритмами класифікація;
- як правило потребують більше часу для навчання моделі ніж інші методи;
- алгоритм немає можливостей до застосування регресії чи прогнозування даних у реальному часі [24].

### 2.1.2 Огляд алгоритму класифікації Deep Learning

Алгоритм відомий під назвою Artificial Neural Networks (ANN) або Deep Learning (DL), є іншим алгоритмом для навчання з вчителем. ANN, він був натхненний дослідженнями системи мозку людини, метод фактично натхненний біологічними системами, здатними виявляти закономірності, і в результаті може робити прогнози [11]. Завдяки своїй особливій здатності до навчання DL здатний витягувати важливі зв'язки або закономірності з аналізу неповних, складних і, можливо, неточних даних, які не можуть бути виявлені іншими обчислювальними методами [11]. Архітектура DL складається з вхідного рівня, прихованих шарів і вихідних шарів, застосування показують, що DL має більшу точність прогнозування, ніж інші методи або люди-експерти для деяких проблем із даними [11]. Метод також має можливість швидкої підгонки нелінійних даних, тому може вирішувати багато проблем, складних для інших методів, проте алгоритм DL розглядається, як чорний ящик, це в свою чергу означає, що його прогнози неможливо пояснити та перевірити хід рішення людиною експертом, це першочергово пов'язано зі складністю архітектури алгоритму.

Переваги:

- один і той самий підхід на основі нейронної мережі можна застосувати до багатьох різних програм і типів даних;
- не потрібно завчасно визначати функції алгоритму;
- забезпечує кращі результати прогнозування ніж усі інші алгоритми, якщо обсяг навчальних даних величезний [13].

Недоліки:

- потребує дуже великої кількості даних, щоб працювати краще, ніж інші методи;
- не існує стандартної теорії, яка б направляла вас у виборі правильних інструментів для глибокого навчання;
- система працює як чорний ящик, тобто невідомо, що відбувається на прихованих шарах [14].

Нейронні мережі у роботі будуть розглянуто детальніше у наступному підрозділі, через їх велику варіативність та специфічність для вирішення конкретних задач.

### **2.1.3 Огляд алгоритму класифікації Support vector machines**

Support vector machines (SVM) – це набір пов’язаних методів навчання з вчителем, які використовуються для вирішення задач класифікації та регресії [25]. Однак алгоритм переважно використовується лише в задачах класифікації [26]. В алгоритмі SVM кожен елемент даних зображується, як точка в  $n$ -вимірному просторі (де  $n$  – кількість потрібних ознак об’єкту), при цьому значення кожної ознаки є значенням певної координати [26]. Потім виконується класифікація, знаходячи гіперплощину, яка краще за всіх розрізняє два чи більше різних класи, як видно на рис. 2.2.

Роздільна гіперплощина – це гіперплощина, яка максимізує відстань між двома паралельними гіперплощинами класів, роблячи припущення, що чим більше відстань між цими паралельними гіперплощинами, тим меншою буде похибка узагальнення класифікатора [27].

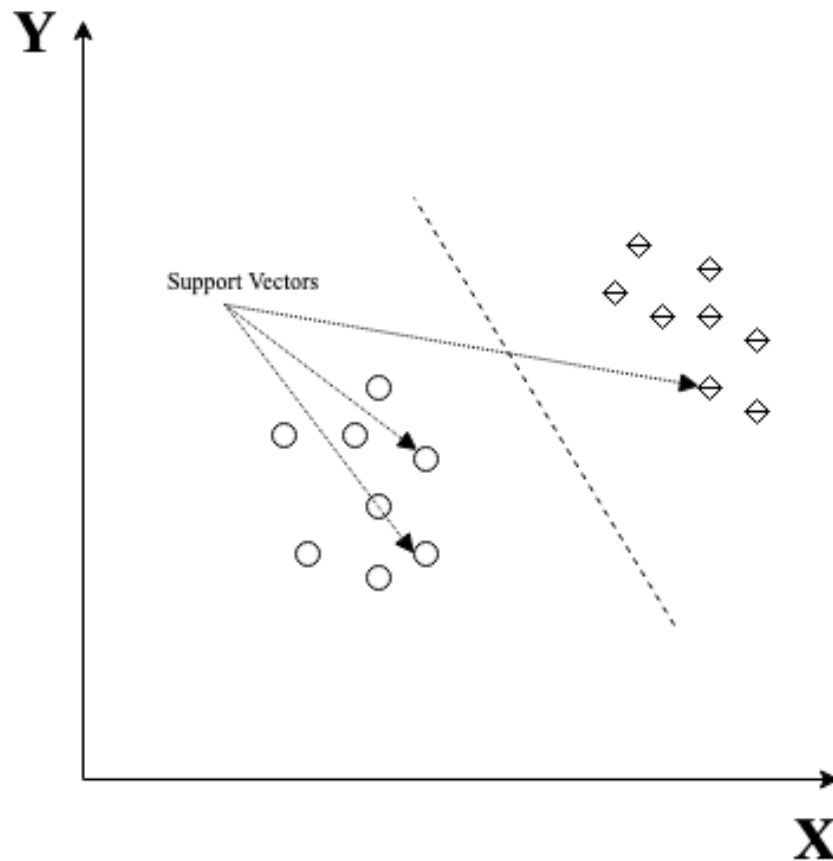


Рисунок 2.2 – Приклад роботи алгоритму SVM

Переваги:

- дуже добре працює коли є велика відстань між даними класів;
- більш ефективний у багатовимірних просторах;
- використовує пам'ять відносно ефективно [28].

Недоліки:

- фактична неможливість використання для великих об'ємів даних;
- дуже чутливий до зашумлених даних під час навчання;
- немає імовірнісної класифікації, тобто метод «жорстко» прогнозує клас об'єкту [28].

#### 2.1.4 Огляд методу класифікації Naïve Bayes

Наївний Байес (NB) є алгоритмом навчання з вчителем, на основі теореми Байеса [11]. Це алгоритм класифікації на основі ймовірності, який класифікує дані

за своїми класами, на основі правила прийняття рішення – «Максимум апостеріорних», у байєсівському налаштуванні, а основне припущення NB полягає в тому, що вплив значення змінної на даний клас не залежить від значень інших змінних [11].

Теорема Байєра:

$$P(B|A) = \frac{P(A|B) P(B)}{P(A)}, \quad (2.1)$$

де  $A$  та  $B$  – події;

$P(A|B)$  – це ймовірність події  $A$ , якщо подія  $B$  істинна (відбулася). Подія  $B$  також називається доказом;

$P(A)$  – є апіорним значенням  $A$  (попередня незалежна ймовірність, тобто ймовірність події до того, як буде видно доказ);

$P(B)$  – є апіорним значенням  $B$ ;

$P(B|A)$  – це ймовірність  $B$  для даної події  $A$ , тобто ймовірність події  $B$  після того, як буде видно доказ  $A$  [29].

Виходячи з цієї моделі можна записати NB алгоритм:

$$P(y|X) = \frac{P(X|y) P(y)}{P(X)}, \quad (2.2)$$

де  $y$  – змінна, що характеризує належність до класу;

$X$  – це вектор залежних ознак розмірністю  $d$ , тобто  $X = (x_1, x_2, x_2, x_d)$ , де  $d$  кількість заданих ознак зразка;

$P(y|X)$  – це ймовірність того, що клас  $y$  буде присвоєно зразку  $X$ ;

$P(X|y)$  – це ймовірність того, що зразку  $X$  буде присвоєно клас  $y$  [29].

Таким чином для того, щоб знайти ймовірність належності обраного зразка  $X$  для всіх можливих значень змінної класу  $y$ , просто потрібно обрати результат з максимальною ймовірністю за формулою (2.3):

$$y = \operatorname{argmax}_y P(y) \prod_{i=1}^n P(x_i | y) \quad (2.3)$$

Зазвичай цей алгоритм сьогодні використовують для автоматичної категоризації тексту [11]. Проте алгоритм має суттєві недоліки, наприклад, існують потреби в методах попередньої обробки даних, які часто виконуються людиною, вручну над документом, перш ніж можна буде розпочати аналіз.

Переваги:

- працює набагато швидше від інших алгоритмів за рахунок того, що не є ітераційний, тобто імовірність прораховуються напряму;
- добре підходить для багатокласових передбачень;
- краще працює з параметрами категорій, ніж з числовими параметрами, на відміну від інших алгоритмів;
- якщо припущення алгоритму про незалежність ознак є правдивим то метод працює набагато краще від інших і потребує в рази менше навчальних даних [30].

Недоліки:

- головне припущення алгоритму дуже рідко є правдивим у реальному світі;
- має велику проблему нульової імовірності, якщо у наборі навчальних даних була відсутня така категорія, яка з'явилась під час передбачення [30].

### 2.1.5 Огляд методу класифікації Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) – це метод, який шукає оптимальний набір дискримінантних проекційних векторів  $W$ , щоб розмістити вихідний простір даних у простір ознак більш низької розмірності, максимізуючи критерій Фішера, що означає, що перекриття між класами в просторі об'єктів низької розмірності є мінімальним [31, 32].

Ідея методу полягала в тому, щоб знайти лінійну комбінацію предикторів, яка показує найбільшу різницю середніх груп по відношенню до дисперсії всередині групи [11]. LDA зазвичай використовується, як метод зменшення розмірності та класифікації, він проектує дані на простір більш низької розмірності, часто це робиться таким чином, щоб максимізувати відношення міжкласової відстані до внутрішньокласної відстані.

Переваги:

- простий, швидкий та портативний алгоритм [33].

Недоліки:

- працює нормально лише за умов нормального розподілу ознак зразків;
- показує погані результати для деяких категоричних змінних [33].

### **2.1.6 Огляд методу класифікації K-Nearest Neighbors**

Основною ідеєю методу KNN є припущення, що подібні речі існують в безпосередній близькості, що в свою чергу означає те, що подібні речі знаходяться близько один до одного [11]. Оскільки навчальні приклади, що потрібні під час виконання повинні бути в пам'яті, цей метод, іноді, також називають класифікацією на основі пам'яті [34].

KNN знаходить відстані між обраною вибіркою даних з усіма вибірками в наборі даних і обирає вказану кількість вузлів (K), найближчих до обраної вибірки даних [11]. KNN простий у використанні та має швидкий час обчислення, крім того, KNN не робить припущень щодо даних, однак точність KNN сильно залежить від якості даних, також присутня проблема, пов'язана з складністю у пошуку оптимального значення вузлів K необхідних для найкращої роботи методу [11].

Це призводить до попереднього етапу про рахування найоптимальнішого числа вузлів, що в свою чергу відображується на швидкості налаштування KNN.

KNN має низьку точність класифікації, якщо класи даних перемішані між собою та знаходиться дуже-дуже близько один до одного.

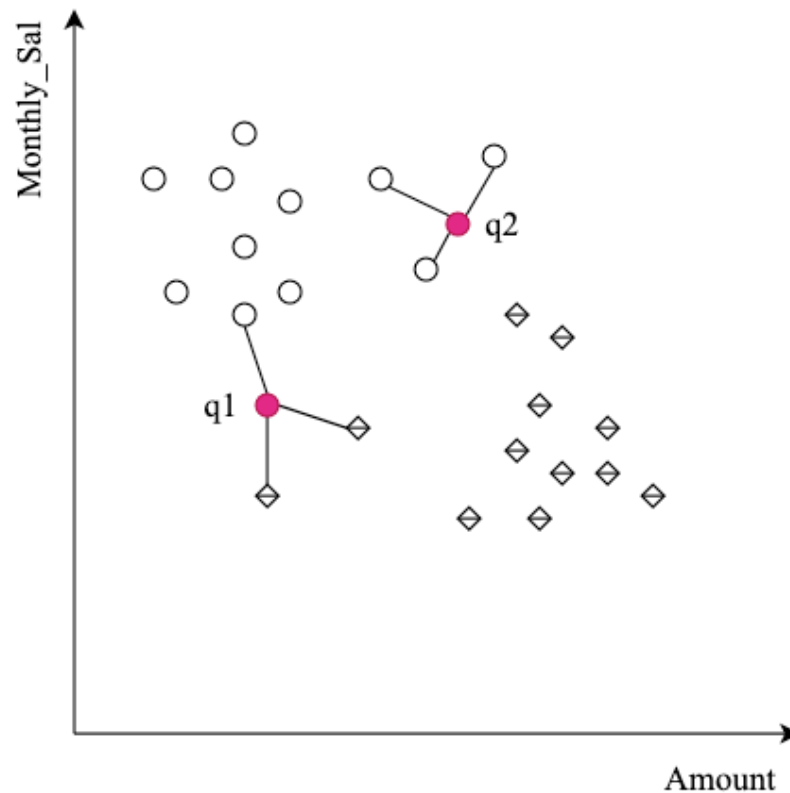


Рисунок 2.3 – Приклад KNN заснований на трьох найближчих сусідах

Переваги:

- простий у реалізації;
- надійний у зашумлених навчальних даних;
- є ефективним, якщо навчальні дані великі [12].

Недоліки:

- необхідність визначати кількість  $K$ ;
- кількість обчислювальних потужностей для перевірки кожного з  $K$  сусідів, якщо це число велике [12].

### 2.1.7 Огляд методу класифікації Self-organizing maps

Self-organizing maps (SOM) або нейронні мережі Кохонена мають високу популярність серед конкурентної галузі нейронних мереж [36]. SOM об'єднують переваги методів кластеризації та прогнозування. В основному, SOM аналізують дані без нагляду, але є також кілька контрольованих варіантів, як-от штучна нейронна мережа зі зустрічним поширенням (CP-ANN), контрольовані мережі кохонена (SKN) і мережі XY-Fused (XYF) [37]. Мережа Кохонена заснована на одному шарі нейронів, які розташовані у двох вимірах [35].

Схематичне зображення моделі зображено на рисунку 2. 4.

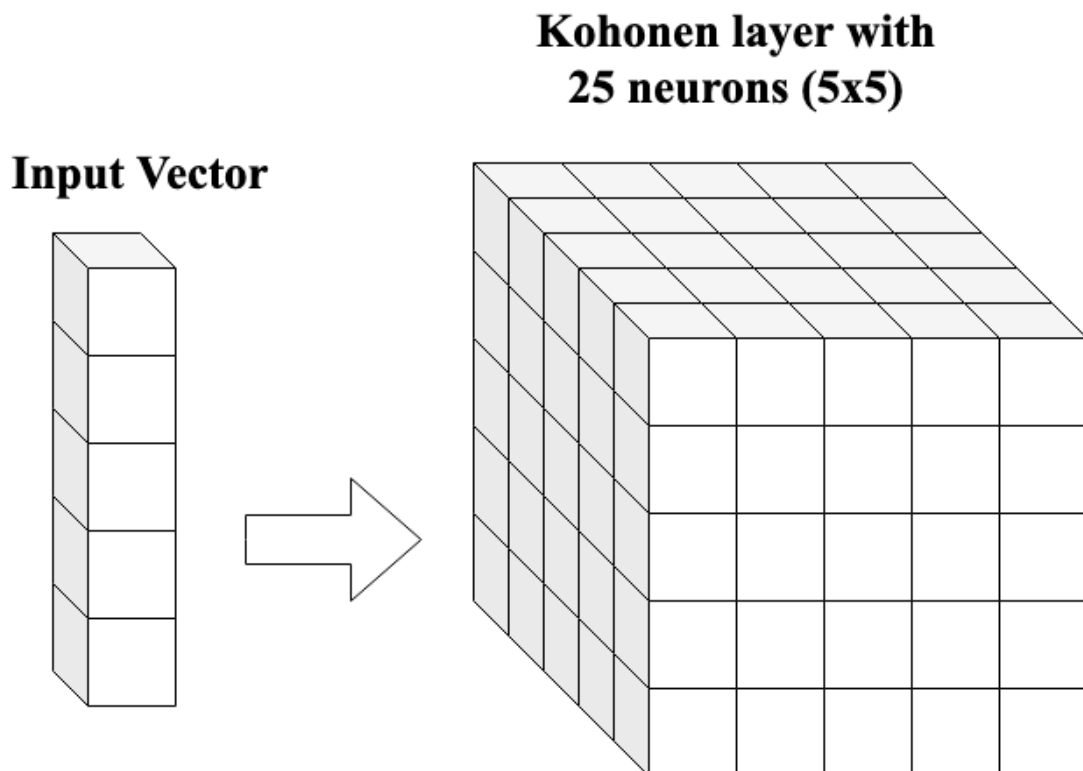


Рисунок 2.4 – Самоорганізуюча карта з 25 нейронами. Кожен нейрон є вектором (вектором ваги) з тим же розміром, що й вхідний вектор

Кожен нейрон на карті включає в себе ваговий вектор, кількість елементів у вагових векторах дорівнює кількості змінних у вхідному векторі, а елементи всіх вагових векторів (нейронів) слід випадково ініціалізувати перед навчанням

мережі [35]. На етапі навчання кожен вхідний вектор представляється мережі, і відбувається пошук найбільш схожого нейрона (нейрон-переможець) на цей вхід, його можна знайти завдяки мінімальній евклідовій відстані між вхідним вектором і ваговими векторами нейронів [35]. Вагові вектори нейрона-переможця та його сусідів оновлюються шляхом додавання різниці між фактичним вхідним вектором і відповідним ваговим вектором до елементів вагових векторів, таким чином, ваговий вектор нейронів-переможців та його сусідів стає більш схожим на вхідний вектор, таким чином, вектор ваги нейрона-переможця стане більш схожим на вхідний, а для більш віддалених нейронів ця схожість зменшиться [35]. Процес корекції ваги повторюється ітераційно, поки всі вектори в навчальному наборі не будуть представлені в мережі достатню кількість разів (epoch).

Переваги:

- простота та швидкість обчислення;
- відсутність параметрів адаптації для налаштування системи;
- детерміновані відтворювані результати;
- гарна кінцева точність передбачення [38].

Недоліки:

- погана організація методу;
- відсутність параметрів адаптації для налаштування системи;
- неможливість візуалізації;
- сильна деградація на великих даних;
- сильна залежність від ініціалізації [38].

### **2.1.8 Огляд методу класифікації Growing neural gas**

Growing neural gas (GNG) є одним із алгоритмів карт, що самоорганізуються (SOM) для навчання без вчителя [39]. Навчання без вчителя – це метод навчання, який працює виключно з вхідними даними і не має інформації про бажаний результат [39]. Вхідні дані послідовно подаються в SOM у вигляді вхідних сигналів, і SOM змінює свою топологічну структуру, щоб відповідати

вхідним даним, використовуючи свій власний специфічний механізм – самоадаптацію, крім того, GNG використовує зростаючий механізм для поступової адаптації та саморегулювання власного розміру [39]. Структура мережі GNG – це граф, що складається з набору вузлів і набору ребер, що з'єднують вузли, до кожного вузла прив'язан ваговий вектор, що відповідає позиції вузла у вхідному просторі, і змінна помилки, призначена для ідентифікації частин мережі, найменш адаптованих до вхідних сигналів, кожне ребро однозначно ідентифікується парою вузлів [39].

GNG працює наступним чином. Після ініціалізації, яка вводить два випадково згенеровані вузла в мережу, дві основні фази чергуються, доки не буде виконано обраний критерій зупинки.

Перша фаза («самоорганізація») — це адаптація, яка виконується п кроків [39]. На кожному кроці генерується випадковий вхідний сигнал, і нейронна мережа адаптується до нього: посилюється (або створюється, якщо він не існує) зв'язок між двома вузлами, найближчими до вхідного сигналу, потім найближчим вузлом і всіма його топологічними сусідами ( вузли, підключені безпосередньо до вузла ребром) рухаються до вхідного сигналу, і помилка найближчого вузла збільшується, це допомагає визначити області, де вузли недостатньо пристосовані до вхідних сигналів [39]. Після цього спрацьовує механізм стирання країв – з мережі видаляються ті краї, які довго не зміцнювались. На останньому кроці адаптації похибка кожного вузла зменшується. Використовуючи цей механізм, нейронна мережа «забуває» старі помилки і, таким чином, може зосередитись на найактуальніших зв'язках.

На другій фазі («зростання») створюється новий вузол, який підключається до мережі, помилка вузла використовується для ідентифікації області, де адаптація була найменш успішною [39]. Після знайдення вузла з найбільшою помилкою та його сусіда з найбільшою помилкою, посередині, між ними створюється новий вузол та похибки цих вузлів зменшуються.

Переваги:

- добре працює з гіпершарами та гіперскладками;

- самомаштабується без додаткових модифікацій;
- не так сильно деградує на великих даних як звичайні SOM [40].

Недоліки:

- складна та ресурсовитратна імплементація;
- погано розрізняє кластери більшої щільності в середині інших кластерів;
- можливість давати збій за рахунок одночасного оновлення великої кількості нейронів;
- погана візуалізація алгоритму [40].

### 2.1.9 Огляд методу класифікації Random forest

Алгоритм Random forest, як випливає з назви, складається з великої кількості окремих дерев рішень, які діють, як ансамбль, у цьому алгоритмі кожне окреме дерево у випадковому лісі дає прогноз класу, а клас, який набрав найбільшу кількість голосів серед дерев, стає прогнозованим класом моделі [11]. Фундаментальною концепцією Random forest є ідея мудрості натовпу. Це випливає з ідеї, що велика кількість відносно некорельованих моделей (дерев), які працюють як комітет, повинна перевершити будь-яку з окремих моделей дерева [11]. Група «слабких учнів» (дерев), якщо їх об'єднати разом, може створити «сильного учня» [11]. Random forest класифікатор не потребує масштабування ознак та є більш надійним ніж звичайне Дерево Рішень для відбору навчальних вибірок у зашумленому навчальному наборі даних.

Переваги:

- зменшення можливості перенасичення даними(over-fitting);
- зазвичай набагато точніше прогнозує клас об'єкта ніж звичайне дерево рішень [12].

Недоліки:

- повільне прогнозування в реальному часі;
- складність реалізації;

- складність самого алгоритму [12].

## **2.2 Методи класифікації засновані на DL**

Методи класифікації за допомогою НМ є дуже різноманітні і існують для вирішення дуже різних задач. Серед найбільш популярних та найбільш доречних до проблематики цієї роботи слід виокремити наступні алгоритми DL:

- RNN (Recurrent Neural Networks).
- CNN (Convolutional Neural Networks);
- LSTM (Long Short Term Memory Networks) [41].

### **2.2.1 Огляд RNN**

Рекурентна нейронна мережа (RNN) — це тип штучної нейронної мережі, яка використовує послідовні дані або дані часових рядів [42]. Ці алгоритми глибокого навчання зазвичай використовуються для класифікації у звичайних(ordinal) або тимчасових проблемах таких, як мовний переклад, обробка природної мови, розпізнавання мовлення, підписання зображень і таке інше, вони включені в популярні програми, такі як Siri, голосовий пошук і Google Translate [42].

RNN відрізняються своєю «пам'яттю», оскільки RNN беруть інформацію з попередніх входів, щоб впливати на поточний вхід і вихід, у той час, як традиційні ANN припускають, що входи та виходи незалежні один від одного, у той час, як вихід рекурентних нейронних мереж залежить від попередніх елементів у послідовності, хоча майбутні події також можуть бути корисними для визначення результату даної послідовності, однонаправлені рекурентні нейронні мережі не можуть врахувати ці події у своїх прогнозах [42].

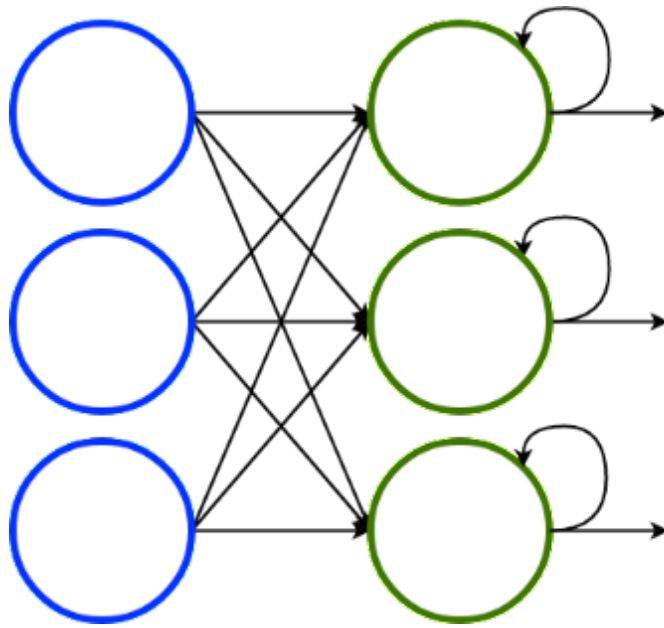


Рисунок 2.5 – Схематичне зображення RNN

Рис. 2.5 схематично зображує, що кожен нейрон прихованого слою звертається до своєї пам'яті, а не тільки передає результати розрахунку далі. У якості прикладу, можна використати систему передбачення слів при написанні тексту на вашому смартфоні, коли базуючись на тому, що ви вже написали система сама пропонує вам продовження вашого речення.

