Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії
(повне найменування інституту, назва факультету (відділення))

Кафедра програмного забезпечення
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту»

Виконав: студент 2-го курсу, групи 1ПІ-22м
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Перебейнос Р.Л.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. ПЗ:
Катєльніков Д. І.
(прізвище та ініціали)
«15» грудня 2023 р.

Опонент к.т.н., доцент каф. ОТ:
Колесник І. С.
(прізвище та ініціали)
«15» грудня 2023 р.

Допущено до захисту
Завідувач кафедри ПЗ
д.т.н, проф. Романюк О.Н.
(прізвище та ініціали)
«15» грудня 2023 р.

ВНТУ – 2023

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти II-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Освітньо-професійна програма – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О. Н.
«19» вересня 2023 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙУ РОБОТУ СТУДЕНТУ

Перебейносу Роману Леонідовичу

1. Тема роботи – Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту.

Керівник роботи: Катєльніков Денис Іванович, к.т.н., доц. кафедри ПЗ, затверджені наказом вищого навчального закладу від
«18» вересня 2023 року № 247.

2. Строк подання студентом роботи
5 грудня 2023 року

3. Вихідні дані до роботи: середовища розробки Visual Studio 2019 та Visual Studio Code, мови розробки C#, операційна система – Windows 10, базові алгоритми для прогнозування результатів на основі статистики.

4. Зміст розрахунково-пояснювальної записки: вступ; аналіз та постановка задачі; аналіз моделей мови; розробка архітектури та алгоритмів системи; розробка методу підготовки вхідних даних; розробка методу прогнозування результатів на основі статистики; розробка програмного модуля для прогнозування результатів на основі статистики з використання штучного

інтелекту, розробка веб-сервісу; тестування додатку; економічна части
висновки; перелік посилань; додатки.

5. Перелік графічного матеріалу: блок-схеми алгоритмів роботи додат
структура нейронної мережі; тестування додатку.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Катєльніков Д. І., к.т.н., доцент кафедри ПЗ	19.09.23	05.12.23
5	Причепа І.В., к.т.н., доцент кафедри ЕПВМ	22.11.23	01.12.23

7. Дата видачі завдання 19 вересня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів магістерської кваліфікаційної Роботи	Строк виконання етапів роботи	Примітк
1	Аналіз моделей нейронних мереж та вибір найбільш доцільної для поставленої задачі	15.09.2023 – 30.09.2023	Вик.
2	Розробка алгоритмів системи	01.10.2023 – 10.10.2023	Вик.
3	Розробка методу отримання вхідних даних	11.10.2023 – 25.10.2023	Вик.
4	Розробка методу прогнозування результатів на основі статистики	26.10.2021 – 15.11.2021	Вик.
5	Економічна частина	22.11.2023 – 01.12.2023	Вик.

Студент _____ Перебейнос Р. Л
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ Катєльніков Д. І
(підпис) (прізвище та ініціали)

Анотація

УДК 004.912.032.26

Перебейнос Р.Л. Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту. Магістерська кваліфікаційна робота зі спеціальності 121 – інженерія програмного забезпечення, освітня програма – інженерія програмного забезпечення. Вінниця: ВНТУ, 2023. 149 с.

На укр. мові. Бібліогр.: 36 назв; рис.: 39; табл. 15.

У магістерській кваліфікаційній роботі проведено детальний аналіз методів прогнозування футбольних матчів на основі статистики.

Запропоновано використовувати логістичну регресію як статистичний метод для прогнозування результату.

Розроблено метод прогнозування футбольних матчів, що використовує удосконалену Dixon-Coles модель та метод обробки вхідних даних, який структурує статистичні данні матчу на окремі сутності, чим вдається покращити надійність роботи системи. Розроблено модуль для прогнозування результатів футбольних матчів з використанням мови програмування C#, бібліотеки ML.NET та середовища розробки Microsoft Visual Studio 2019. Розроблено веб-сервіс, що використовує модуль прогнозування результатів футбольних матчів. При розробці використано мову програмування C#, фреймворк ASP.NET Core . та бібліотеку ML.NET.

Отримані в магістерській кваліфікаційній роботі результати можуть використовувати букмекери компанії для формування коефіцієнтів на футбольну подію.

Ключові слова: нейронні мережі, Dixon-Coles модель , веб-технології, прогнозування футбольних матчів.

Abstract

Perebeinos R.L. Development of methods and software tools for predicting the results of football matches based on artificial intelligence models. Vinnitsa: VNTU, 2023. – 149 p.

In Ukrainian language. Bibliographer: 36 titles; fig.: 39; tabl. 15.

In the master's qualification work, a detailed analysis of the methods of predicting football matches based on statistics was carried out.

It is proposed to use logistic regression as a statistical method for predicting the outcome.

A football match prediction method has been developed that uses the improved Dixon-Coles model and an input data processing method that structures the statistical data of the match into separate entities, which makes it possible to improve the reliability of the system. A module for predicting the results of football matches has been developed using the C# programming language, the ML.NET library and the Microsoft Visual Studio 2019 development environment. A web service using the module for predicting the results of football matches has been developed. The C# programming language, the ASP.NET Core framework was used in the development. and the ML.NET library.

The results obtained in the master's qualification work can be used by the company's bookmakers to form coefficients for a football event.

Keywords: neural networks, Dixon-Coles model, web technologies, prediction of football matches.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. У сучасному світі спорт, зокрема футбол, займає важливе місце в житті людей. Футбол стає не лише розвагою, але й предметом наукових досліджень та аналізу. Прогнозування результатів футбольних матчів стає актуальним завданням, яке викликає інтерес як у фахівців із різних галузей, так і у широкої аудиторії. Враховуючи набуті попередні знання та постійний розвиток технологій, нові методи аналізу та прогнозування результатів футбольних матчів мають великий потенціал для подальшого вивчення та впровадження [1].

Використання штучного інтелекту дозволяє зібрати велику кількість даних про минулі матчі, аналізувати сильні та слабкі сторони команд, а також враховувати індивідуальні характеристики гравців. У результаті, створення таких моделей сприяє підвищенню точності прогнозів та забезпеченню кращих результатів для команд, тренерів та вболівальників[2].

Враховуючи, з одного боку, популярність такого явища як професійний футбол і ресурси, які в ньому обертаються, і, з другого боку, приймаючи до уваги відсутність надійних методів прогнозування, які отримують достовірні значущі результати, можна зробити висновок, що розробка подібного методу прогнозування з використанням новітніх наробок в галузі штучного інтелекту залишається актуальною задачею.

Зв’язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення

Мета роботи є підвищення точності прогнозування результатів футбольних матчів за рахунок використання удосконаленої моделі штучного інтелекту Dixon-Coles.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

1. Провести аналіз існуючих методів прогнозування результатів футбольних матчів та розглянути їх переваги та недоліки.

2. На основі аналізу переваг відомих підходів запропонувати власний підхід до прийняття рішень і обґрунтовано його ефективність.

3. Розробити архітектуру системи прогнозування.

4. Розробити та дизайн веб-ресурсу.

5. Реалізувати інформаційну систему прогнозування футбольних матчів та забезпечити доступ до неї через мережу Інтернет.

6. Провести тестування та дослідження розробленої системи.

Об'єкт дослідження це процес прогнозування результатів футбольних матчів на основі накопиченої статистики.

Предметом дослідження є методи та засоби прогнозування результатів матчів на основі футбольної статистики та моделей штучного інтелекту.

Методи дослідження. У процесі досліджень використовувались методи дослідження: теорія баз даних при організації зберігання інформації; теорія нейронних мереж та машинне навчання - для аналізу великої кількості даних та виявлення закономірностей; теорія розподілених систем для побудови веб-сервісу, який надає доступ до модуля прогнозування результатів футбольних матчів; комп’ютерне моделювання для аналізу та перевірки отриманих теоретичних положень.

Наукова новизна отриманих результатів.

1. Подальшого розвитку отримав метод прогнозування результатів футбольних матчів на основі моделі Dixon-Coles [4], що базується на дискретному розподілі Пуасона [5], у якому, на відміну від існуючого, використовується спеціальний коефіцієнт загального потенціалу команди, який дозволяє підвищити точність прогнозу результатів матчів.

2. Уперше запропоновано метод визначення спеціального коефіцієнту потенціалу команди, особливість якого полягає у врахуванні індивідуальної статистики кожного гравця на відповідній позиції. Застосування цього спеціального коефіцієнту потенціалу команди в моделі Dixon-Coles дає можливість підвищити точність прогнозу.

Практична цінність отриманих результатів полягає в тому, що на основі отриманих в роботі теоретичних положень запропоновано алгоритми та розроблено програмні засоби веб-сервісу для прогнозування результатів футбольних матчів.

Особистий внесок здобувача. Усі наукові результати, викладені у магістерській кваліфікаційній роботі, отримані автором особисто. У науковій роботі, опублікованій у співавторстві, автору належать такі результати: використання моделей штучного інтелекту для прогнозування результатів футбольних матчів [6]; розробка програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту [7].

Апробація матеріалів магістерської кваліфікаційної роботи. Результати магістерської кваліфікаційної роботи доповідалися та обговорювалися на:

- міжнародній науково-практичній Інтернет конференції «Електронні інформаційні ресурси: створення, використання, доступ.» (Вінниця/Суми 2023);
- III Всеукраїнській науково-технічній конференції молодих вчених, аспірантів та студентів «Комп'ютерні Ігри і мультимедіа як інноваційний підхід» (Одеса 2023).

Публікації. Основні результати досліджень опубліковано в 2 наукових працях – у матеріалах конференцій.

1 АНАЛІЗ СТАНУ ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ФУТБОЛЬНИХ МАТЧІВ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Аналіз стану прогнозування результатів футбольних матчів

Футбол є одним з найпопулярніших видів спорту у світі, який об'єднує мільйони людей різних культур, національностей та вікових груп. Цей феномен нашого життя не тільки пропонує захоплюючі спортивні події та розваги, але й сприяє формуванню дружби, командного духу та змагальності серед гравців та вболівальників. Футбол втілює у собі силу спорту, яка перетворює його на глобальне явище, що відображає людську пристрасть до змагань, майстерності та солідарності.

Штучний інтелект (ШІ) відіграє важливу роль в сучасному світі, зокрема в прогнозуванні результатів спортивних змагань, як-от футбольних матчів. Метою аналізу є вивчення та оцінка сучасних методів прогнозування результатів футбольних матчів з використанням ШІ [7].

Використання штучного інтелекту в прогнозуванні результатів спортивних змагань стало дедалі популярнішим у останні роки. Це пояснюється тим, що ШІ дозволяє аналізувати великі обсяги даних та робити прогнози на основі виявлених закономірностей. У порівнянні з традиційними методами прогнозування, такими як експертні оцінки або статистичні методи, ШІ може пропонувати більш точні та об'єктивні результати.

Один з прикладів використання ШІ в прогнозуванні результатів спортивних змагань - це система подання ставок на основі машинного навчання, яка використовує історичні дані про результати матчів, статистику гравців та інші фактори для побудови прогнозів. Така система може враховувати різні фактори, що впливають на результати матчів, та адаптуватися до нових умов.

Ще одним прикладом може бути використання нейронних мереж для прогнозування результатів футбольних матчів. Нейронні мережі - це моделі, які

навчаються розпізнавати та аналізувати шаблони даних. Вони можуть виявити складні залежності між різними факторами та адаптуватися до нових умов. Такі моделі можуть виявитися корисними для аналізу впливу різних стратегій або тактик на результати матчів [8].

При оцінці стану прогнозування результатів футбольних матчів на основі штучного інтелекту важливо врахувати їхню ефективність та точність. Деякі методи можуть мати високу точність в одних умовах, але працювати менш ефективно в інших. Тому важливо проводити аналіз та порівняння різних методів, щоб визначити найбільш ефективні та точні підходи.

Також варто зазначити, що результати прогнозів можуть залежати від якості вхідних даних. У зв'язку з цим, важливо забезпечити надійність та актуальність даних, які використовуються в процесі прогнозування.

Футбол та прогнозування результатів тісно пов'язані, оскільки вони допомагають вболівальникам, експертам та ставках на спорт розуміти тенденції та можливі результати майбутніх матчів. Прогнозування результатів футбольних матчів засноване на аналізі статистичних даних, таких як попередні результати команд, форма гравців, взаємодія на полі та інші фактори. Використання різних методів, зокрема штучного інтелекту, дозволяє забезпечити більш точні та об'єктивні прогнози, що можуть полегшити прийняття рішень та підвищити інтерес до гри [9].

У цілому цей напрямок має значні можливості для розвитку та покращення. Застосування штучного інтелекту у цій сфері може відкрити нові горизонти для аналізу даних, адаптації до змінних умов та оцінки впливу різних стратегій на результати гри. Це, в свою чергу, може сприяти більш точним прогнозам, які будуть корисними для команд, експертів, ставок на спорт та шанувальників футболу загалом. Однак, для досягнення успіху в цьому напрямку, необхідно зосередитись на неперервному вдосконаленні технік, забезпеченні якості даних

1.2 Порівняльний аналіз аналогів

Як правило, системи прогнозування результатів футбольних матчів є частинами програмного комплексу, які прогнозують результати багатьох спортивних змагань, не тільки футболу. Серед таких систем найбільш відомими є:

- ELO Ratings
- SciSports
- Kickoff.ai
- Wyscout
- xG Philosophy

ELO Ratings – це система рейтингів, яка враховує силу команди на основі її попередніх результатів [10]. Бали ELO набуваються або втрачаються після кожного матчу, що дозволяє оновлювати рейтинг команди на постійній основі (рисунок 1.1).

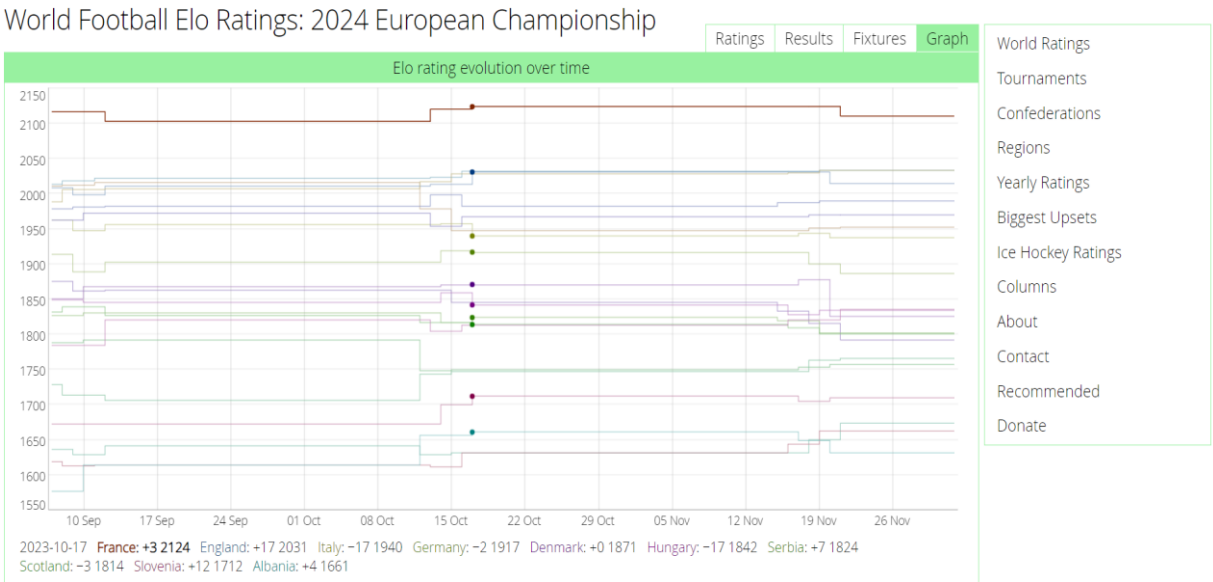


Рисунок 1.1 – Інтерфейс веб-додатку SciSports

До переваг сервісу відноситься простота розуміння, швидкодія, підтримка багатьох мов. До недоліків можна віднести не врахування багатьох статистичних факторів.

SciSports - це компанія, яка використовує машинне навчання та аналіз даних для прогнозування результатів спортивних змагань, включаючи футбол [11]. Ними була розроблена модель, заснована на великих даних, яка аналізує статистику гравців, команд і попередні матчі (рисунок 1.2).

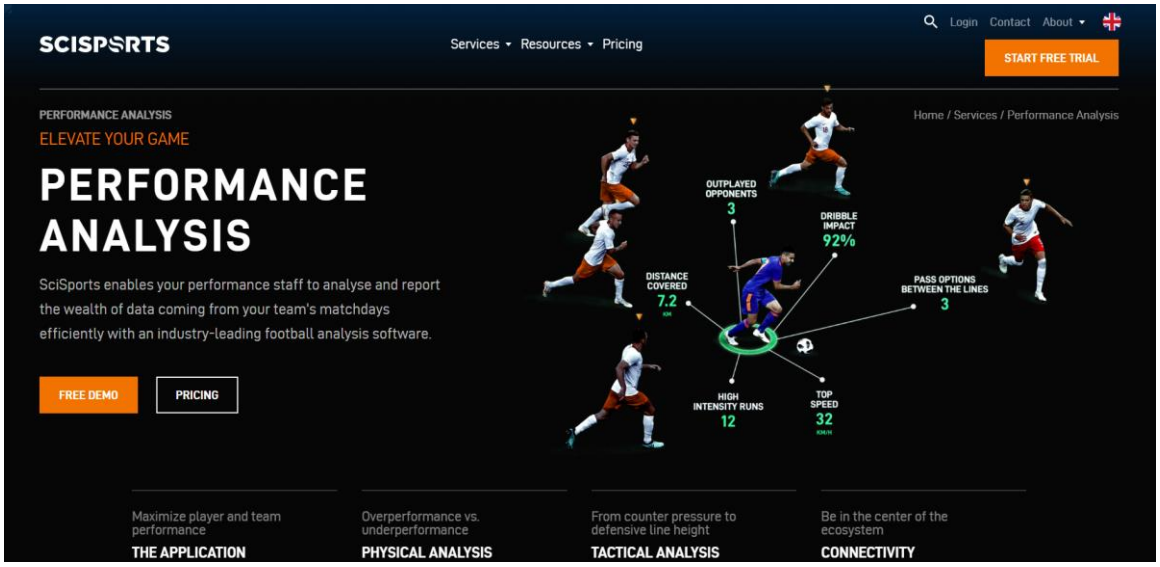


Рисунок 1.2 – Інтерфейс веб-додатку SciSports

Kickoff.ai - це додаток, який використовує моделі машинного навчання для прогнозування результатів футбольних матчів [12]. Вони використовують різні алгоритми та методи, такі як глибоке навчання та опорні вектори, для аналізу великої кількості даних та створення прогнозів (рисунок 1.3).

Wyscout - це платформа для аналізу футболу, яка використовує моделі машинного навчання та статистичний аналіз для прогнозування результатів матчів [13]. Вони пропонують детальний аналіз команд, гравців, тактики та інших факторів, що впливають на результати матчів (рисунок 1.4).

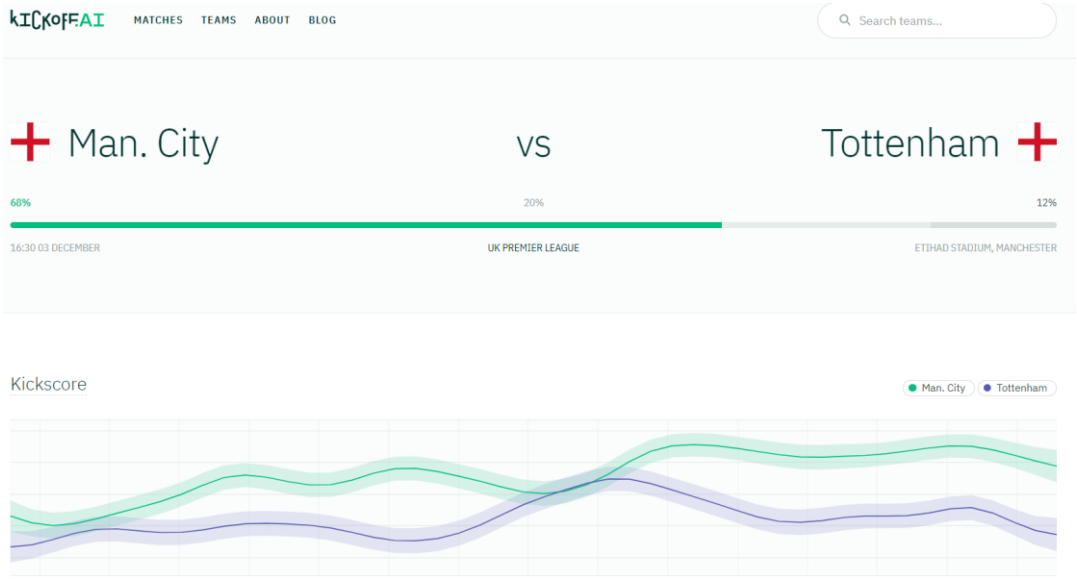


Рисунок 1.3 – Інтерфейс веб-додатку Kickoff.ai

Перевагами Kickoff.ai є використання різних алгоритмів машинного навчання та безкоштовна версія для користувачів, недоліками – обмежена чи неточна інформація та можливість не врахування всіх важливих факторів матчів.

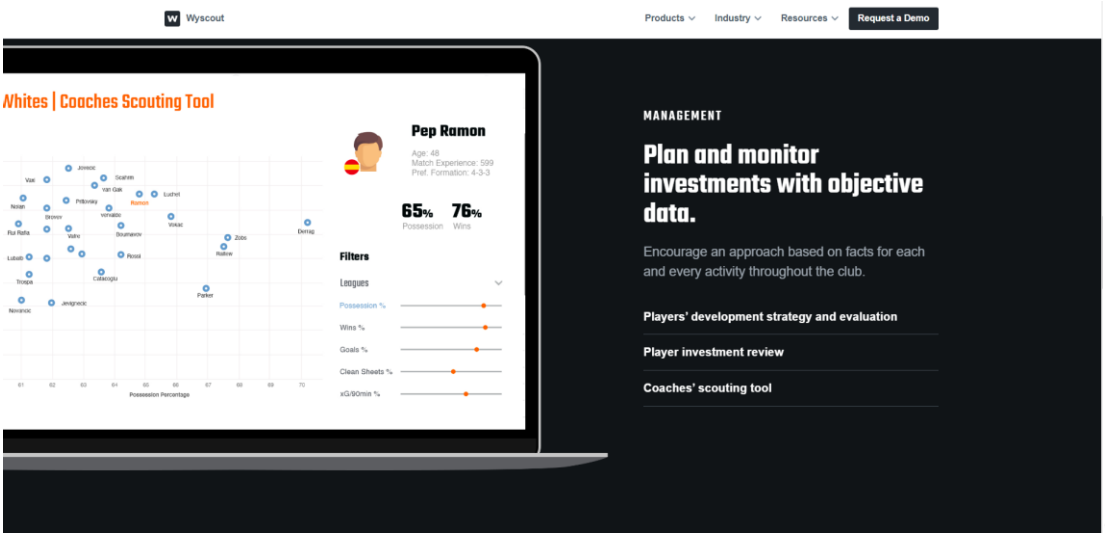


Рисунок 1.4 – Інтерфейс веб-додатку Wyscout

Перевагами Wyscout є детальний аналіз команд, гравців, тактики та інших факторів та широкий спектр статистичних даних, недоліками – обмежений доступ до інформації без платної підписки.

xG Philosophy - це платформа, яка розробляє моделі на основі очікуваних голів (xG) за допомогою машинного навчання [14]. Вони аналізують дані про удари, створення гольових моментів та інші фактори, щоб створити прогнози результатів матчів.