Дивлячись на рис. 2.6, ліва («згорнута») частина рисунку представляє всю нейронну мережу, а точніше всю передбачену фразу, як-от «feeling under the weather», права («Розгорнута») частина рисунку представляє окремі шари, або часові кроки нейронної мережі, кожен шар RNN відповідає одному слову у цій фразі, наприклад, «weather» [42]. Попередні вхідні дані, такі як «feeling» та «under», будуть представлені, як прихований стан на третьому кроці часу, щоб передбачити наступне слово послідовності – «the» [42].

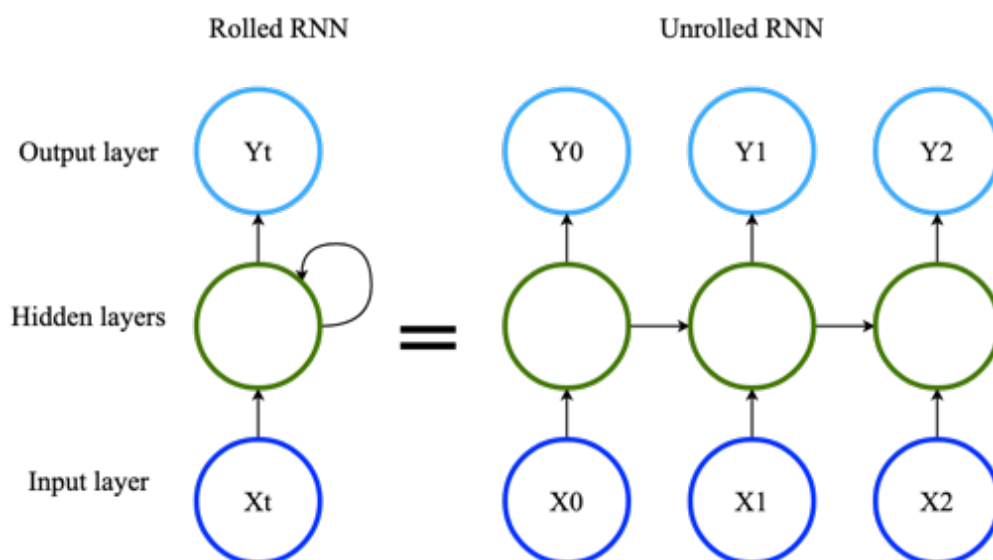


Рисунок 2.6 – Розширене представлення RNN.

Іншою відмінною характеристикою рекурентних мереж є те, що вони мають спільні параметри ваги на кожному рівні мережі, у той час, як мережі прямого зв'язку мають різні ваги в кожному вузлі. Рекуррентні нейронні мережі мають однаковий параметр ваги на кожному шарі мережі, тим не менш, ці ваги все ще коригуються в процесі зворотного поширення та градієнтного спуску, щоб полегшити навчання [42].

RNN, як правило, стикаються з двома проблемами, відомими, як градієнти, що вибухають, і градієнти, що зникають [42]. Ці проблеми визначаються розміром градієнта, який є нахилом функції втрат вздовж кривої помилки, коли градієнт занадто малий, він продовжує зменшуватися, оновлюючи параметри ваги, доки вони не стануть незначними, тобто 0 [42]. Коли це відбувається, алгоритм повністю перестає навчатись.

Вибухові градієнти з'являються коли градієнт стає занадто великим та тягне за собою збільшення значення параметра ваги у нейронах. В кінцевому випадку вони будуть представлені як NaN [42].

Одним із можливих рішень цих проблем є зменшення кількості прихованих шарів, чи зменшення кількості вузлів у шарах НМ, усуваючи частину складності моделі RNN.

Недоліки та переваги збігаються з такими у ANN (пункт 2.1.2).

### 2.2.2 Огляд CNN

CNN – це багатошарові штучні нейронні мережі зі здатністю виявляти складні елементи в даних, наприклад, витягувати ознаки в графічних і текстових даних. CNN в основному використовуються в задачах комп'ютерного зору таких, як класифікація зображень, виявлення об'єктів і сегментація зображення, однак останнім часом CNN почали застосовувати ще й для вирішення текстових проблем [43].

Деякі дослідження CNN, щодо класифікації медичних зображень, досягли результатів, які конкурують з людьми експертами, наприклад, CheXNet, CNN із 121 шаром, навчений на наборі даних із понад 100 000 рентгенівських знімків грудної клітки (ChestX-ray 14), досяг кращої продуктивності, ніж середня продуктивність чотирьох рентгенологів [44].

Основною структурною відмінністю CNN від інших НМ є два невід'ємних прихованих шара мережі:

- шар згортки (convolution) для отримання ознак із даних;
- шар об'єднання (pooling) для зменшення розміру карти об'єктів [43].

Згортка — це процес, за допомогою якого отримують ознаки [43]. Операція згортки відповідає за виявлення найважливіших ознак, результат, якої відомий, як Карта ознак, Згорнута функція або Карта активації [43]. Карта ознак обчислюється за допомогою застосування детектора ознак до вхідних даних, у деяких фреймворках глибокого навчання детектор функцій також називають ядром або фільтром, наприклад, цей фільтр може бути матрицею 3 на 3 [43]. Є можливість обрати будь-який фільтр, який підходить для вирішення проблеми.

Карта ознак обчислюється за допомогою поелементного множення ядра та матричного представлення вхідних даних, це гарантує, що карта ознак, передана CNN, буде менша, але буде містити усі важливі ознаки [43]. Фільтр робить це шляхом ковзання (sliding), крок за кроком, через кожен елемент у вхідних даних [43]. Кожна повна ітерація використання фільтру для обрахунку одного

значення карти ознак називається «великим кроком», і їх можна визначити під час створення CNN.

Візуалізацію процесу створення Карти ознак зображено на рис. 2.7.

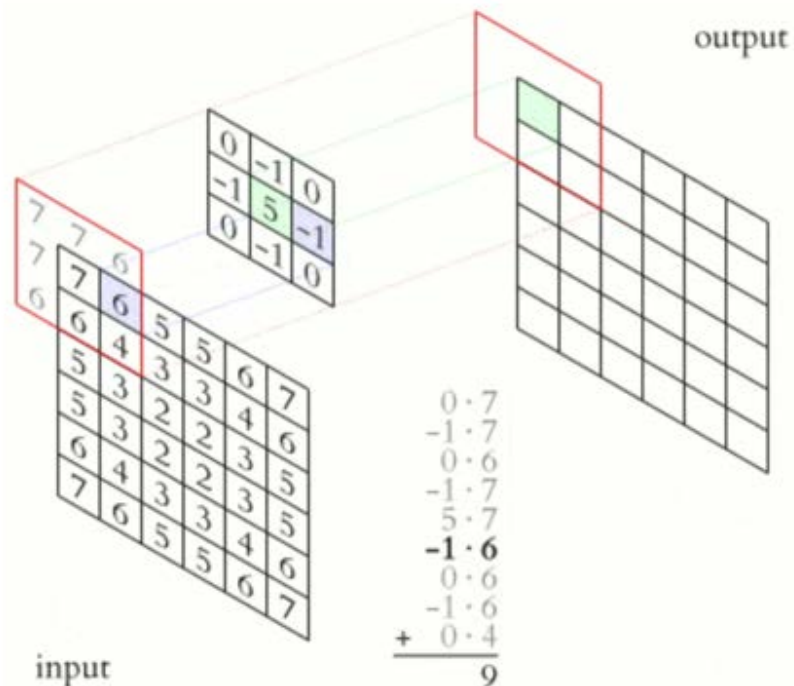


Рисунок 2.7 – Візуалізація використання фільтра для створення карти ознак

Наступним важливим кроком алгоритму є Об'єднання. Об'єднання працює шляхом розміщення матриці, скажімо, матриці 2 на 2, на карті ознак і виконання певної операції, наприклад, при max-об'єднанні обирається максимальне число, що потрапляє в цю матрицю, у aver-об'єднанні обчислюється середнє значення чисел, що підпадають під цю матрицю, а у min-об'єднанні використовується мінімальне число, що потрапляє в цю матрицю, а результат називається Об'єднана карта ознак [43]. Об'єднання в пул гарантує, що мережа може виявляти ознаки незалежно від їх розташування, об'єднання також гарантує, що розмір даних, що передаються в CNN, ще більше зменшується [43].

Візуалізований приклад max-об'єднання зображено на рис. 2.8.

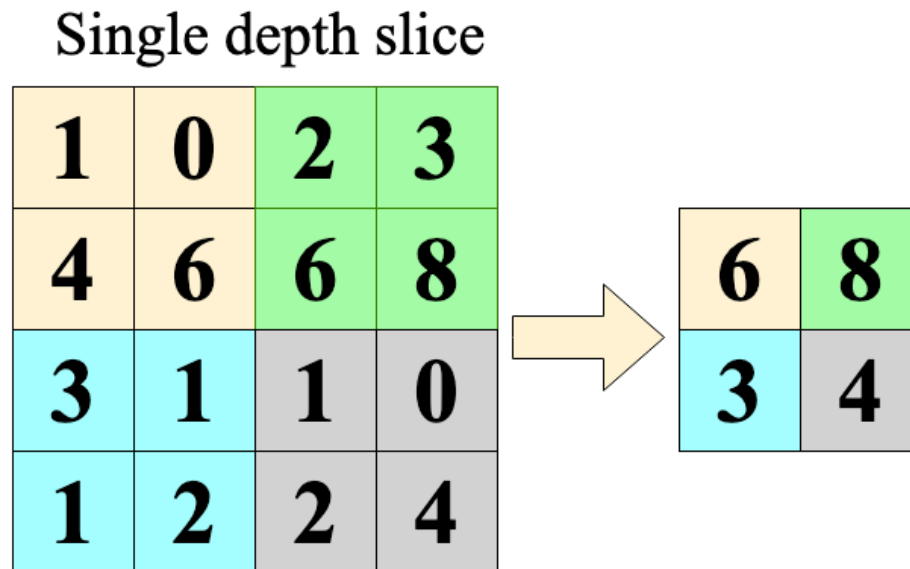


Рисунок 2.8 – Приклад використання max-об’єднання для одного зрізу, для створення об’єднаної карти ознак

Після чого використовується звичайний шар НМ для виводу передбачення CNN.

Як вже було сказано вище CNN зазвичай використовують для класифікації зображень, проте по при гарні результати вже навчена система є сильно чутливою до незначного шуму зображення, яке навіть не помітне людському оку [45]. Це наглядно видно на прикладі приведенному Goodfellow[46] у статті [45], що зображено на рис. 2.9.

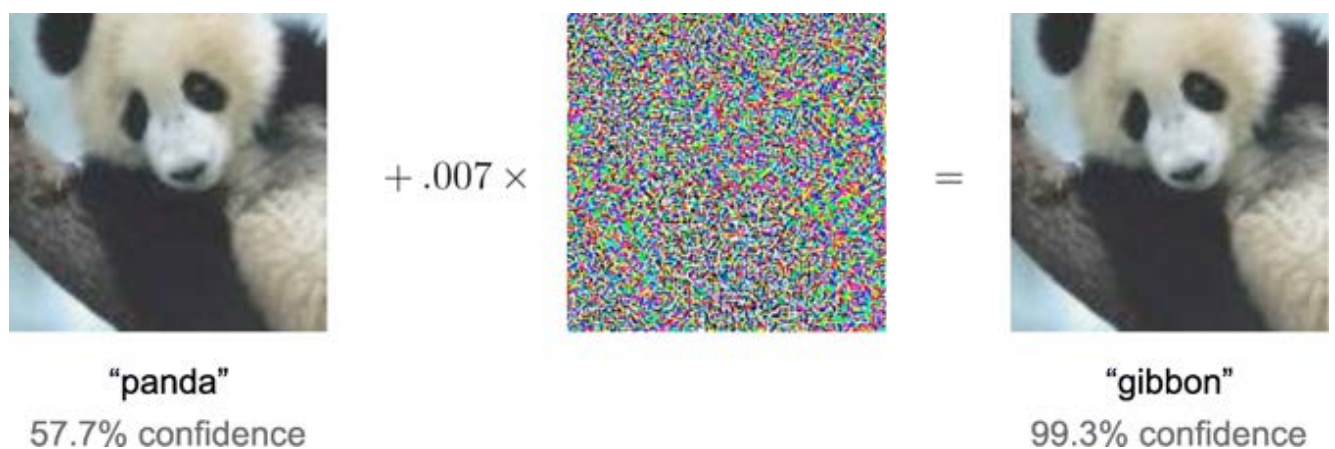


Рисунок 2.9 – Результат роботи CNN до та після додавання мізерного шуму до зображення [45]

Мережа класифікувала об'єкт на зображенні зліва як «панду» з достовірністю 57,7%. Додавши дуже невелику кількість ретельно сконструйованого шуму (середнє зображення), мережа неправильно класифікувала об'єкт як «гібон» з 99,3% впевненістю щодо правильного зображення без видимих змін для людини [46].

Переваги:

- дуже добре працює з проблемами текстової класифікації та класифікації зображень;
- може виокремлювати важливі особливості об'єкту без нагляду вчителя.

Недоліки:

- дуже повільне прогнозування за рахунок кількості необхідних обчислень;
- потребує гігантську навчальну вибірку для коректної роботи;
- чутлива до непомітних для людини, але специфічних шумів;
- потребує великих обчислювальних потужностей.

### 2.2.3 Огляд LSTM

LSTM був розроблений для подолання проблем простої повторюваної мережі (RNN), дозволяючи мережі зберігати дані в пам'яті, до якої вона може отримати доступ пізніше [47]. LSTM — це особливий тип рекурентної нейронної мережі (RNN), яка може вивчати довгострокові моделі [47]. Загальний вид LSTM зображено на рис. 2.10.

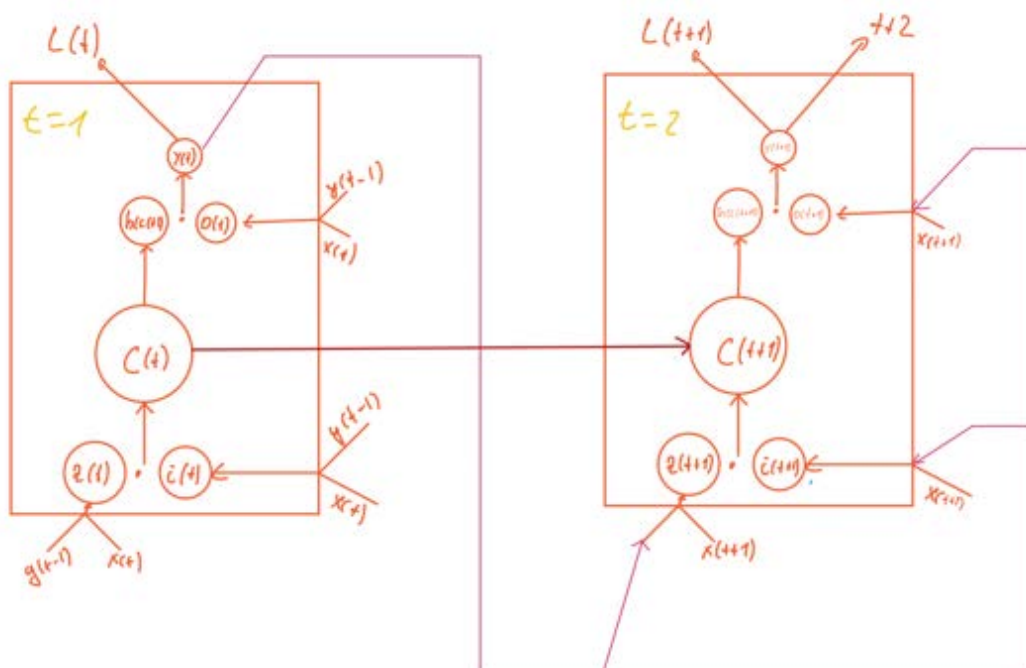


Рисунок 2.10 – Загальний вид LSTM моделі

Ключем до LSTM є стан комірки – горизонтальна лінія, що проходить через середню частину діаграми, як зображено на рисунку 2.11.

Стан клітини нагадує конвеєрну стрічку, він проходить прямо по всьому ланцюжку, лише з деякими незначними лінійними взаємодіями, інформації дуже легко просто текти по ній без змін [47].

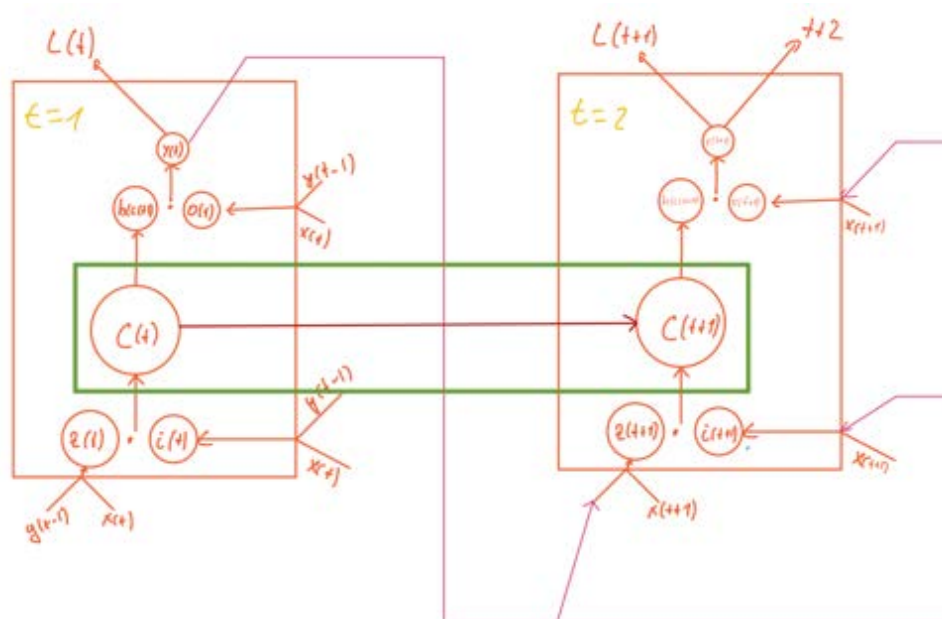


Рисунок 2.11 – Шлях даних, що перетікають з ітерації в ітерацію

Якщо розглядати кожну ітерацію загалом то можна побачити, що метод складається з 3 основних кроків/вузлів:

- крок забування;
- крок введення даних;
- крок виведення даних.

Крок забування.

Перший крок – вирішити, яку інформацію ми збираємось забути (викинути) у стані клітинки, це рішення приймається сигмовидним шаром, який називається «Forget Gate Layer» [48].

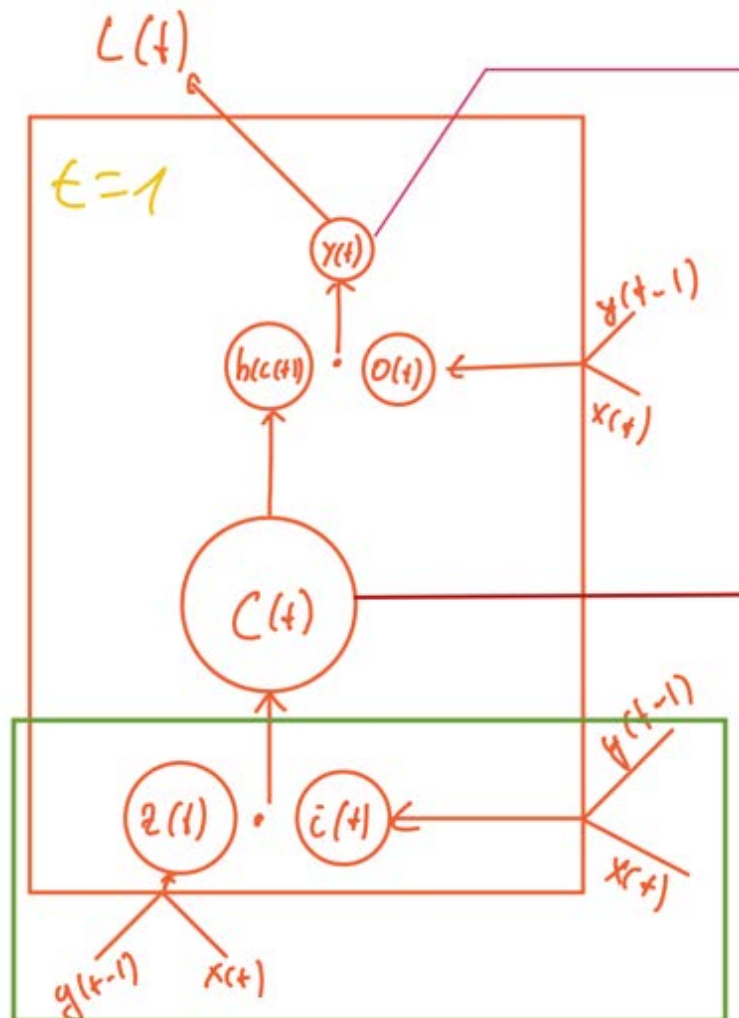


Рисунок 2.12 – Крок забування

Крок введення даних.

Наступний крок – вирішити, яку нову інформацію ми будемо зберігати в стані клітинки [48]. Крок розділяється на декілька частин. По-перше, сигмовидний шар під назвою «Input Gate layer» вирішує, які значення ми будемо оновлювати, далі шар  $\tanh$  створює вектор нових значень-кандидатів, які можна додати до стану [48].

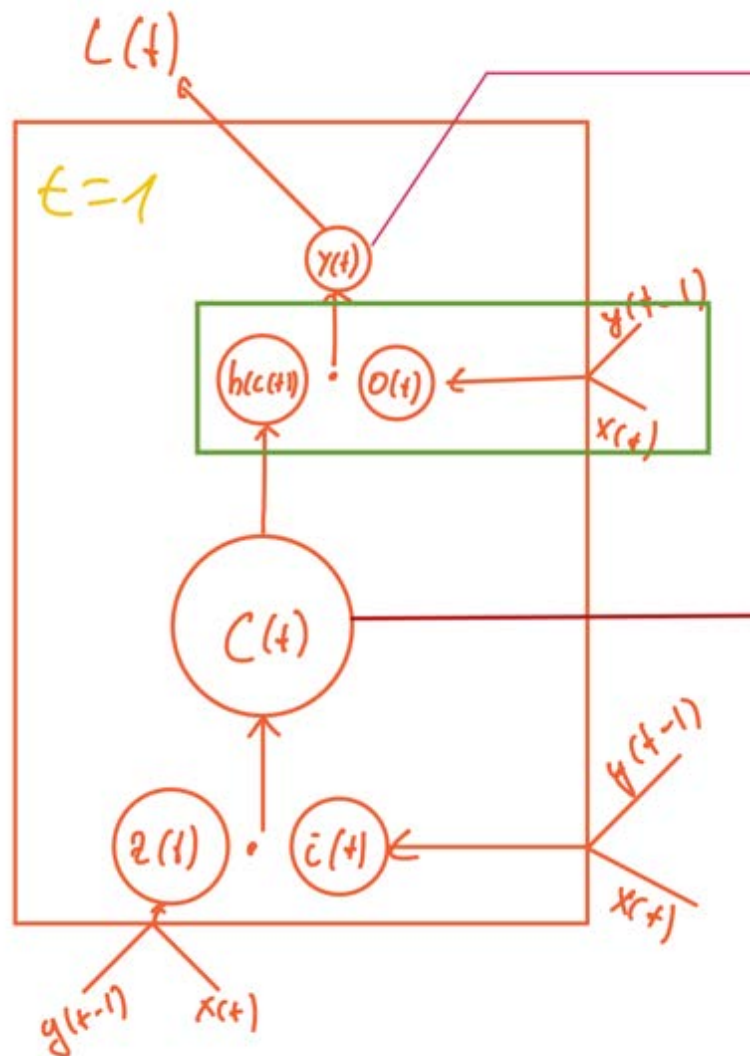


Рисунок 2.13 – Крок введення даних

Тепер треба об'єднати перших два кроки, щоб створити оновлення стану клітинки, після чого потрібно змінити старий стан комірки пам'яті на новий [48].