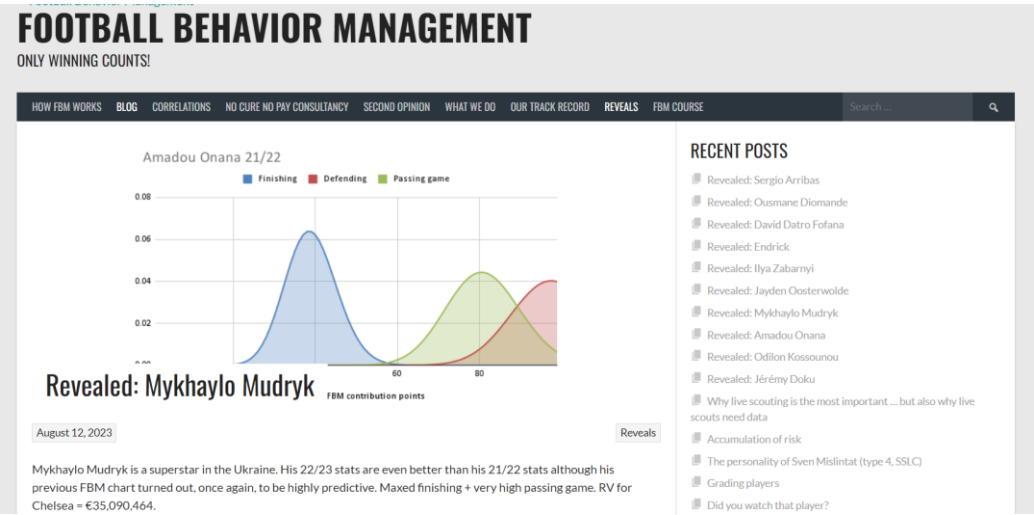


Рисунок 1.5 – Інтерфейс веб-додатку xG Philosophy

Перевагами xG Philosophy є фокус на аналізі очікуваних голів (xG), що допомагає враховувати створені гольові можливості, недоліками – обмежена модель оскільки вона зосереджується в основному на статистиці ударів.

Використання цих додатків може допомогти футбольним професіоналам, аналітикам та ентузіастам більше зрозуміти сильні та слабкі сторони команд та гравців, спланувати тактику та розробити стратегії для перемоги. Азартні люди також їх використовують, додатки є корисними для ставок, оскільки вони надають прогнози на основі різних алгоритмів та статистичних даних, що можуть допомогти користувачам приймати обґрунтовані рішення.

В цілому, ці додатки відіграють важливу роль у сучасному футболі, допомагаючи різним учасникам гри отримати доступ до цінних даних та

аналітичних інструментів для підвищення ефективності та успішності на футбольному полі.

Таблиця 1.1 – Порівняльні характеристики програмних продуктів

Веб-сайт	Базовий принцип	Підписка	Переваги	Недоліки	Точність
SciSports	ШІ	Так	Використання великих даних, аналіз статистики гравців та команд	Обмежений доступ до інформації без платної підписки	Змінна
Stratagem	ШІ	Так	Різні моделі машинного навчання, аналіз команд та гравців	Обмежений доступ та можливість використання	Змінна
Kickoff.ai	ШІ	НІ	Використання різних алгоритмів, таких як глибоке навчання та опорні вектори	Інформація може бути обмеженою чи неточною	Змінна
Wyscout	ШІ	Так	Детальний аналіз команд, гравців, тактики та інших факторів	Обмежений доступ до інформації без платної підписки	Змінна
xG Philosophy	ШІ	Ні	Аналіз на основі очікуваних голів (xG), статистика ударів	Може не враховувати всі важливі фактори впливу на матч	Змінна

Жодний з додатків не може гарантувати 100% точність прогнозів, тому важливо використовувати ці ресурси з обережністю та враховувати власні дослідження та знання.

Таблиця порівняння характеристик демонструє, що створення програмного рішення є виправданим. У результаті отримаємо продукт, який

вирішує недоліки наявних альтернатив та забезпечує більш ефективні результати порівняно з безкоштовними аналогами.

1.3 Аналіз методів розв’язання поставленої задачі

Варіантів для розв’язання задачі з прогнозування футбольних матчів є декілька:

- регресійний аналіз;
- методи машинного навчання;
- експертні системи;
- аналіз часових рядів;
- системи на основі знань.
- глибоке навчання.

Одним із перших та найбільш очевидних варіантів є регресійний аналіз. Даний метод полягає у встановленні залежності між змінними, такими як кількість забитих голів, очок команди та інших статистичних даних. Регресійний аналіз допомагає визначити важливість кожної змінної та її вплив на результат матчу. Недоліком є його лінійна природа, яка не враховує нелінійні залежності між змінними та обмежена здатність враховувати випадкові події та взаємозв'язки між різними факторами [15].

Використовуючи алгоритми машинного навчання, такі як дерева рішень, опорні векторні машини та нейронні мережі, можна створити моделі, які передбачають результати футбольних матчів на основі навчальних даних. Основним недоліком методів машинного навчання є потенційна надлишкове навчання моделей, коли вони занадто точно налаштовані на навчальних даних і погано працюють на реальних даних [16].

Експертні системи використовують знання та досвід експертів у галузі футболу для прогнозування результатів матчів. Експертні системи можуть враховувати такі фактори, як форма команди, травми гравців, статистика гравців

та інше. Недоліком є залежність від експертного досвіду та знань, а також можливість упередженості та обмеженості у знаннях експертів [17].

Аналіз часових рядів передбачає аналіз попередніх результатів команд з метою прогнозування майбутніх результатів. Він може включати авторегресійні моделі та інші техніки аналізу часових рядів. Одним з недоліків аналізу часових рядів є його залежність від стабільності і структури даних у часі, що може призвести до неточних прогнозів у разі змін у закономірностях або виняткових подіях [18].

Системи на основі знань використовують експертні знання та статистичні дані для створення бази знань, яка може бути використана для прогнозування результатів футбольних матчів. Недоліки включають складність у підтримці та оновленні бази знань, а також обмеженість враховувати дані з різних джерел інформації [19].

Глибоке навчання передбачає використання нейронних мереж з багатьма шарами для виявлення складних шаблонів у даних. Глибоке навчання може виявити взаємозв'язки між різними факторами, які впливають на результати матчів, та передбачити результати з високою точністю. Головним недоліком є висока складність моделей та велика кількість ресурсів, необхідних для їх навчання та оптимізації [20].

Штучний інтелект відкриває нові можливості для прогнозування результатів футбольних матчів. Різноманітні методи розв'язання цієї задачі, такі як регресійний аналіз, машинне навчання, експертні системи, аналіз часових рядів, системи на основі знань та глибоке навчання, дозволяють створювати точні та ефективні прогнози результатів футбольних матчів.

Також розглянемо наявні статистичні алгоритми.

Poisson Distribution – метод який, ґрунтується на статистичному розподілі Пуассона, який використовується для прогнозування числа голів, які заб'ють обидві команди в матчі [5]. Вихідні дані для розподілу Пуассона зазвичай

включають середню кількість голів, забитих та пропущених командами в попередніх матчах.

Dixon-Coles модель – статистична модель, яка вдосконалює модель розподілу Пуассона, враховуючи залежності між голами, забитими та пропущеними командами, а також зменшуючи вплив голів, забитих на останніх хвилинах гри [4].

Google's PageRank Algorithm – алгоритм, який вперше був використаний для ранжування веб-сторінок, може бути адаптований для прогнозування результатів футбольних матчів. Команди представляються як вузли у мережі, а перемоги - як зв'язки між ними. PageRank вимірює "авторитетність" кожної команди на основі якості її перемог [21].

Для вирішення питання прогнозування результатів футбольних матчів було обрано “Dixon-Coles Model”, який відноситься до методів статистичного моделювання та базуються на аналізі часових рядів. Даний метод забезпечить високу ефективність та надійність роботи програмного додатку.

Важливо пам'ятати, що прогнозування результатів футбольних матчів завжди має випадковість, і навіть найкращі моделі не можуть гарантувати 100% точність. Однак використання комбінації різних методів та вдосконалення алгоритмів машинного навчання може допомогти зробити прогнози більш точними та інформативними для футбольних експертів та вболівальників.

1.4 Аналіз Dixon-Coles моделі

Dixon-Coles Model - це статистична модель, розроблена Марком Діксоном та Стюартом Коулзом у 1997 році, спеціально для прогнозування результатів футбольних матчів. Модель враховує атакуючі та оборонні характеристики команд, а також специфічні особливості розподілу голів у футболі [4].

Основна ідея моделі полягає в тому, щоб оцінити ймовірність кожного можливого результату матчу, враховуючи силу атаки та оборони кожної команди. Модель використовує такі параметри, як:

– Атакуючі параметри для кожної команди.

– Оборонні параметри для кожної команди.

– Загальний середній рівень атаки та оборони в чемпіонаті.

Ці параметри оцінюються на основі історичних даних про результати матчів та кількість забитих та пропущених голів кожної команди (рисунок 1.6).

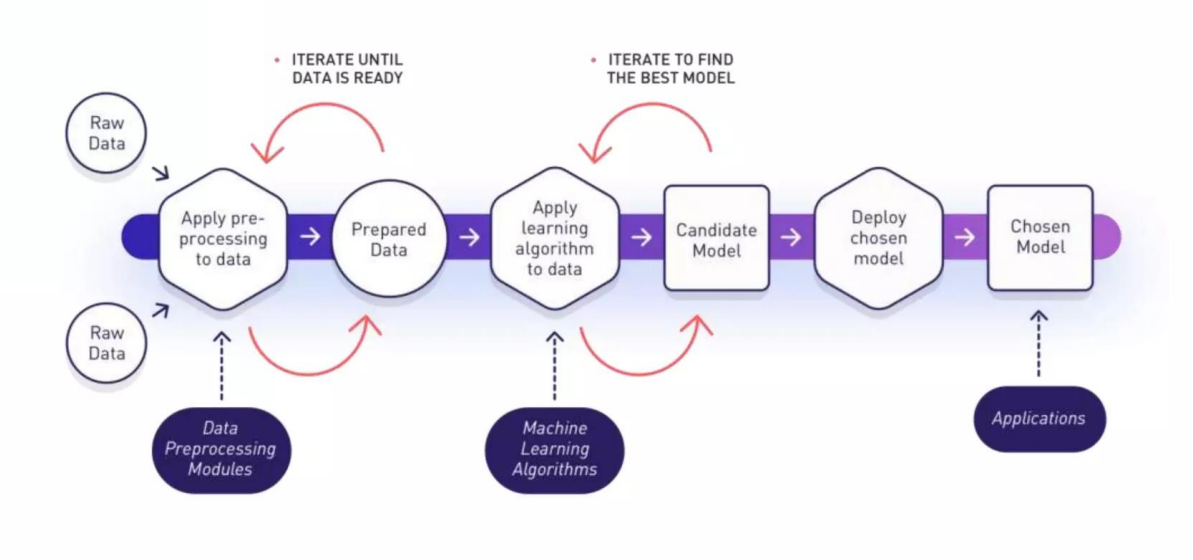


Рисунок 1.6 – Принцип роботи Dixon-Coles Model

Особливістю Dixon-Coles Model є врахування специфічного розподілу голів у футболі. Традиційно, для аналізу часових рядів використовують модель Пуассона, однак вона має тенденцію переоцінювати ймовірність нічийних результатів та підрахунку низької кількості голів. Dixon-Coles Model вносить корективи в цей розподіл для врахування цієї особливості футбольних матчів.

Розподіл Пуассона є дискретним ймовірнісним розподілом, який описує кількість подій, що відбуваються за фіксований проміжок часу або простору, за умови, що середній швидкість подій стала, та події відбуваються незалежно одна від одної (рисунок 1.7). Розподіл Пуассона названо на честь французького математика Сімеона Дени Пуассона [5].

Математично розподіл Пуассона визначається за допомогою ймовірності кількості подій (k) за певний проміжок часу, що розраховується за формулою:

$$P(k) = (e(-\lambda) * \lambda k) / k! \quad (1.1)$$

де P(k) - ймовірність спостереження k подій, λ (лямбда) - середній швидкість подій у заданому інтервалі (часі або просторі), e - основа натурального логарифму (приблизно дорівнює 2.71828), і k! - факторіал числа k (добуток всіх цілих чисел від 1 до k).

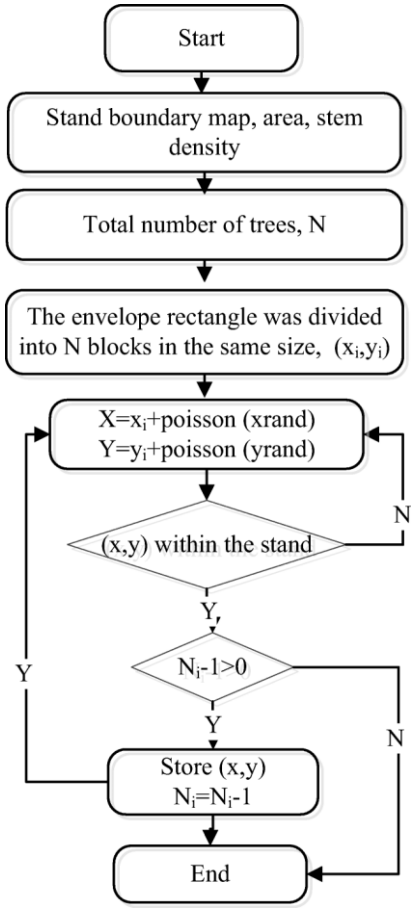


Рисунок 1.7 – Блок-схема алгоритму роботи розподілу Пуасона

Виділяють наступні особливості розподілу Пуассона:

Дискретний – використовується для опису дискретних подій, тобто тих, які можна підрахувати цілими числами (наприклад, кількість дзвінків у кол-центрі за годину).

Безмежний – відноситься до безмежних розподілів, оскільки теоретично може бути будь-яка кількість подій за заданий проміжок часу.

Незалежність подій – події, описані розподілом Пуассона, вважаються незалежними, тобто настання однієї події не впливає на ймовірність настання іншої.

Стала середня швидкість подій – розподіл Пуассона припускає, що середня швидкість подій (λ) є сталою за заданий проміжок часу або простору.

Розподіл Пуассона має широке застосування у математиці, статистиці та наукових дослідженнях для моделювання випадкових подій, таких як кількість звірів, які народжуються у популяції за певний час, або кількість автомобілів, що проїжджають через перехрестя за день. У спортивних прогнозах, зокрема у футболі, розподіл Пуассона також використовується для оцінки ймовірності кількості забитих голів.

Після оцінки параметрів моделі, можна розрахувати ймовірність кожного можливого результату матчу. Ці прогнози можуть бути використані для різних цілей, таких як ставки, аналіз змагань та планування стратегій командами.

Проаналізуємо штучні нейронні мережі.

Прості нейронні мережі застосовуються для базових завдань класифікації, наприклад, коли потрібно зробити прогноз на основі заданих параметрів. Для більш складних випадків, таких як класифікація зображень, обробка природної мови та інше, використовуються глибокі нейронні мережі [22]. Основна відмінність глибоких нейронних мереж від простих полягає в наявності більше одного прихованого рівня, що дозволяє наближати значно складніші функції. Серед глибинних нейронних мереж виділяють два основних види:

- згорткові нейронні мережі (з англ. Convolutional Neural Network або CNN);
- рекурентні нейронні мережі (з англ. Recurrent Neural Network або RNN).

Зазвичай у нейронних мережах кожен вузол пов'язаний з усіма вузлами наступного рівня. Під час навчання мережі ваги зв'язків між вузлами

регулюються, що дозволяє визначити ступінь впливу кожної характеристики на імовірність певного результату.

Базові штучні нейронні мережі, як правило, складаються з 3 шарів:

– шар входу;
– прихований шар;
– шар виходу.

Загальне представлення такої мережі зображено на рисунку 1.8.

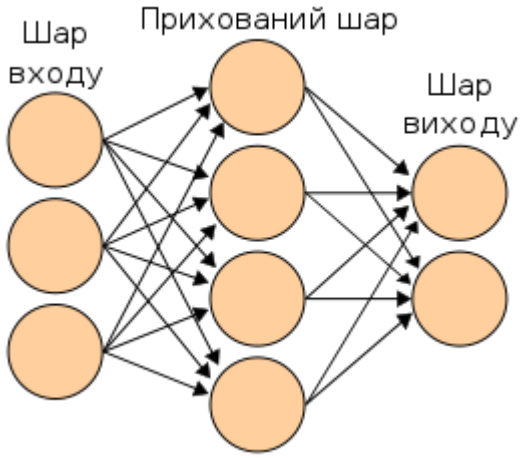


Рисунок 1.8 – Архітектура базової нейронної мережі

Згорткові нейронні мережі – це вид глибоких нейронних мереж, які широко застосовуються у задачах обробки зображень. CNN мають три типи шарів: [23]

– pooling layer (шар об’єднання);
– convolutional layer (шар згортки);
– fully-connected layer (шар повного об’єднання).

Шар згортки обробляє вхідні дані за допомогою набору фільтрів, отримуючи набір характеристик у вигляді n-мірного масиву. Цей масив передається в шар об'єднання, який скорочує кількість характеристик через виконання певних операцій з масивом, що прискорює роботу алгоритму. Послідовність шарів згортки та об'єднання повторюється кілька разів, залежно від типу нейронної мережі. Наприкінці, остаточний набір характеристик

передається до шару повного з'єднання, де аналізуються всі отримані характеристики та визначається кінцевий результат. Приклад архітектури згорткової нейронної мережі наведено на рисунку 1.9.

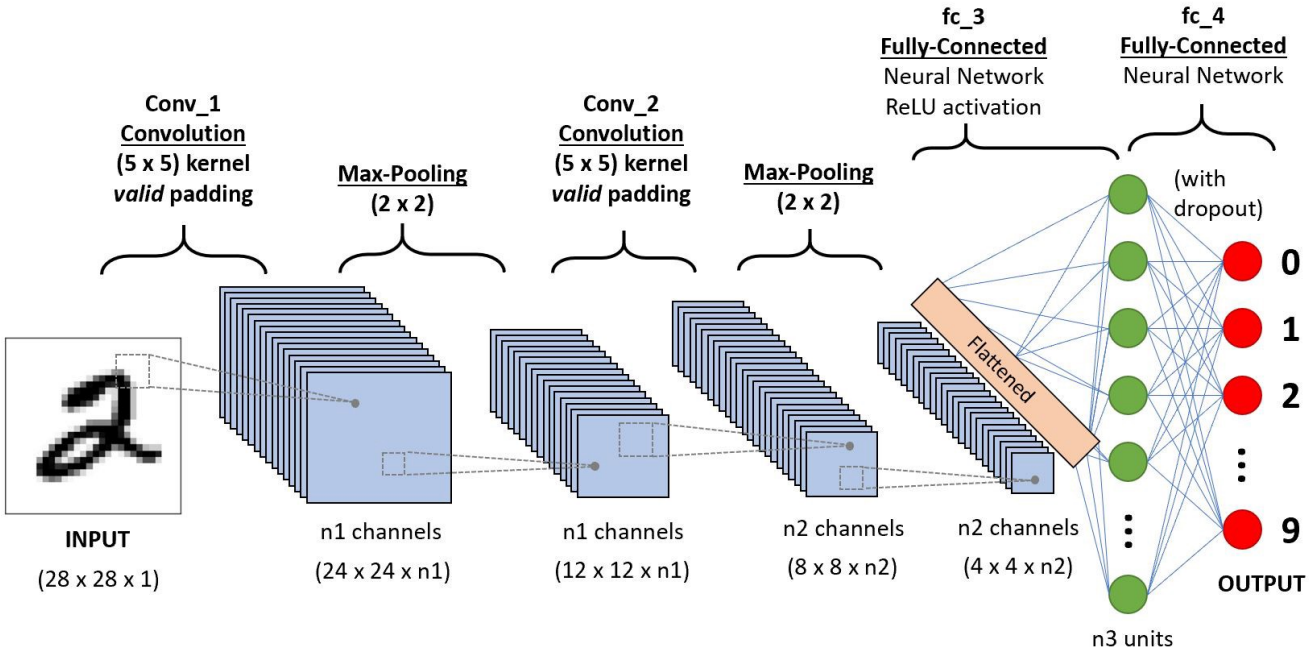


Рисунок 1.9 – Приклад згорткової нейронної мережі

Рекурентні нейронні мережі представляють собою тип нейронних мереж, що здатні обробляти вхідні дані, враховуючи контекстуальну інформацію з попередніх або наступних даних. Вони відрізняються високою ефективністю при роботі з послідовностями, які мають взаємозв'язки, такими як текстові дані, відеоматеріали, аудіозаписи та інші подібні типи інформації [24].

Оскільки рекурентні нейронні мережі можуть опрацьовувати послідовності даних, вони забезпечують розуміння довготермінових залежностей та забезпечують кращий аналіз відносин між елементами послідовності. Це робить їх надзвичайно корисними у таких сферах, як машинний переклад, розпізнавання мови, аналіз емоцій в тексті та генерація тексту. Також вони допомагають у прогнозуванні часових рядів, аналізі відео та аудіо і виявленні аномалій у складних даних.Загальна архітектура таких мереж представлена на рисунку 1.10.

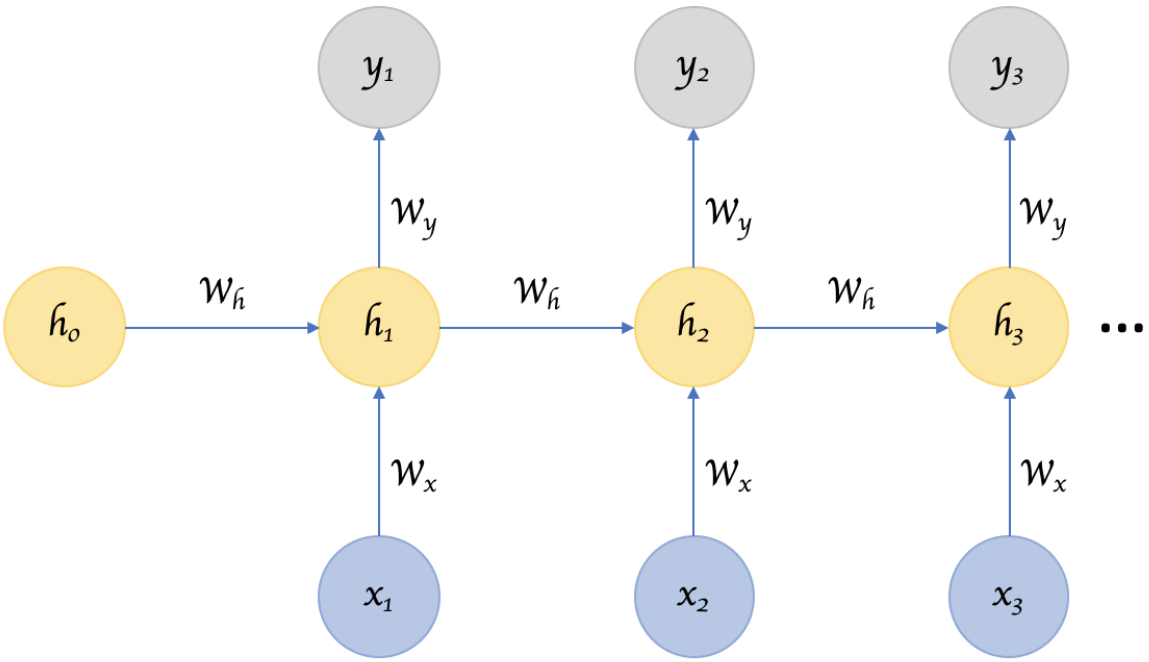


Рисунок 1.10 – Архітектура рекурентної нейронної

Нейронні мережі зазвичай обробляють вхідні дані фіксованого розміру, що ставить обмеження на їх застосування у задачах, де потрібно обробляти послідовності непередбачуваних розмірів, таких як текст. Однак, рекурентні нейронні мережі розроблені таким чином, що вони можуть працювати з послідовностями змінного розміру.

1.5 Постановка задачі

Після аналізу проблеми розробки систем для прогнозування результатів футбольних матчів на основі штучного інтелекту, було визначено завдання, які необхідно виконати для розробки програмного додатку:

- вибрати найефективніший підхід для прогнозування результатів футбольних матчів;
- розробити методику збору та підготовки вхідних даних, таких як статистика команд та гравців;
- створити метод прогнозування результатів футбольних матчів, використовуючи відповідні алгоритми штучного інтелекту;

- розробити модуль для прогнозування результатів, який включає вибраний алгоритм;
- створити веб-сервіс для взаємодії з модулем прогнозування результатів футбольних матчів;
- провести тестування програмного застосунку і оцінити його точність та ефективність.

1.6 Висновки

У першому розділі було розглянуто стан питання прогнозування результатів футбольних матчів на основі моделей штучного інтелекту на сьогоднішній день.

Проаналізовано існуючі аналоги та проведено їх порівняння. У результаті порівняння було наведені їх переваги і недоліки, з чого випливає доцільність розробки власного програмного додатку.

Проведено аналіз існуючих підходів до вирішення поставленої задачі. У результаті аналізу було обрано “Dixon-Coles Model” метод статистичного моделювання який базуються на аналізі часових рядів.

Сформульовано основні завдання, які необхідно виконати для розробки програмного додатку.

2 РОЗРОБКА МЕТОДІВ ТА ЗАСОБІВ РЕАЛІЗАЦІЇ СИСТЕМИ ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ФУТБОЛЬНИХ МАТЧІВ

2.1 Розробка архітектури системи

Першим етапом розробки системи є проектування її загального алгоритму роботи. Розроблювана система працюватиме як веб-сервіс. Таким чином користувач буде взаємодіяти із системою вибираючи матчі, які його цікавлять та натискаючи на кнопку прогнозувати результат. Після натискання кнопки буде відправлятись HTTP запит на сервер, який працюватиме з вхідними даними, статистикою команд, та за допомогою “Dixon-Coles Model” спрогнозує результат. Загальний алгоритм роботи системи зображено на рисунку 2.1.

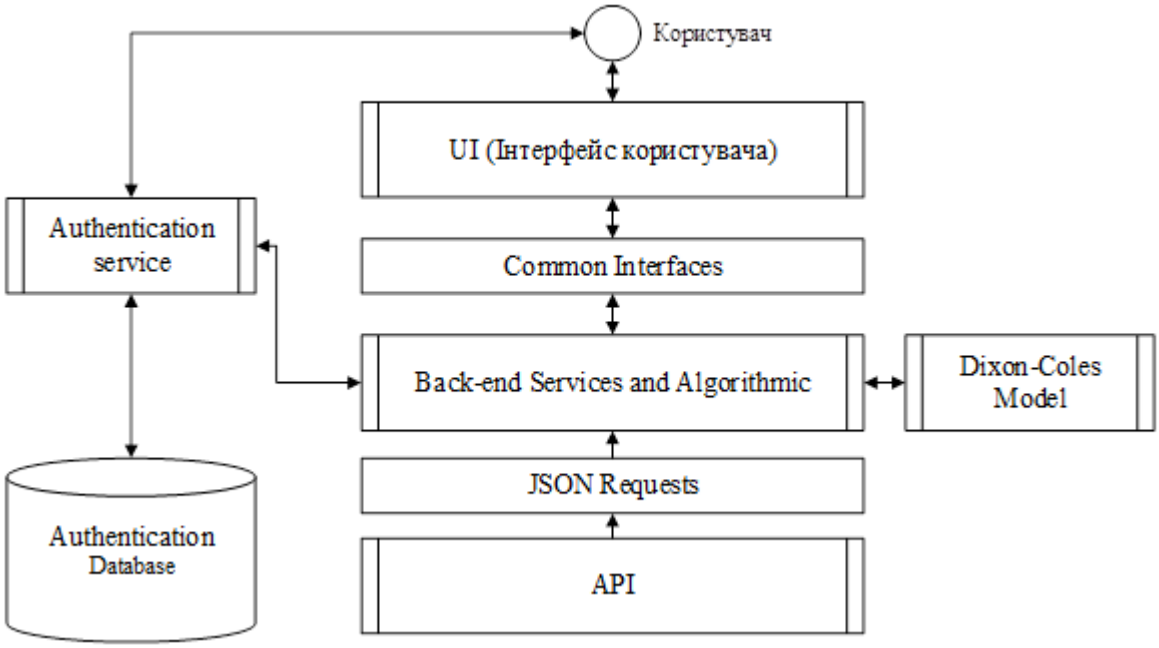


Рисунок 2.1 – Загальний алгоритм роботи системи

Архітектура системи містить п’ять ключових програмних компонентів:

– API;
– Back-end Services та Algorithmic;
– UI (User Interface);

– Authentication service.

– Dixon-Coles model service.

Дана архітектурна модель забезпечує чітке розподілення функцій та ролей між компонентами системи.

Football API - це компонент, що забезпечує отримання великих обсягів статистичних даних та передачу їх у форматі JSON для подальшої обробки [25].

Back-end Services та Algorithmic - це основний функціональний модуль системи, який включає логіку обробки отриманих статистичних даних, сервіси для взаємодії з базою даних, взаємодію з сервісом прогнозування результатів матчу, формує структуру веб-ресурсу та забезпечує виконання функціоналу для користувача.

UI (User Interface) - відповідає за відображення інформації користувачу та дизайн веб-ресурсу.

Authentication service - забезпечує процес верифікації користувача, зокрема, введення імені та пароля при вході на веб-сайт. Правильне введення даних дозволяє користувачеві використовувати функціонал веб-сайту.

Dixon-Coles Model service – відповідає за прогнозування результату матчів, отримує на вхід відсортовані статистичні дані, виконуючи математичне моделювання, повертає результат прогнозу.

Розбиття системи на менші ієрархічні компоненти спрощує процес розробки та допомагає визначити ролі та відповідальності.

Далі розглянемо основні компоненти архітектури системи та засоби їх реалізації.

На рисунку 2.2 представлена структура ключового компонента системи - бек-енду.

Бек-енд включає всі модулі, які реалізують системну логіку через програмні рішення. Він забезпечує проміжне програмне забезпечення для взаємодії з іншими компонентами системи:

– отримання та обробка даних від API;

– зберігання та редагування інформації в базі даних;
– обробка статистичних запитів від користувача;
– реалізація алгоритмів для прогнозування результатів матчів;
– керування користувацьким інтерфейсом.

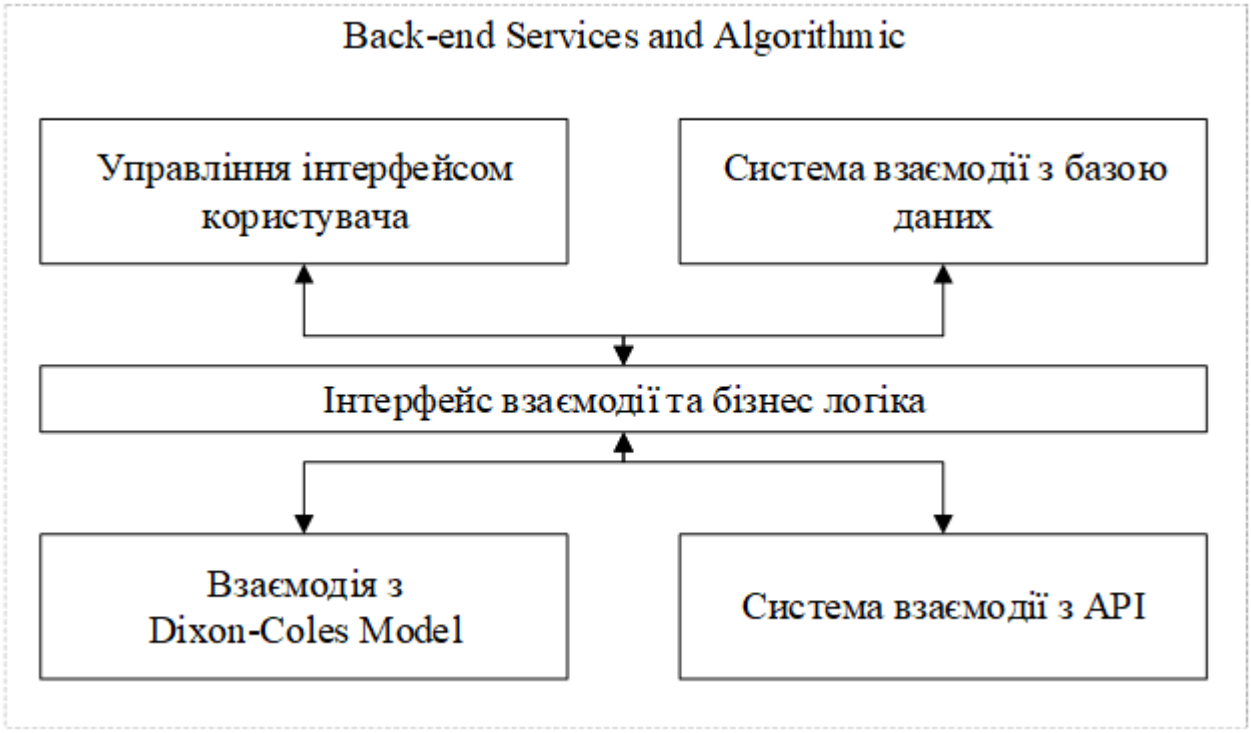


Рисунок 2.2 – Структура та основі функції Бек-енду

Реалізація внутрішніх модулів ґрунтується на принципах проектування MVC та використанні формату JSON для обміну даними між системою та API. Алгоритми обробки статистичної інформації забезпечують аналіз статистичних даних та розрахунок відповідних коефіцієнтів.

На рисунку 2.3 представлені основні компоненти Dixon-Coles моделі для прогнозування результатів футбольних матчів, розглянемо з яких компонентів вона складається:

– оцінка атакуючих та захисних здібностей команд;
– врахування часового фактору;
– параметри домашньої/гостьової переваги;

– максимізація правдоподібності;
– прогнозування результатів;
– оцінка точності прогнозів.

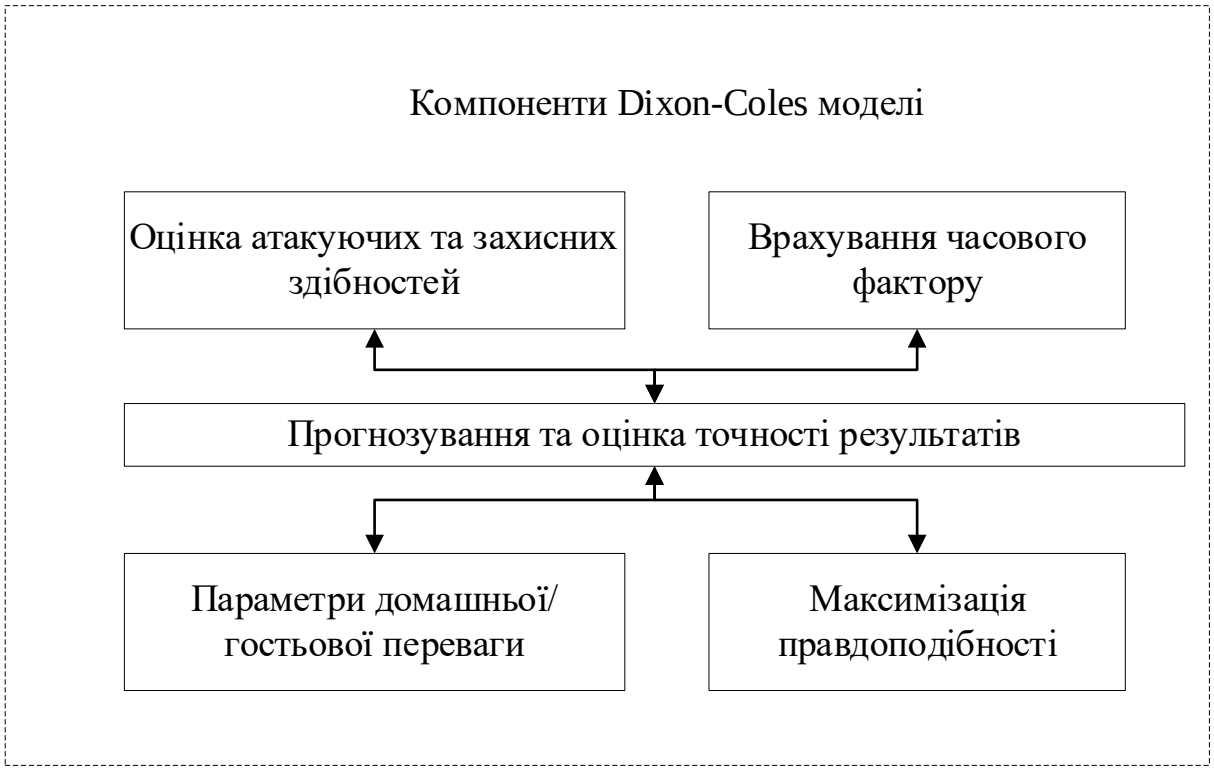


Рисунок 2.3 – Основні компоненти Dixon-Coles моделі

Оцінка атакуючих та захисних здібностей команд враховує історичні дані про голи, забиті та пропущені кожною командою, для визначення їх сили в атакувальній та захисній діяльності.

Врахування часового фактору включає форму команд, яка може змінюватися з часом, модель може включати згладжування часових рядів або інші методи, щоб враховувати актуальну інформації.

Параметри домашньої/гостьової переваги відображають те, що команди, як правило, грають краще на своєму стадіоні, ніж на стадіоні суперника.

Максимізація правдоподібності – це статистичний процес, який використовується для визначення оптимальних параметрів моделі на основі існуючих даних.

Прогнозування результатів на основі оцінки атакуючих та захисних здібностей команд, часового фактору та параметрів домашньої/гостьової переваги, модель Dixon-Coles розраховує ймовірність різних сценаріїв результатів матчу (наприклад, перемога, нічия або поразка).

Для оцінка точності прогнозів можна використовувати різні методи, такі як крос-валідація або порівняння з іншими моделями прогнозування.

2.2 Розробка структури веб-сервісу

Структура веб-сайту визначає його складові частини та їх розташування на сайті. Від неї залежить зручність користувача при пошуку необхідної інформації та комфорт перебування на сайті. Умовну структуру веб-сайту можна поділити на зовнішню та внутрішню.

Веб-сайти створюються з різними метами та цілями, і це значною мірою впливає на їх внутрішню структуру. Отже, для правильного формування внутрішньої структури сайту, необхідно визначити його цілі, мету, цільову аудиторію та забезпечити зручність користувачів під час використання веб-сайту.

Рисунок 2.4 – Структура веб-ресурсу

Хоча веб-сторінки можуть суттєво відрізнятися одна від одної, вони зазвичай містять схожі стандартні компоненти. Ці компоненти включають:

– заголовок;
– навігаційну панель;
– основну робочу область або зміст сторінки;
– бічну панель;
– футер.

Головна сторінка служить візитівкою веб-сайту, і коли користувач переходить на неї, він оцінює її дизайн та функціональність. Меню та основна робоча область є важливими елементами. Зміст робочої області залежить від обраного елемента меню.

Крім того, при відвідуванні сайту, якщо жоден елемент меню не вибрано, відображаються спортивні новини. Для отримання додаткової інформації про новини можна перейти за посиланням, що надається після натискання на новину.

Розглянемо головну сторінку веб-ресурсу. Зазвичай, вгорі веб-сторінки розташована широка смуга з великим заголовком, логотипом та, можливо, слоганом. Зазвичай ці елементи залишаються незмінними при переході з однієї сторінки на іншу.

Навігаційна панель відповідає за посилання на основні розділи веб-сайту. Вона також, як правило, залишається стабільною на різних сторінках. Деякі веб-дизайнери вважають навігаційну панель частинкою заголовка, а не окремим елементом, хоча це не є обов'язковим.

Центральна робоча область містить унікальний зміст веб-сторінки, такий як відео, заголовки новин тощо. Це єдина частина веб-сайту, яка змінюється від сторінки до сторінки.

Бічна панель забезпечує доступ до додаткової інформації, посилань, цитат, оголошень і т.д.

Головна сторінка веб-сайту в цьому контексті має назву "Головна", схематичний вигляд якої представлений на рисунку 2.5.

Рисунок 2.5 – Структура головної сторінки

Нижній колонтитул, або футер, зазвичай містить текст меншого розміру, інформацію про авторські права або контакти. Зазвичай, ця інформація не є важливою для функціонування веб-сайту. Футер також може використовуватися для SEO-цілей, розміщуючи посилання на популярний контент для швидкого доступу.

Після того, як користувач обирає пункт меню "Прогнозування", формат робочої області змінюється. У прикладі наведено список турнірів, де на початку розташовані популярні змагання. У даному випадку користувач вибрав "Англійську Прем'єр-лігу" та в розділі інформації про турнір обрав "Календар".

На рисунку 2.6 схематично зображена сторінка, яка відображається при виборі пункту меню, наприклад «Прогнозування».

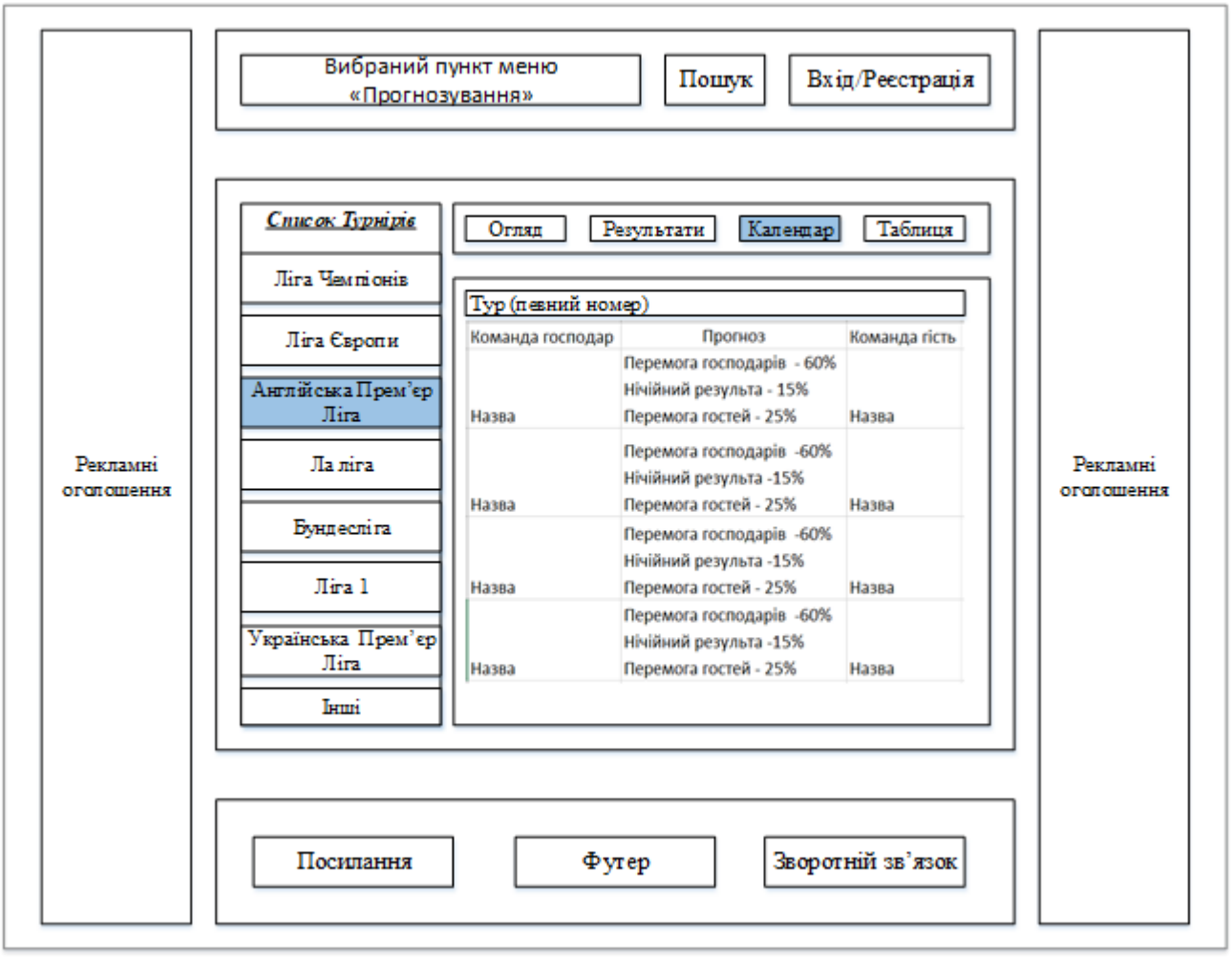


Рисунок 2.6 – Структура сторінки при виборі пункту меню

Календар відображає наступну інформацію: список матчів, час проведення матчу, прогноз, команду-господар та команду-гість.