Крок виведення даних.

Вихід буде заснований на стані клітинки, але буде відфільтрованою версією, спочатку запускається сигмовидний шар, який вирішує, які частини стану клітинки треба вивести, потім переноситься стан комірki через  $\tanh$  (щоб нормалізувати значення між “-1” і “1”) і множиться на вихід сигмовидних воріт, щоб передати лише потрібні частини даних [48].

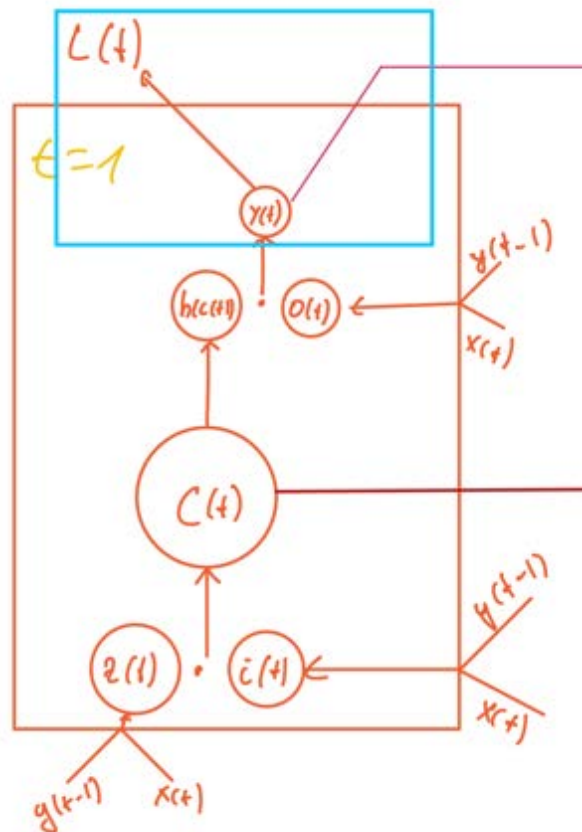


Рисунок 2.14 – Крок виведення даних

Переваги:

- надає змогу забувати помилкові ваги, що в свою чергу веде до покращення результатів передбачення;
- може вирішувати нетривіальні задачі, як наприклад написання музики;
- гарно підходить для передбачень у реальному часі;
- добре справляється з проблемою перенаповнення.

Недоліки:

- потребує великих обчислювальних потужностей;
- гірше справляється з тривіальними задачами класифікації.

### 2.3 Порівняння розглянутих методів для технічного діагностування

Порівняння розглянутих методів для вирішення задачі цієї дипломної кваліфікаційної роботи зображено у табл. 2.1.

Таблиця 2.1 – переваги та недоліки розглянутих методів

| Метод                        | Переваги                                                                                                                       | Недоліки                                                                                                                                               |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Decision tree                | Просто перевірити людиною-експертом; Не потребує багато попередньої обробки даних.                                             | Потребує забагато обчислювальних потужностей для класифікації.                                                                                         |
| Support vector machines      | Відносна ефективність використання пам'яті                                                                                     | Фактична неможливість використання для великих об'ємів даних; дуже чутливий до зашумлених даних під час навчання.                                      |
| Naïve Bayes                  | Працює дуже швидко.                                                                                                            | Головне припущення методу про те, що вплив значення змінної на даний клас не залежить від значень інших змінних, не є правдивим у випадку цієї роботи. |
| Linear Discriminant Analysis | Простий, швидкий та портативний.                                                                                               | Показує незадовільні результати класифікації.                                                                                                          |
| K-Nearest Neighbors          | Ефективний у зашумлених даних та коли навчальні дані великі.                                                                   | Потребує багато обчислень для оптимального числа K.                                                                                                    |
| Self-organizing maps         | Простота та швидкість обчислення;                                                                                              | Відсутність параметрів адаптації для налаштування системи; сильна деградація на великих даних.                                                         |
| Growing neural gas           | Добре працює з гіпершарами та гіперскладками; самомаштабується без додаткових модифікацій; не така сильна деградація як у SOM. | Складна та ресурсовитратна імплементація; погано розрізняє кластери більшої щільності в середині інших кластерів;                                      |

Продовження таблиці 2.1

| Метод         | Переваги                                                                                                                                                    | Недоліки                                                                                                   |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| Random forest | Зменшення можливості перенасичення даними(over-fitting).                                                                                                    | Складність алгоритму; повільне прогнозування.                                                              |
| RNN           | Не потрібно завчасно визначати функції алгоритму; забезпечує кращі результати прогнозування ніж усі інші алгоритми, якщо обсяг навчальних даних величезний. | Потребує дуже великої кількості даних, щоб працювати краще, ніж інші методи; немає довгострокової пам'яті. |
| CNN           | Може виокремлювати важливі особливості об'єкту без нагляду вчителя.                                                                                         | Дуже повільне прогнозування за рахунок кількості необхідних обчислень; чутливий до специфічних шумів.      |
| LSTM          | Добре справляється з проблемою перенаповнення; надає змогу забувати помилкові ваги.                                                                         | Потребує великих обчислювальних потужностей.                                                               |

Як стає зрозуміло з табл. 2.1, більшість розглянутих методів класифікації мають суттєві недоліки для вирішення конкретно задачі цієї дипломна кваліфікаційної роботи. Однією з великих вад є занадто велика чутливість до зашумлених даних. Деякі алгоритми які добре справляються з таким типом даних мають владу у використанні занадто великої кількості обчислювальних потужностей, що в свою чергу призводить до значного сповільнення, як навчання так і прогнозування стану КП.

Найбільш ефективними виявились алгоритми DL та звичайні, які є стійкими до сильно зашумлених даних. Проте для виконання задачі класифікації у цій роботі був обрано алгоритм LSTM, через його велику перевагу у можливості запам'ятовувати дані та забувати непотрібні ваги.

## 2.4 Використані матеріали у роботі

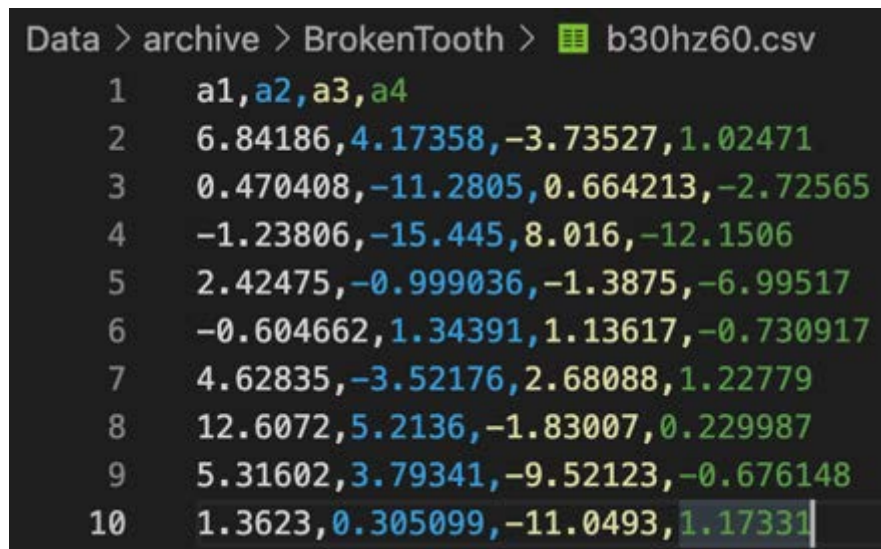
У роботі було використано вибірку даних з сайту kaggle.com, а саме датасет Gearbox Fault Diagnosis [49].

Набір даних діагностики несправностей коробки передач включає набір даних вібрації, записаний за допомогою симулятора діагностики несправностей коробки передач SpectraQuest [49].

Набір даних був записаний за допомогою 4-х вібраційних датчиків, розміщених у чотирьох різних напрямках, при зміні навантаження від «0» до «90» відсотків та включає два різних стана КП:

- справний стан;
- зламаний зуб [49].

Всього є 20 файлів, 10 для працездатної КП і 10 для зламаної. Кожен файл відповідає заданому навантаженню від 0% до 90% з кроком 10% [49].



```
Data > archive > BrokenTooth > b30hz60.csv
1  a1,a2,a3,a4
2  6.84186,4.17358,-3.73527,1.02471
3  0.470408,-11.2805,0.664213,-2.72565
4  -1.23806,-15.445,8.016,-12.1506
5  2.42475,-0.999036,-1.3875,-6.99517
6  -0.604662,1.34391,1.13617,-0.730917
7  4.62835,-3.52176,2.68088,1.22779
8  12.6072,5.2136,-1.83007,0.229987
9  5.31602,3.79341,-9.52123,-0.676148
10 1.3623,0.305099,-11.0493,1.17331
```

Рисунок 2.15 – Фрагмент початкових даних для КП зі зламаними зубцями

Як зображено на рис. 2.15, данні представлені у вигляді десятиного числа з плаваючою точкою. Дані не є відсортованими та мають різну кількість знаків після зап'ятої, також зображено, що присутні, як від'ємні так і позитивні числа. Кожен з двадцяти файлів налічує приблизно сто тисяч записів, тобто усього представлено приблизно два мільйони записів у якості вхідних даних.

Первинне дослідження даних показало, що коефіцієнт кореляції параметрів з класом об'єкту (справний/зламаний зубці) дуже низький, майже відсутній.

Таблиця 2.2 – Коефіцієнт кореляції параметрів до класу у випадку 80% завантаженості

| Назва параметру | Коефіцієнт кореляції |
|-----------------|----------------------|
| a1              | -0.001300            |
| a2              | 0.000311             |
| a3              | -0.002035            |
| a4              | 0.001612             |
| faulty          | 1.000000             |

Також слід зауважити, що дані є дуже зашумленими, це зображено на рисунку 2.16. Задля створення цього графіку чотирьохвимірні дані було зведено до двовимірних за допомогою алгоритму PCA. На рисунку зеленими крапками відмічаються дані з справних КП, а червоними дані з КП у яких присутні зламані зубці.

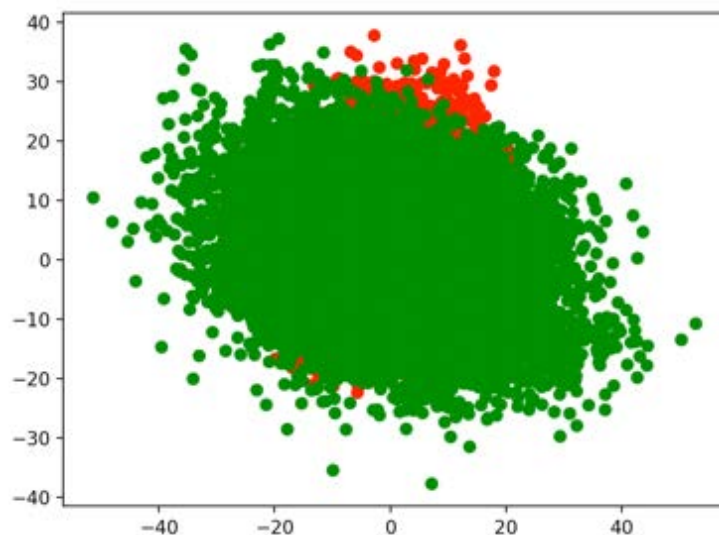


Рисунок 2.16 – Двовимірний графік розподілення екземплярів вибірки

На рисунку 2.16 зображено, що дані перекривають одні-одних та є дуже зцентрованими. Тобто можна зробити висновок, що зашумленість даних є величезною, бо майже усі екземпляри КП зі зламаними зубцями всередині знаходяться під справними екземплярами КП.

SpectraQuest є провідним розробником і виробником комплексних систем без додаткових налаштувань для навчання та діагностування з аналізу вібрації машини, балансування ротора та центрування валу/муфти [50].



Рисунок 2.17 – Вигляд апаратури MFS, на якій проводилось збирання даних [50]

Пристрій сконструйований для глибокого розуміння різних вібраційних сигнатур. Він імітує реальні машини. Хоча діагностування однієї несправності механізму може бути корисним, існує багато випадків, коли аналіз взаємодії між динамічною жорсткістю, резонансом і швидкістю є важливим для розуміння спектрів вібрацій реального світу [50]. MFS дозволяє розвинути та покращити досвід, необхідний для діагностики проблем промислового обладнання в добре контрольованих експериментах [50]. На заводі, який працює в повному обсязі, практично неможливо зрозуміти кінетику та динаміку обладнання без негативного впливу на виробництво та прибуток: MFS дає можливість автономного навчання та експериментів, без простої виробництва, що значною мірою зменшує витрати на реалізацію нововведень у виробництво.

## **2.5 Запропонований метод діагностики стану КП**

На рис. 2.18 зображено схему запропонованого методу.



Рисунок 2.18 – Метод визначення стану КП

Формування вибірки даних.

Основною ідеєю цього етапу є об'єднання та структуризація усіх наявних навчальних даних в один великий набір даних, та виокремлення лише важливих для діагностики параметрів.

Під час практичного застосування, для збереження оперативної пам'яті, краще усі параметри діагностики привести до типу float 32 та додати параметри завантаженості та поламки у форматі int16. Після чого обрати дані, згідно з потребами користувача, як зображено на рис. 2.19.

|       | a1        | a2        | a3        | a4        | load | failure |
|-------|-----------|-----------|-----------|-----------|------|---------|
| 88320 | 1.260410  | -1.357260 | -1.916330 | 1.845700  | 10.0 | 1       |
| 88321 | -0.126204 | -2.272830 | 0.536155  | 1.530920  | 10.0 | 1       |
| 88322 | -0.903164 | -1.042040 | -0.741337 | 1.650110  | 10.0 | 1       |
| 88323 | -1.693560 | 2.792950  | -0.066998 | 2.017350  | 10.0 | 1       |
| 88324 | -3.556980 | 2.114110  | 0.225473  | 2.329820  | 10.0 | 1       |
| 88325 | -2.932810 | 0.317006  | 0.426616  | 3.086060  | 10.0 | 1       |
| 88326 | -3.131050 | 0.505054  | 1.176690  | 0.415654  | 10.0 | 1       |
| 88327 | -1.759770 | 0.680900  | -0.095417 | 0.528074  | 10.0 | 1       |
| 88328 | -1.593960 | -0.156572 | 0.278035  | 1.607740  | 10.0 | 1       |
| 88329 | -1.703730 | -1.681700 | -0.030991 | 1.351480  | 10.0 | 1       |
| 88330 | 0.758765  | -0.592594 | -0.709882 | 1.926550  | 10.0 | 1       |
| 88331 | -1.440220 | 0.174783  | -2.486230 | 2.456370  | 10.0 | 1       |
| 88332 | -1.175900 | 0.942161  | -1.881420 | 3.496110  | 10.0 | 1       |
| 88333 | -2.199410 | 0.085301  | -1.705390 | 2.739000  | 10.0 | 1       |
| 88334 | -2.831860 | 1.445570  | -0.648491 | 3.259020  | 10.0 | 1       |
| 88335 | -4.340590 | 1.637680  | -0.879570 | 3.345640  | 10.0 | 1       |
| 88336 | -2.692370 | 1.777600  | 2.125710  | -1.512060 | 10.0 | 1       |
| 88337 | -0.250224 | -0.275339 | 1.072260  | -3.653520 | 10.0 | 1       |
| 88338 | -0.188349 | 0.767806  | -0.464730 | 0.234628  | 10.0 | 1       |
| 88339 | -0.405724 | 3.389630  | -0.108247 | 2.180650  | 10.0 | 1       |

Рисунок 2.19 – Перші 20 рядків об'єднаних даних для завантаження 10

Розділення на маленькі фрагменти з лише зламаними КП та з лише здоровими КП.

Етап розділення передбачає формування структури масивів навчальних даних, де кожному двовимірному масиву присвоюється параметр класифікації.

Приклад реалізації цього етапу зображено на рис. 2.20.

```
[[-1.69381e-01 -1.28208e+00 3.30282e+00 -1.55699e+00]
 [ 3.94582e+00 -2.20907e-01 -3.48517e-03 -1.74652e-01]
 [ 8.88728e-01 6.94251e-01 -3.54913e-02 -4.70260e-01]
 [-2.33282e+00 -1.57887e+00 1.30601e+00 -1.29554e+00]
 [ 1.75551e+00 2.08489e+00 3.48722e-01 5.02751e-01]
 [ 2.48702e+00 -6.24662e-01 9.41114e-01 2.55715e-01]
 [-5.93818e-01 -3.19066e-01 1.87537e+00 -1.14449e+00]
 [ 1.83684e-01 9.59308e-01 2.63464e-01 -8.74251e-01]
 [ 9.49517e-01 -2.71343e-01 -2.17044e-01 -1.67690e+00]
 [ 8.92798e-01 -1.09268e+00 -1.72897e-01 -9.65629e-01]], 0
```

Рисунок 2.20 – Приклад фрагменту з виключно працездатними КП

Побудова правильної вимірності навчальних даних.

Через те що LSTM очікує на вході тривимірні дані визначеного формату, треба налаштувати правильну вимірність навчальних даних, а саме послідовність:

- загальна кількість наборів;

- кількість екземплярів у наборі;
- кількість вимірів в кожному екземплярі.

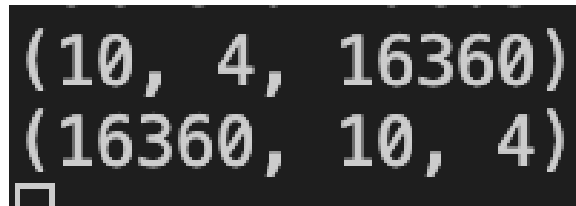


Рисунок 2.21 – Вимірність навчальних даних, до та після налаштування

### Побудова LSTM моделі Many-to-one

Під час цього етапу відбувається створення та налаштування моделі LSTM, а саме варіанту Many-to-one.

Модель складається з 2 шарів:

- LSTM – вхідний шар моделі з 16 нейронами та активатором «tanh»;
- Dense – вихідний шар моделі з 1 нейроном та активатором «sigmoid».

Для моделі був обраний оптимізатор Adam з величиною кроку 0.01.

Функція забування – Бінарна кросентропія.

Модель навчається впродовж десяти епох. Параметр `batch_size` встановлюється у значення 10.

Навчання створеної LSTM моделі.

Цей етап передбачає завантаження навчальних даних у модель та сам процес навчання.

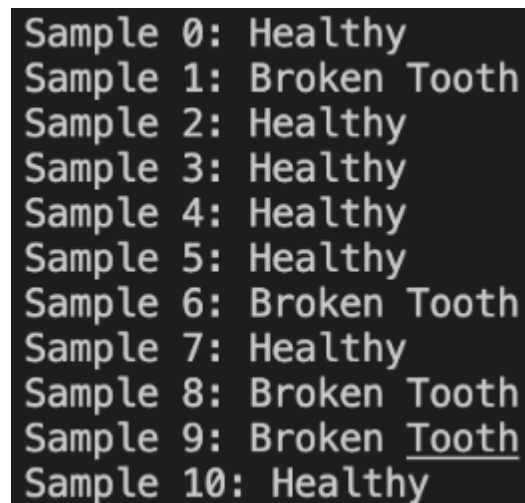
Діагностування стану КП.

Після того, як модель завершила навчання у ней треба завантажити набори даних користувача для діагностики стану КП, такого самого формату як і при навчанні.

Зведення отриманих даних до зрозумілого людині стану.

Після того, як модель поверне діагностований стан, він буде також тривимірним, тому його треба перетворити на одновимірну змінну та округлити до найближчого стану, тобто до 0 або до 1. Тепер треба зробити його зрозумілим

для кінцевого користувача то надати назву згідно з вердиктом діагностики після округлення, наприклад, так, як показано на рис. 2.22.



```

Sample 0: Healthy
Sample 1: Broken Tooth
Sample 2: Healthy
Sample 3: Healthy
Sample 4: Healthy
Sample 5: Healthy
Sample 6: Broken Tooth
Sample 7: Healthy
Sample 8: Broken Tooth
Sample 9: Broken Tooth
Sample 10: Healthy
  
```

Рисунок 2.22 – Результат роботи запропонованого методу

## 2.6 Висновки за розділом 2

У розділі 2 було розглянуто усі найпопулярніші методи класифікації за допомогою, як машинного навчання так і алгоритмів глибокого навчання. Було наведено їх переваги та недоліки. У розділі розглянуто наступні методи:

- а) дерево рішень (Decision tree);
- б) нейронні мережі (Artificial neural networks):
  - 1) RNN;
  - 2) CNN;
  - 3) LSTM;
- в) опорні векторні матриці (Support vector machines);
- г) наївний Байєс (Naïve Bayes);
- д) лінійний дискримінантний аналіз (Linear discriminant analysis);
- е) К-найближчих сусідів (K-Nearest Neighbours);
- ж) карти, що самоорганізуються (Self-organizing maps);
- з) нервовий газ, що розширюється (Growing neural gas);
- і) випадкові ліси (Random forest).

Було запропоновано власний метод вирішення задачі технічної діагностики, заснований на НМ LSTM, що в свою чергу може стати підставою для визначення стану КП без повного розбору автомобіля.

## **3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

### **3.1 Вибір засобів реалізації програмного забезпечення**

Зважаючи на мету дипломної кваліфікаційної роботи, можна зрозуміти, що хоча основною задачею і є розробка ефективного методу технічного діагностування, але також є задача розробки вебзастосунку з відповідною функціональністю та базою даних (БД). На сьогоднішній момент часу, на ринку є багато мов програмування, які можуть одразу вирішити всі задачі, та мови, що можна комбінувати між собою для вирішення кожної з задач окремо. Для досягнення мети цієї роботи треба обрати наступні засоби програмного забезпечення: мову програмування, фреймворк для створення вебзастосунку, систему управління базами даних, засоби реалізації запропонованого методу та засоби попередньої обробки даних.

#### **3.1.1 Вибір мови програмування**

Серед найбільш популярних мов з можливістю зручної обробки даних можна виділити наступні: Python, R, MATLAB, C# та Java. Кожна з цих мов має свої переваги та недоліки і кожна має свій унікальний стиль вирішення проблеми роботи.

Одразу можна сказати, що попри зручну роботу з даними та наявність бібліотек машинного навчання, дуже складно буде побудувати вебзастосунок на базі MATLAB, проте комбінуючи його з іншими мовами програмування, наприклад Java, можна розділити повноваження та побудувати модульний, мультимовний ПЗ.

В табл. 3.1 наведено порівняння цих мов програмування між собою базуючись на потребах під час розробки проєкту, де «+» – це повна реалізація критерію, «+/-» – часткове вирішення, або вирішення за допомогою супутніх засобів, «-» – відсутня можливість реалізації критерію.

Таблиця 3.1 – Порівняльна таблиця мов програмування [51-53]

| Критерій \ Мова                        | Python | R   | MATLAB | C#  | Java |
|----------------------------------------|--------|-----|--------|-----|------|
| Наявність бібліотек машинного навчання | +      | +   | +      | +/- | +/-  |
| Підтримка спільноти                    | +      | +/- | -      | +   | +    |
| Засоби аналізу даних                   | +      | +   | +/-    | +   | +/-  |
| Високорівнева мова                     | +      | +   | +/-    | -   | -    |
| Засоби створення вебзастосунку         | +      | -   | -      | +   | +    |
| Засоби зручної роботи з БД             | +      | +/- | -      | +   | +    |

Як можна зрозуміти з табл. 3.1 усім потрібним критеріям для розробки ПЗ повністю відповідає лише мова Python. Наступною за нею можна вважати C#, але за рахунок того, що C# не має такої кількості та якості бібліотек для створення і налаштування нейронних мереж, наявність зручного фреймворку ASP.NET для створення вебзастосунку не переважає недоліків. Також ще однією проблемою у даному випадку є низькорівневність мови C#, що в свою чергу призводить до збільшення написаного програмістом коду та збільшення часу на розробку.

Такі мови програмування, як MATLAB взагалі не дають можливості розробити повноцінний застосунок, що в свою чергу призводить до необхідності використання додаткової мови програмування такої, як наприклад Java. Це також збільшує кількість модулів, що необхідно розробити, а це призводить до витрати зайвого часу на розробку.

Тому, виходячи з усього вищесказаного, мовою програмування для реалізації ПЗ була обрана мова Python. Також треба відмітити, що під час розробки використовувалась мова JavaScript, але виключно для роботи з деякими елементами інтерфейсу вебзастосунку.