Кількість матчів у турі залежить від кількості команд, що беруть участь у турнірі. У АПЛ 20 команд, тому в одному турі проводиться 10 матчів. Українська Прем'єр-ліга має 16 команд, отже, в турі проводиться 8 матчів.

За допомогою "Календаря" користувач може дізнатись про майбутні матчі своєї улюбленої команди та отримати прогноз на результати.

Веб-сайт надає всю необхідну інформацію, оскільки інформаційне забезпечення є одним з ключових етапів розробки веб-сторінок. Створення структури та дизайну спрямоване на привертання уваги користувачів до вигляду

сайту, тоді як контент фокусується на обсязі та якості інформації, отриманої відвідувачем.

У підсумку, структурування веб-сторінки відіграє важливу роль у створенні зручного, ефективного та привабливого сайту для користувачів. Наявність чіткої інформаційної архітектури, яка включає такі стандартні компоненти, як заголовок, навігаційна панель, основна робоча область, бічна панель та нижній колонтитул, сприяє зручному пошуку та доступу до потрібної інформації.

Окрім того, врахування цілей, мети та цільової аудиторії сайту при формуванні внутрішньої структури допомагає забезпечити зручність користування та задоволення потреб користувачів. Комбінація чудового дизайну, ефективної структури та якісного контенту забезпечує успіх веб-сайту та підтримує лояльність користувачів.

2.3 Розробка алгоритмів роботи системи

Розробка алгоритму роботи системи потрібна для визначення послідовності кроків та операцій, які система виконує для досягнення конкретної мети. Це допомагає забезпечити ефективність, коректність та оптимальність роботи системи, а також полегшує процес розробки, налагодження та масштабування системи [26].

Система прогнозування результатів футбольних матчів виконуватиме дві головні функції:

– отримання, обробку і представлення статистичної інформації про турніри, команди, матчі, події в матчі, гравців та їх статистику;
– прогнозування результатів самих матчі за допомогою Dixon-Coles моделі.

Умовно класифікуємо алгоритми системи наступним чином:

– алгоритм для здобуття статистичних даних через API;
– алгоритм для передбачення результатів матчів;
– алгоритми обробки та відображення інформації.

Для отримання інформації потрібно розробити алгоритм взаємодії з API. При формуванні запитів до API слід враховувати потребу в конкретній інформації та аналізувати її структуру. Привабливість API полягає у тому, що цей сервіс є великою базою даних з детальною документацією.

Комунікація з API відбувається за допомогою HTTP-запитів. Один зі способів надсилання HTTP-запиту до сервера - це використання методу GET, який є найпопулярнішим та найчастіше використовується. Метод GET, що в перекладі з англійської означає "отримувати", буде основним при виконанні запитів. Найдоступніший спосіб створити запит методом GET - це ввести URL-адресу в адресний рядок веб-браузера.

https://apiv2.apifootball.com/?action=get_teams&APIkey=“value”

Дане посилання (URL) складається з наступних частин:

1. Протокол: https - вказує на використання безпечного протоколу передачі гіпертексту (Secure Hypertext Transfer Protocol).
2. Домен: apiv2.apifootball.com - це адреса сервера, на якому розташований API.
3. Параметри запиту:
   – action=get_teams - вказує на дію, яку потрібно виконати, в даному випадку - отримати список країн.
   – APIkey=«value» - це унікальний ключ доступу до API, який видається розробникам для авторизації та відстеження використання сервісу.

Параметри запиту розділені символом & і починаються з символу ?. Усі ці частини разом формують URL, який використовується для взаємодії з API.

API-ключ є індивідуальним ключем адміністратора системи, який забезпечує конфіденційність даних APIfootball. Це обмежує можливість

незареєстрованих користувачів надсилати запити до їх сервісу, гарантуючи захист інформації [27].

Щоб отримати ключ, потрібно пройти реєстрацію та обрати одну з доступних платних підписок.

Розглянемо детально основні класи даних FootballAPI. Класи даних відображені у таблиці 2.1.

Таблиця 2.1 – Класи даних FootballAPI

Класи	Дія	Опис	Приклад запиту
Countries	**get_countries**	Надає перелік підтримуваних країн	https://apiv2.apifootball.com/?action=get_countries
Competitions	**get_leagues** country_id	Надає перелік підтримуваних змагань	https://apiv2.apifootball.com/?action=get_leagues&country_id=41
Teams	**get_teams** team_id league_id	Надає список відомих команд	https://apiv2.apifootball.com/?action=get_teams&league_id=148&
Players	**get_players** player_id player_name	Надає інформацію про доступних гравців	https://apiv2.apifootball.com/?action=get_players&player_name=ronaldocristiano&
Standings	get_standings league_id	Відображає турнірну таблицю ліг	https://apiv2.apifootball.com/?action=get_standings&league_id=148
Events (Results Fixtures)	get_events timezone from / to	Представляє події та результати матчів	https://apiv2.apifootball.com/?action=get_events&from=2019-04-01&to=2019-04-03&league_id=148
Lineups	get_lineups match_id	Надає склад команд для окремої події	https://apiv2.apifootball.com/?action=get_lineups&match_id=24562

Statistics	**get_statistics** match_id	Відображає статистику окремої події	https://apiv2.apifootball.com/?action=get_statistics&match_id=24562
H2H	**get_H2H** timezone firstTeam secondTeam	Показує останні зустрічі між командами та недавні ігри кожної з них	https://apiv2.apifootball.com/?action=get_H2H&firstTeam=Chelsea&secondTeam=Arsenal

При виконанні запиту результат буде отримано в JSON форматі. На рисунку 2.7 наведений приклад JSON відповіді.


```
[
  {
    "player_key": 3183500916,
    "player_name": "Ronaldo Cristiano",
    "player_number": "7",
    "player_country": "Portugal",
    "player_type": "Forwards",
    "player_age": "34",
    "player_match_played": "31",
    "player_goals": "21",
    "player_yellow_cards": "3",
    "player_red_cards": "0",
    "team_name": "Juventus",
    "team_key": "4187"
  }
]
```


Рисунок 2.7 – JSON відповідь

Інформація наведена у форматі: ("key": "value"). Проте для звичайного користувача, що використовує веб-ресурс, така інформація може здатися складною та незрозумілою. Тому рекомендується обробити цю інформацію та подати її у вигляді, зручному для сприйняття користувачем.

Після дослідження можливостей та сутностей, які надає API, був розроблений алгоритм для отримання статистичної інформації (рисунок 2.8).

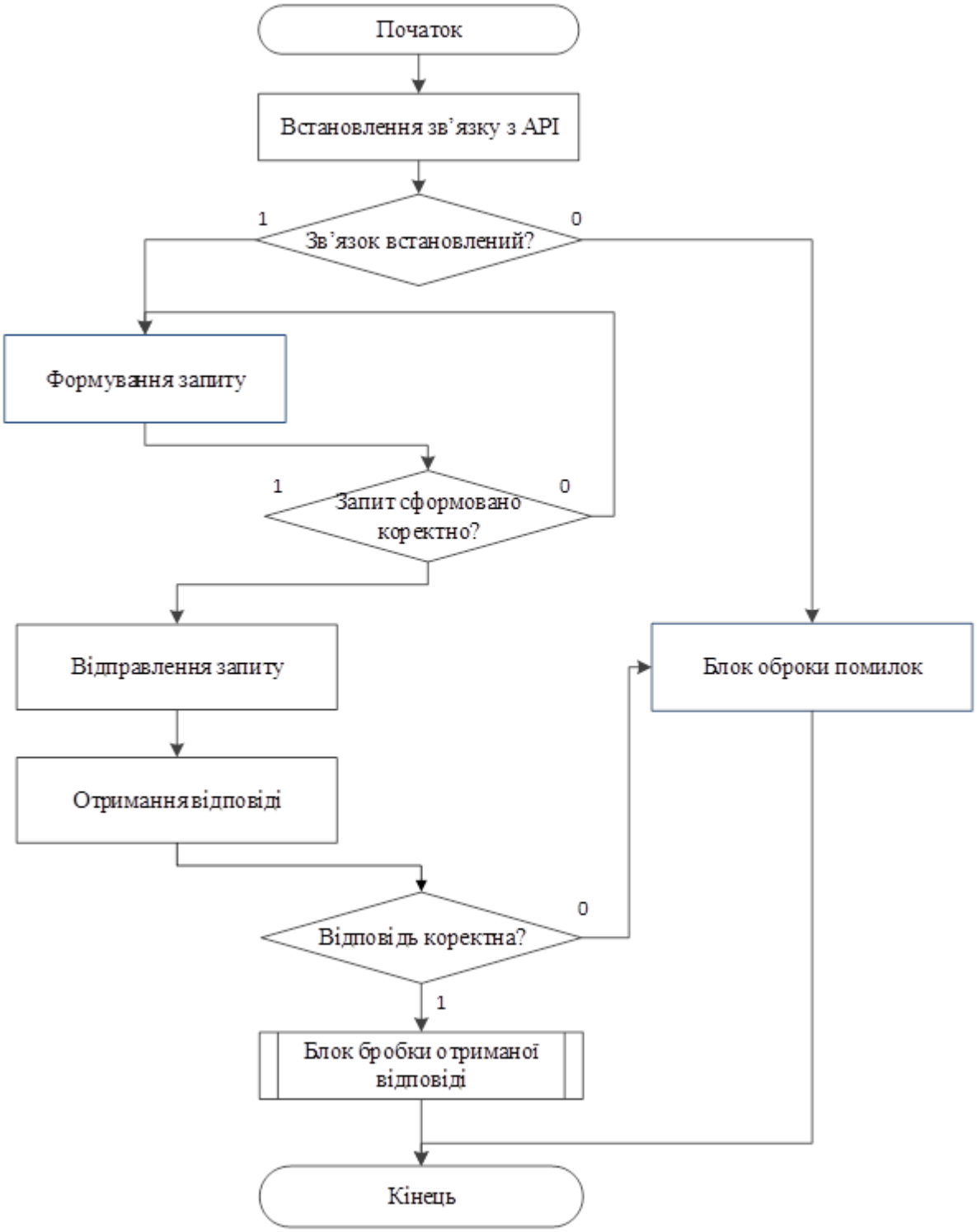


Рисунок 2.8 – Алгоритм для отримання статистичної інформації

Розглянемо основні функції цього алгоритму.

Встановлення зв'язку з API є початковим кроком алгоритму, для цього необхідно надіслати запит на сервер із даними для реєстрації на сервері.

Якщо зв'язок успішно встановлено, формується запит, залежно від потрібної сутності (див. таблицю 1.2), до якого додаються відповідні параметри. Якщо зв'язок не встановлено, алгоритм переходить до блоку обробки помилок, який визначає конкретну помилку.

Якщо запит сформовано правильно, він відправляється на сервер; в інакшому випадку, формування запиту починається знову.

Після надсилання запиту API надсилає дані у форматі JSON. Частина алгоритму, відповідальна за отримання відповіді, перевіряє її на відповідність формату JSON та, якщо все вірно, передає її до блоку обробки отриманої відповіді. У разі невідповідності формату, алгоритм переходить до блоку обробки помилок.

Розглянемо детальніше алгоритм модифікації та зберігання даних. На рисунку 2.8 представлено універсальний алгоритм модифікації інформації, отриманої від API. Для кожної сутності він містить певні зміни, оскільки отримується структурно різна інформація.

Алгоритм модифікації залежно від отриманої інформації, створює об'єкт або список об'єктів з певними параметрами для подальшої взаємодії. Створені об'єкти використовуються, як статистичні дані для прогнозування результатів.

Залежно від виду інформації, яку отримує сервіс, алгоритм виконує функції зберігання даних у базі даних системи для оптимізації кількості запитів до API.

Приклад: Користувач хоче переглянути турнірну таблицю АПЛ і натискає відповідну кнопку на сайті. Система розпізнає цю дію, аналізує необхідну інформацію та надсилає запит до API. Після отримання інформації алгоритм конвертує JSON Result у об'єкт.

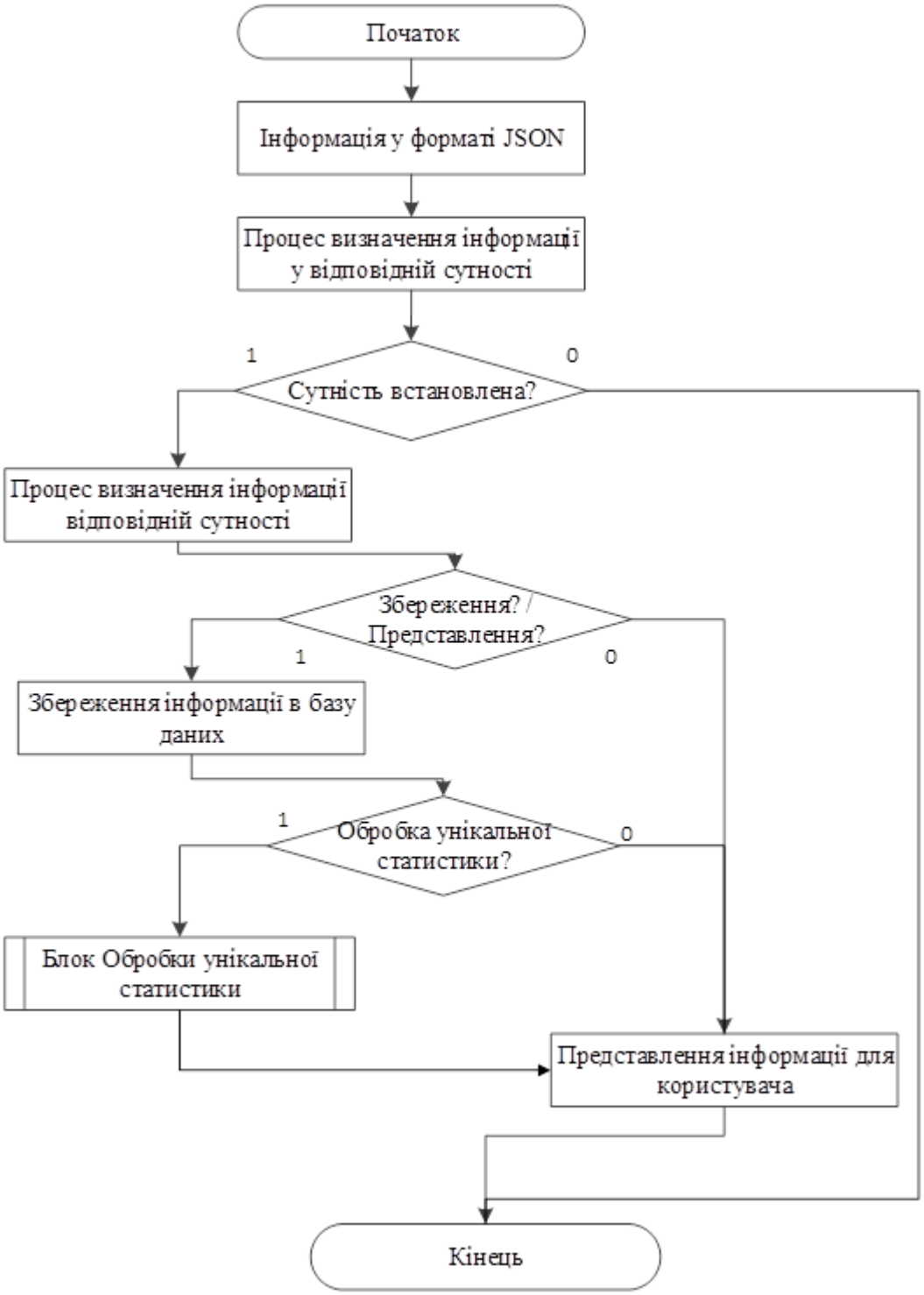


Рисунок 2.9 – Алгоритм для модифікації та збереження інформації

Турнірна таблиця АПЛ є сутністю API, тому її можна відобразити на сайті без збереження в базі даних і без використання алгоритму прогнозування результату матчу.

Якщо ж користувач відкриває календар матчів, або сам матч виконуються наступні дії:

1. Виконується запити до API та отримуються необхідні статистичні дані для прогнозування результату.
2. Далі формуються відформатований набір статистичних даних, такі як форма команди, фактор домашнього поля, статистика за останні 5 матчів кожної команди, статистика зустрічей між командами, турнірне положення, орієнтований склад на матч, суддівська бригада.
3. Відформатовані статистичні дані використовуються, як вхідні параметри для моделі Dixon-Coles.
4. Модель аналізує данні, та повертає ймовірність трьох результатів (перемога, нічия, поразка). Сума ймовірностей дорівнює 1.

Наприклад: перемога – 0.65, нічия – 0.2, поразка – 0.15.

Інший приклад: Користувач хоче дізнатися, яка команда забила найбільше голів у турнірі. Для розрахунку цієї інформації потрібно зробити кілька запитів до API, обробити та зберегти дані в базі даних. Потім, використовуючи відповідний алгоритм, визначити команду та відобразити дані для користувача.

2.4 Розробка методу визначення спеціального коефіцієнту потенціалу команди

Розглянемо алгоритми, які використовують статистику для визначення загального потенціалу команди у матчі. Окрім звичайної статистики, були розроблені алгоритми, які розраховують силу команди у кожній лінії, з метою підвищення точності прогнозування результату матчу.

Для кожної позиції встановлено статистичні критерії, за якими оцінюється кожен гравець у стартовому складі.

Спочатку необхідно визначити схему, за якою буде побудована команда. Враховуючи сучасні тенденції щодо позиційної гри та команд, які віддають

перевагу домінуючому футболу, була обрана схема 4-3-3. Тобто 4 захисника, 3 півзахисника та 3 нападника.

На рисунку 2.10 наведений алгоритм визначення загального потенціалу команди.

Рисунок 2.10 – Алгоритм визначення загального потенціалу команди

Для нападників основним показником є кількість забитих голів, а також важливою є кількість гольових передач. Оскільки для деяких нападників одного точного удару достатньо для забиття голу, а іншим потрібно більше десяти, буде враховано співвідношення ударів і голів. Введемо коефіцієнт ефективності нападника, який позначимо як ЕН.

$$ЕН = Г * 0.6 + П * 0.4 - (\frac{Г}{У} * 0.1) \quad (2.1)$$

де ЕН коефіцієнт ефективність нападника;

Г – кількість м'ячів, забитих протягом сезону;

П – кількість результативних передач.

У командах, що надають перевагу ефектному футболу, півзахисник може виконувати роль "організатора гри" (ПП) або опорного півзахисника (ОП). В даній схемі з трьома півзахисниками будуть два організатори гри та один опорний півзахисник.

Основне завдання ПП полягає у керуванні грою та розробці гольових ситуацій для нападників. Також враховується кількість вирішальних передач за матч. Вирішальна передача - це передача партнеру у сприятливому положенні для забиття гола (вихід один на один, положення перед порожніми воротами та ін.).

Введемо коефіцієнт ефективності плеймейкера, який позначимо як ЕПМ.

$$ЕПМ = Г * 0.2 + П * 0.5 + \frac{КП}{І} * 0,3 \quad (2.2)$$

де ЕПМ - коефіцієнт успішності півзахисника-організатора гри;

Г - кількість голів, забитих протягом сезону;

П - кількість результативних передач;

КП - кількість вирішальних передач;

Основне завдання ОП полягає у відслідковуванні та контролю руху гравців противника, відборі м'яча від суперника в центральній зоні та точному передаванні м'яча партнерам у команді.

Введемо коефіцієнт ефективності опорного півзахисника, який позначимо як ЕОП.

$$ЕОП = \frac{В}{І} - \frac{Ф}{І} + ТП * 0,0015 \quad (2.3)$$

де ЕОП – коефіцієнт ефективності опорного півзахисника;

В – кількість успішних відборів протягом сезону;

Ф – кількість порушень правил;

ТП – відсоток точних передач;

І – кількість зіграних матчів за сезон.

Основне завдання захисників - запобігти забиттю гола командою суперника. Захисники поділяються на центральних та крайніх. Зазвичай у схемі 4:3:3 є два центральних захисника і два крайніх захисника. Для центрального захисника характерні такі навички, як гра головою, стрибки та підкати. Крайнього захисника можна визначити як універсального гравця, він виконує функції як у захисті, так і в нападі.

Введемо коефіцієнт ефективності центрального захисника, як ЕЦЗ.

$$ЕЦЗ = \frac{І}{П} + Є * 0{,}01 \quad (2.4)$$

де ЕОП – коефіцієнт ефективності захисника;

П – кількість пропущених голів команди;

Є – відсоток успішної боротьби;

І – кількість зіграних матчів за сезон.

Введемо коефіцієнт ефективності крайнього захисника, як ЕКЗ.

$$ЕКЗ = \frac{І}{П} + ГП * 0{,}2 \quad (2.5)$$

де ЕКЗ – коефіцієнт ефективності крайнього захисника;

П - кількість пропущених м'ячів командою;

ГП - кількість результативних передач;

І - кількість зіграних матчів за сезон.

Основне завдання голкіпера - запобігти забиттю гола опонентом у свої ворота. Важливим параметром для воротаря є відношення кількості ударів у рамку воріт до кількості пропущених м'ячів. Відбитий удар воротарем називається сейвом. Однак, протягом однієї гри команда суперника може не

завдати жодного удару у рамку воріт. Тому вводиться поняття матчу, зіграного на "нуль".

Позначимо коефіцієнт успішності для голкіпера, як ЕВ.

$$\text{ЕВ} = \frac{\text{І}}{\text{П}} + \frac{\text{Н}}{\text{І}} + \frac{\text{П}}{\text{С}} \tag{2.6}$$

де ЕВ - коефіцієнт ефективності воротаря;

П - кількість пропущених м'ячів;

Н - кількість зіграних матчів на "нуль";

С - кількість сейвів;

І - кількість зіграних матчів за сезон.

Визначивши коефіцієнти для усіх гравців команди, додамо їх та порівняємо з командою суперником. Удосконалені статистичні дані передаються в модель Dixon-Coles.

2.5 Розробка покращеного методу прогнозування результатів футбольних матчів на основі моделі Dixon-Coles.

Діксон і Коулз визначили двомірну модель із граничним розподілом Пуассона. Відповідна функція спільної ймовірності маси (pmf) визначається як:

$$P(X_{i,j} = x, Y_{j,i} = y) = \tau_{\lambda,\mu}(x)\frac{e^{-\lambda}\lambda^{x}}{x!}\frac{e^{-\mu}\mu^{y}}{y!} \text{ де } \lambda = \alpha_i\beta_j\gamma\mu \tag{2.7}$$

$$= \alpha_j\beta_i\tau_{\lambda,\mu}(x,y) = \begin{cases} 1-\lambda\mu\rho & \text{if } x = y = 0 \\ 1-\lambda\rho & \text{if } x = 0, y = 1 \\ 1+\mu\rho & \text{if } x = 0, y = 1 \\ 1-\rho & \text{if } x = y = 1 \\ 1 & \text{інакше} \end{cases}$$

де λ1 та λ2 – середні значення двох граничних розподілів Пуассона, а τλ1,λ2 (·, ·) вимірює кореляцію між балами. Для x1 > 1 і x2 > 1 ймовірності залишаються

незмінними від добутку граничних розподілів, оскільки ймовірності зсуваються лише між чотирма парами (0, 0), (1, 0), (0, 1) і (1, 1). Величина зсуву залежить від параметра залежності ˜ω, який має задовольняти нерівність:

$$\max(-1/\lambda 1, -1/\lambda 2) \leq \tilde{\omega} \leq \min(1/\lambda 1\lambda 2, 1) \qquad (2.8)$$

Таким чином, ˜ω може приймати позитивні або негативні значення, але це обмежений діапазон для відповідних засобів λ1 та λ2. Випадок де ˜ω = 0 відповідає незалежним оцінкам (рисунок 2.11). Зокрема, для зростаючих значень ˜ω ймовірності під Модель Діксона та Коулза зсувається від (0, 0) та (1, 1) до пар (0, 1) та (1, 0) у пропорційний спосіб [4].

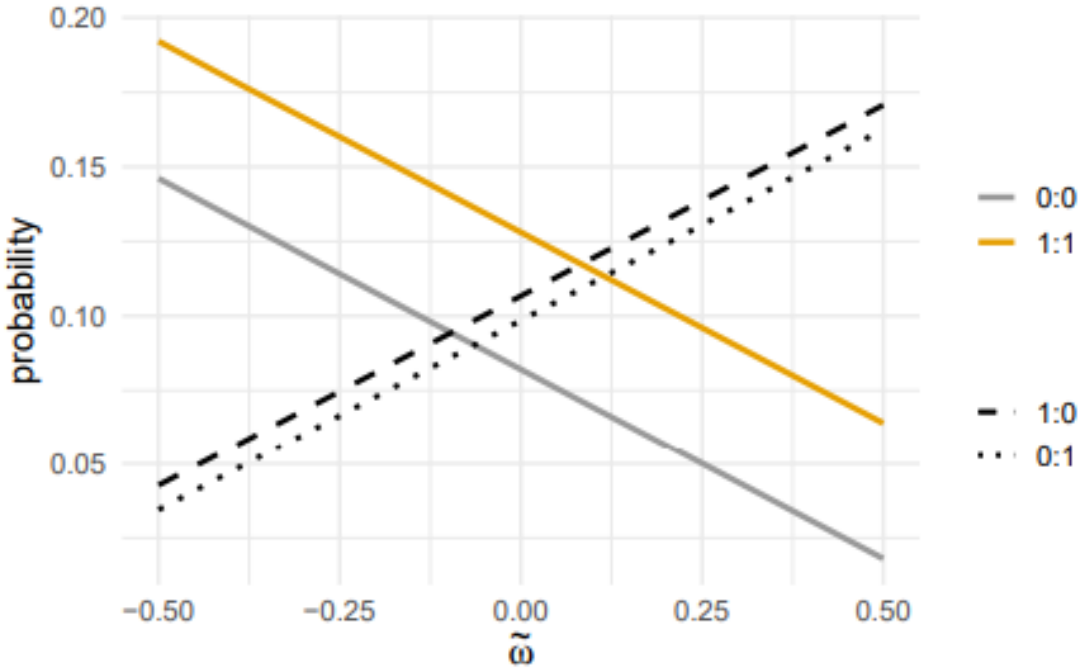


Рисунок 2.11 – Вплив параметра залежності ˜ω в моделі Діксона і Коулза на ймовірності для пар (0, 0), (1, 0), (0, 1) і (1, 1).

У моделі Діксона-Коулза рівняння Пуассона використовується для моделювання кількості голів, забитих командами під час матчу. Рівняння Пуассона є дискретним розподілом ймовірностей, який описує ймовірність

фіксованої кількості подій, які відбуваються протягом певного проміжку часу або простору, якщо ці події мають сталу середню швидкість і незалежні між собою.

У контексті футбольних матчів, рівняння Пуассона використовується для прогнозування кількості голів, які заб'ють команди на основі їх атакуючих та захисних здібностей. Відповідно до моделі, атакуючі здібності однієї команди комбінуються з захисними здібностями іншої команди для визначення очікуваної кількості голів, які будуть забиті під час матчу.

Розглянемо удосконалений алгоритм моделі Діксона-Коулза на рисунку 2.12.

Рисунок 2.12 – Алгоритм формування прогнозу результату матчу на основі моделі Діксона-Коулза

Розглянемо кожен етап детально. На основі отриманих та відформатованих даних від API формуються вхідні дані. До них відноситься статистичні дані минулих матчів (датасет), розроблений коефіцієнт потенціалу команд та параметри оптимізації.

Наступний етап це підготовка даних, в даному етапі виконується визначення унікальності кожної команди, перевірка яка команда є господарем, а яка гостем.

Далі виконується ініціалізація параметрів моделі, а саме формуються атакуючі та захисні коефіцієнти команд, які корегуються розробленим коефіцієнтом потенціалу команди. Також враховуються коефіцієнти корекції рахунку (rho) та домашньої переваги (gamma).

Наступним етапом є виклик функції логарифмічної правдоподібності, яка виконує наступні дії:

– обчислення очікуваних голів для домашньої та виїзної команд;
– обчислення правдоподібності результату матчу за розподілом Пуассона;
– обчислення коефіцієнта корекції rho;
– обчислення загальної логарифмічної правдоподібності.

Отже, рівняння Пуассона в даному випадку дозволяє розрахувати ймовірність того, що команда заб'є певну кількість голів на основі їх атакуючих та захисних параметрів.

Далі необхідно виконати оптимізацію на основі наступних параметрів:

– мінімізація від'ємної логарифмічної правдоподібності;
– урахування обмежень та параметрів оптимізації.

Останнім етапом є генерація вихідних даних. Формуються оптимальні параметри атаки та захисту для кожної команди і повертається результат прогнозу. Після розрахунку ймовірностей для різних кількостей голів, які можуть бути забиті командами, модель може визначити найбільш ймовірні результати матчу, а саме ймовірність перемоги, поразки і нічиї.

2.6 Розробка алгоритму авторизації

Авторизація - це система безпеки, яка встановлює рівні доступу або права користувача/клієнта, пов'язані з ресурсами системи, включаючи файли, сервіси, програми, дані та функції додатків. Цей процес дозволяє чи обмежує доступ до мережевого ресурсу, надаючи користувачам можливість отримати доступ до різних ресурсів на основі встановленої ідентифікації користувача [28].

Більшість систем веб-безпеки працюють на основі двофазного процесу. Перший крок полягає в автентифікації, яка забезпечує визначення особи користувача, а другий крок - авторизація, яка надає користувачам доступ до різних ресурсів залежно від їх ідентифікації. Сучасні операційні системи покладаються на добре розроблені процеси авторизації для спрощення розгортання та керування додатками. Важливими факторами є тип користувача, номер та облікові дані, які потрібно перевірити на відповідність діям та ролям.

Керування доступом у комп'ютерних системах та мережах засноване на політиці доступу, який поділяється на дві стадії:

1. Етап встановлення політики, коли доступ допускається.
2. Етап додержання політики, під час якого запити на доступ дозволяються або забороняються.

Отже, авторизація є складовою етапу встановлення політики, який передує етапу додержання політики, коли запити на доступ схвалюються або відхиляються на основі попередньо встановлених дозволів. Алгоритм реєстрації наведено на рисунку

Контроль доступу також використовує процес автентифікації для підтвердження особистості користувача. Якщо користувач намагається отримати доступ до ресурсу, система контролю доступу перевіряє, чи має користувач право на використання цього ресурсу. Сервер безпеки реалізує послуги авторизації, які можуть регулювати доступ на рівні окремих файлів чи програм.

Реєстрація - це процес зберігання даних користувача, зокрема імені користувача та паролю, з метою надання доступу до функціональності веб-сайту. Велика кількість веб-сайтів вимагає реєстрації користувачів на платформі. У цьому випадку зареєстровані користувачі мають змогу отримати доступ до формування індивідуальної статистики [29].

Розглянемо алгоритм реєстрації, який використовує система прогнозування результатів футбольних матчів (рисунок 2.13).

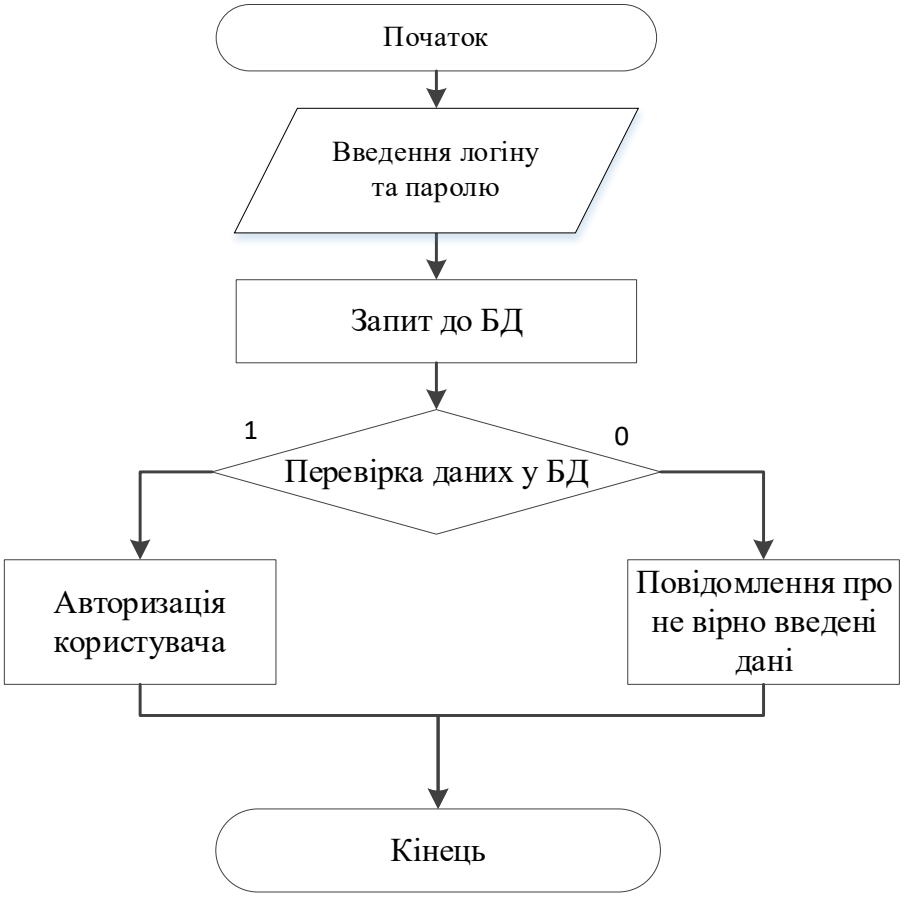


Рисунок 2.13 – Алгоритм реєстрації на веб-ресурсі

Отже, авторизація та реєстрація важливі для забезпечення безпеки та контролю доступу до ресурсів веб-сайту чи системи. Вони дозволяють відокремлювати користувачів за рівнями доступу, забезпечуючи захист від

несанкціонованого доступу та забезпечуючи індивідуальні налаштування для кожного користувача.

2.6 Висновки

У другому розділі було розроблено архітектуру та структуру веб-сервісу для прогнозування результатів футбольних матчів на основі штучного інтелекту.

Розроблено загальні алгоритми роботи додатку, такі як алгоритми отримання, обробки і представлення статистичної інформації про турніри, команди, матчі, події в матчі, гравців та їх статистику. Алгоритми прогнозування результатів матчів за допомогою удосконаленої моделі Діксона-Коулза.

Розроблено метод визначення загального потенціалу команди, який удосконалює модель Діксона-Коулза, що на відміну від існуючих враховує індивідуальну статистику кожного гравця на відповідній позиції для підвищення точності прогнозу.

Запропоновано представлення не тільки статистичної інформації про матч для користувача, а комбіноване – що надає можливість переглядати статистику і прогноз на матч.

3 РОЗРОБКА СИСТЕМИ ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ФУТБОЛЬНИХ МАТЧІВ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного засобу.

Вибір відповідних технологій для створення будь-якого програмного рішення є важливою частиною процесу розробки, оскільки відповідна технологія може значно спростити процес втілення. У процесі створення додатку було вирішено використати єдину мову програмування, а саме C# на платформі .NET 6.0.

Станом на сьогодні, мова програмування C# є однією з найбільш потужних, динамічно розвиваючих у сфері IT. За допомогою цієї мови створюються різні програми - від компактних десктопних застосунків до великих веб-порталів та веб-сервісів, які обслуговують мільйони користувачів щодня [30].

Мова програмування C# має зручний синтаксис, а .NET 6.0 надає потужну бібліотеку класів, що спрощують розробку та реалізацію різних функцій веб-ресурсу. Що дозволяє швидше розробляти веб-застосунки та проводити їх тестування та налагодження.

Завдяки оптимізації у .NET 6.0, веб-ресурси створені за допомогою мови програмування C# є продуктивними та ефективними.

Для розробки веб-сервісу прогнозування результатів футбольних матчів доцільно використовувати технологію ASP.NET Core MVC. Це дозволить зробити процес обробки та відображення статистичної інформації оптимізованим.

ASP.NET Core MVC - це сучасний фреймворк для створення веб-застосунків, розроблений компанією Microsoft на базі платформи .NET Core. MVC - це абревіатура від Model-View-Controller, що означає використання

архітектурного шаблону проектування, який спрощує розробку та підтримку веб-застосунків [31].

Основні компоненти ASP.NET Core MVC:

Модель (Model) - відповідає за представлення даних та бізнес-логіку застосунку. Модель включає в себе класи, які представляють об'єкти доменної моделі, та логіку для роботи з даними та взаємодії з базами даних чи іншими зовнішніми сервісами.

Перегляд (View) - відповідає за відображення даних користувачеві у вигляді HTML-сторінок. Види можуть використовувати різні технології для створення динамічного контенту, такі як Razor, JavaScript чи CSS.

Контролер (Controller) - є центральною частиною архітектури MVC, керує взаємодією між моделями та переглядами. Контролер обробляє вхідні HTTP-запити, виконує бізнес-логіку та передає дані між моделями та переглядами.

Особливості ASP.NET Core MVC:

Дозволяє легко додавати нові функції та компоненти до застосунку за допомогою модульної архітектури та підтримки зовнішніх пакетів.

Фреймворк пропонує потужну систему маршрутизації, яка дозволяє налаштовувати правила для обробки URL-адрес та спрямування запитів до відповідних контролерів та дій.

ASP.NET Core MVC має вбудовану систему впровадження залежності (Dependency Injection), що спрощує управління компонентами застосунку та забезпечує гнучкість.

Фреймворк підтримує модульне тестування та інтеграційне тестування, що дозволяє створювати надійні та стійкі до помилок веб-застосунки. Також даний фреймворк включає ряд вбудованих механізмів безпеки, таких як захист від - кросплатформеного скриптінгу (XSS), кросплатформеної підміни запитів (CSRF) та автентифікацію та авторизацію користувачів.

Для роботи з моделями машинного навчання доцільно використовувати бібліотеку ML.NET.

ML.NET - це відкрита, крос-платформна бібліотека машинного навчання для розробників .NET, розроблена Microsoft. ML.NET дозволяє розробникам легко інтегрувати моделі машинного навчання безпосередньо в .NET-додатки без необхідності використання інших мов програмування або зовнішніх бібліотек [32]. Переваги використання даної бібліотеки:

- Крос-платформеність надає можливість бути використаною на різних платформах, таких як Windows, Linux та macOS, завдяки підтримці .NET Core.
- Широкий спектр вбудованих алгоритмів машинного навчання, які покривають різні завдання, включаючи класифікацію, регресію, ранжування, кластеризацію, виявлення аномалій, прогнозування часових рядів та рекомендації.
- Модельні трансформації які дозволяють обробляти та підготовлювати дані перед тренуванням моделі. Це включає кодування категоріальних даних, масштабування числових даних, обробку тексту, зменшення розмірності та багато іншого.
- Легкість використання без необхідності глибоких знань в цій області. Вона пропонує інтуїтивний API, який забезпечує простоту використання та швидке створення та впровадження моделей машинного навчання.
- Інтеграція з іншими бібліотеками та фреймворками дозволяє працювати з популярними бібліотеками машинного навчання, такими як TensorFlow та ONNX.

ML.NET є потужним та гнучким інструментом для розробників .NET, які хочуть використовувати машинне навчання в своїх додатках, не залежачи від їх досвіду в галузі штучного інтелекту.

З урахуванням усіх зазначених вище характеристик та переваг, мова програмування C# виявляється найбільш підходящою для створення веб-складової системи пошуку іменованих сутностей.

3.2 Розробка веб-сервісу

Розробка веб-сервісу являється одним із головних завдань, адже для отримання прогнозу на певний матч, користувачу потрібно взаємодіяти з певним зручним інтерфейсом. Для розробки веб-сервісу було обрано використання мови програмування – C#, в середовищі Microsoft Visual Studio 2019 [33].

Використання архітектурного шаблону ASP.NET Core MVC дозволяє легко налаштувати систему маршрутизації та впровадження залежностей (Dependency Injection). Головна мета веб-сервісу це обробка та відображення футбольної статистики та прогнозу на матчі.

Загальна структура проекту в середовищі Microsoft Visual Studio 2019 наведена на рисунку 3.1

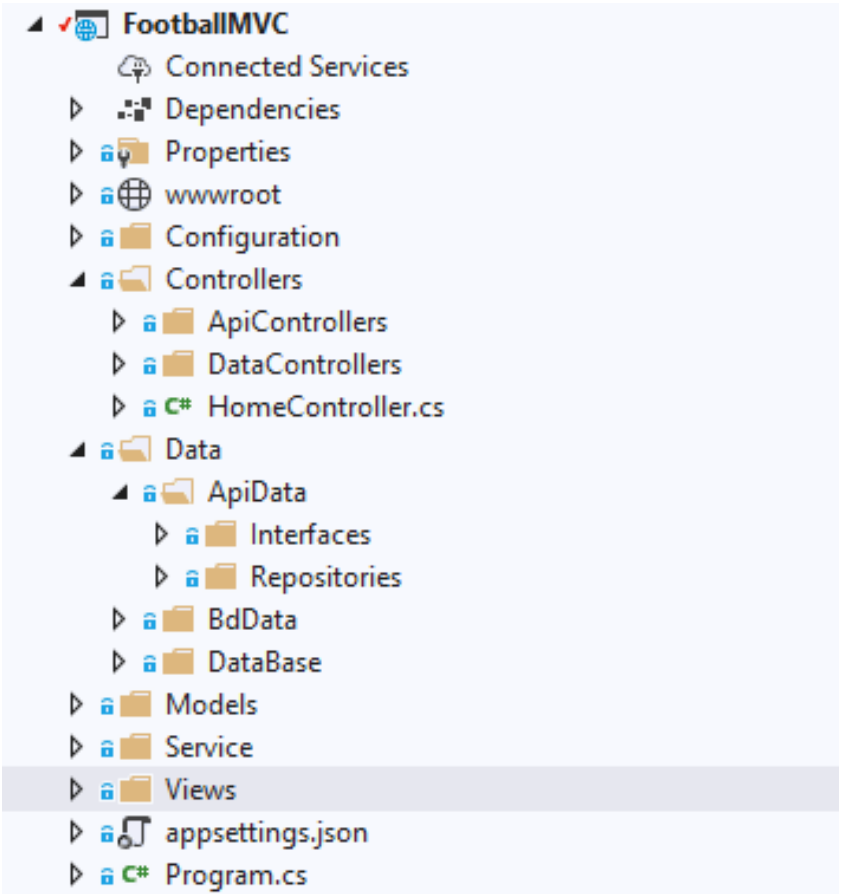


Рисунок 3.1 – Загальна структура проекту

Клас Program.cs є вихідною точкою виконання для будь-якого проекту на платформі .NET. Він відповідає за ініціалізацію та запуск веб-додатку. У версії .NET 6.0 клас Program.cs було спрощено, згорнувши його до одного файлу з меншим обсягом коду порівняно з попередніми версіями.

Даний клас має наступні головні методи:

- CreateBuilder()
- Build();
- Run().

Метод Main() є вихідною точкою для будь-якого проекту на .NET. Він створює та запускає веб-хост та конфігурує його з використанням налаштувань із файлів конфігурації та середовища.

CreateBuilder() метод створює новий екземпляр WebApplicationBuilder, який використовується для налаштування та створення веб-додатку. Він автоматично застосовує налаштування із файлів конфігурації та змінних середовища.

Метод Build() створює екземпляр веб-додатку (WebApplication) з поточними налаштуваннями та конфігурацією. Він також реєструє всі необхідні сервіси та об'єкти у контейнері залежностей.

Метод Run() запускає веб-додаток, відкриває порт для прослуховування вхідних запитів і переходить у режим очікування.

Для взаємодії з базою даних за допомогою Entity Framework потрібен контекст даних - клас, який унаслідується від класу Microsoft EntityFramework DbContext. Таким чином, було розроблено клас AppDBContext.cs. Цей клас містить ключову властивість DbSet, яка представляє собою набір об'єктів, пов'язаних з відповідною таблицею в базі даних.

Наприклад, для таблиці Players в базі даних є відповідна властивість: public DbSet<Player> Players { get; set; }. Оперуючи властивістю Players, можна отримувати, додавати та змінювати дані в базі даних. Клас AppDBContext.cs успадковується від інтерфейсу DbContext.