### 3.1.2 Вибір бібліотек Python для класифікації та прогнозування

У мові програмування Python існує декілька бібліотек, здатних до вирішення задачі класифікації різними шляхами. Для реалізації запропонованого методу у пункті 2.4 більше за всіх підходять бібліотеки: tensorflow та scikit-learn. Обидві бібліотеки є безкоштовними та мають свої переваги та недоліки.

Scikit-learn – це бібліотека машинного навчання з відкритим вихідним кодом, яка підтримує навчання з вчителем та без нього, вона надає різні інструменти для підгонки моделі, попередньої обробки даних, вибору й оцінки моделі та багато інших утиліт [54]. Серед її переваг можна виділити просту, майже однострокову реалізацію багатьох алгоритмів. Наприклад реалізація методу класифікації KNN звісно потребує попередньої обробки даних, але сама модель реалізується і навчається однією строчкою коду. Проте ця перевага інколи є її недоліком, така простота реалізації не надає потрібної гнучкості для впровадження власного методу класифікації. Також серед недоліків можна виділити погану роботу з налаштуванням систем DL. Натомість бібліотека представляє широкий пакет можливостей для попередньої обробки даних.

TensorFlow – це прозора платформа, яка спрощує створення та розгортання моделей машинного навчання [55]. TensorFlow пропонує кілька рівнів абстракції, щоб розробник міг обрати потрібний для своїх задач, наприклад, високорівневе API Keras полегшує роботу з TensorFlow та машинним навчанням [55]. Серед переваг TensorFlow можна виділити те, що вона дійсно надає змогу швидко та гнучко створити та налаштувати модель для нейронної мережі. Надає змогу додати будь-яку кількість шарів будь-якого типу у НМ з можливістю налаштування кожного аспекту, кожного шару, наприклад, можливість обрати активатор для кожного окремого прихованого шару. Проте, ця бібліотека вимагає апаратного прискорення від GPU, що в свою чергу є проблематичним при використанні новітньої архітектури ARM. Також велика гнучкість системи інколи може бути більше проблемою ніж перевагою, наприклад, неправильно

налаштована система може використовувати усі потужності комп'ютера та унеможливити роботу за ним.

Обидві ці бібліотеки добре підходять для вирішення задач машинного навчання (МН), тому для вирішення проблеми цієї роботи використовувались вони обидві. TensorFlow використовувався безпосередньо для створення моделі НМ та налаштування її. Бібліотека Scikit-learn, в свою чергу, використовувалась для попередньої обробки даних та перевірки похибки системи. Таким чином можна сказати, що в роботі було використано кращі сторони обох представлених бібліотек. Також треба відзначити, що бібліотека Scikit-learn значно спростила перевірку багатьох методів МН для вирішення задачі роботи.

### **3.1.3 Вибір засобів Python для попередньої обробки даних**

Для роботи з даними в Python також представлено багато різних бібліотек. Для вирішення задач цієї роботи була обрана бібліотека Pandas. Pandas – це високорівнева бібліотека для аналізу даних [56].

Бібліотека створена для швидкої та зручної роботи з даними, вона має змогу забирати дані з CSV та TSV файлів, також є можливість отримувати дані напряду з SQL баз даних. Pandas дозволяє швидко та просто створювати датасети придатні до роботи з алгоритмами МН. У роботі було використано цю бібліотеку для об'єднання усіх файлів годин та додавання необхідних параметрів до первинних даних.

Також, як було сказано у розділі вище, для первинної обробки даних використовувалась бібліотека Scikit-learn, а саме для нормалізації початкових даних.

Окрім того слід відмітити, що для розуміння характеру початкових даних використовувались бібліотека побудови графіків. Це була бібліотека Matplotlib, що надає широкий спектр різноманітних графічних відображення даних. Проте безпосередньо для розробки вебзастосунку вона не використовувалась.

### 3.1.4 Вибір фреймворку Python для створення вебзастосунку

У мові програмування Python є два найпопулярніших фреймворки для створення вебзастосунків. Перший фреймворк це Django, другий це Flask.

Django — це високорівневий вебфреймворк Python, який заохочує швидкий розвиток і чистий, прагматичний дизайн [57]. Створений досвідченими розробниками, він вирішує значну частину задач веброзробки, тому програміст можете зосередитись на написанні програми без необхідності створювати все заново, також цей фреймворк є безкоштовним та з відкритим вихідним кодом [57].

Django без додаткових налаштувань може:

- реалізувати систему автентифікації користувача;
- система адміністрування вмісту;
- систему псевдо URL для зручної навігації;
- RSS-канали;
- забезпечує надійну систему збереження паролів користувача;
- запобігає проблемі «ін'єкція SQL»;
- запобігає підробці міжсайтових запитів;
- надає змогу легкого масштабування застосунку;
- вбудований вебсервер для відладки та тестування застосунку [57].

Як можна побачити зі списку вище, Django дійсно є універсальною системою для побудови будь-яких вебзастосунків, що має вбудовані рішення для типових проблем. Також треба відзначити, що Django для створення вебзастосунків використовує концепцію Model-View-Controller(MVC), що в свою чергу значно спрощує розробку сайту. Треба додати, що Моделі в Django мають вбудовану можливість працювати з багатьма базами даних за допомогою міграцій, це забезпечує швидку та безпечну роботу з даними у системі. Іншою гарною особливістю фреймворка є його модульність, тобто, якщо треба додати якийсь сторонній допоміжний фреймворк, то це не буде вимагати багато часу.

Натомість Flask – це мікрофреймворк для створення вебзастосунків [58]. «Мікро» називають легковаговий, мінімалістичний фреймворк, що свідомо надає можливість реалізації лише найбазовіших функцій вебзастосунку [58].

За замовчанням функціональність фреймворку дуже мала. Є лише власний вебсервер, шаблонізатор коду та підтримка запитів RESTful, окрім найбазовіших функцій.

Проте слід визначити, що фреймворк є також модульним і суспільство розробило багато додаткових мікрозастосунків для цього фреймворку. Наприклад, модуль flask-login, який за своєю суттю повторює функціональність автентифікації користувача з Django. Треба також додати, що фреймворк не має концепції для створення вебзастосунку такої, як MVC, а також не має системи об'єктно-реляційної проєкції для зручної роботи з базами даних.

Flask добре підходить для створення малих вебзастосунків або мікро-сервісів, проте теоретично має все необхідне для створення великих вебзастосунків, якщо використовувати потрібні модулі. Також він надає можливість гнучкого налаштування системи під конкретні вимоги розробника.

Виходячи з конкретної мети цієї роботи для розробки вебзастосунку був обраний фреймворк Django через те, що це дуже сильно спростить та пришвидшить процес розробки ПЗ.

### **3.1.5 Вибір СУБД**

Задля розробки ПЗ була обрана система управління базою даних (СУБД) PostgreSQL. PostgreSQL – це вільна об'єктно-реляційна система управління базами даних, що базується на мові SQL [59].

СУБД відрізняється високою надійністю та гарною продуктивністю. PostgreSQL підтримує транзакції (ACID), реплікація реалізована вбудованими механізмами, при цьому система розширюється – можна створювати свої типи даних та індексів, а також розширювати поведінку за допомогою мов програмування [59].

Основними перевагами PostgreSQL є:

- підтримка БД необмеженого розміру;
- потужні та надійні механізми транзакцій та реплікації;
- розширювана система вбудованих мов програмування та підтримка завантаження C-сумісних модулів;
- успадкування;
- легка розширюваність [59].

Саме через ці переваги було обрано СУБД PostgreSQL.

### 3.1.6 Вибір редактору коду

Visual Studio Code — це спрощений редактор коду з повною підтримкою всіх стадій розробки таких, як відлагодження, виконання завдань і контроль версій [16]. Редактор спрямований на надання інструментів, які потрібні розробнику для швидкого циклу створення коду та відлагодження, він залишає більш складні робочі процеси для повнофункціональних IDE [16]. VS Code розповсюджується повністю за безкоштовною моделлю, як для персонального використання, так і для комерційного [16].

Особливостями саме цього редактору є:

- дозволяє розробляти, як консольні програми, так і програми з графічним інтерфейсом, у тому числі з підтримкою технології Windows Forms, а також вебсайти, вебпрограми, вебслужби, як у рідному, так і в керованому кодах на всі платформи;
- у редакторі присутні вбудований відладчик, інструменти для роботи з Git та засоби рефакторингу, навігації за кодом, автодоповнення типових конструкцій та контекстної підказки;
- продукт підтримує розробку для платформ ASP.NET і Node.js, і вважається легковажним рішенням, яке дозволяє обійтися без повного інтегрованого середовища розробки;

- редактор підтримує велику кількість різних мов програмування таких, як C++, C#, Python, PHP, JavaScript та інші [17].

### **3.2 Функціональні вимоги до програмного забезпечення**

Метою цієї дипломної кваліфікаційної роботи є створення програмного вебзастосунку для поліпшення моніторингу технічного стану автомобілів з можливістю діагностування технічного стану КП. У рамках цього проекту розробляються вебзастосунок з наступними функціональними вимогам:

- можливість створення облікового запису;
- можливість автентифікації на сайті;
- можливість додавання автомобілів до облікового запису користувача;
- безпечна система збереження даних користувача;
- можливість редагування та видалення доданих користувачем даних;
- можливість додавати до кожного автомобіля свою історію обслуговування;
- можливість діагностування технічного стану КП за чотирма параметрами вібросенсорів;
- можливість адміністрування вебзастосунку.

Діаграму прецедентів наведено на рис. 3.1.



Рисунок 3.1 – Діаграма прецедентів користувача

### 3.3 Проектування бази даних

Повну схему БД зображено на рис. 3.2. Для генерування ER діаграми БД та візуальної роботи з БД було використано безкоштовну версію застосунку DBreaver. Застосунок значно спростив моніторинг БД. DBreaver є програмою для візуального керування БД, що в свою чергу надає змогу легко змінювати системні параметри БД та генерувати звіти.

Основна частина бази даних була створена за допомогою фреймворку Django, через те, що, як описувались вище, фреймворк бере на себе реалізацію найбазовіших функцій вебзастосунку. Усі таблиці, що починаються з «cars», було створено за допомогою міграції. Усі інші було створено автоматично, лише невеликі модифікації проводились із ними.

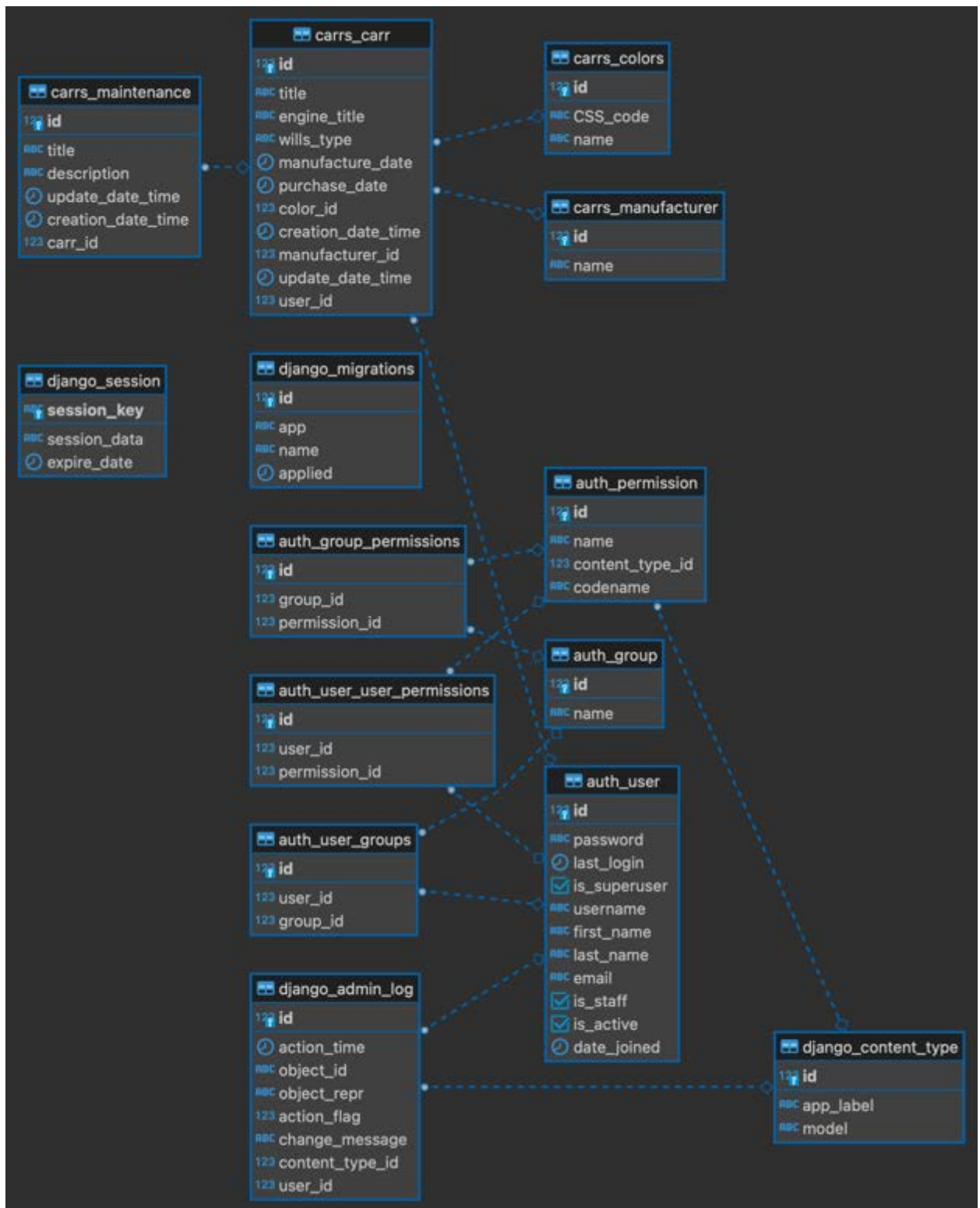


Рисунок 3.2 – Повна схема БД

Основними таблицями вебзастосунку є таблиці «carr» та «maintenance», в них містяться дані про автомобіль та про сервісне обслуговування відповідно. Також було створено дві допоміжні таблиці для зручного та систематизованого

збереження статичних даних таких, як колір автомобілів та виробник. Django створив автоматично декілька важливих таблиць, які треба розглянути детальніше. Наприклад, таблиці «user» та «session». «User» – це таблиця, завдяки якій фактично і відбувається автентифікація користувача на сайті. До неї було додано поле email, щоб надати користувачу можливість відновити пароль. Таблиця «session» в свою чергу надає можливість безпечного доступу до даних користувача, тобто унеможлиблює змінення даних конкретного користувача будь-яким іншим користувачем. Також важливою таблицею є «migrations», тут зберігається уся історія змін бази даних, тобто, якщо щось піде не так, БД можна буде відновити до попередньої версії, або просто подивитись історію додавання нової функціональності та автора.

Усі назви та типи полів зображено на рис. 3.2, також там представлені усі залежності та типи цих залежностей. Змінні названо таким чином, щоб вони пояснювали, за що вони відповідають.

Вся робота саме з таблицями в БД проводилась виключно за допомогою міграції.

### **3.4 Структура програмного забезпечення**

У процесі створення ПЗ за основу файлової структури був узят основний шаблон програмування для фреймворку Django, а саме MVC.

Головну файлову структуру розробленого вебзастосунку зображено на рисунку 3. 3.

Папка `car_management_site` є кореневою папкою проєкту та містить у собі весь розроблений вебзастосунок. Папка, що має таку саму назву, але знаходиться у каталозі головної є «проєктом». Так під час використання фреймворку Django називається папка, де знаходяться всі налаштування. Також на тому самому рівні знаходиться папка `car`, вона є папкою «додатку». Концепція Django передбачає широку модульність та легке масштабування розробленого застосунку, тому

кожен додаток теоретично повинен бути повністю автономним, що і було реалізовано у роботі.

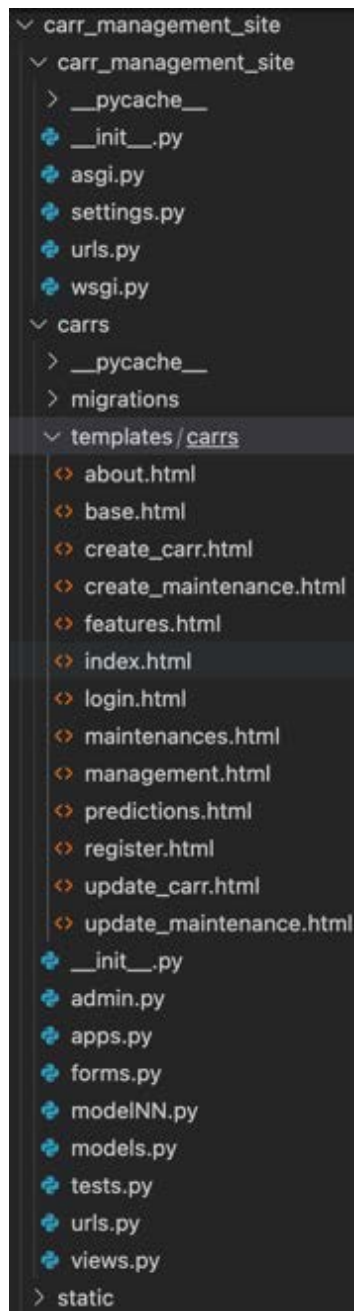


Рисунок 3.3 – Головна структура файлів ПЗ

Фактично можна побачити, що структура сайту складається з лише одного «додатку», тому можна сказати, що цей додаток і є розробленим проєктом.

Опис почнемо з файлів налаштувань у папці «проекту»:

- `settings.py` – відповідає за усі налаштування у вебзастосунку такі, як підключені модулі Django та «додатки», також він відповідає за підключення до БД, підключення зовнішніх модулів та за статичні змінні;
- `urls.py` – містить у собі налаштування загальних URL посилань, наприклад, посилання на адміністративну панель;
- інші файли у каталозі були згенеровані автоматично та розробником не змінювались.

Тепер детальніше розглянемо папку «додатку». Опис треба почати з важливих каталогів, що є в додатку, – таких, як `migrations` та `templates/carrs`.

Папка міграції напряму розробником ніколи не змінюється, проте в ній зберігаються дуже важливі дані, а саме усі маніпуляції з моделями у системі. Фактично модель у системі еквівалентна таблиці у БД. Це зроблено задля зручності та вирішення багатьох проблем «марджингу», тобто не треба співставляти базу даних з моделями, у системі це робиться автоматично за допомогою міграції.

Папка `templates/carrs` фактично є папкою `carrs` у папці `templates`, це зроблено навмисно, саме через модульність фреймворку. У папці зберігаються усі HTML шаблони на, яких ми зупинимось трішки пізніше. Така структура каталогу дає змогу усім папкам з шаблонами фактично знаходитись в одній папці після реалізації застосунку. Тобто після запуску програми, Django шукає папку `templates` у всіх «додатках» та збирає всі знайдені шаблони в одну велику папку `templates`. Цей спосіб зберігання є дуже зручним, якщо застосунок містить більше ніж один додаток, це дає змогу викликати шаблони іншого додатку використовуючи зручний і короткий псевдошлях. Проте саме під час реалізації цього ПЗ, ця особливість скоріше була просто необхідністю ніж гарною перевагою.

Опис файлів у папці «додатку»:

- `admin.py` – це файл адміністрування, в ньому зазвичай реєструються моделі, данні яких в ручному режимі може змінювати адміністратор;

- `apps.py` – це файл налаштування додатку, там зберігається така інформація, як назва додатку в системі та інші параметри;
- `forms.py` – це файл, де створюється і зберігаються класи форм, що будуть використані у застосунку, як приклад форма реєстрації користувача;
- `modelNN.py` – це файл, де зберігається запропонована нейронна мережа та знаходиться механізм завантаження заздалегідь створених після навчання вагів, це зроблено заради покращення швидкодії вебзастосунку;
- `models.py` – це файл, де зберігаються класи всіх моделей вебзастосунку та відношення між ними;
- `tests.py` – це файл для збереження автоматичних тестів системи;
- `urls.py` – це файл, де зберігається налаштування зручних URL адресів, а також функція `view`, що за них відповідає;
- `views.py` – це файл, де зберігаються усі функції представлення вебзастосунку, тобто функції, які відповідають за те, що треба вивести користувачу на екран та на якій сторінці;

Далі треба розібрати HTML шаблони, проте багато з них є майже однаковими та мають незначні відмінності, тому вони будуть об'єднані у групи задля наглядності.

Каталог `templates/carrs` складається з:

- `about.html` – це шаблон де знаходяться інформація для сторінки `Contacts`;
- `base.html` – це базовий шаблон вебзастосунку, він містить у собі `header` та `footer`, який використовується майже на кожній сторінці сайту, окрім домашньої сторінки;
- `create_carr.html` та `update_carr.html` – містять у собі реалізацію форум для додавання та редагування моделі `Carr` відповідно;
- `create_maintenance.html` та `update_maintenance.html` – містять у собі реалізацію форум для додавання та редагування моделі `Maintenance` відповідно;
- `features.html` – це шаблон, де знаходиться інформація для сторінки `Features`;

- index.html – це шаблон головною домашній сторінка, вона є унікальною і не наслідує нічого від base.html;
- login.html та register.html – містять у собі шаблони для входження в обліковий запис та реєстрації нового облікового запису відповідно;
- maintenances.html – це шаблон для відображення сервісних робіт, що проводились з автомобілем користувача;
- management.html – це шаблон менеджменту автомобілів користувача у системі;
- predictions.html – це шаблон, створений для реалізації візуального інтерфейсу діагностування запропонованого методу.

Також є папка static, яку можна знайти на рис. 3.3 у самому кінці переліку, проте в ній зберігаються дуже важливі файли, тому її треба розглянути детальніше (рис.3.4).

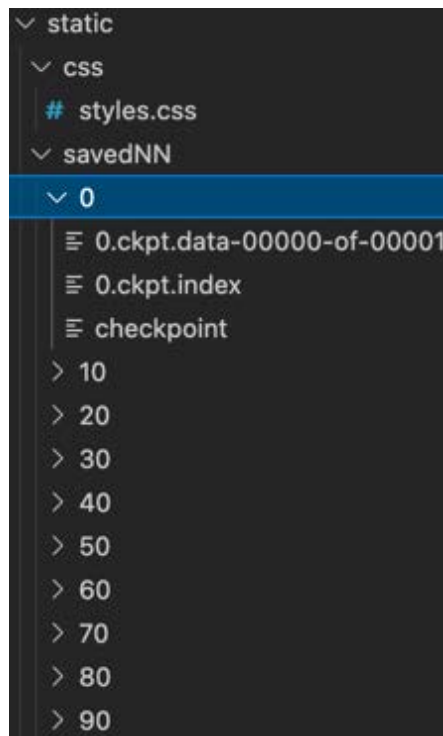


Рисунок 3.4 – Папка static

Окрім важливого файлу CSS стилів – style.css, що використовується у розробленому проєкті, тут також зберігається папка savedNN, в якій знаходяться збережені ваги для кожної окремої завантаженості КП.

Саме до цієї папки та файлів в ній звертається модель в файлі modelNN.py задля отримання вагів під обрану користувачем завантаженість.

### **3.5 Висновки за розділом 3**

У розділі було розглянуто та порівняно можливі мови програмування для вирішення задач, встановлених у цій дипломній кваліфікаційній роботі кваліфікаційні роботи: розглянуто мови програмування Python, R, MATLAB, C# та Java, порівняні їх переваги та недоліки. Також було розглянуто можливі фреймворки для реалізації вебзастосунку. Було розглянуто засоби первинної обробки даних, засоби візуалізації та засоби створення алгоритмів МН.

Остаточного було обрано мову програмування Python, як основну та єдину мову для реалізації вебзастосунку. Було обрано фреймворк Django через його значущі переваги перед іншими аналогами. Було обрано бібліотеки Pandas, Scikit-learn та Matplotlib для передньої обробки даних та їх візуалізації. Також було обрано бібліотеку TensorFlow для реалізації запропонованого методу технічного діагностування. В якості СУБД було обрано PostgreSQL.

Також у розділі було визначено функціональні вимоги для ПЗ та створення діаграми прецедентів для користувача. Окрім того було розроблено структуру БД та розглянуто основну структуру реалізованого ПЗ.

## 4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

### 4.1 Алгоритм функціонування програмного забезпечення

Розроблений під час дипломної кваліфікаційної роботи вебзастосунок загалом надає користувачеві наступні можливості:

- автентифікація у вебзастосунку;
- додавання автомобілів та відомостей про них до свого облікового запису;
- додавання відомостей про сервісне обслуговування автомобілів, що належить до облікового запису користувача;
- діагностування стану КП автомобіля;
- перегляд інформації про вебзастосунок та про метод діагностування стану КП.