На рисунку 3.2 зображено діаграма класів для роботи з Entity Framework

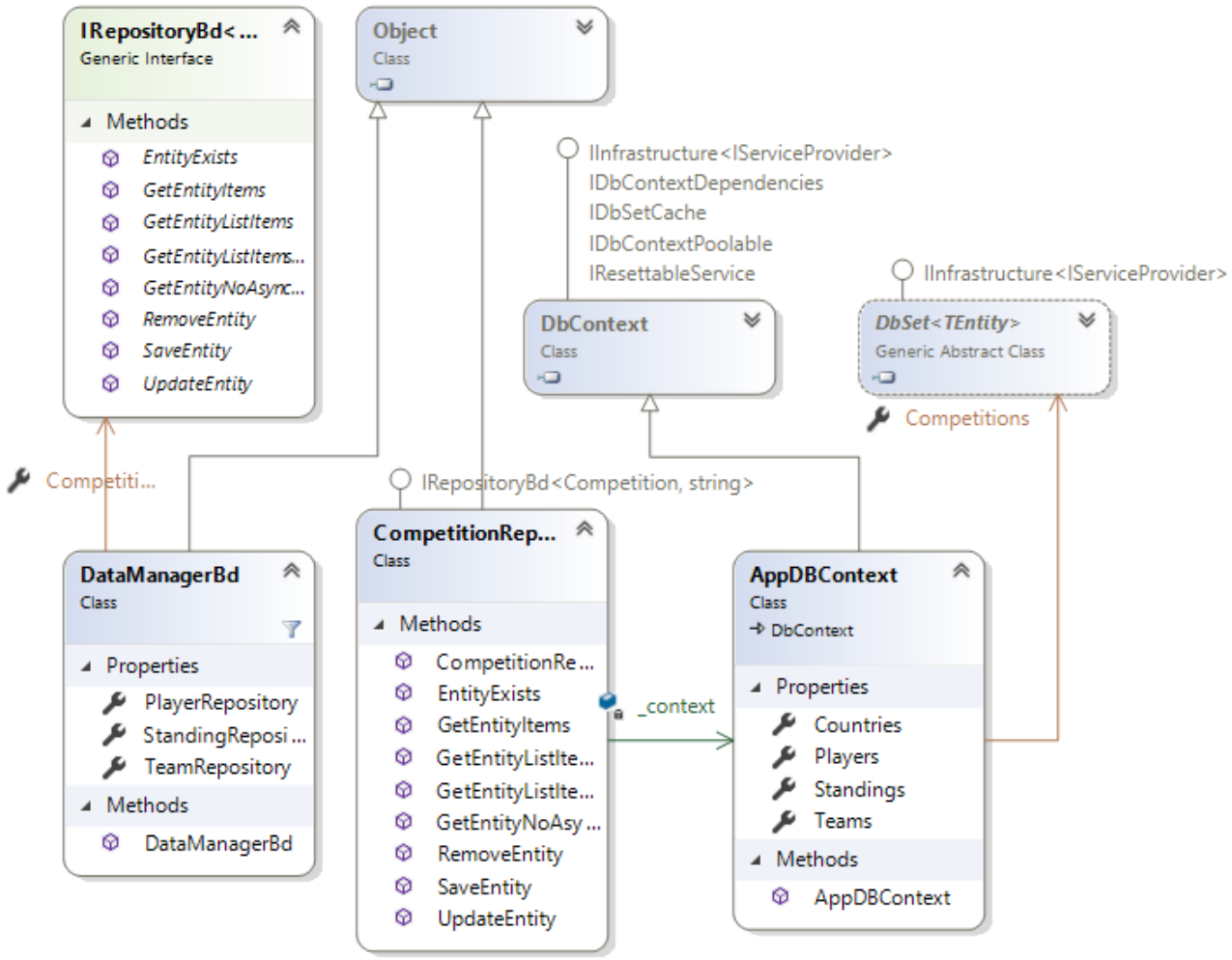


Рисунок 3.2 – Діаграма класів для роботи з Entity Framework

Інтерфейс IRepoDB визначає методи, які дозволяють виконувати різні запити до бази даних. Наприклад, на рисунку 3.3 представлено клас CompetRepoDB, який реалізує ці методи.

Наступним завдання є взаємодія з API. API - це інтерфейс програмування застосунків (API), створений для спрощення розробки веб-служб, які взаємодіють через HTTP. Він дозволяє розробникам створювати легкі, швидкі та масштабовані RESTful веб-сервіси, що надають доступ до ресурсів (даних,

функцій тощо) для різних клієнтських додатків, таких як веб-сторінки, мобільні або настільні програми.

Основне завдання це сформувати та відправити коректний HTTP запит, оскільки виконується взаємодія зі стороннім API, в документації якого описаний формат відправки запитів. В основному це HTTP GET запити.

Для кожної сутності статистики було розроблено клас, який містить в собі методи для формування запиту, його відправки та валідації отриманої відповіді.

На рисунку 3.3 зображено діаграму класів для спілкування зі стороннім API.

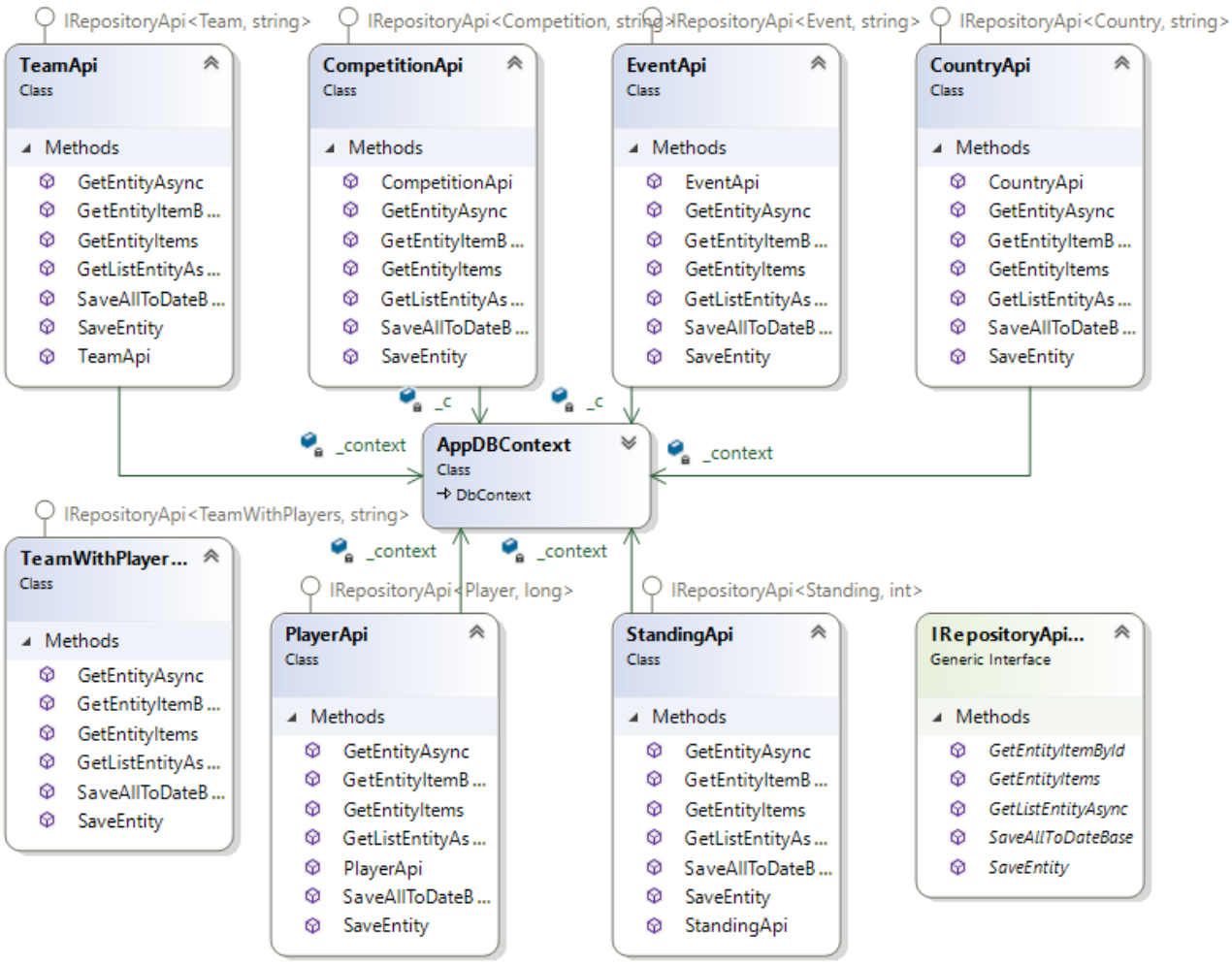


Рисунок 3.3 – Діаграма класів для спілкування з API

При розробці веб-ресурсів важливо дотримуватись, принципів зрозумілого коду, для можливості його масштабування. Тому класи, для взаємодіють з API імплементують інтерфейс IRepositoryApi<T,K>, методи якого відображені в таблиці 3.1.

Таблиця 3.1 – Методи інтерфейсу IRepositoryApi<T,K>

№	Назва методу	Перший аргумент	Пояснення	Другий аргумент	Пояснення	Повертає значення	Пояснення
1	GetEntityAsync	string path	Шлях до потрібного ресурсу	HttpClient client	Клієнт для здійснення HTTP запитів	Task<T>	Асинхронне завдання, що повертає екземпляр типу T
2	GetListEntityAsync	string path	Шлях до потрібного ресурсу	HttpClient client	Клієнт для здійснення HTTP запитів	Task<List<T>>	Асинхронне завдання, що повертає список екземплярів типу T
3	SaveAllToDateBase	List<T> elements	Список елементів типу T для збереження	-	-	List<T>	Список елементів типу T, збережених в базі даних
4	GetEntityItems	-	-	-	-	IQueryable<T>	Колекція елементів типу T, отриманих з бази даних
5	GetEntityItemById	K id	Ідентифікатор елемента типу K	-	-	T	Екземпляр типу T, який відповідає переданому ідентифікатору
6	SaveEntity	T entity	Екземпляр типу T для збереження	-	-	Task	Асинхронне завдання для збереження екземпляра типу T в базі даних

public Standing GetEntityItemById(int id) - метод, який приймає на вхід ідентифікатор (id) та повертає об'єкт Standing, який відповідає цьому ідентифікатору, шляхом пошуку в контексті бази даних _context.Standings.

public async Task<List<Standing>> GetListEntityAsync(string path, HttpClient client) - асинхронний метод, який приймає шлях до API-ресурсу та об'єкт HttpClient, а повертає список об'єктів Standing. Метод виконує запит до API за допомогою HttpClient, перевіряє, чи є відповідь успішною, а потім десеріалізує JSON-рядок, отриманий з відповіді, у список об'єктів Standing. У разі помилки повертається null.

Кожен клас який унаслідує інтефейс повинен імплементувати всі його методи. В таблиці 3.2 наведені реалізовані методи класу StandingAPI, та описано функції, за які вони відповідють

Таблиця 3.2 – Методи класу StandingAPI

№	Назва методу	Призначення
1	StandingApi(AppDBContext context)	Конструктор класу, ініціалізує контекст бази даних для роботи з таблицею Standing
2	GetEntityAsync(string path, HttpClient client)	Завантажує один об'єкт типу Standing з API за вказаним шляхом та клієнтом HTTP, асинхронно
3	GetEntityItemById(int id)	Знаходить та повертає об'єкт типу Standing з бази даних за заданим ідентифікатором (int id)
4	GetEntityItems()	Повертає всі об'єкти типу Standing з бази даних у вигляді IQueryable<Standing>
5	GetListEntityAsync(string path, HttpClient client)	Завантажує список об'єктів типу Standing з API за вказаним шляхом та клієнтом HTTP, асинхронно
6	SaveAllToDateBase(List<Standing> standings)	Зберігає всі об'єкти типу Standing у базі даних, які відсутні в таблиці, та повертає список збережених об'єктів
7	SaveEntity(Standing entity)	Зберігає один об'єкт типу Standing у базі даних, асинхронно

В архітектурному патерні MVC важливим елементом є моделі. Розглянемо діаграму класів предметної області на рисунку 3.4

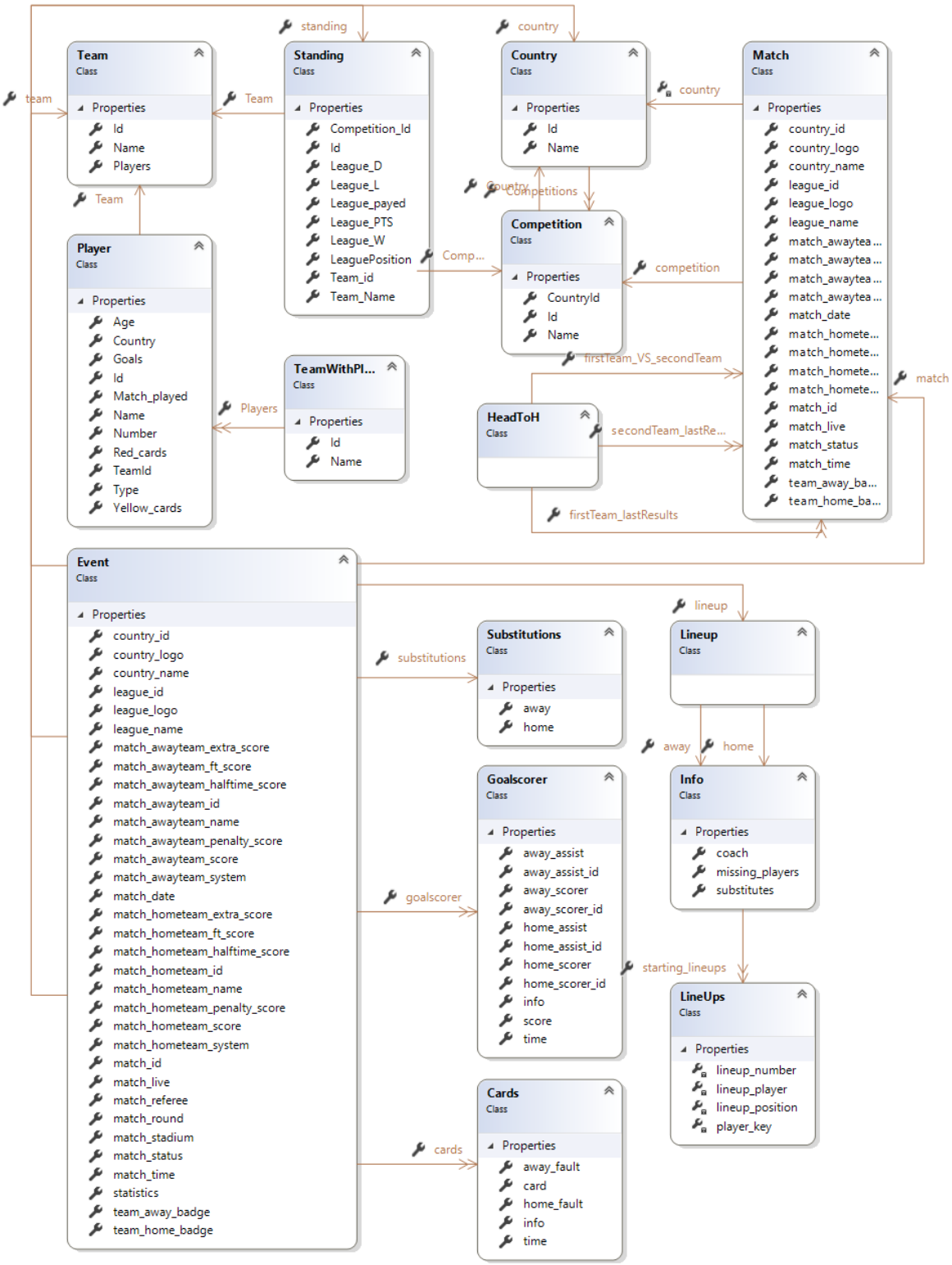


Рисунок 3.4 – Діаграма класів футбольної статистики

Моделі відповідають за зберігання, обробку та отримання даних, а також за інтеграцію з базами даних чи іншими джерелами даних. Вони слугують основою для представлень та контролерів, які забезпечують відображення даних та управління користувацькими діями відповідно. Моделі дозволяють відокремити логіку предметної області від інтерфейсу користувача, сприяючи більшій гнучкості та легкості управління кодом.

Діаграма класів, які взаємодіють з контролером CompetitionApiController зображено на рисунку 3.5.

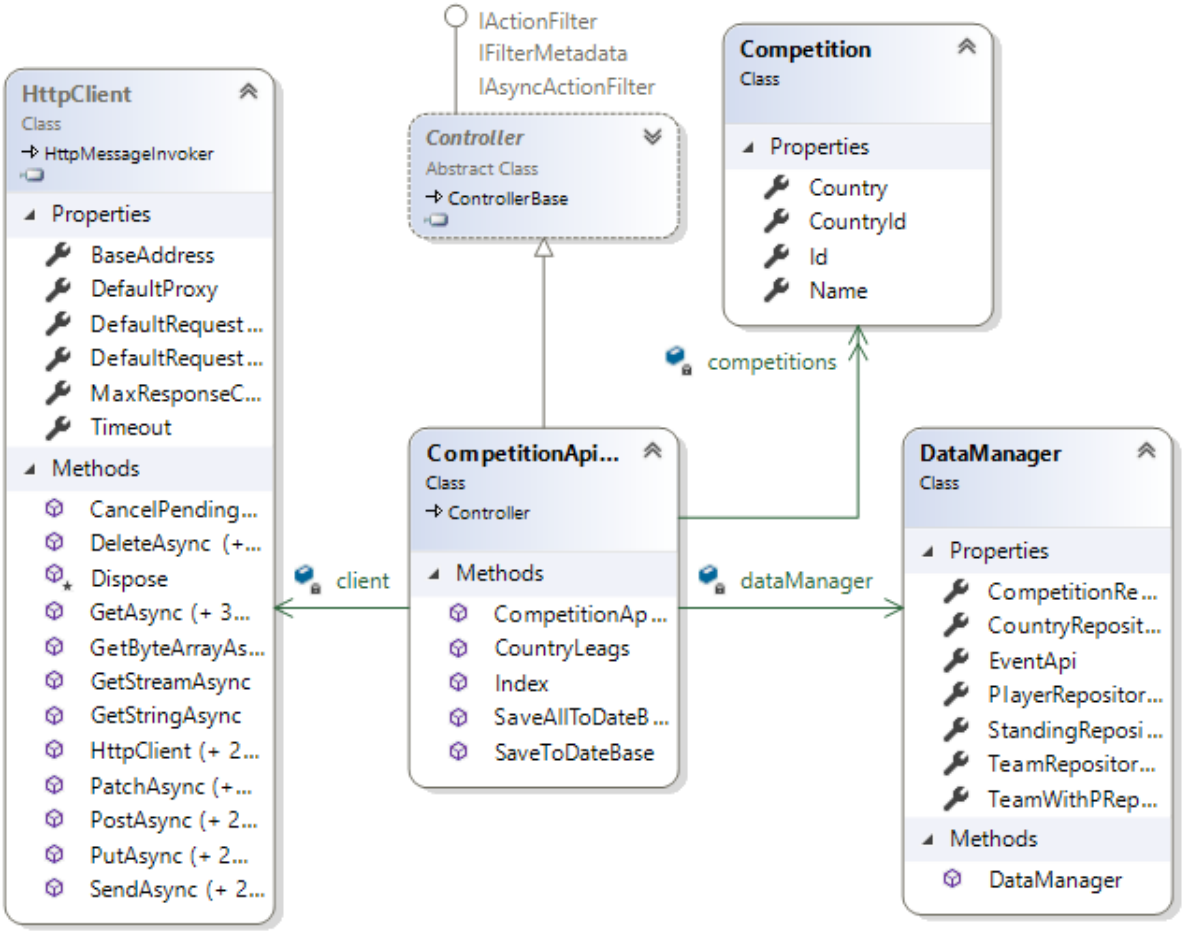


Рисунок 3.5 – Діаграма класів взаємодії сервісів з контролером

Контролер відповідає за обробку користувацьких взаємодій та управління потоком додатку. Завдяки контролеру користувацькі запити обробляються,

відповідні моделі оновлюються, а потім відповідні представлення вибираються для відображення даних користувачеві. Контролер допомагає забезпечити розділення логіки доменної області (моделі) та інтерфейсу користувача (представлення), сприяючи чистоті коду та полегшуючи розробку та супровід додатку.

Представлення (View) в моделі Model-View-Controller (MVC) - це компонент, який відображає інформацію користувачеві та забезпечує візуальний інтерфейс для взаємодії. Відповідає за відображення даних, отриманих від моделі, і передає дії користувача (наприклад, кліки на кнопки) до контролера. Представлення зазвичай реалізовано за допомогою шаблонів та стилів, які визначають структуру, вигляд і оформлення сторінки. Вони не містять бізнес-логіки, а лише забезпечують зручний спосіб відображення даних на екрані. Завдяки розділенню обов'язків між моделлю, контролером і представленням, MVC дозволяє створювати гнучкі та масштабовані програми з чіткою структурою.

Для відображення статистичної інформації та відповідно самий прогноз на матч, було створено набір представлень, які зберігаються в розділі Views. В таблиці 3.3 відображено список види представлень, та файли в середині них.

Таблиця 3.3 – Перелік представлень

Вид представлення	Файли
CompetitionApi	CountryLeags.cshtml, Index.cshtml, SaveAllToDateBase.cshtml, SaveToDateBase.cshtml
Competitions	Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, Index.cshtml
Countries	Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, Index.cshtml
CountryApi	Index.cshtml, SaveAllToDateBase.cshtml, SaveToDateBase.cshtml
Home	Index.cshtml, Privacy.cshtml

Продовження таблиці 3.3

Main	CompetitionsInfo.cshtml, DetailsMatch.cshtml, GetAllCountries.cshtml, GetListCompetitionsByCountry.cshtml, Index.cshtml, LastResults.cshtml, ViewStanding.cshtml, ViewTemsByLeagID.cshtml, _Menu.cshtml
PlayerApi	Index.cshtml, SaveAllToDateBase.cshtml, ViewPlayersByTeamId.cshtml
Players	Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, Index.cshtml
Shared	Error.cshtml, SidebarPartial.cshtml, _CssPartial.cshtml, _Footer.cshtml, _Header.cshtml, _Layout.cshtml, _Meta.cshtml, _ValidationScriptsPartial.cshtml
Shared\Components\Standings	Default.cshtml
StandingApi	Index.cshtml, SaveAllToDateBase.cshtml, ViewStandingByLeagID.cshtml
Standings	Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, Index.cshtml
StatisticAPI	Index.cshtml
TeamApi	Index.cshtml, SaveAllToDateBase.cshtml, SaveToDateBase.cshtml, ViewTemsByLeagID.cshtml
Teams	Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, Index.cshtml

У патерні MVC, файл Layout.cshtml відіграє роль макета, який визначає загальну структуру та дизайн веб-сторінки. Цей файл використовується для визначення спільних елементів інтерфейсу, таких як верхня панель навігації, бокове меню, футер та інші ресурси, які будуть використовуватися на більшості сторінок веб-додатку. _Layout.cshtml дозволяє уникнути повторення коду, оскільки спільні елементи описуються в одному місці та вбудовуються в різні представлення, що полегшує управління та розробку додатку. Окрім

_Layout.cshtml, також важливими є файли _Header.cshtml, _Sidebar.cshtml та _Footer.cshtml (таблиця 3.4).

Таблиця 3.4 – Перелік представлень, які знаходяться в папці Shared

Назва файлу	Опис
Error.cshtml	Сторінка для відображення повідомлень про помилки.
SidebarPartial.cshtml	Часткове представлення для змісту бічної панелі.
_CssPartial.cshtml	Часткове представлення для завантаження файлів CSS.
_Footer.cshtml	Часткове представлення для нижнього колонтитула сторінки.
_Header.cshtml	Часткове представлення для верхнього колонтитула сторінки.
_Layout.cshtml	Основний макет для додатка.
_Meta.cshtml	Часткове представлення для метатегів та SEO.
_ValidationScriptsPartial.cshtml	Часткове представлення для завантаження скриптів

Отже, було розроблено веб-сервіс для прогнозування результатів футбольних матчів. Також додатковою перевагою є можливість перегляду статистики усіх команд.