Базуючись на цих можливостях, було визначено та побудовано блок-схему, що відображає алгоритму роботи програми (рис. 4.1)

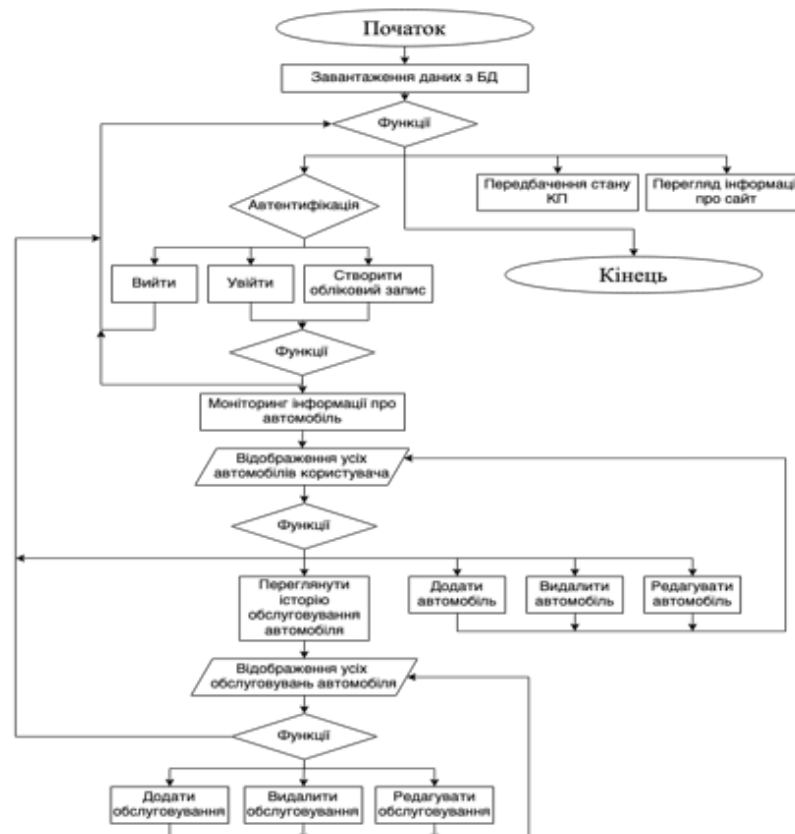


Рисунок 4.1 – Схема функціонування ПЗ

## 4.2 Опис модулів реалізованої системи

### 4.2.1 Опис модулю для роботи з БД

Опис створених моделей для роботи з БД зображено у табл. 4.1.

Таблиця 4.1 – Опис моделей у модулі models.py

| № | Найменування моделі         | Опис моделі                                                                                               |
|---|-----------------------------|-----------------------------------------------------------------------------------------------------------|
| 1 | Carr(models.Model):         | Модель, що містить у собі всі поля задля створення та збереження даних про автомобіль для користувача.    |
| 2 | Maintenance(models.Model):  | Модель, що містить у собі всі поля задля створення та збереження даних про обслуговування для автомобіля. |
| 3 | Colors(models.Model):       | Модель, що містить у собі поля задля збереження та зручного використання можливих кольорів автомобілів.   |
| 4 | Manufacturer(models.Model): | Модель, що містить у собі поля задля збереження та зручного використання можливих виробників автомобілів. |

### 4.2.2 Опис модулю для роботи з формами

Опис створених форм для роботи вебзастосунку зображено у табл. 4.2.

Таблиця 4.2 – Опис форм у модулі forms.py

| № | Найменування форми             | Опис форми                                                                                      |
|---|--------------------------------|-------------------------------------------------------------------------------------------------|
| 1 | NewUserForm(UserCreationForm): | Форма створена задля додавання нового облікового запису у систему.                              |
| 2 | CarrForm(forms.ModelForm):     | Форма створена задля додавання нового автомобіля та редагування існуючого автомобілю у системі. |

## Продовження таблиці 4.2

| № | Найменування форми                | Опис форми                                                                                              |
|---|-----------------------------------|---------------------------------------------------------------------------------------------------------|
| 3 | MaintenanceForm(forms.ModelForm): | Форма створена задля додавання нового обслуговування та редагування існуючого обслуговування у системі. |
| 4 | PredictionForm(forms.Form):       | Форма створена задля передачі даних у модель для подальшого діагностування стану КП                     |

## 4.2.3 Опис модулю для роботи з view

Опис створених view для роботи вебзастосунку зображено у таблиці 4.3.

Таблиця 4.3 – Опис функцій у модулі views.py

| № | Найменування функції | Опис функції                                                                                                                  |
|---|----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| 1 | index(request):      | Функція, що відображує HTML шаблон головної сторінки.                                                                         |
| 2 | contacts(request):   | Функція, що відображує HTML шаблон сторінки з інформацією про розробника.                                                     |
| 3 | features(request):   | Функція, що відображує HTML шаблон сторінки з інформацією про роботу вебзастосунку.                                           |
| 4 | management(request): | Функція, що відображує HTML шаблон сторінки з інформацією про автомобілі користувача, який використовує сайт.                 |
| 5 | signup(request):     | Функція, що відображує форму реєстрації користувача та створює новий обліковий запис, якщо отримані данні валідні.            |
| 6 | signin(request):     | Функція, що відображує форму входження користувача та надає користувачу можливість увійти на сайт, якщо такий присутній у БД. |
| 7 | logoutt(request):    | Функція, що надає змогу користувачеві вийти з облікового запису.                                                              |

## Продовження таблиці 4.3

| №  | Найменування функції                                   | Опис функції                                                                                                                                                               |
|----|--------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8  | <code>create_Carr(request):</code>                     | Функція, що відображує форму створення нового автомобіля для користувача та створює його, якщо отримані дані валідні.                                                      |
| 9  | <code>update_Carr (request, pk):</code>                | Функція, що відображує форму редагування даних автомобіля та оновлює їх, якщо отримані дані валідні.                                                                       |
| 10 | <code>delete_Carr(request, pk):</code>                 | Функція, що видаляє автомобіль з усіма обслуговуваннями з БД.                                                                                                              |
| 11 | <code>maintences_Car(request, pk):</code>              | Функція, що відображує HTML шаблон сторінки з інформацією про обслуговування автомобіля, які належать обраному автомобілю.                                                 |
| 12 | <code>create_Maintenance(request, pk):</code>          | Функція, що відображує форму створення нового обслуговування автомобіля та створює його, якщо отримані дані валідні.                                                       |
| 13 | <code>update_Maintenance(request, carr_id, pk):</code> | Функція, що відображує форму редагування даних обслуговування автомобіля та оновлює їх, якщо отримані дані валідні.                                                        |
| 14 | <code>delete_Maintenance(request, carr_id, pk):</code> | Функція, що видаляє обслуговування з БД.                                                                                                                                   |
| 15 | <code>predictions(request):</code>                     | Функція, що відображує форму введення даних для діагностування стану КП та передає їх на подальшу обробку у розроблену модель. Також відображує результати діагностування. |

**4.2.4 Опис модулю діагностування стану КП**

Модуль діагностування стану КП знаходиться у файлі `modelNN.py` та у функції «`predict(x_test, load)`». У функцію передаються дані для діагностування, а саме 10 наборів даних з датчиків одного автомобіля, другим параметром є параметр навантаження при зборі даних.

Функція складається з декількох частин. У першій частині отримані дання приводяться до тривимірного виду, щоб передати їх у модель для подальшого діагностування стану та обирається найактуальніший набір вагів для моделі, що відповідає обраному параметру навантаження, що обрав користувач

Код описаного вище сегменту:

```
labels = ['Working Ok', 'Need Maintanse']
x_test = np.expand_dims(x_test,0)
BASE_DIR = Path(__file__).resolve().parent.parent
checkpoint_path = os.path.join(BASE_DIR.as_posix(),
f"static/savedNN/{load}/{load}.ckpt")
checkpoint_dir = os.path.dirname(checkpoint_path)
latest = tf.train.latest_checkpoint(checkpoint_dir)
```

Наступним сегментом є частина налаштування самої LSTM моделі код, якої зображено нижче:

```
model = Sequential([
    layers.LSTM(16),
    layers.Dense(1,activation='sigmoid')
])
model.compile(
    optimizer=keras.optimizers.Adam(learning_rate= 0.01),
    loss=keras.losses.BinaryCrossentropy(),
    metrics=['accuracy']
)
model.load_weights(latest)
```

У останньому рядку коду в сегменті вище відбувається завантаження вагів у модель. Це робиться задля значного прискорення роботи вебзастосунку та економії обчислювальних потужностей серверу. Через цей етап зникає етап тренування моделі, бо вона вже натренована заздалегідь.

Останнім сегментом функції є викликання самого діагностування та обробка його до зрозумілого людині результату.

Код сегменту:

```
prediction = model.predict(x_test)
prediction = np.squeeze(prediction,axis=1)
prediction = np.around(prediction, decimals=0).astype(int)
return labels[prediction[0]]
```

Функція повертає текст який потім зображується на сторінці діагностування.

#### 4.2.5 Опис модулю URL маршрутизації

Файл `urls.py` містить у собі список «`urlpatterns`», який складається з об'єктів типу `URLPattern`. У списку перший параметр відповідає за зручний вид URL для користувача, другий за функцію `view`, що викликає перехід по цьому URL, а третій за назву сторінки у коді вебзастосунок. Список розроблених `URLPattern` представлений нижче:

- `path("", views.index, name='home');`
- `path('contacts', views.contacts, name='contacts');`
- `path('features', views.features, name='features');`
- `path('management', views.management, name='management');`
- `path('register', views.signup, name='signup');`
- `path('login', views.signin, name='signin');`
- `path('logout', views.logoutt, name='signout');`
- `path('create_car', views.create_Carr, name='create_car');`
- `path('update_car/<str:pk>', views.update_Carr, name='update_car');`
- `path('delete_car/<str:pk>', views.delete_Carr, name='delete_car');`
- `path('maintenances_car/<str:pk>', views.maintences_Car, name = 'maintenances_car');`
- `path('create_maintenance/<str:pk>', views.create_Maintence, name = 'create_maintenance');`

- `path('update_maintenance/<str:carr_id>/<str:pk>', views.update_Maintenance, name = 'update_maintenance');`
- `path('delete_maintenance/<str:carr_id>/<str:pk>', views.delete_Maintenance, name = 'delete_maintenance');`
- `path('predictions', views.predictions, name='predictions').`

Також є зовнішній файл маршрутизації, який за ієрархією є вищим від додатку Django, він так само називається `urls.py` та містить:

- `path('admin/', admin.site.urls)` – маршрутизація до загальної адміністрації панель вебзастосунку;
- `path(' ', include('carrs.urls'))` – переадресація порожнього рядку на додаток Django.

#### 4.2.6 Опис модулю HTML шаблонів

Детальний опис файлів та те за, що кожен з них відповідає було представлено у розділі 3, у цьому пункті буде доречним конкретизувати лише деякі аспекти розроблених шаблонів. Під час розробки активно використовувався вбудований шаблонізатор фреймворку Django – Jinja2. Завдяки використанню цього шаблонізатору була створена архітектура, при якій є базовий файл з загальними елементами для всього сайту, а саме файл `base.html`, наприклад навігаційна панель. Усі інші файли розробленого вебзастосунку, окрім файла головної сторінки, лише розширюють базовий шаблон, що надає змогу значно зменшити написаний код та час розробки. Також це дає багато інших переваг, наприклад посилання на інші сторінки розробленого вебзастосунку відбуваються завдяки псевдо посиланням таким, як наприклад:

```
<a class="nav-link {% if url_name == 'features' %}active{% endif %}" href="{% url 'features' %}">
```

У фрагменті коду вище зображено, що окрім використання псевдо посилання `href="{% url 'features' %}"` було використано також конструкцію



#### 4.3 Висновки за розділом 4

У четвертому розділі дипломної кваліфікаційної роботи було розроблено та побудовано блок-схему функціонування створеного вебзастосунку. Було проведено опис усіх важливих модулів розробленого вебзастосунку, а саме:

- модуль для роботи з БД;
- модуль для роботи з формами;
- модуль для роботи з view;
- модуль діагностування стану КП;
- модуль HTML шаблонів.

Було розглянуто функції та методи, які є головними у кожному з цих модулів. Було конкретизовано, за що відповідає кожен з них.

## **5 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ**

### **5.1 Призначення програми**

Розроблений під час виконання кваліфікаційної роботи вебзастосунок призначений для зручного зберігання та управління інформацією про технічний стан автомобіля. Застосунок надає змогу зберігати інформацію, як про автомобілі так і про обслуговування, що з ними проводились. Також присутня можливість діагностування стану КП транспортного засобу.

Областю застосування вебзастосунку є менеджмент стану автомобіля. Користувачеві надається змога входити у свій або створювати новий обліковий запис, додавати, видаляти чи редагувати інформацію про автомобілі, а також додавати, видаляти чи редагувати інформацію про сервісне обслуговування кожного з автомобілів користувача, також надається можливість діагностування стану КП за допомогою десяти різних наборів даних, зібраних з чотирьох різних, протилежно направлених вібраційних сенсорів поблизу КП.

### **5.2 Умови виконання програми**

Умовами виконання програми для налаштування серверної частини є наявність встановленого на машині Python версії 3.8.12, наявність необхідного набору встановлених бібліотек, а саме:

- Django;
- Sklearn;
- Tensorflow;
- Numpy;
- crispy\_bootstrap5.

Та наявність усіх допоміжних пакетів, що потрібні для коректної роботи вищезазначених бібліотек. Також необхідна 64 бітна операційна система, 512Мб вільного дискового простору, СУБД PostgreSQL 14, та наявність GPU.

Умовами виконання програми для кінцевого користувача є наявність будь-якого браузеру з підтримкою HTML5. Розроблений вебзастосунок призначений для роботи як на комп'ютерах (стаціонарних та ноутбуках), так і на смартфонах під управлінням будь-якої операційної системи.

### 5.3 Виконання програми

Розроблений вебзастосунок працює з обліковим записом користувача, тому функціональність без нього дуже обмежена. Головною сторінкою вебзастосунку є сторінка привітання та можливих варіантів автентифікації, а саме реєстрації чи входу в обліковий запис, як видно на рис. 5.1.



Рисунок 5.1 – Домашня сторінка вебзастосунку

Як можна чітко побачити на рис. 5.1 окрім можливих варіантів автентифікації надається також змога навігації по сайту, а саме можливість переходу на сторінки Management, Predictions, Features та Contacts. Останні три

сторінки не вимагають наявності облікового запису у користувача. Задля прикладу розглянемо сторінку Features (рис. 5.2).

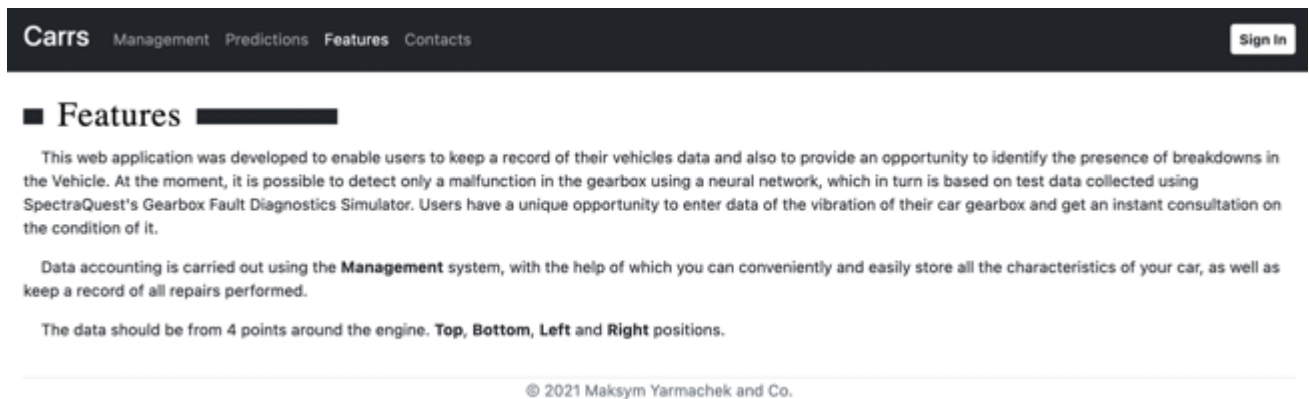


Рисунок 5.2 – Сторінка Features

На рис. 5.2 окрім інформації про розроблений сайт та його можливості зображено, що користувачеві надається можливість навігації по вебзастосунку, яка буде представлена на усіх інших сторінках сайту. Також можна побачити, що в правому верхньому кутку сторінки знаходиться кнопка Sing In, вона свою чергу надає користувачеві можливість увійти у свій обліковий запис. Якщо натиснути на цю кнопку, то користувач опиниться на сторінці входу (рис. 5.3).

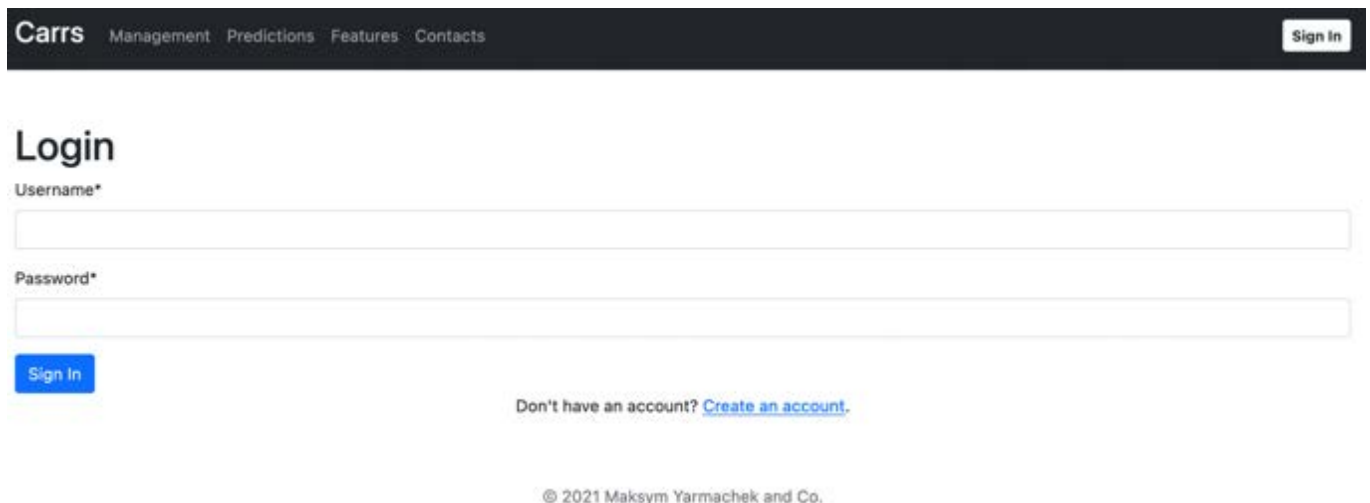


Рисунок 5.3 – Сторінка входу в обліковий запис

Проте, якщо у користувача наразі немає облікового запису у вебзастосунку то йому надається можливість перейти на сторінку створення такого, а саме за допомогою посилання «Create an account» (рис. 5.4).

Рисунок 5.4 – Сторінка створення облікового запису

На цій сторінці також присутня можливість перейти на сторінку «входу», якщо користувач все ж таки має обліковий запис. Після проходження реєстрації або входу на сайт користувача буде автоматично переведено на сторінку Management, а кнопка Sing In у правому верхньому кутку заміниться на кнопку Sing Out з інформацією про наявного користувача, що надає змогу виходу з облікового запису (рис. 5.5).

The screenshot shows a web application interface for 'Carrs Management'. The top navigation bar includes links for 'Carrs', 'Management', 'Predictions', 'Features', and 'Contacts'. On the right, there is a user profile '@testuser' and a 'Sign Out' button. The main content area is titled 'New Carr' and displays the following information:

- Manufacturer: **Ford**
- Manufacturer Date: **Jan. 5, 2021**
- Purchase Date: **Nov. 10, 2021**
- Color: **Red**

Below the information, there are three buttons: 'Full Info' (green), 'Maintenances' (blue), and 'Delete' (red). At the bottom left, there is a blue 'Add Carr' button. At the bottom center, there is a small copyright notice: '© 2021 Maksym Yarmachek and Co.'

Рисунок 5.5 – Сторінка Management

Як зображено на рис. 5.5, саме тут буде представлена можливість переглядати та додавати нові автомобілі до облікового запису. Кнопки Full Info, Maintenances та Delete присутні для кожного доданого автомобіля користувача та відповідають за редагування автомобіля, перегляд обслуговування автомобіля, видалення автомобіля та всієї інформації про нього з системи відповідно. Також на сторінці присутня кнопка Add Carr, яка відповідає за створення нового автомобіля у системі. Розглянемо форму створення нового автомобіля детальніше (рис. 5.6). На рис. 5.6 представлені усі наявні користувачеві поля для заповнення нової інформації про автомобіль, поля відмічені значком зірочки є обов'язковими для заповнення. У формі представлено декілька видів полів, саме поля звичайного тексту, поля вибору та поля дати. Треба зупинитись трішки детальніше на полях дати та вибору. Поля вибору напряду беруть дані з БД та надають змогу обрати колір чи виробника, які представлені у системі. А поля введення дати обладнані спеціальним селектором дати для зручності введення користувачем (рис 5.7).

**Carrs** Management Predictions Features Contacts @testuser Sign Out

## Add New Car

Title\*

Engine Title

Wills Type

Manufacturer\*

Manufacture Date\*

Purchase Date\*

Color\*

Add Car

© 2021 Maksym Yarmachek and Co.

Рисунок 5.6 – Сторінка створення нового автомобіля

Manufacture Date\*

December 2021

| Su | Mo | Tu | We | Th | Fr | Sa |
|----|----|----|----|----|----|----|
|    |    |    | 1  | 2  | 3  | 4  |
| 5  | 6  | 7  | 8  | 9  | 10 | 11 |
| 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| 26 | 27 | 28 | 29 | 30 | 31 |    |

Рисунок 5.7 – Селектор дати

Після додавання всієї необхідної інформації та проходження валідації даних, користувача буде переадресовано на сторінку Management, де він може побачити тільки-но створений їм новий автомобіль.

Після натискання на кнопку Full Info буде показано екран схожий на той, що зображено на рис. 5.6, проте кнопка Add Car буде замінена на Update, та дані обраного автомобіля просто оновляться.

Розглянемо сторінку Maintenances детальніше (рис. 5.8).

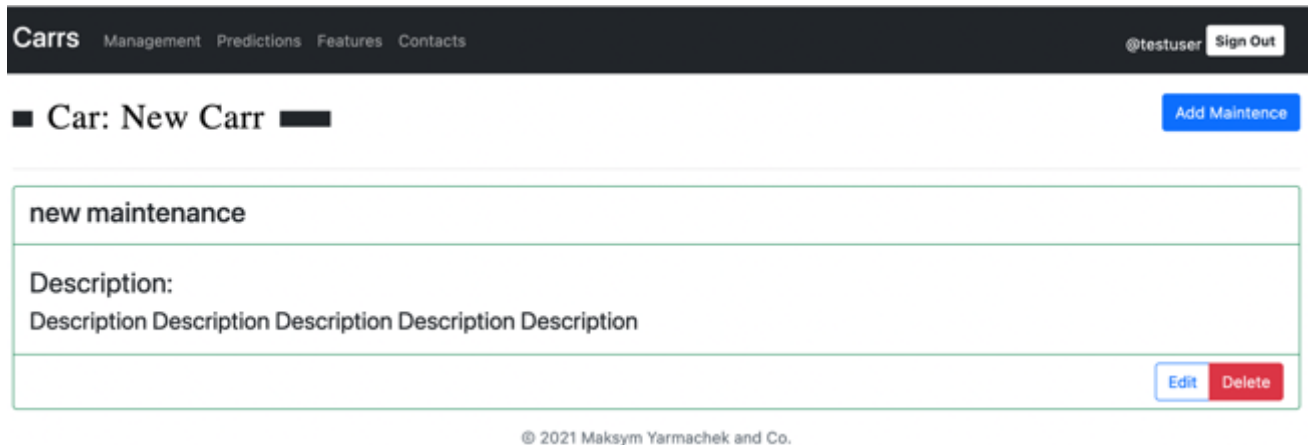


Рисунок 5.8 – Сторінка Maintenances

На рис. 5.8 представлено загальний вид сторінки, де знаходиться інформація про всі обслуговування автомобіля. На сторінці присутні загально кнопка Add Maintenances, після натискання якої користувач опиниться на сторінці додавання інформації про нове обслуговування автомобіля (рис. 5.9).