3.3 Розробка моделі прогнозування

Розробка удосконаленої моделі Діксона-Коулза з використанням розробленого методу визначення загального потенціалу команди є важливим модулем веб-сервісу.

Першим етапом буде завантаження та оброка даних про матч. Необхідно зібрати дані про минулі матчі, та відповідну статистику. Оскільки отримання статистичних даних реалізовано задопомогою веб-сервісу, необхідно їх відфільтрувати.

Оброблені статистичні данні будуть зберігатись в класі MatchData.cs. Опис параметрів класу наведено в таблиці 3.5.

Таблиця 3.5 – Опис параметрів класу

Змінні	Призначення	Діапазон значень
HomeTeam	Ідентифікатор домашньої команди	String (не більше 100 знаків)
AwayTeam	Ідентифікатор гостьової команди	String (не більше 100 знаків)
MatchDate	Дата матчу	DateTime
HomeTeamGoals	Кількість голів, забитих домашньою командою	int (0 - 1000)
AwayTeamGoals	Кількість голів, забитих гостьовою командою	int (0 - 1000)
HomeTeamShots	Кількість пострілів домашньою командою	int (0 - 1000)
AwayTeamShots	Кількість пострілів гостьовою командою	int (0 - 1000)
HomeTeamShotsOnTarget	Кількість пострілів у створ воріт домашньою командою	int (0 - 1000)
AwayTeamShotsOnTarget	Кількість пострілів у створ воріт гостьовою командою	int (0 - 1000)
HomeTeamCorners	Кількість кутових ударів домашньою командою	int (0 - 1000)
AwayTeamCorners	Кількість кутових ударів гостьовою командою	int (0 - 1000)
HomeTeamFouls	Кількість фолів домашньою командою	int (0 - 1000)
AwayTeamFouls	Кількість фолів гостьовою командою	int (0 - 1000)
HomeTeamYellowCards	Кількість жовтих карток домашньою командою	int (0 - 1000)
AwayTeamYellowCards	Кількість жовтих карток гостьовою командою	int (0 - 1000)
HomeTeamRedCards	Кількість червоних карток домашньою командою	int (0 - 1000)
AwayTeamRedCards	Кількість червоних карток гостьовою командою	int (0 - 1000)

Наступним етапом є визначення функції втрат, яка моделює ймовірність голів, забитих кожною командою в матчі. У ML.NET визначемо власну функцію втрат, створивши естімейтор та перетворювач даних. В таблиці 3.6 наведено назву та опис основних методів класу DixonColesLikelihoodTransformer.cs

Таблиця 3.6 – Основні методи класу DixonColesLikelihoodTransformer.cs

Назва	Призначення
Options	Вкладений клас, який містить налаштування для
Lambda	Параметр регуляризації, який використовується в моделі Діксона-Колза.
InputColumnNames	Масив назв вхідних колонок, які трансформатор використовує для обчислення ймовірностей.
OutputColumnNames	Масив назв вихідних колонок, які трансформатор створює після обчислення ймовірностей.
DixonColesLikelihoodTransformer(MLContext context, string[] inputColumnNames, string[] outputColumnNames, Options options)	Конструктор класу, який приймає контекст машинного навчання, вхідні та вихідні колонки та налаштування.
IsRowToRowMapper	Властивість, яка вказує, чи є трансформатор мапером рядка до рядка.
GetOutputSchema(SchemaShape inputSchema)	Метод, який отримує схему вихідних даних на основі вхідної схеми.
GetRowToRowMapper(SchemaShape inputSchema)	Метод, який повертає мапер рядка до рядка (не реалізований для цього трансформатора).
GetRowTransforms()	Метод, який повертає масив перетворень рядків даних, які використовуються трансформатором.
CalculateDixonColesLikelihood(DataViewRow row, int i, ref float score)	Метод, який обчислює ймовірності за моделлю Діксона-Колза на основі вхідних даних.

DixonColesLikelihoodTransformer.cs – перетворювач, який реалізує інтерфейс Itransformer.cs. Основними методами якого є public SchemaShape

GetOutputSchema(SchemaShape inputSchema) який повертає new SchemaShape (columns).

GetRowTransforms який викликає обчислення функції правдоподібності Діксона-Коулза, враховує параметри моделі, такі як сили атаки та захисту команд, а також гіперпараметри, як lambda.

DixonColesLikelihoodEstimator.cs – естімейтор який реалізує інтерфейс IEstimator<DixonColesLikelihoodTransformer> . Основними методами якого є DixonColesLikelihoodTransformer Fit (IDataView input) та SchemaShape GetOutputSchema (SchemaShape inputSchema) В таблиці 3.7 наведено назву та опис основних методів класу DixonColesLikelihoodTransformer.cs

Таблиця 3.7 – Основні методи класу DixonColesLikelihoodEstimator.cs

Назва	Призначення
_context	Приватне поле, яке зберігає контекст машинного навчання.
_inputColumnNames	Приватне поле, яке зберігає масив назв вхідних колонок.
_outputColumnNames	Приватне поле, яке зберігає масив назв вихідних колонок.
_options	Приватне поле, яке зберігає налаштування для трансформатора DixonColesLikelihoodTransformer.
DixonColesLikelihoodEstimator (MLContext context, string[] inputColumnNames, string[] outputColumnNames, DixonColesLikelihoodTransformer. Options options)	Конструктор класу, який приймає контекст машинного навчання, вхідні та вихідні колонки та налаштування.

Продовження таблиці 3.7

Fit(IDataView input)	Метод, який створює трансформатор DixonColesLikelihoodTransformer на основі вхідних даних.
GetOutputSchema(SchemaShape inputSchema)	Метод, який отримує схему вихідних даних на основі вхідної схеми.

Було створено власний перетворювач та естімейтор для обчислення функції правдоподібності Діксона-Коулза та адаптовано інтеграцію з ML.NET.

Далі необхідно застосувати регресію Пуассона для тренування моделі на основі історичних даних про матчі. Для цього використаємо можливості бібліотеки ML.NET. Викличимо функцію Derivative (float output, float label);

Далі необхідно оцінити ефективність моделі за допомогою набору даних для перевірки та внести необхідні корективи. Тому викличимо метод

TrainModel(IEnumerable<MatchData> historicalMatches)

Використаємо навчену модель для прогнозування результатів майбутніх матчів. Для цього потрібно викликати метод CalculateMatchOutcomeProbability (MatchData match) який повертає результату матчу (перемога, нічия або поразки)

3.4 Висновки

У третьому розділі було обґрунтовано вибір мов програмування та технологій, які будуть використовуватися при розробці програмного продукту та наведено основні їх переваги.

У результаті аналізу було обрано мову програмування C# та технологію ASP.NET Core для розробки веб-сервісу.

Для розробки модуля прогнозування результатів футбольних матчів було обрано бібліотеку ML.NET для виклику регресії Пуасона. Було проведено розробку веб-ресурсу та його модулів.

4 ТЕСТУВАННЯ ПРОГРАМНОГО ДОДАТКУ

4.1 Тестування системи

Для тестування системи веб-ресурсу, яка прогнозує результати футбольних матчів на основі моделей штучного інтелекту необхідно виконати наступні кроки.

Спочатку необхідно поділити зібрані історичні дані на тренувальні, валідні та тестові набори. Тренувальний набір використовується для навчання моделі Dixon-Coles, валідний набір для налаштування параметрів моделі, а тестовий набір для оцінки точності моделі на нових даних (рисунок 4.1).

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	0.5427	0.189	2.878	0.004	0.173	0.912
team[T.Bournemouth]	-0.4886	0.189	-2.580	0.010	-0.860	-0.117
team[T.Brighton]	-0.7769	0.207	-3.745	0.000	-1.183	-0.370
team[T.Burnley]	-0.7349	0.203	-3.612	0.000	-1.134	-0.336
team[T.Chelsea]	-0.1912	0.172	-1.109	0.268	-0.529	0.147
team[T.Crystal Palace]	-0.4949	0.189	-2.614	0.009	-0.866	-0.124
team[T.Everton]	-0.5143	0.191	-2.697	0.007	-0.888	-0.141
team[T.Huddersfield]	-0.9673	0.222	-4.354	0.000	-1.403	-0.532
team[T.Leicester]	-0.2703	0.177	-1.523	0.128	-0.618	0.078
team[T.Liverpool]	0.1134	0.160	0.710	0.478	-0.200	0.426
team[T.Man City]	0.3351	0.152	2.208	0.027	0.038	0.633
team[T.Man United]	-0.1090	0.168	-0.648	0.517	-0.439	0.221
team[T.Newcastle]	-0.6466	0.198	-3.263	0.001	-1.035	-0.258
team[T.Southampton]	-0.6901	0.202	-3.422	0.001	-1.085	-0.295
team[T.Stoke]	-0.7334	0.205	-3.569	0.000	-1.136	-0.331
team[T.Swansea]	-0.9693	0.222	-4.364	0.000	-1.405	-0.534
team[T.Tottenham]	-0.0159	0.165	-0.096	0.923	-0.339	0.307
team[T.Watford]	-0.5080	0.191	-2.664	0.008	-0.882	-0.134
team[T.West Brom]	-0.8674	0.214	-4.049	0.000	-1.287	-0.448

Рисунок 4.1 – Тренувальний набір даних

Далі потрібно навчити модель Dixon-Coles на тренувальному наборі даних, використовуючи метод максимальної правдоподібності для оцінки параметрів

моделі. Необхідно відслідковувати метрики точності (рисунок 4.2), як-от сума квадратів помилок (MSE) або коефіцієнт детермінації (R^2), для оцінки якості прогнозів.

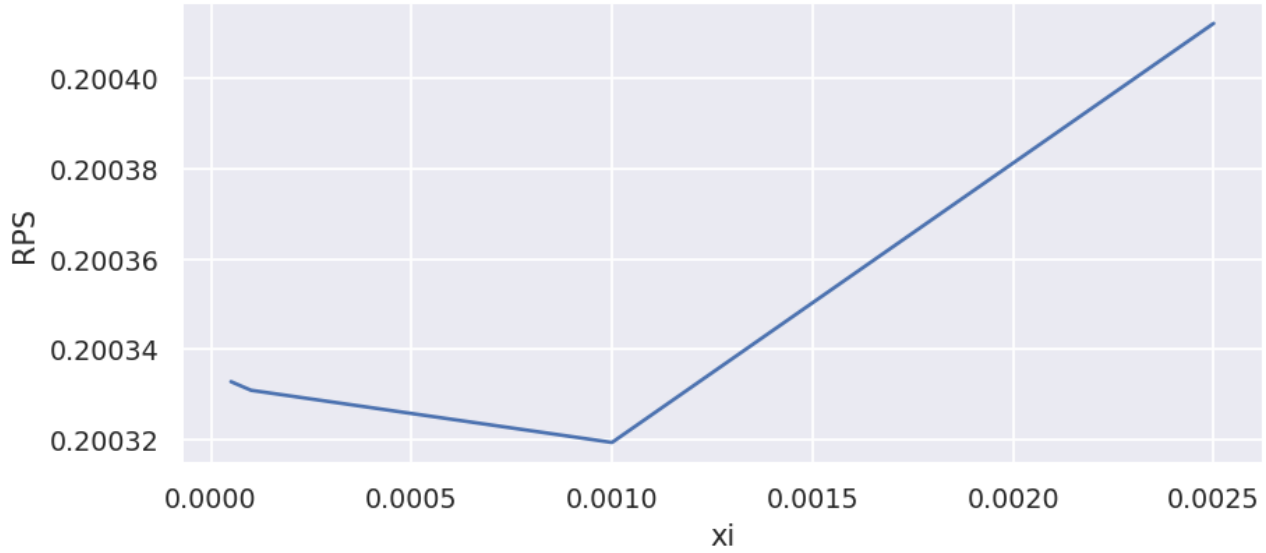


Рисунок 4.2 – Метрики точності

Оцінки рангової ймовірності є мірою похибки в наших прогнозах, тому що менше значення, то ефективніша модель. Схоже, ми найкраще працюємо з xi приблизно 0,001. Менші значення xi, ймовірно, надто сильно впливають на старі дані, тоді як більші значення xi надто сильно впливають на останні дані.

У статті Діксона та Коула автори повідомляють, що використовують значення xi 0,0065. Однак їхні одиниці часу становлять пів тижня, а наші – дні. Якщо ми розділимо 0,0065 на 3,5 (кількість днів у половині тижня), то отримаємо 0,00186, що відповідає нашому значенню 0,001.

Потім необхідно перевірити модель на варіаційному наборі даних, щоб визначити оптимальні параметри моделі. Використовуйте методи, такі як крос-валідація або решітчастий пошук (Grid Search), для систематичного відбору найкращих параметрів.

Наступним кроком буде оцінка фінальної моделі на тестовому наборі даних. Це допоможе вам оцінити, як ваша модель працює на реальних, раніше невідомих даних. Зверніть увагу на метрики точності та порівняйте їх з результатами валідації, щоб виявити можливі проблеми з перенавчанням (overfitting) (рисунок 4.2).

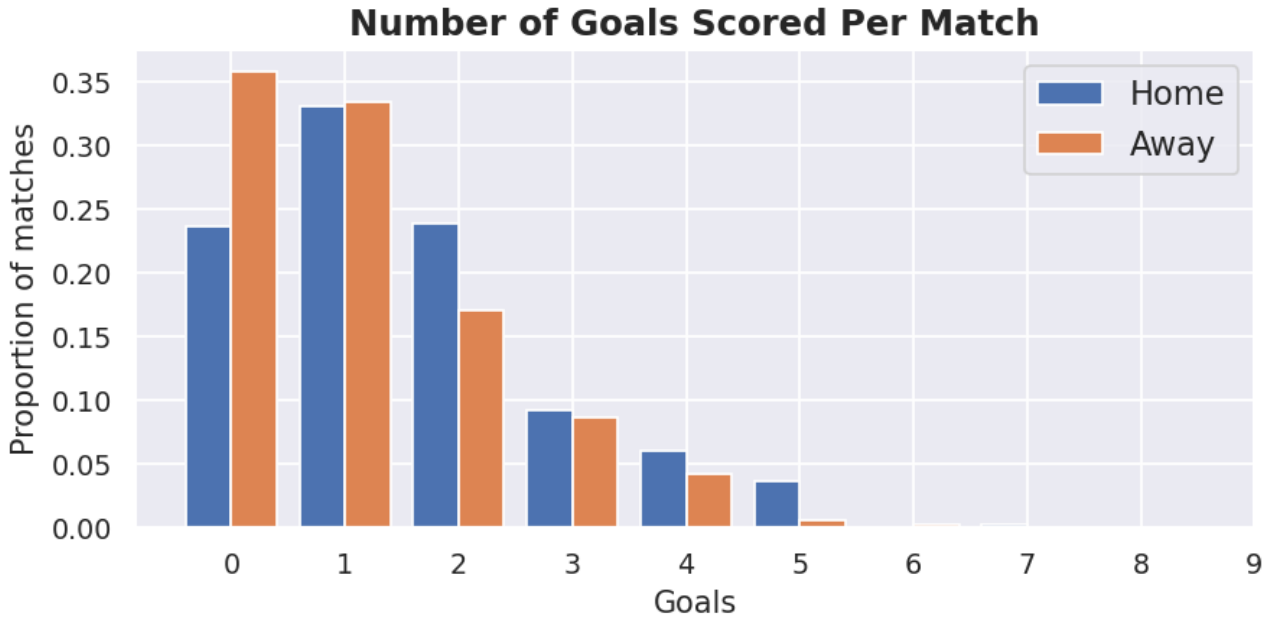


Рисунок 4.3 – Оцінка фінальної моделі

Після того проведемо дослідження випадків, коли модель Dixon-Coles робить некоректні прогнози. Для цього виявимо спільні характеристики цих випадків або причини помилок, щоб зрозуміти, чи модель не враховує важливі фактори, або деякі дані містять шум.

Згідно з результатами аналізу помилок, необхідно застосувати зміни до моделі Dixon-Coles. Це може включати зміни в прогнозуванні або налаштуванні параметрів, а також в додаванні додаткових змінних або параметрів до моделі для покращення точності прогнозів.

Проведемо повторне тестування на тестовому наборі даних після внесення змін до моделі. Переконаємось, що модель працює належним чином і точність прогнозів покращилася.

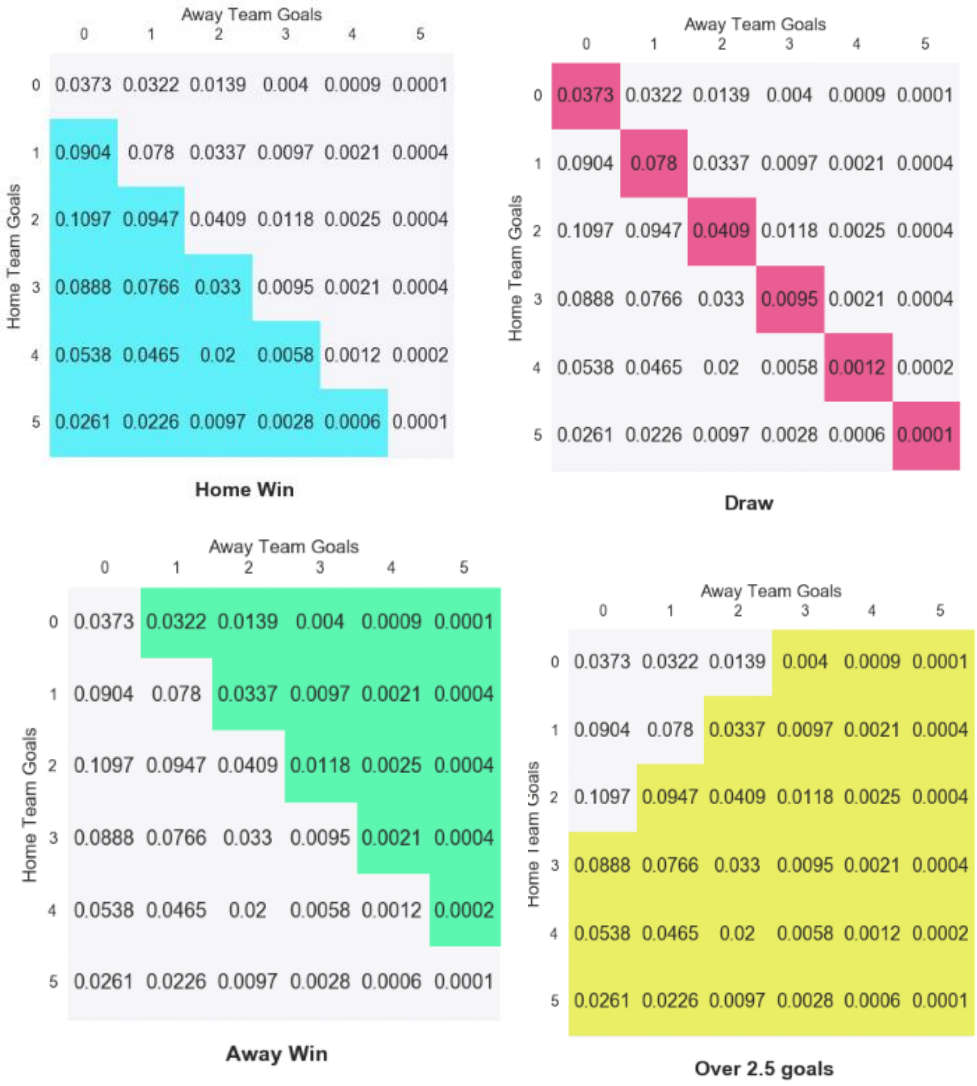


Рисунок 4.4 – Отримані результати прогнозування

Перейдемо до дослідження стабільності прогнозів моделі Dixon-Coles за різних обставин, вивчаючи роботу моделі при різних комбінаціях команд, сценаріях гри та інших факторах, що можуть вплинути на результати матчів.

Запустимо систему в режимі онлайн для користувачів, щоб вони могли отримувати прогнози результатів футбольних матчів (рисунок 4.5).

Match	Home	Draw	Away
Arsenal v Everton	71.4 %	18 %	12 %
Burnley v West Ham	42 %	28 %	31 %
Chelsea v Sunderland	88.5 %	9 %	3 %
Hull v Tottenham	11 %	17 %	71.9 %
Leicester v Bournemouth	53.5 %	24 %	23 %
Liverpool v Middlesbrough	87.7 %	9 %	4 %
Man Utd v C Palace	42 %	29 %	30 %
Southampton v Stoke	57.1 %	24 %	19 %
Swansea v West Brom	43 %	29 %	29 %
Watford v Man City	5 %	10 %	85.5 %

Рисунок 4.5 – Отримані результати прогнозування в зручній для користувача формі

Проаналізувавши ймовірність перемоги користувач може використати дану інформацію в своїх цілях, таких як розуміння перебігу матчу або зробити ставку на певну команду.

Також необхідно постійно слідкувати за роботою системи, аналізувати точність прогнозів на нових даних та проводити необхідні корективи для забезпечення найвищої якості прогнозів після запуску системи.

Для тестування та впровадження системи прогнозування результатів футбольних матчів на основі моделі Dixon-Coles, було виконано такі основні кроки: розділення даних, навчання та валідація моделі, тестування моделі, аналіз помилок, удосконалення моделі та повторне тестування. Після успішного проходження цих етапів, система була протестовна в режимі онлайн для користувачів.

4.2 Розробка інструкції користувача

Для того щоб взаємодіяти з системою прогнозування результатів футбольних матчів, необхідно описати можливості, та відповідні кроки, які необхідно виконати користувачу.

Відкриємо головну сторінку (рисунок 4.6).