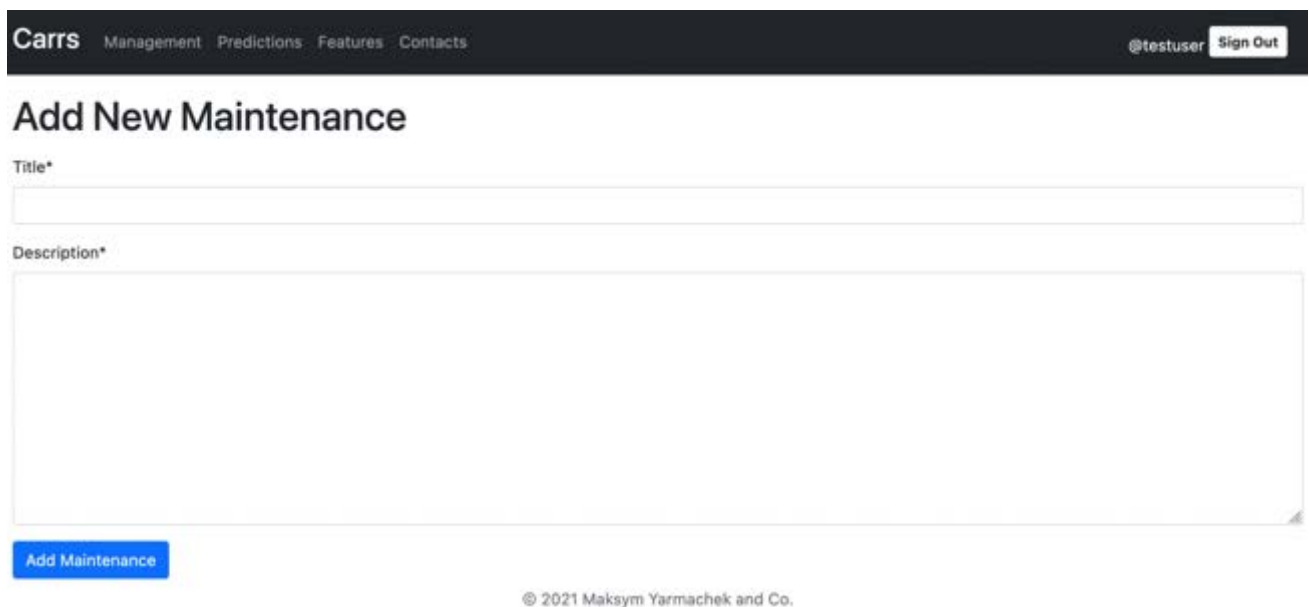


Рисунок 5.9 – Сторінка створення нового обслуговування

Також на рис. 5.8 присутні дві кнопки, що будуть представлені для кожного доданого обслуговування, а саме кнопки Edit та Delete. Кнопка Edit переведе користувача на сторінку оновлення інформації про обслуговування.

Сторінка заснована на тій самій формі, що й сторінка додавання нового обслуговування, тому візуально вона відрізняється лише назвою сторінки та назвою кнопки, все інше залишається таким самим (рис. 5.10).

**Carrs** Management Predictions Features Contacts @testuser Sign Out

## Edit Maintenance

Title\*

new maintenance

Description\*

Description Description Description Description Description

Edit Maintenance

© 2021 Maksym Yarmachek and Co.

Рисунок 5.10 – Оновлення інформації обслуговування

Кнопка Delete відповідає за видалення обслуговування з системи.

У розробленому вебзастосунку також присутня сторінка діагностування стану КП (рис. 5.11).

**Carrs** Management Predictions Features Contacts @testuser Sign Out

## Predict gear box state:

Engine Load\*

0%

Vibration Data\*

5.70718,-7.20074,-4.54852,-5.52796  
 -13.561,-4.68453,1.3053,-7.91893  
 3.52773,-0.159441,3.04621,-0.923214  
 12.5466,-8.62812,11.9866,-5.01071  
 6.94914,-0.179236,8.2617,-3.23649  
 5.80352,-1.55319,-2.47599,-6.307  
 -1.08165,3.97097,-7.4898,-6.00217  
 6.60639,-5.2884,9.67397,4.21278  
 -1.19753,5.92426,-3.85929,1.47045  
 0.645542,7.96934,-5.79981,7.4855

Enter 10 data samples. 1 per row without spaces

Predict

© 2021 Maksym Yarmachek and Co.

Рисунок 5.11 – Сторінка діагностування стану КП

Сторінка діагностування не потребує автентифікації користувача у системі через те, що вона не прив'язана до конкретного автомобіля, та будь-яка людина з потрібними даними вібраційних сенсорів може скористатися запропонованим методом. На рис. 5.11 представлено два поля, обидва є обов'язковими для заповнення, перше є селектором навантаження автомобіля, а друге полем для введення даних вібраційних сенсорів. У полі Vibration Data наявна підказка, яка показує, яким саме чином дані повинні бути введені у поле.

Після введення усіх потрібних даних та натискання на кнопку Predict сторінка оновиться та у верхній частині з'явиться додатковий текст, як показано на рис. 5.12.

## Predict gear box state:

**NN Prediction is: "Your gearbox Working Ok".**

Рисунок 5.12 – Діагностування стану КП

Присутні два варіанти відповіді для користувача перший – «Working OK», що свідчить про те, що з КП все гаразд (рис. 5.12) та другий – «Need Maintains», що свідчить про те, що КП потрібен ремонт через те, що в ній присутні зламані зубці.

### 5.4 Експериментальне дослідження розробленого методу

Як було описано в розділі 2.4 матеріалом для роботи стала вибірка даних з симулятора SpectraQuest. Було створено десять наборів вагів для кожного з представлених можливих навантажень. Задля перевірки точності діагностування різних методів вибірка для кожного з можливих навантажень розділялась у співвідношенні 80% та 20%, де 80% – це навчальні дані, а останні 20% – це дані для перевірки методу.

Задля порівняння різних методів класифікації використовувалась вибірка даних з навантаженням 30%. Для перевірки точності використовувався параметр загальної точності (ассигасу), який в свою чергу був реалізований за допомогою бібліотеки Sklearn, а саме методу ассигасу\_score. Деякі методи розглянуті у розділі 2 не були протестовані під час експериментального дослідження, через їх значущі недоліки саме для вирішення конкретно задачі цієї роботи. У табл. 5.1 наведено результати класифікації за допомогою усіх протестованих методів.

Таблиця 5.1 – Порівняння методів за точністю діагностування

| Метод                | Загальна точність |
|----------------------|-------------------|
| Запропонований метод | 0.96              |
| ANN                  | 0.68              |
| KNN                  | 0.63              |
| LSTM                 | 0.59              |
| RF                   | 0.58              |
| NB                   | 0.56              |
| RNN                  | 0.55              |
| SVM                  | 0.50              |

З таблиці 5.1 можна безперечно зробити висновок, що запропонований метод працює якнайкраще для класифікації стану КП. Загальна точність запропонованого методу склала 96%, коли найближчим за точністю методом став звичайний ANN з точністю діагностування 68%. Наявна перевага запропонованого методу складає 28%. В абсолютному вираженні

З таблиці вище можна також зробити висновок, що найкращими методами класифікація для вирішення задачі цієї роботи є методи машинного навчання, проте у трійці лідерів також знаходиться метод KNN, який не є таким. Це відбулося через велику зашумленість початкових даних, яку було описано в розділі 2. Метод KNN є малочутливим до дуже зашумлених даних на відміну від інших.

У розглянутих в розділі 1 аналогах запропонованого вебзастосунку та у джерелах, виділених під час дослідження методів класифікації в розділі 2, не було виявлено повноцінних аналогів запропонованого методу для вирішення задачі класифікації стану КП.

Загалом реалізації дієвого методу технічного діагностування КП вимагає більше досліджень та застосування напрацювань цієї роботи у комерційному секторі. Результати цієї роботи можуть бути використані для подальших досліджень.

Покращити результати запропонованого методу можна шляхом:

- суттєвого збільшення кількості навчальних даних, що надасть змогу будувати набори більшої кількості, що в свою чергу дієво відобразиться на результатах діагностування;
- збільшення кількості джерел навчальних даних, наприклад дані з реальних автомобілів.

## **5.5 Висновки за розділом 5**

У 5-му розділі було визначено призначення розробленого вебзастосунку, визначено умови виконання розробленого ПЗ, як для налаштування серверної частини, так і для кінцевого користувача. Також у розділі було наведено та проілюстровано приклад виконання програми. В останній частині розділу було наведено експериментальне дослідження запропонованого методу технічного діагностування, було порівняно розроблений метод з іншими, було наведено можливі поліпшення запропонованого методу та області його використання.

## ВИСНОВКИ

В ході виконання кваліфікаційної роботи магістра було розглянуто проблему технічного діагностування деталей автомобілів, а саме стану КП. Було наведено актуальність цієї проблеми, представлені аналоги, що вирішують схожі задачі технічної діагностики. Також було зроблено висновки про функціональність та працездатність розглянутих аналогів.

У процесі роботи було проаналізовано усі найпопулярніші методи класифікації технічного стану об'єкту, було виявлено їх переваги та недоліки, та порівняно їх між собою задля вирішення задачі цієї роботи. Методи, які було розглянуто в ході цієї роботи: Decision tree, Artificial neural networks, Support vector machines, Naïve Bayes, Linear discriminant analysis, K-Nearest Neighbours, Self-organizing maps, Growing neural gas, Random forest. Було розглянуто та проаналізовано матеріали, що були використані у ході виконання цієї роботи, також було наведено власний метод вирішення поставленої задачі класифікації технічного стану КП. Запропонований метод базується на методі МН, а саме LSTM, проте значно відрізняється у частині обробки даних, що проходять через мережу LSTM.

Було порівняно та проаналізовано усі найпопулярніші мови програмування, що задовольняють вимогам цієї дипломної кваліфікаційної роботи. Було обрано мову програмування Python, як мову, за допомогою якої можна одразу побудувати вебзастосунок та реалізувати запропонований метод. Було визначено необхідні бібліотеки для виконання задачі цієї роботи, а саме: Scikit-learn, Tensorflow, Numpy. Було обрано фреймворк, за допомогою якого було реалізовано вебзастосунок, а саме Django. Було обрано СУБД PostgreSQL для управління БД та обрано редактор коду для реалізації ПЗ.

У ході роботи було визначено функціональні вимоги до розробленого застосунку, представлено діаграму прецедентів, розроблено структуру БД та розроблено структуру самого вебзастосунку.

Також під час виконання завдання дипломної кваліфікаційної роботи було розроблено та представлено бсхему функціонування вебзастосунку. Було розроблено та описано всі основні модулі ПЗ, наведено приклади їх функціонування та особливості роботи деяких з них.

У ході роботи також було визначено призначення розробленого вебзастосунку, було описано умови виконання програми, як для серверної частини так і для кінцевого користувача. Також було проведено експериментальне дослідження запропонованого методу та виявлено його значущу перевагу над іншими методами.

Наукова новизна дипломної роботи полягає у розробленому методі технічного діагностування деталей автомобілів, що характеризується використанням моделі класифікації на основі нейронної мережі архітектури LSTM з визначенням спеціальної попередньої обробки даних, що дозволяє зменшити витрати на виявлення несправностей у КП, включаючи часові та фінансові витрати.

Розроблений метод дозволяє розв'язати задачу технічного діагностування стану КП автомобіля без фізичного відділення КП від автомобіля.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. The principals of technical diagnostics [Electronic resource]. – Access mode : <https://articlekz.com/en/article/29158>
2. Selig, M. A modern diagnostic approach for automobile systems condition monitoring [Electronic resource] / M. Selig, Z. Shi, A. Ball and K. Schmidt // Journal of Physics : Conference Series. – 2012. – Vol. 364. – P 1–6. Access mode : <https://iopscience.iop.org/article/10.1088/1742-6596/364/1/012013/pdf>
3. XDS 1000 [Electronic resource]. – Access mode : <https://www.avlditest.com/index.php/en/vehicle-diagnostics-xds-1000.html>
4. Folder-AVL-DiTEST-Diagnostics-PA7416E-2018 [Electronic resource]. – Access mode : <https://www.avlditest.com/index.php/en/vehicle-diagnostics-mds-drive-188.html?file=files/content/downloads/Folder%20und%20Prospekte/prospekte/prospekte-2018/Folder-AVL-DiTEST-Diagnostics-PA7416E-2018.pdf>
5. ScanTool [Electronic resource]. – Access mode : <https://www.autoenginuity.com/scantool/>
6. OBD-II diagnostic software for Windows, Mac and Linux | OBD Auto Doctor [Electronic resource]. – Access mode : <https://www.obdautodoctor.com/tour>
7. Presentation of the OBD and the OBD2 diag port – Outils OBD Facile [Electronic resource]. – Access mode : <https://www.outilsobdfacile.com/obd-presentation.php>
8. PCMSCAN – Palmer Performance Engineering, Inc. [Electronic resource]. – Access mode : <https://www.palmerperformance.com/products/pcmscan/index.php>
9. Do It Yourself – Model S | Tesla [Electronic resource]. – Access mode : <https://www.tesla.com/support/do-it-yourself-model-s>
10. Duer, S. Training of an artificial neural network in the diagnostic system of a technical object [Text] / S. Duer, K. Zajkowski, I. Płocha // Neural Computing & Applications. – 2013. – Vol. 22. – P. 1581–1590.
11. Uduak Idio, Akpan Review of classification algorithms with changing inter-class distances [Electronic resource] / Uduak Idio Akpan, Andrew Starkey //

Machine Learning with Applications. – 2021. – Vol. 4. P. 1-9. Access mode : [https://www.sciencedirect.com/science/article/pii/S2666827021000128?dgcid=raven\\_sd\\_aip\\_email](https://www.sciencedirect.com/science/article/pii/S2666827021000128?dgcid=raven_sd_aip_email)

12. 7 Types of Classification Algorithms [Electronic resource]. – Access mode : <https://analyticsindiamag.com/7-types-classification-algorithms/>

13. Advantages of Deep Learning | disadvantages of Deep Learning [Electronic resource]. – Access mode : <https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-Deep-Learning.html>

14. The advantages and disadvantages of deep learning [Electronic resource]. – Access mode : <https://smebook.eu/knowledge-base/deep-learning/the-advantages-and-disadvantages-of-deep-learning/>

15. Технічна діагностика автомобілів. Основні поняття і означення [Електронний ресурс]. – Режим доступу : [https://studopedia.su/2\\_41163\\_tehnichna-diaagnostika-avtomobiliv.html](https://studopedia.su/2_41163_tehnichna-diaagnostika-avtomobiliv.html)

16. Visual Studio Code [Electronic resource]. – Access mode : <https://code.visualstudio.com/docs/supporting/faq>

17. Visual Studio Code: обзор, отзывы, аналоги, интеграция, сайт [Электрон. ресурс]. – Режим доступа : <https://bizzapps.ru/p/vs-code/>

18. Jijo Bahzad Classification Based on Decision Tree Algorithm for Machine Learning [Text] / Jijo Bahzad, Mohsin Abdulazeez, Adnan // Journal of Applied Science and Technology Trends. – 2021. – Vol. 2. – P. 20-28.

19. Damanik I. S. Decision Tree Optimization in Algorithm Using Genetic Algorithms [Text] / I. S. Damanik, A. P. Windarto, A. Wanto, S. R. Andani, W. Saputra // Journal of Physics: Conference Series. – 2019. – Vol. 1225 – P. 205-213.

20. Gavankar S. S. Eager decision tree [Text] / S. S. Gavankar, S. D. Sawarkar, // 2nd International Conference for Convergence in Technology (I2CT) : Mumbai. – 2017. – pp. 837–840.

21. Swain, P. H. The decision tree classifier: Design and potential [Text] / P. H. Swain, H. Hauska // IEEE Transactions on Geoscience Electronics. – 1977. – Vol. 15, No. 3. – P. 142-147.

22. Dey, A. Machine learning algorithms: a review [Text] / A. Dey // International Journal of Computer Science and Information Technologies. – 2016. – Vol. 7, No. 3. – P. 1174–1179
23. Mrva, J. Decision Support in Medical Data Using 3D Decision Tree Visualisation [Text] / J. Mrva, Š. Neupauer, L. Hudec, J. Ševcech, P. Kapec // E-Health and Bioengineering Conference. – 2019. – Vol. 5. – P. 21-24.
24. Top 5 advantages and disadvantages of Decision Tree Algorithm [Electronic resource]. – Access mode : <https://dhirajkumarblog.medium.com/top-5-advantages-and-disadvantages-of-decision-tree-algorithm-428ebd199d9a>
25. Srivastava Durgesh. Data classification using support vector machine [Electronic resource] / Srivastava Durgesh, Bhambhu Lekha // Journal of Theoretical and Applied Information Technology. – 2010. – Vol. 12. – P. 1-7. Access mode : [https://www.researchgate.net/publication/285663733\\_Data\\_classification\\_using\\_support\\_vector\\_machine](https://www.researchgate.net/publication/285663733_Data_classification_using_support_vector_machine)
26. SVM | Support Vector Machine Algorithm in Machine Learning [Electronic resource]. – Access mode : <https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/>
27. Vapnik, V. The Nature of Statistical Learning Theory / V. Vapnik // Springer-Verlag : NY. – 1995.
28. Top 4 advantages and disadvantages of Support Vector Machine or SVM [Electronic resource]. – Access mode : <https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107>
29. Text Classification Using Naive Bayes: Theory & A Working Example [Electronic resource]. – Access mode : <https://towardsdatascience.com/text-classification-using-naive-bayes-theory-a-working-example-2ef4b7eb7d5a>
30. Naive Bayes Explained: Function, Advantages & Disadvantages, Applications in 2021 [Electronic resource]. – Access mode : <https://www.upgrad.com/blog/naive-bayes-explained/>
31. Classification depend on linear discriminant analysis using desired outputs [Text] / S. Nasar Aldian, A. Nuredin, J. Ibrahim, A. Omar // 17th International

Conference on Sciences and Techniques of Automatic Control and Computer Engineering. – 2016. – Vol. 10. – P. 328–332.

32. Enhanced Direct Linear Discriminant Analysis for Feature Extraction on High Dimensional Data [Text] / A. K. Qin, S. Y. M. Shi, P. N. Suganthan, M. Loog // Proceedings of the Twentieth National Conference on Artificial Intelligence. – 2005. – P. 851–855.

33. Pros and Cons of popular Supervised Learning Algorithms [Electronic resource]. – Access mode : <https://medium.com/analytics-vidhya/pros-and-cons-of-popular-supervised-learning-algorithms-d5b3b75d9218>

34. Padraig, C. k-Nearest neighbour classifiers [Electronic resource] / C. Padraig, D. Sarah // ACM Computing Surveys. – 2007. – Vol. 54. – P. 1–17. Access mode : [https://www.researchgate.net/publication/228686398\\_k-Nearest\\_neighbour\\_classifiers](https://www.researchgate.net/publication/228686398_k-Nearest_neighbour_classifiers)

35. Mahdi, V. Classification Ability of Self Organizing Maps in Comparison with Other Classification Methods [Text] / V. Mahdi, Kompany-Zareh Mohsen // Communications in Mathematical and in Computer Chemistry. – 2013. – Vol. 70. – P. 29–44.

36. Kohonen, T. Self-Organizing Maps / T. Kohonen, // Springer : New York. – 2001.

37. Melssen, W. Supervised Kohonen networks for classification problems [Text] / W. Melssen, R. Wehrens, L. Buydens // Chemom. Intl. Lab. – 2006. – Vol. 83. – P. 99–113.

38. Fort, Jean-Claude Advantages and drawbacks of the Batch Kohonen algorithm [Electronic resource] / Jean-Claude Fort, P. Letremy, M. Cottrell // Institut Elie Cartan. – 2013. – P. 1–10. Access mode : <https://samos.univ-paris1.fr/archives/ftp/preprints/samos158.pdf>

39. Fišer, D. Growing neural gas efficiently [Text] / D. Fišer, J. Faigl, M. Kulich // Neurocomputing. – 2013. – Vol. 104. – P. 72–82.

40. Нестандартная кластеризация 5: Growing Neural Gas [Электрон. ресурс]. – Режим доступа : <https://habr.com/ru/post/340360/>

41. Top 10 Deep Learning Algorithms You Should Know in 2021 [Electronic resource]. – Access mode : <https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm>
42. What are Recurrent Neural Networks? | IBM [Electronic resource]. – Access mode : <https://www.ibm.com/cloud/learn/recurrent-neural-networks>
43. NLP Essential Guide: Convolutional Neural Network for Sentence Classification [Electronic resource]. – Access mode : <https://cnvrg.io/cnn-sentence-classification/>
44. Yadav, S. S. Deep convolutional neural network based medical image classification for disease diagnosis [Electronic resource] / S. S. Yadav, S. M. Jadhav, // Journal of Big Data. – 2019. – Vol. 6. No. 113. – P. 1–18. Access mode : <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0276-2>
45. Yamashita, R. Convolutional neural networks: an overview and application in radiology [Text] / R. Yamashita, M. Nishio, R. K. G. Do // Insights into Imaging. – 2018. – Vol. 9. – P. 611–629.
46. Goodfellow, I. J. Explaining and harnessing adversarial examples [Electronic resource] / I. J. Goodfellow, J. Shlens, C. Szegedy // Published as a conference paper at ICLR. – 2015. – Vol. 3. – P. 1–11. Access mode : <https://arxiv.org/pdf/1412.6572.pdf>
47. Multi Class Text Classification with Keras and LSTM [Electronic resource]. – Access mode : <https://djajafer.medium.com/multi-class-text-classification-with-keras-and-lstm-4c5525bef592>
48. Understanding LSTM Networks [Electronic resource]. – Access mode : <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>
49. Gearbox Fault Diagnosis [Electronic resource]. – Access mode : <https://www.kaggle.com/brjapon/gearbox-fault-diagnosis>
50. SpectraQuest Inc. [Electronic resource]. – Access mode : <https://spectraquest.com/>
51. Спажев, О. М. Порівняльний аналіз найпопулярніших мов програмування в області штучного інтелекту, машинного навчання та глибинного

навчання / Спажев Олександр Миколайович. – Вінницький національний технічний університет, 2021. – 3 с.

52. 11 Найпопулярніших програмних засобів машинного навчання в 2021 році [Електронний ресурс]. – Режим доступу : <https://uk.myservername.com/11-most-popular-machine-learning-software-tools-2021>

53. Різниця між машинним навчанням Python та R (Технологія) | Різниця між схожими об'єктами, словами і термінами [Електронний ресурс]. – Режим доступу : <https://ua.sawakinome.com/articles/technology/difference-between-python-and-r-machine-learning-3.html>

54. Getting Started — scikit-learn 1.0.1 documentation [Electronic resource]. – Access mode : [https://scikit-learn.org/stable/getting\\_started.html](https://scikit-learn.org/stable/getting_started.html)

55. Why TensorFlow [Electronic resource]. – Access mode : <https://www.tensorflow.org/about>

56. Введение в pandas: анализ данных на Python [Электрон. ресурс]. – Режим доступа : <https://khashtamov.com/ru/pandas-introduction/>

57. The web framework for perfectionists with deadlines [Electronic resource]. – Access mode : <https://www.djangoproject.com/>

58. Плюсы и минусы фреймворка Flask [Электрон. ресурс]. – Режим доступа : <https://tproger.ru/articles/pochemu-vam-stoit-vybrat-frejmwork-flask-dlja-svoego-sledujushhego-proekta/>

59. PostgreSQL — популярная свободная объектно-реляционная система управления базами данных [Электрон. ресурс]. – Режим доступа : <https://web-creator.ru/articles/postgresql>

**ДОДАТОК А**  
**Технічне завдання**

### **A.1 Підстава для розробки**

Підставою для розробки служить завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація методів технічного діагностування деталей автомобілів», затверджене на засіданні кафедри програмних засобів за протоколом №435 від 5 листопада 2021 року.

### **A.2 Ціль розробки та її призначення**

Розробка направлена на створення вебзастосунку, що надає змогу користувачеві зручно та швидко вести облік наявних автомобілів та історію проведених сервісних обслуговувань з ними, з можливістю діагностування технічного стану КП автомобіля без розбору останнього. Призначенням вебзастосунку, що розробляється, є надання користувачеві зручного засобу моніторингу стану автомобіля та надання засобів швидкого технічного діагностування стану КП транспортного засобу.

### **A.3 Основні вимоги до програмного продукту**

Програмне забезпечення, що розробляється, повинне:

- функціонувати для будь-якого браузеру з підтримкою HTML5;
- надавати можливість додавання, видалення та редагування інформації про автомобілі користувача;
  - надавати можливість додавання, видалення та редагування інформації про сервісні обслуговування, що проводились з автомобілем;
- надавати можливість зручної навігації по вебзастосунку;
- дані користувача повинні бути захищені від перегляду та змінення будь-яким іншим користувачем;
- вебзастосунок повинен мати систему автентифікації користувача;

- повинен надавати змогу використання запропонованого методу діагностування деталей автомобіля.

#### **A.4 Вимоги до видів забезпечення і компонентів**

Для серверної частини вебзастосунку висувається відповідність наступним вимогам:

- 64-бітна операційна система;
- наявність GPU;
- наявність 512Мб вільного простору на накопичувачі пристрою;
- встановлений Python 3.8.12, бібліотек Django, Sklearn, Tensorflow, Numpy, crispy\_bootstrap5, та усіх супутніх бібліотек.

Для користувача:

- наявність браузеру.

#### **A.5 Обмеження на установку і виконання програми**

Програму, що розробляється в даній роботі, можна встановити і запустити на будь-якому комп'ютері під керуванням 64-х бітної операційної системи.

Запуск програми здійснюється шляхом старту серверу через файл `manage.py`.

Користувач отримує доступ до вебзастосунку за допомогою посилання, і браузеру, що присутній у його системі.

Додаткових апаратних і програмних засобів для роботи програми не потрібно.

### **A.6 Вимоги до транспортування та зберігання**

Програмний засіб не вимагає транспортування та зберігання, оскільки отримати повний комплект програмного забезпечення можна тільки через мережу інтернет

### **A.7 Вимоги до програмної документації**

Програмна документація разом з застосунком не постачається.

### **A.8 Умови експлуатації**

Програма повинна функціонувати під управлінням будь-якої 64-х бітної операційної системи.

**ДОДАТОК Б**  
**Текст програми**

## Б1. Текст файла settings.py

```

"""
Django settings for carr_management_site project.

Generated by 'django-admin startproject' using Django 3.2.9.

For more information on this file, see
https://docs.djangoproject.com/en/3.2/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.2/ref/settings/
"""

from pathlib import Path
import os

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-7s0*3t8-taw9zo56lw2v#2-0(zhs5-kg-@4iqcx3!7^egzjf0a'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'crispy_forms',
    "crispy_bootstrap5",
    'carrrs',
]
CRISPY_ALLOWED_TEMPLATE_PACKS = "bootstrap5"
CRISPY_TEMPLATE_PACK = "bootstrap5"

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'carr_management_site.urls'

```

```

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [str(BASE_DIR.joinpath('templates'))],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'carr_management_site.wsgi.application'

# Database
# https://docs.djangoproject.com/en/3.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'car_managment',
        'USER': 'Django',
        'PASSWORD': '123',
        'HOST': '127.0.0.1',
        'PORT': '5432',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

# Internationalization
# https://docs.djangoproject.com/en/3.2/topics/i18n/

```

```

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC_URL = '/static/'
STATICFILES_DIRS = [
    os.path.join(BASE_DIR, "static"),
]

# Default primary key field type
# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

```

## Б2. Текст файла `admin.py`

```

from django.contrib import admin

from .models import Carr, Colors, Manufacturer, Maintenance

# Register your models here.

admin.site.register(Carr)
admin.site.register(Maintenance)
admin.site.register(Colors)
admin.site.register(Manufacturer)

```

## Б3. Текст файла `forms.py`

```

from django import forms
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User
from django.forms import fields
from .models import Carr, Maintenance

class NewUserForm(UserCreationForm):
    email = forms.EmailField(required=True)

    class Meta:
        model = User
        fields = ("username", "email", "password1", "password2")

    def save(self, commit=True):
        user = super(NewUserForm, self).save(commit=False)
        user.email = self.cleaned_data['email']
        if commit:
            user.save()

```

```

        return user

class CarrForm(forms.ModelForm):
    class Meta:
        model = Carr
        exclude = ['user']

class MaintenanceForm(forms.ModelForm):
    title = forms.TextInput()
    description = forms.CharField(widget=forms.Textarea())

    class Meta:
        model = Maintenance
        exclude = ['carr']

LOAD_CHOICES = (
    ("0", "0%"),
    ("10", "10%"),
    ("20", "20%"),
    ("30", "30%"),
    ("40", "40%"),
    ("50", "50%"),
    ("60", "60%"),
    ("70", "70%"),
    ("80", "80%"),
    ("90", "90%")
)

class PredictionForm(forms.Form):
    load = forms.ChoiceField(label="Engine Load", choices=LOAD_CHOICES,
required=True)
    samples = forms.CharField(label="Vibration Data",
required=True,widget=forms.Textarea(attrs={'placeholder': '5.70718,-7.20074,-4.54852,-
5.52796\n-13.561,-4.68453,1.3053,-7.91893\n3.52773,-0.159441,3.04621,-
0.923214\n12.5466,-8.62812,11.9866,-5.01071\n6.94914,-0.179236,8.2617,-
3.23649\n5.80352,-1.55319,-2.47599,-6.307\n-1.08165,3.97097,-7.4898,-6.00217\n6.60639,-
5.2884,9.67397,4.21278\n-1.19753,5.92426,-3.85929,1.47045\n0.645542,7.96934,-
5.79981,7.4855',
'rows':10})),
    help_text="Enter 10 data samples. 1 per row
without spaces",
)

```

#### Б4. Текст файла modelNN.py

```

import numpy as np
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from sklearn.utils import shuffle
from sklearn.preprocessing import MinMaxScaler, normalize
from sklearn.metrics import mean_squared_error
import os
from django.template.tags.static import static
from pathlib import Path

def predict(x_test, load = 60):

```

```

np.random.seed(777)
labels = ['Working Ok', 'Need Maintains']

x_test = np.expand_dims(x_test,0)

BASE_DIR = Path(__file__).resolve().parent.parent

checkpoint_path = os.path.join(BASE_DIR.as_posix(),
f"static/savedNN/{load}/{load}.ckpt")

checkpoint_dir = os.path.dirname(checkpoint_path)

latest = tf.train.latest_checkpoint(checkpoint_dir)

model = Sequential([
    layers.LSTM(16),
    layers.Dense(1,activation='sigmoid')
])
model.compile(
    optimizer=keras.optimizers.Adam(learning_rate= 0.01),
    loss=keras.losses.BinaryCrossentropy(),
    metrics=['accuracy']
)
model.load_weights(latest)

prediction = model.predict(x_test)
prediction = np.squeeze(prediction,axis=1)
prediction = np.around(prediction, decimals=0).astype(int)

return labels[prediction[0]]

```

## Б5. Текст файлу models.py

```

from django.db import models
from django.contrib.auth.models import User

# Create your models here.
class Carr(models.Model):
    title = models.CharField('Title', max_length = 255)
    engine_title = models.CharField('Engine Title', max_length = 50, blank = True,
null = True)
    wills_type = models.CharField('Wills Type', max_length= 50, blank = True, null
= True)
    manufacturer = models.ForeignKey("Manufacturer", on_delete=models.CASCADE)
    manufacture_date = models.DateField('Manufacture Date')
    purchase_date = models.DateField('Purchase Date')
    update_date_time = models.DateTimeField('Update Date Time',auto_now=True)
    creation_date_time = models.DateTimeField('Creation Date
Time',auto_now_add=True)
    color = models.ForeignKey('Colors', on_delete= models.CASCADE)
    user = models.ForeignKey(User, on_delete=models.CASCADE)

    def __str__(self):
        return self.title

class Maintenance(models.Model):
    title = models.CharField('Title', max_length=50, blank=False, null=False)

```

```

        description = models.CharField('Description', max_length=1000, blank=False,
null=False)
        carr = models.ForeignKey('Carr', on_delete= models.CASCADE)
        update_date_time = models.DateTimeField('Update Date Time', auto_now=True)
        creation_date_time = models.DateTimeField('Creation Date
Time', auto_now_add=True)

    def __str__(self):
        return self.title

class Colors(models.Model):
    name = models.CharField('Name', max_length= 30)
    CSS_code = models.CharField('CSS Code', max_length=7)

    def __str__(self):
        return self.name

class Manufacturer(models.Model):
    name = models.CharField('Manufacturer Name', max_length=30)

    def __str__(self):
        return self.name

```

## Б6. Текст файла urls.py

```

from django import urls
from django.urls import path
from . import views

urlpatterns = [
    path('', views.index, name='home'),
    path('contacts', views.contacts, name='contacts'),
    path('features', views.features, name='features'),
    path('management', views.management, name='management'),
    path('register', views.signup, name='signup'),
    path('login', views.signin, name='signin'),
    path('logoutt', views.logoutt, name='signout'),
    path('create_car', views.create_Carr, name='create_car'),
    path('update_car/<str:pk>', views.update_Carr, name='update_car'),
    path('delete_car/<str:pk>', views.delete_Carr, name='delete_car'),
    path('maintenances_car/<str:pk>', views.maintences_Car,
name='maintenances_car'),
    path('create_maintenance/<str:pk>', views.create_Maintence,
name='create_maintenance'),
    path('update_maintence/<str:carr_id>/<str:pk>', views.update_Maintence,
name='update_maintence'),
    path('delete_maintence/<str:carr_id>/<str:pk>', views.delete_Maintence,
name='delete_maintence'),
    path('predictions', views.predictions, name='predictions')]

```

## Б7. Текст файла views.py

```

from django.shortcuts import render, redirect
import numpy as np

```

```

from .models import Carr, Maintenance
from .forms import MaintenanceForm, NewUserForm, CarrForm, PredictionForm
from .modelNN import predict
from django.contrib.auth import login, authenticate, logout
from django.contrib import messages
from django.contrib.auth.forms import AuthenticationForm

from django.views.generic.edit import UpdateView

# Create your views here.
def index(request):
    return render(request, 'carrrs/index.html')

def contacts(request):
    return render(request, 'carrrs/about.html')

def features(request):
    return render(request, 'carrrs/features.html')

def management(request):
    if request.user.is_anonymous:
        context = {
            'is_anonymous': 1
        }
    else:
        context = {
            'carrrs': Carr.objects.filter(user=request.user)
        }
    return render(request, 'carrrs/management.html', context)

def signup(request):
    storage = messages.get_messages(request)
    storage.used = True
    if request.method == "POST":
        form = NewUserForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)
            messages.success(request, "Registration successful." )
            return redirect('management')
        messages.error(request, form.error_messages)
    form = NewUserForm()
    return render(request, 'carrrs/register.html',
context={"register_form":form})
    # return render(request, 'carrrs/signup')

def signin(request):
    storage = messages.get_messages(request)
    storage.used = True
    if request.method == "POST":
        form = AuthenticationForm(request, data=request.POST)
        if form.is_valid():
            username = form.cleaned_data.get('username')
            password = form.cleaned_data.get('password')
            user = authenticate(username=username, password=password)
            if user is not None:
                login(request, user)
                messages.info(request, f"You are now logged in as {username}.")

```

```

        return redirect('management')
    else:
        messages.error(request,"Invalid username or password.")
    else:
        messages.error(request,"Invalid username or password.")
    form = AuthenticationForm()
    return render(request=request,          template_name="carrrs/login.html",
context={"login_form":form})

def logoutt(request):
    storage = messages.get_messages(request)
    storage.used = True
    logout(request)
    messages.info(request, 'You have successfully logged out.')
    return redirect('home')

def create_Carr(request):
    storage = messages.get_messages(request)
    storage.used = True
    if request.method == 'POST':
        form = CarrForm(request.POST)
        if form.is_valid():
            carr = form.save(commit=False)
            carr.user = request.user # The logged-in user
            carr.save()
            return redirect('management')
    else:
        form = CarrForm()
    return render(request, 'carrrs/create_carr.html', {'carr_form': form})

def update_Carr(request, pk):
    storage = messages.get_messages(request)
    storage.used = True
    try:
        carr = Carr.objects.get(id=pk)
        form = CarrForm(instance=carr)
        if request.method == 'POST':
            form = CarrForm(request.POST, instance=carr)
            if form.is_valid():
                form.save()
                return redirect('update_car',pk = pk)
    except carr.DoesNotExist:
        messages.error(request,"Carr has not found!")
        return redirect('management')
    return render(request, 'carrrs/update_carr.html', {'carr_form': form})

def delete_Carr(request, pk):
    storage = messages.get_messages(request)
    storage.used = True
    try:
        carr = Carr.objects.get(id=pk)
        carr.delete()
        return redirect('management')
    except carr.DoesNotExist:
        messages.error(request,"Carr has not found!")
        return redirect('management')

def maintenances_Car(request, pk):
    carr = Carr.objects.get(id=pk)

    context = {

```

```

        'carr_title': carr.title,
        'carr_id': carr.id,
        'maintences': Maintenance.objects.filter(carr=pk)
    }
    return render(request, 'carrs/maintenances.html', context)

def create_Maintenance(request, pk):
    storage = messages.get_messages(request)
    storage.used = True
    if request.method == 'POST':
        form = MaintenanceForm(request.POST)
        if form.is_valid():
            maintenance = form.save(commit=False)
            maintenance.carr = Carr.objects.get(id=pk) # Current car
            maintenance.save()
            return redirect('maintenances_car', pk)
    else:
        form = MaintenanceForm()
    return render(request, 'carrs/create_maintenance.html', {'maintenance_form':
form})

def update_Maintenance(request, carr_id, pk):
    storage = messages.get_messages(request)
    storage.used = True
    try:
        maintenance = Maintenance.objects.get(id=pk)
        form = MaintenanceForm(instance=maintenance)
        if request.method == 'POST':
            form = MaintenanceForm(request.POST, instance=maintenance)
            if form.is_valid():
                form.save()
                return redirect('maintenances_car', carr_id)
        else:
            return render(request, 'carrs/update_maintenance.html',
{'maintenance_form': form})
    except maintenance.DoesNotExist:
        messages.error(request, "Maintenance has not found!")
        return redirect('maintenances_car', carr_id)

def delete_Maintenance(request, carr_id, pk):
    storage = messages.get_messages(request)
    storage.used = True
    try:
        maintenance = Maintenance.objects.get(id=pk)
        maintenance.delete()
        return redirect('maintenances_car', carr_id)
    except maintenance.DoesNotExist:
        messages.error(request, "Maintenance has not found!")
        return redirect('maintenances_car', carr_id)

def predictions(request):
    storage = messages.get_messages(request)
    storage.used = True
    if request.method == 'POST':
        form = PredictionForm(request.POST)

        samples = []
        data = request.POST.get("samples").splitlines()
        for sample in data:
            samples.append(sample.split(', '))

```

```

samples = np.array(samples,dtype=np.float)

prediction = predict(samples,
    request.POST.get("load"))

context = {
    'prediction': prediction,
    'prediction_form': form
}
return render(request, 'carrrs/predictions.html', context)
else:
    form = PredictionForm()
    context = {'prediction_form': form }

return render(request, 'carrrs/predictions.html', context)

```

## Б8. Текст файлу model.py

```

import numpy as np
import os
import pandas as pd
from sklearn.utils import shuffle
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
import tensorflow.keras as keras
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from matplotlib import pyplot as plt

np.random.seed(777)

labels = ['Healthy', 'Broken Tooth']

def Aggregate_DataSets():
    h30hz = []
    bt30hz = []

    filesList = []

    for dirname, __, filenames in os.walk('Data/archive'):
        for filename in filenames:
            filesList += [os.path.join(dirname, filename)]

    filesList.sort()
    for file in filesList:
        if '.DS_Store' in file:
            filesList.remove(file)

    for file in filesList:
        if 'Healthy' in file:
            h30hz += [pd.read_csv(file)]
        else:
            bt30hz += [pd.read_csv(file)]

    for index, item in enumerate(h30hz):
        item = item.astype('float64')
        item['load'] = index * 10 * np.ones((len(item.index),1),'float64')

```

```

failureArray = np.zeros((len(item.index),1),np.int8)
item['failure'] = failureArray
h30hz[index] = item

for index, item in enumerate(bt30hz):
    item = item.astype('float64')
    item['load'] = index * 10 * np.ones((len(item.index),1),'float64')
    failureArray = np.ones((len(item.index),1),np.int8)
    item['failure'] = failureArray
    bt30hz[index] = item

healthy_df = pd.concat(h30hz, axis=0,ignore_index=True)
broken_df = pd.concat(bt30hz,axis=0,ignore_index=True)

broken_health_aggregated = pd.concat([broken_df,healthy_df], axis =0)
return broken_health_aggregated

aggregated_DataSet = Aggregate_DataSets()

scaler = MinMaxScaler(feature_range=(-1,1))
train_ratio = 0.80
chunk_size = 10

load = 30

tmp_DF = aggregated_DataSet.copy()
tmp_DF = tmp_DF[tmp_DF.load == load]

healthy_df = tmp_DF[tmp_DF.failure == 0]
healthy_df = healthy_df.drop(['load','failure'], axis=1)
broken_df = tmp_DF[tmp_DF.failure == 1]
broken_df = broken_df.drop(['load','failure'], axis=1)

healthy_chunks_split = (len(healthy_df)//chunk_size)*chunk_size
broken_chunks_split = (len(broken_df)//chunk_size)*chunk_size

healthy_chunks      =      np.split(healthy_df.to_numpy()[0:healthy_chunks_split],
healthy_chunks_split // chunk_size , axis=0)[0:-1]
broken_chunks      =      np.split(broken_df.to_numpy()[0:broken_chunks_split],
broken_chunks_split // chunk_size , axis=0)[0:-1]

healthy_X, healthy_Y = [i for i in healthy_chunks], [0 for _ in healthy_chunks]
broken_X, broken_Y = [i for i in broken_chunks], [1 for _ in broken_chunks]

X = healthy_X + broken_X
y = healthy_Y + broken_Y

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42, shuffle = True)

X_train = np.dstack(X_train)
X_train = np.rollaxis(X_train,-1)
X_test = np.dstack(X_test)
X_test = np.rollaxis(X_test,-1)

y_train = np.asarray(y_train)
y_test = np.asarray(y_test)

```



```

        <td class="spacer"></td>
        <td class="notedly">Yarmachek</td>
    </tr>
    <tr>
        <td class="">Email:</td>
        <td class="spacer"></td>
        <td class="notedly">max.yarmachek@gmail.com</td>
    </tr>
    <tr>
        <td class="">Phone:</td>
        <td class="spacer"></td>
        <td class="notedly">+380XXXXXXXX</td>
    </tr>
</table>
</div>
</div>
{% endblock content %}

```

## Б10. Текст файла base.html

```

{% load static %}
{% with request.resolver_match.url_name as url_name %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link
        rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
        crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{% static 'css/styles.css' %}">
    <link rel="stylesheet" href="//code.jquery.com/ui/1.13.0/themes/base/jquery-
ui.css">
    <script src="https://code.jquery.com/jquery-3.6.0.js"></script>
    <script src="https://code.jquery.com/ui/1.13.0/jquery-ui.js"></script>
    <title>{% block title %}{% endblock title %}</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top|fixed-
bottom|sticky-top">
        <div class="container-lg">
            <a class="navbar-brand" href="{% url 'home' %}"><h3>Carrrs</h3></a>
            <button class="navbar-toggler hidden-lg-up btn-light" type="button" data-
bs-toggle="collapse" data-bs-target="#collapsibleNavId" aria-controls="collapsibleNavId"
            aria-expanded="false" aria-label="Toggle navigation"></button>
            <div class="collapse navbar-collapse" id="collapsibleNavId">
                <ul class="navbar-nav me-auto mt-2 mt-lg-0">
                    <li class="nav-item">
                        <a class="nav-link {% if url_name == 'management' %}active{% endif
%}" href="{% url 'management' %}">
                            Management <span class="visually-hidden">(current)</span></a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link {% if url_name == 'predictions' %}active{% endif
%}" href="{% url 'predictions' %}">
                            Predictions <span class="visually-hidden">(current)</span></a>
                    </li>

```

```

        <li class="nav-item">
            <a class="nav-link {% if url_name == 'features' %}active{% endif %}"
href="{% url 'features' %}">
                Features <span class="visually-hidden">(current)</span></a>
            </li>
            <li class="nav-item">
                <a class="nav-link {% if url_name == 'contacts' %}active{% endif %}"
href="{% url 'contacts' %}">
                    Contacts <span class="visually-hidden">(current)</span></a>
                </li>
            </ul>
            <li class="nav-item">
                {% if user.is_authenticated %}
                    <div class="nav-link text-light" style="padding-left: 0px">
                        <span class="align-bottom">@{{user.username}}</span>
                        <a class="btn btn-sm btn-secondary fw-bold border-white bg-
white text-dark" href="{% url 'signout' %}">Sign Out</a>
                    </div>
                {% else %}
                    <a class="btn btn-sm btn-secondary fw-bold border-white bg-
white text-dark" href="{% url 'signin' %}">Sign In</a>
                {% endif %}
            </li>
        </div>
    </div>
</nav>

{% block content %}{% endblock content %}

<div class="container">
    <footer class="">
        <p class="text-center text-muted">© 2021 Maksym Yarmachek and Co.</p>
    </footer>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>

</body>
</html>

{% endwith %}

```

## Б11. Текст файла create\_carr.html

```

{% extends 'carrs/base.html' %}
{% block title %} Add New Car {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Create Car-->
<div class="container py-2">
    <h1 class="py-2">Add New Car</h1>
    {% if messages %}

```

```

        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ carr_form|crispy }}
        <button class="btn btn-primary" type="submit">Add Car</button>
    </form>
</div>

<script>
$(document).ready(function() {
    $('#dateinput').datepicker({ format: "yyyy/mm/dd" });
});
</script>
{% endblock content %}

```

## Б12. Текст файла create\_maintenance.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Add New Maintenance {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Create Maintenance-->
<div class="container py-2">
    <h1 class="py-2">Add New Maintenance</h1>
    {% if messages %}
        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ maintenance_form|crispy }}
        <button class="btn btn-primary" type="submit">Add Maintenance</button>
    </form>
</div>
{% endblock content %}

```

## Б13. Текст файла features.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Features {% endblock title %}

{% block content %}

<div class="container-lg">
    <div class="d3">
        <h2 class="lable"><span class="sp">Features</span></h2>
    <p>

```



```

        <!-- <link rel="apple-touch-icon" href="/docs/5.1/assets/img/favicons/apple-
touch-icon.png" sizes="180x180">
        <link rel="icon" href="/docs/5.1/assets/img/favicons/favicon-32x32.png"
sizes="32x32" type="image/png">
        <link rel="icon" href="/docs/5.1/assets/img/favicons/favicon-16x16.png"
sizes="16x16" type="image/png">
        <link rel="manifest" href="/docs/5.1/assets/img/favicons/manifest.json">
        <link rel="mask-icon" href="/docs/5.1/assets/img/favicons/safari-pinned-
tab.svg" color="#7952b3">
        <link rel="icon" href="/docs/5.1/assets/img/favicons/favicon.ico"> -->
        <meta name="theme-color" content="#7952b3">
        <style>
            .bd-placeholder-img {
                font-size: 1.125rem;
                text-anchor: middle;
                -webkit-user-select: none;
                -moz-user-select: none;
                user-select: none;
            }

            @media (min-width: 768px) {
                .bd-placeholder-img-lg {
                    font-size: 3.5rem;
                }
            }
        </style>

        <!-- Custom styles for this template -->
        <!-- <link href="cover.css" rel="stylesheet"> -->

    </head>

    <body class="d-flex h-100 text-center text-white bg-dark">

        <div class="cover-container d-flex w-100 h-100 p-3 mx-auto flex-column">
            <header class="mb-auto">
                <div>
                    <h3 class="float-md-start mb-0">Carrrs</h3>
                    <div class="container-fluid"></div>
                    <nav class="nav nav-masthead navbar-custom justify-content-center float-
md-end">
                        <a class="nav-link" href="{% url 'management' %}">Management</a>
                        <a class="nav-link" href="{% url 'predictions' %}">Predictions</a>
                        <a class="nav-link" href="{% url 'features' %}">Features</a>
                        <a class="nav-link" href="{% url 'contacts' %}">Contacts</a>
                    </nav>
                </div>
            </header>

            <main class="px-3">
                {% if user.is_authenticated %}
                <h1>WELCOME @{{user.username}}</h1>
                <p class="lead">
                    Know you can use all site features!
                </p>
                <p class="lead">
                    <a href="{% url 'signout' %}" class="btn btn-lg btn-secondary fw-bold
border-white bg-white text-dark">Sing Out</a>
                </p>
                {% else %}

```

```

        <h1>WELCOME</h1>
        <p class="lead">Start using this App right now!</p>
        <p class="lead">
            <a href="{% url 'signin' %}" class="btn btn-lg btn-secondary fw-bold
border-white bg-white text-dark">Sing In</a>
            <a href="{% url 'signup' %}" class="btn btn-lg btn-secondary fw-bold
border-white bg-white text-dark">Sing Up</a>
        </p>
    {% endif %}
</main>

<footer class="mt-auto text-white-50">
    <p class="text-center text-muted">© 2021 Maksym Yarmachek and Co.</p>
</footer>
</div>

<!-- JavaScript Bundle with Popper -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>

```

## Б15. Текст файла login.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Sign In {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Login-->
<div class="container py-5">
    <h1>Login</h1>
    {% if messages %}
        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ login_form|crispy }}
        <button class="btn btn-primary" type="submit">Sign In</button>
    </form>
    <p class="text-center">Don't have an account? <a href="{% url 'signup' %}">Create an account</a>.</p>
</div>

{% endblock content %}

```

## Б16. Текст файла maintenances.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Maintenances {% endblock title %}

{% block content %}

```



```


Engine



>{{carr.engine_title}}



Wills Type:



>{{carr.wills_type}}



Manufacturer:



>{{carr.manufacturer}}



Manufacturer



>{{carr.manufacture_date}}



Purchase



>{{carr.purchase_date}}



Color:



>{{carr.color}}


```



```

    </form>
</div>
{% endblock content %}

```

## Б19. Текст файла register.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Sign Up {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Register-->
<div class="container py-5">
    <h1>Register</h1>
    {% if messages %}
        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ register_form|crispy }}
        <button class="btn btn-primary" type="submit">Sign Up</button>
    </form>
    <p class="text-center">If you already have an account, <a href="{% url
'signin' %}">login</a> instead.</p>
</div>

{% endblock content %}

```

## Б20. Текст файла update\_carr.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Car Info {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Update Car-->
<div class="container py-2">
    <h1 class="py-2">Car Info</h1>
    {% if messages %}
        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ carr_form|crispy }}
        <button class="btn btn-primary" type="submit">Update</button>
    </form>
</div>

```

```

<script>
$(document).ready(function() {
    $('#dateinput').datepicker({ format: "yyyy/mm/dd" });
});
</script>
{% endblock content %}

```

## Б21. Текст файла update\_maintenance.html

```

{% extends 'carrrs/base.html' %}
{% block title %} Edit Maintenance {% endblock title %}

{% block content %}
{% load crispy_forms_tags %}

<!--Edit Maintenance-->
<div class="container py-2">
    <h1 class="py-2">Edit Maintenance</h1>
    {% if messages %}
        {% for message in messages%}
            <div class="alert alert-danger">
                {{message}}
            </div>
        {% endfor %}
    {% endif %}
    <form method="POST">
        {% csrf_token %}
        {{ maintenance_form|crispy }}
        <button class="btn btn-primary" type="submit">Edit Maintenance</button>
    </form>
</div>
{% endblock content %}

```

## Б22. Текст файла styles.css

```

.nav-link {
    color: #f5faff;
}
.nav-link :focus :hover{
    text-decoration: underline !important;
    color: firebrick !important;
}

.a .nav-link :focus :hover{
    text-decoration: firebrick !important;
    color: firebrick !important;
}

.btn-light{
    background-color: whitesmoke;
}

.four {
    background: #F4F7EE;
    padding: 50px 20px;
    text-align: left;
}

.four h1 {
    font-family: 'Merriweather', serif;
}

```

```

    position: relative;
    color: #C44737;
    font-size: 50px;
    font-weight: normal;
    padding: 8px 20px 7px 20px;
    border-top: 4px solid;
    border-left: 4px solid;
    display: inline-block;
    margin: 0;
    line-height: 1;
  }
  .four h1:before {
    content: "";
    position: absolute;
    width: 28px;
    height: 28px;
    top: -28px;
    left: -28px;
    border: 4px solid #C44737;
    box-sizing: border-box;
  }
  @media (max-width: 450px) {
    .four h1 {font-size: 36px;}
    .four h1:before {
      width: 20px;
      height: 20px;
      top: -20px;
      left: -20px;
    }
  }
}

.d3 {
  padding: 20px 0;
  border-bottom: 2px solid rgba(0,0,0,.05);
  text-align: left;
}

.lable {
  font-family: 'Merriweather', serif;
  font-size: 35px;
  letter-spacing: 1px;
  max-width: 320px;
  padding-top: 5px;
  width: 100%;
  position: relative;
  display: inline-block;
  color: #050505;
}

.d3 h2 {
  text-align: left;
}

.d3 h2:before {
  content: "";
  position: absolute;
  top: calc(50% - 6px);
  left: 0;
  right: 0;
  height: 16px;
  background: #212529;
  z-index: -1;
}

```

```

.d3 .sp {
    background: white;
    padding: 0 15px;
    margin-left: 20px;
}
.notedly{
    font-weight: bold;
}

.fl111111{
    display: block !important;
    float: right !important;
}

.spacer{
    width: 25px;
    color: #ffffff;
}

.div-table {
    display: table;
    width: auto;
    border-spacing: 5px; /* cellspacing:poor IE support for this */
}
.div-table-row {
    display: table-row;
    width: auto;
    clear: both;
}
.div-table-col {
    float: left; /* fix for buggy browsers */
    display: table-column;
    width: 200px;
}

```

**ДОДАТОК В**  
**Слайди презентації**

# Національний університет «Запорізька Політехніка»

## кафедра програмних засобів

Кваліфікаційна робота магістра

на тему: ДОСЛІДЖЕННЯ ТА ПРОГРАМНА  
РЕАЛІЗАЦІЯ МЕТОДІВ ТЕХНІЧНОГО  
ДІАГНОСТУВАННЯ ДЕТАЛЕЙ АВТОМОБІЛІВ

Виконав: студент групи КНТ-210м

Ярмачек М. В.

Керівник: к.т.н., доцент

Льовкін В. М.

2021 р.

Рисунок В.1 – Титульний слайд

## Мета роботи, об'єкт та предмет дослідження

- ♦ Мета роботи – створення програмного вебзастосунку технічного діагностування деталей автомобілів для спрощення та підвищення ефективності моніторингу стану автомобіля.
- ♦ Об'єкт дослідження – процес технічного діагностування деталей автомобілів.
- ♦ Предмет дослідження – нейронні мережі як метод технічного діагностування деталей автомобілів.

Рисунок В.2 – Слайд «Мета роботи, предмет та об'єкт дослідження»

## Задачі роботи

- ❖ Розглянути існуючі методи технічного діагностування деталей;
- ❖ Визначити вимоги до програмного забезпечення;
- ❖ Обрати мову і засоби розробки;
- ❖ Провести проектування архітектури програмного забезпечення;
- ❖ Розробити реалізацію програмного забезпечення;
- ❖ Провести експериментальне дослідження розробленого програмного забезпечення.

3

Рисунок В.3 – Слайд «Задачі роботи»

## Огляд існуючих методів класифікації

| Метод                        | Переваги                                                                                                                                                       | Недоліки                                                                                                                                                                                              |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Decision tree                | Просто перевірити людиною-експертом; Не потребує багато попередньої обробки даних.                                                                             | Потребує багато обчислювальних потужностей для класифікації.                                                                                                                                          |
| Support vector machines      | Відносна ефективність використання пам'яті                                                                                                                     | Фактична неможливість використання для великих об'ємів даних; дуже чутливий до зашумлених даних під час навчання.                                                                                     |
| Naive Bayes                  | Працює дуже швидко.                                                                                                                                            | Головне припущення методу про те, що вплив значення змінної на даний клас не залежить від значень інших змінних, не є правильним у випадку цієї роботи. Показує незадовільні результати класифікації. |
| Linear Discriminant Analysis | Простий, швидкий та портативний.                                                                                                                               |                                                                                                                                                                                                       |
| K-Nearest Neighbors          | Ефективний у зашумлених даних та коли навчальні дані великі.                                                                                                   | Потребує багато обчислень для оптимального числа K.                                                                                                                                                   |
| Self-organizing maps         | Простота та швидкість обчислення;                                                                                                                              | Відсутність параметрів адаптації для налаштування системи; сильна деградація на великих даних.                                                                                                        |
| Growing neural gas           | Добре працює з гіпершарами та гіперкладками; самонаштовбується без додаткових модифікацій; не така сильна деградація як у SOM.                                 | Складна та ресурсовитратна імплементація; погано розрізняє кластери більшої щільності в середині інших кластерів.                                                                                     |
| Random forest                | Зменшення можливості перенасичення даними (overfitting).                                                                                                       | Складність алгоритму; повільне прогнозування.                                                                                                                                                         |
| RNN                          | Не потрібно заздалегідь визначати функції алгоритму; забезпечує кращі результати прогнозування ніж усі інші алгоритми, якщо обсяг навчальних даних величезний. | Потребує дуже великої кількості даних, щоб працювати краще, ніж інші методи; немає довгострокової пам'яті.                                                                                            |
| CNN                          | Може виокремлювати важливі особливості об'єкту без нагляду вчителя.                                                                                            | Дуже повільне прогнозування за рахунок кількості необхідних обчислень; чутливий до специфічних шумів.                                                                                                 |
| LSTM                         | Добре справляється з проблемою перенасичення; надає змогу забувати помилкові ваги.                                                                             | Потребує великих обчислювальних потужностей.                                                                                                                                                          |

4

Рисунок В.4 – Слайд «Огляд існуючих методів класифікації»

## Метод визначення технічного стану коробки передач



5

Рисунок В.5 – Слайд «Метод визначення технічного стану коробки передач»

## Вибір мови програмування

| Критерій \ Мова                        | Python | R   | MATLAB | C#  | Java |
|----------------------------------------|--------|-----|--------|-----|------|
| Наявність бібліотек машинного навчання | +      | +   | +      | +/- | +/-  |
| Підтримка спільноти                    | +      | +/- | -      | +   | +    |
| Засоби аналізу даних                   | +      | +   | +/-    | +   | +/-  |
| Високорівнева мова                     | +      | +   | +/-    | -   | -    |
| Засоби створення веб-застосунку        | +      | -   | -      | +   | +    |
| Засоби зручної роботи з БД             | +      | +/- | -      | +   | +    |

6

Рисунок В.6 – Слайд «Вибір мови програмування»

## Діаграма прецедентів користувача

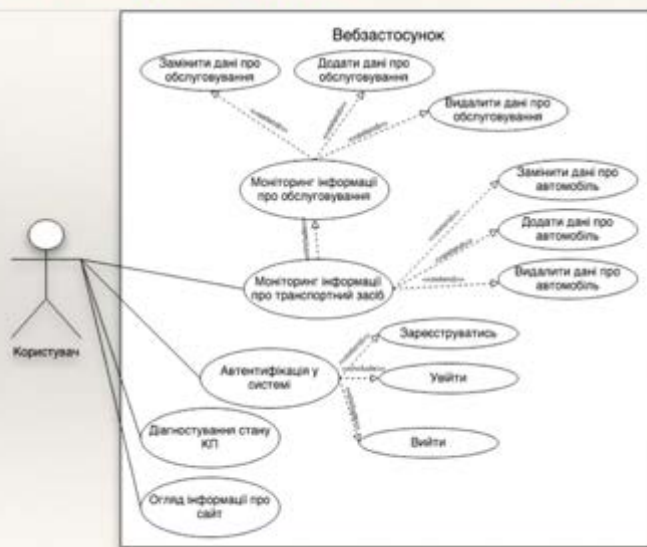


Рисунок В.7 – Слайд «Діаграма прецедентів користувача»

## Файлова структура застосунку

```

static
├── css
│   └── styles.css
├── savedNN
│   └── 0
│       ├── 0.ckpt.data-00000-of-00001
│       ├── 0.ckpt.index
│       └── checkpoint
│           ├── 10
│           ├── 20
│           ├── 30
│           ├── 40
│           ├── 50
│           ├── 60
│           ├── 70
│           ├── 80
│           └── 90

```

```

car_management_site
├── car_management_site
│   ├── __pycache__
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   ├── wsgi.py
│   └── cars
│       ├── __pycache__
│       ├── migrations
│       └── templates / cars
│           ├── about.html
│           ├── base.html
│           ├── create_car.html
│           ├── create_maintenance.html
│           ├── features.html
│           ├── index.html
│           ├── login.html
│           ├── maintenances.html
│           ├── management.html
│           ├── predictions.html
│           ├── register.html
│           ├── update_car.html
│           ├── update_maintenance.html
│           ├── __init__.py
│           ├── admin.py
│           ├── apps.py
│           ├── forms.py
│           ├── modelNN.py
│           ├── models.py
│           ├── tests.py
│           ├── urls.py
│           └── views.py
└── static

```

Рисунок В.8 – Слайд «Файлова структура застосунку»

## Схема Базы Даних



Рисунок В.9 – Слайд «Схема базы даних»

## Схема функціонування застосунку

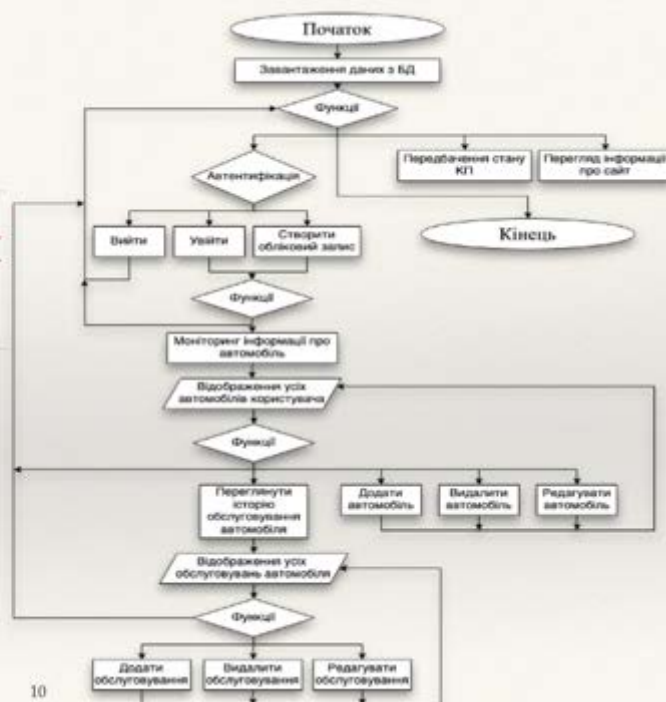


Рисунок В.10 – Слайд «Схема функціонування застосунку»

## Сторінка створення облікового запису

Carrs Management Predictions Features Contacts Sign In

### Register

Username\*

Required: 100 characters or fewer. Letters, digits and @/./+/-/\_ only.

Email\*

Password\*

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation\*

Enter the same password as before, for verification.

Sign Up

If you already have an account, [login](#) instead.

© 2021 Maksym Yarmachuk and Co.

11

Рисунок В.11 – Слайд «Сторінка створення облікового запису»

## Приклад розроблених сторінок

Carrs Management Predictions Features Contacts Sign In

### New Carr

Manufacturer: Ford

Manufacturer Date: Jan. 5, 2021

Purchase Date: Nov. 10, 2021

Color: Red

Put info Maintenance Delete

Add Car

© 2021 Maksym Yarmachuk and Co.

### Сторінка Management

Carrs Management Predictions Features Contacts Sign In

Car: New Carr Add Maintenance

new maintenance

Description: Description Description Description Description

Edit Delete

© 2021 Maksym Yarmachuk and Co.

### Сторінка Maintenance

Carrs Management Predictions Features Contacts Sign In

### Add New Car

Title\*

Engine Title

Mile Type

Manufacturer\*

Manufacture Date\*

Purchase Date\*

Color\*

Add Car

© 2021 Maksym Yarmachuk and Co.

Сторінка додавання нового автомобілю

12

Рисунок В.12 – Слайд «Приклади розроблених сторінок»

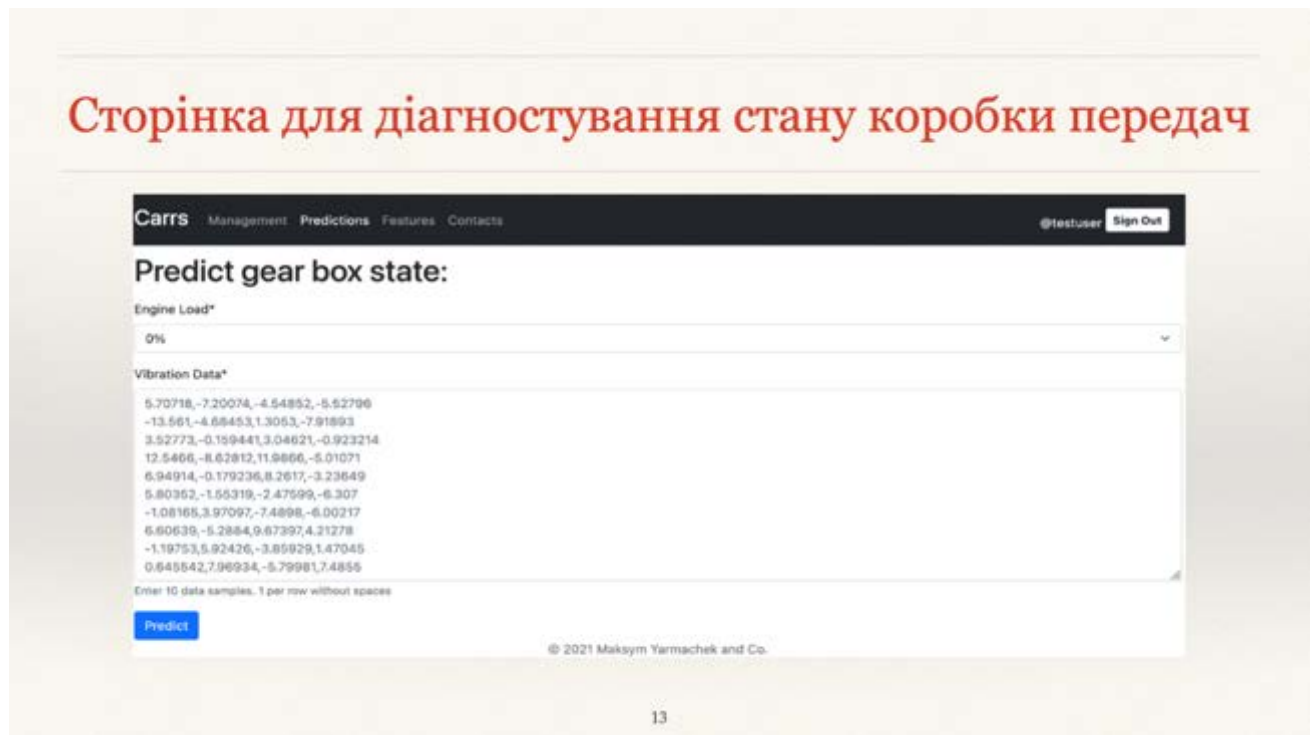


Рисунок В.13 – Слайд «Сторінка для діагностування стану коробки передач»

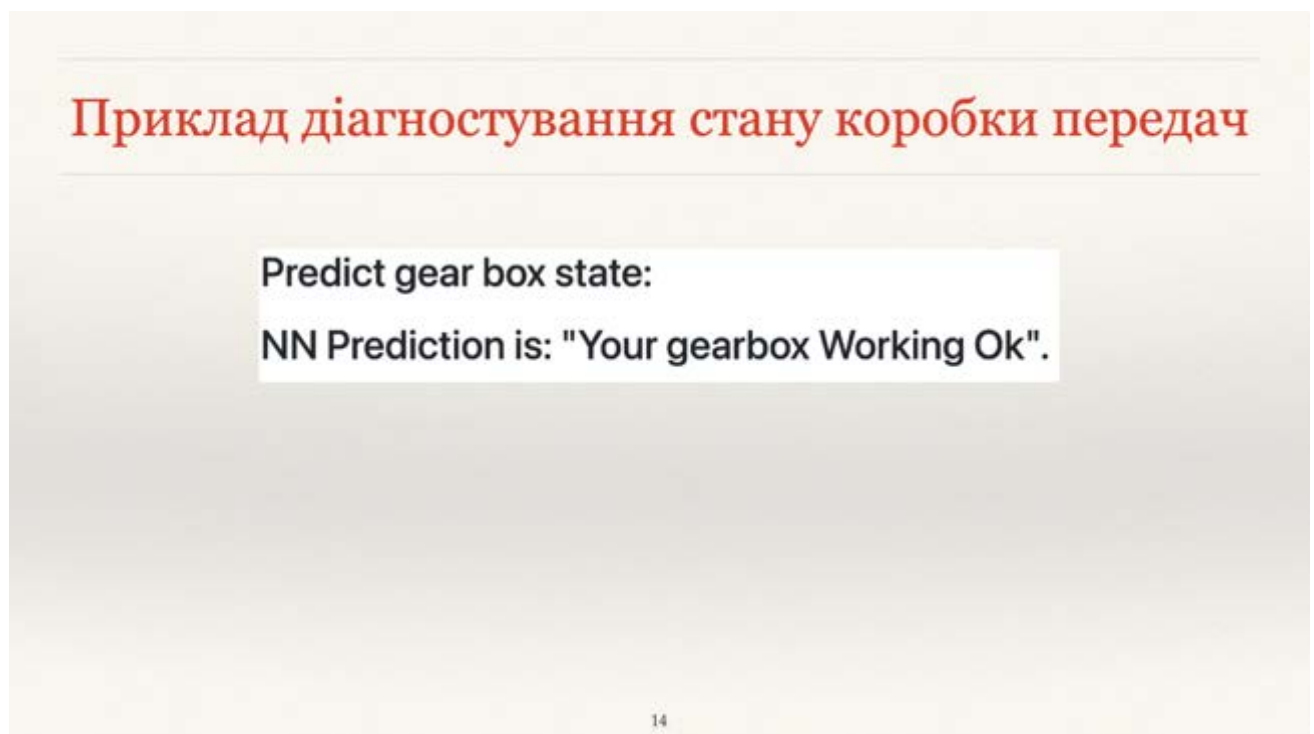


Рисунок В.14 – Слайд «Приклад діагностування стану коробки передач»

## Результати експериментального дослідження

| Метод                | Загальна точність |
|----------------------|-------------------|
| Запропонований метод | 0.96              |
| ANN                  | 0.68              |
| KNN                  | 0.63              |
| LSTM                 | 0.59              |
| RF                   | 0.58              |
| NB                   | 0.56              |
| RNN                  | 0.55              |
| SVM                  | 0.50              |

15

Рисунок В.15 – Слайд «Результати експериментального дослідження»

## Висновки

- ♦ Проаналізовано існуючі методи вирішення задачі технічного аналізу;
- ♦ Запропоновано метод діагностування стану КП;
- ♦ Обрано засоби розробки вебзастосунку;
- ♦ Зроблено проектування системи;
- ♦ Розроблено вебзастосунок;
- ♦ Проведено експериментальне дослідження запропонованого методу.

Розроблено метод технічного діагностування деталей автомобілів, що характеризується використанням моделі класифікації на основі нейронної мережі архітектури LSTM з визначенням спеціальної попередньої обробки даних, що дозволяє зменшити витрати на виявлення несправностей у коробці передач, включаючи часові та фінансові витрати.

16

Рисунок В.16 – Висновки

## Перелік публікацій

1. Льовкін, В.М. Розробка мобільної 3D-гри за допомогою середовища Unity [Електронний ресурс] / В.М. Льовкін, М.В. Ярмачек // Тиждень науки-2020. Факультет комп'ютерних наук і технологій. Тези доповідей науково-практичної конференції, Запоріжжя, 13–17 квітня 2020 р. [Електронний ресурс] / Редкол. : В. В. Наумик (відпов. ред.) Електрон. дані. – Запоріжжя : НУ«Запорізька політехніка», 2020. – С. 203-204
2. Ярмачек, М.В. Розробка мобільної 3d-гри за допомогою середовища Unity / М.В. Ярмачек, В.М. Льовкін // Матеріали XII міжнародної науково-практичної конференції «Free and Open Source Software», Харків, 17-19 листопада 2020 р. – Харків : ХНУБА, 2020. – С. 39.

Рисунок В.17 – Перелік публікацій