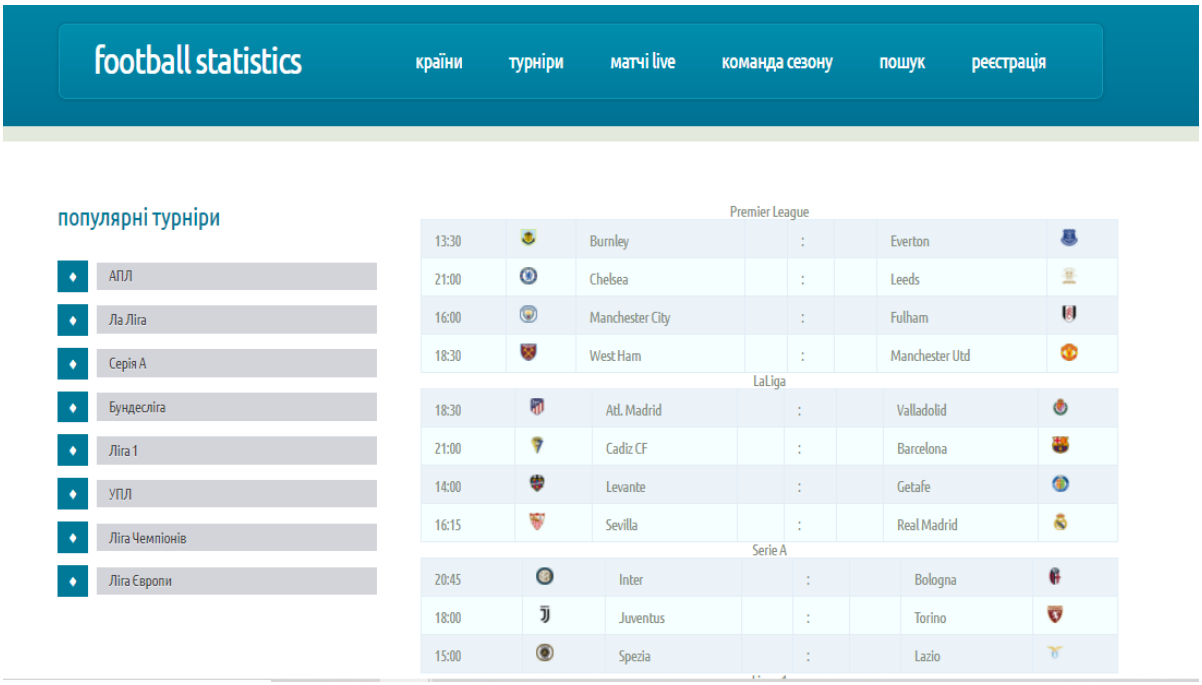


Рисунок 4.6 – Головна сторінка веб-ресурсу

Для користувача відкривається широкий спектр можливостей, які допоможуть йому в аналізі та прогнозуванні результатів футбольних матчів.

Користувачу надається можливість отримати найсвіжіші та точні прогнози результатів матчів на основі передової моделі Dixon-Coles, ознайомитись з детальними статистичними даними про команди та гравців, що допоможуть йому глибше зрозуміти тенденції та патерни футболу.

Створення особистого аккаунта та персональні налаштування, щоб отримувати індивідуальні рекомендації та відслідковувати улюблені команди.

Розглянемо головне меню, яке дозволяє знайти легкий доступ до основних розділів та можливостей (рисунок 4.7).

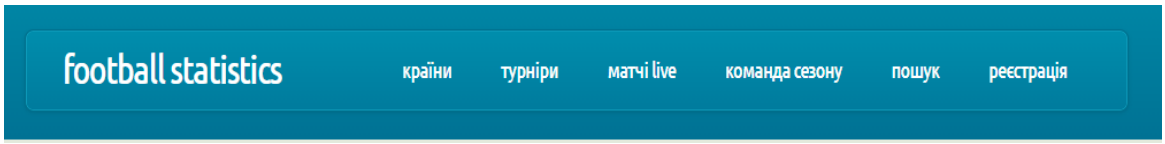


Рисунок 4.7 – Головне меню веб-ресурсу

Виділимо наступні розділи, такі як турніри та матчі live. Дані розділи відповідають за статистичні дані про команди, гравців та матчі.

За допомогою пошуку користувач зможе знайти будь яку команду або турнір.

Також надається можливість для реєстрації в системі, для перегляду унікального контенту.

Зі сторони розташована панель швидкого доступу, яка включає розділ "Популярні змагання". У переліку змагань представлені п'ять головних європейських ліг, європейські кубкові змагання, а також українська Прем'єр-лігу.

Рисунок 4.8 – Панель популярних змагань

Перейшовши в розділ країни, користувачу надається можливість переглянути список країн (рисунок 4.9).

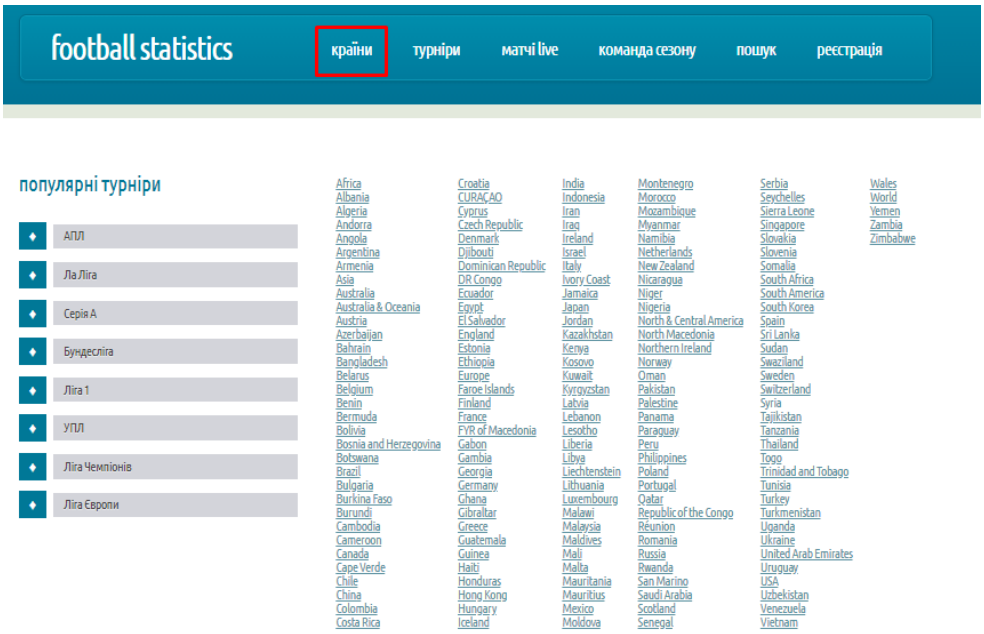


Рисунок 4.9 – Список країн

Натиснувши на певну країну переглянути список футбольних турнірів (рисунок 4.10).

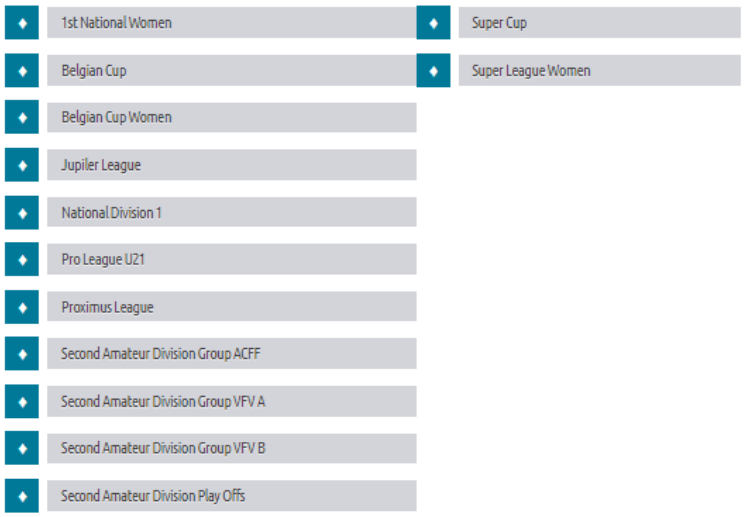


Рисунок 4.10 – Список турнірів певної країни

Наступним розглянемо розділ турніри. На рисунку 4.11 наведено турнірну таблицю Бельгійської прем’єр ліги.

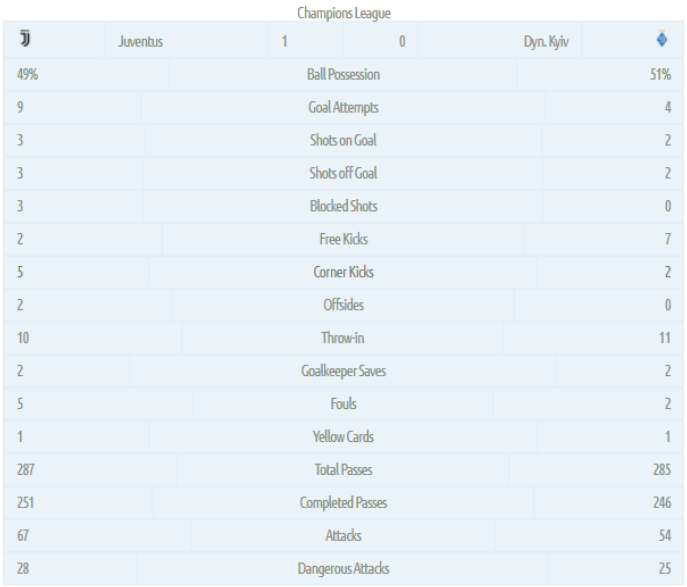
Таблиця Календар Команди Результати

Позиція в таблиці	Назва	Кількість зіграних матчів	Кількість виграних матчів	Кількість нічиїх	Кількість програних матчів	Кількість очків
1	Beerschot VA	14	9	1	4	28
2	Genk	14	8	4	2	28
3	Club Brugge KV	14	8	3	3	27
4	Antwerp	14	7	4	3	25
5	St. Liege	14	6	6	2	24
6	Charleroi	14	7	2	5	23
7	Anderlecht	14	5	7	2	22
8	Kortrijk	14	5	4	5	19
9	Leuven	13	5	4	4	19
10	Cercle Brugge KSV	14	6	0	8	18
11	Gent	14	5	1	8	16
12	Oostende	14	4	4	6	16
13	Eupen	13	3	7	3	16
14	Waregem	14	4	2	8	14
15	KV Mechelen	13	3	3	7	12
16	Waasland-Beveren	14	3	3	8	12
17	St. Truiden	14	2	5	7	11

Рисунок – 4.11 – Турнірна таблиця певного турніру

Для того щоб переглянути статистику матчу необхідно його відкрити. На рисунку 4.11 зображено статистику матчу. В статистиці наведені такі позиції, як кількість голів, відсоток володіння м’ячом кожною із команд, гольові моменти,

удари, штрафні, жовті карточки, відсоток успішності пасів, кількість небезпечних атак і тд.

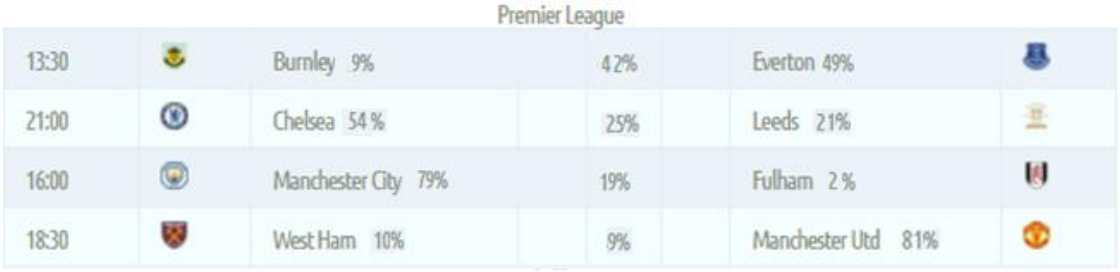


Champions League

Juventus		Dyn. Kyiv
1		0
49%	Ball Possession	51%
9	Goal Attempts	4
3	Shots on Goal	2
3	Shots off Goal	2
3	Blocked Shots	0
2	Free Kicks	7
5	Corner Kicks	2
2	Offsides	0
10	Throw-in	11
2	Goalkeeper Saves	2
5	Fouls	2
1	Yellow Cards	1
287	Total Passes	285
251	Completed Passes	246
67	Attacks	54
28	Dangerous Attacks	25

Рисунок 4.13 – Футбольна статистика матчу

На рисунку 4.13 наведено список матчів різних турнірів в певний день.



Premier League

13:30	Burnley 9%	42%	Everton 49%	
21:00	Chelsea 54%	25%	Leeds 21%	
16:00	Manchester City 79%	19%	Fulham 2%	
18:30	West Ham 10%	9%	Manchester Utd 81%	

Рисунок 4.13 – Список матчів

Для кожного матчу користувач бачить відсоткову перемогу кожної з команд. В даному випадку ймовірність того, що в матчі Вест Хем – Манчестер Юнайтед, перемогу здобудить команда господар 10%, нічия – 9 %, перемога гостей – 81%.

4.3 Висновки

У четвертому розділі було проведено тестування прогнозування результату матчів з використання моделі штучного інтелекту, було проведення навчання та валідація даних, також доведено його повну працездатність та відповідність поставленому технічному завданню.

Розроблено інструкцію користувача по використанню програмного продукту.

5 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота за темою «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

5.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Розробка методів і програмних засобів прогнозування результатів

футбольних матчів на основі моделей штучного інтелекту» є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання комерційного потенціалу розробки будемо здійснювати за дванадцятьма критеріями, рекомендованими Державним комітетом України з питань науки, інновацій та інформатики (2010 р.), які наведені в таблиці 5.1.

Для визначення загального рівню комерційного потенціалу розробки, скористаємося даними таблиці 5.3, в якій наведено рекомендації щодо можливих рівнів комерційного потенціалу будь-якої розробки.

Таблиця 5.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					

Продовження таблиці 5.1

6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 5.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	5	4	4
2. Ринкові переваги (наявність аналогів)	4	4	4
3. Ринкові переваги (ціна продукту)	4	3	3
4. Ринкові переваги (технічні властивості)	4	4	3
5. Ринкові переваги (експлуатаційні витрати)	3	3	3
6. Ринкові перспективи (розмір ринку)	3	4	3
7. Ринкові перспективи (конкуренція)	3	3	4
8. Практична здійсненність (наявність фахівців)	3	3	3
9. Практична здійсненність (наявність фінансів)	3	3	4
10. Практична здійсненність (необхідність нових матеріалів)	3	3	3
11. Практична здійсненність (термін реалізації)	3	3	4
12. Практична здійсненність (розробка документів)	3	3	3
Сума балів	41	40	41
Середньоарифметична сума балів $СБ_c$	40,7		

За результатами розрахунків, наведених в таблиці 5.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 5.3 [34].

Таблиця 5.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту» становить 40,7 балів, що, відповідно до таблиці 5.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

5.2 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

5.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [34]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (5.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дні;

$T_р$ – середнє число робочих днів в місяці, $T_р$=22 дні.

$$З_о = 30000{,}00 \cdot 60 / 22 = 81818{,}18 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	30000	1363,64	60	81818,18
Інженер-розробник програмного забезпечення	28000	1272,73	60	76363,64
Консультант	22000	1000,00	30	30000,00
Всього				188181,82

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \quad (5.2)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_р \cdot t_{зм}} , \quad (5.3)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=6700,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [34];

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$Т_р$ – середнє число робочих днів в місяці, приблизно $Т_р$ = 22 дні;

$t_{зм}$ – тривалість зміни, год.

$С_1 = 6700,00 \cdot 1,10 \cdot 1,65 / (22 \cdot 8) = 69,09$ грн.
$З_{р1} = 69,09 \cdot 8,00 = 552,75$ грн.

Таблиця 5.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Підготовка робочого місця дослідника	8	2	1,1	69,09	552,75
Інсталяція програмного забезпечення	3	5	1,7	106,78	320,34
Всього					873,09

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод} = (188181,82 + 873,09) \cdot 11 / 100\% = 20796,04$ грн.

5.2.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

$З_н$ = (188181,82 + 873,09+ 20796,04) · 22 / 100% = 46167,21 грн.

5.2.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень.

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} H_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} B_j \cdot Ц_{вj} , \qquad (5.6)$$

де H_j – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

B_j – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$М_1$ = 2 · 210,00 · 1,1 - 0,000 · 0,00 = 462,0 грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за од	Норма витрат, од	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Офісний папір А4-500-80	210	2	0	0	462
Папір для записів А5	110	1	0	0	121
Органайзер офісний	210	1	0	0	231
Канцелярське приладдя (набір офісного працівника)	175	3	0	0	577,5
Картридж для принтера	1100	1	0	0	1210
Flesh-пам'ять 32 GB	180	1	0	0	198
Тека для паперів	82	1	0	0	90,2
					2889,7

5.2.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту» відсутні.

5.2.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \quad , \qquad (5.7)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань устаткування.

$B_{спец} = 10000 \cdot 3 \cdot 1,1 = 33000$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.7 – Витрати на придбання устаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Монітор 25" Samsung Odyssey G4 S25BG40 (LS25BG400EIXCI) IPS Full HD	3	10000	33000
Ноутбук HP Pavilion 15-eg3022ua (826L7EA)	3	40000	132000
Маршрутизатор	1	2000	2200
Всього			167200

5.2.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (5.8)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, (K_i = *1,10…1,12);*

k – кількість найменувань програмних засобів.

$B_{прг} = 11600{,}00 \cdot 3 \cdot 1{,}1 = 38976$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.8 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
ОС Windows 11	3	11600	38976
Прикладний пакет Microsoft Office 2019	3	5500	18480
Всього			57456

5.2.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12} , \qquad (5.9)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (132000{,}00 \cdot 3) / (5 \cdot 12) = 6600$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.9 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук HP Pavilion 15-eg3022ua (826L7EA) 3 шт	132 000	5	3	6600,00
Монітор 25" Samsung Odyssey G4 S25BG40 (LS25BG400EIXCI) IPS Full HD	33 000	5	3	1650,00
Робоче місце дослідника	15000	5	3	750,00
Оргтехніка	6000	4	3	375,00
ОС Windows 11 (3 пак)	38976	3	3	3248,00
Прикладний пакет Microsoft Office 2019 (3 пак)	18480	3	3	1540,00
Всього				14163,00

5.2.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію ($B_е$) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (5.10)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e = 7,50$ грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,065 \cdot 480,0 \cdot 7,50 \cdot 0,95 / 0,97 = 687,52$ грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.10 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук HP Pavilion 15-eg3022ua (826L7EA) 3 шт	0,065	480	687,52
Робоче місце дослідника	0,15	480	528,87
Оргтехніка	0,45	20	66,11
Всього			1282,50

5.2.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (5.11)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$B_{св}$ = (188181,82 + 873,09) · 20 / 100% = 37810,98 грн.

5.2.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (5.12)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 35%.

$B_{сп}$ = (188181,82 + 873,09) · 35 / 100% = 66169,22 грн.

5.2.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$І_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (5.13)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$І_в$ = (188181,82 + 873,09) · 50 / 100% = 94527,46 грн.

5.2.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків;

витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.14)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 120%.

$В_{нзв}$ = (188181,82 + 873,09) · 120 / 100% = 226 865,89 грн.

Витрати на проведення науково-дослідної роботи розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (5.15)$$

$В_{заг}$= 924382,91 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (5.16)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,9.

ЗВ = 924382,91 / 0,9 = 1027092,13 грн.

5.3 Розрахунок економічної ефективності науково-технічної розробки

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN – збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

1-й рік – 5000 користувачів/рік;

2-й рік – 6000 користувачів/рік;

3-й рік – 3500 користувачів/рік.

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 20000 користувачів;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 1000,00 грн /за рік користування;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 500,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 3-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [34]:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (5.17)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Приймемо ρ =30%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (20000{,}00 \cdot 500{,}00 + 1200 \cdot 5000) \cdot 0{,}83 \cdot 0{,}3 \cdot (1 - 0{,}18/100\%) = 6454917$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (20000{,}00 \cdot 500{,}00 + 1200 \cdot (5000 + 6000)) \cdot 0{,}83 \cdot 0{,}3 \cdot (1 - 0{,}18/100\%) = 10963113$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (20000{,}00 \cdot 500{,}00 + 1200 \cdot (5000 + 6000 + 3500)) \cdot 0{,}83 \cdot 0{,}3 \cdot (1 - 0{,}18/100\%) =$ 14006145,3 грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.18)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, $\tau = 0{,}25$;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$ПП = 6454917 / (1+0{,}25)^1 + 10963113 / (1+0{,}25)^2 + 14006145{,}3 / (1+0{,}25)^3 =$ 19351472,31 грн.

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \tag{5.19}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 1027092,13грн.

$PV = k_{інв} \cdot ЗВ$ = 2· 1027092,13= 2054184,252грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \tag{5.20}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 19351472,31 грн;

PV – теперішня вартість початкових інвестицій, 2054184,252 грн.

$E_{абс} = ПП - PV$ = 19351472,31 - 2054184,252= 17297288,06 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{5.21}$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 17297288,06 грн;

PV – теперішня вартість початкових інвестицій, 2054184,252грн;

$T_{ж}$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$E_{в} = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+17297288{,}06/2054184{,}252)^{1/3} = 1{,}11.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \tag{5.22}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,11;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,11+0,25 = 0,36 < 1,11 свідчить про те, що внутрішня економічна дохідність інвестицій $E_{в}$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_{в}}, \tag{5.23}$$

де $E_{в}$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 1,11 = 0,9 року.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

За нашими розрахунками, результати нашої розробки можуть бути впроваджені з 1 січня 2024 року, а її результати будуть виявлятися протягом 2024-го , 2025-го, 2026-го років.

Прогноз зростання попиту на нашу розробку складає по роках:

1-й рік після впровадження (2024р.) – приблизно 350 шт.;

2-й рік після впровадження (2024 р.) – приблизно 250 шт.;

3-й рік після впровадження (2026р.) – приблизно 150 шт.;

У 2026 р. не планується отримання прибутків для потенційних інвесторів, оскільки високою є ймовірність, що з'являться нові, більш якісні та ефективні розробки.

Розрахуємо очікуване збільшення прибутку $\Delta П_i$, що його може отримати потенційний інвестор від впровадження результатів нашої розробки, для кожного із років, починаючи з першого року впровадження:

$$\Delta П_i = \sum_{1}^{n} (\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\upsilon}{100}\right), \quad (5.12)$$

де $\Delta Ц_o$ – покращення основного оціночного показника від впровадження результатів розробки у даному році. Зазвичай таким показником є збільшення ціни нової розробки, грн.; у нашому випадку $\Delta Ц_o = 5$ тис. грн.;

N – основний кількісний показник, який визначає обсяг діяльності у даному році до впровадження результатів наукової розробки; було встановлено, що $N = 220$ шт;

ΔN – покращення основного кількісного показника від впровадження результатів розробки;

$Ц_о$ – основний оціночний показник, який визначає обсяг діяльності у даному році після впровадження результатів розробки, грн.; $Ц_о = 23$ тис. грн.;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки;

λ – коефіцієнт, який враховує сплату податку на додану вартість. У 2020 році ставка податку на додану вартість встановлена на рівні 20%, а коефіцієнт $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати ρ= 0,2...0,3; візьмемо ρ= 0,25;

υ – ставка податку на прибуток. У 2020 році υ = 18%.

Тоді, збільшення чистого прибутку для потенційного інвестора $\Delta П_1$ протягом першого року від реалізації нашої розробки (2021 р.) складатиме:

$$\Delta П_1 = [4000 \cdot 220 + 23000 \cdot 350] \cdot 0{,}8333 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 2\,248\,500 \text{ грн.}$$

Протягом другого (2022 р.) року складатиме:

$$\Delta П_2 = [4000 \cdot 220 + 23000 \cdot (350 + 250)] \cdot 0{,}8333 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 3\,226000 \text{ грн.}$$

Протягом третього (2023 р.) року складатиме:

$$\Delta П_3 = [4000 \cdot 220 + 23000 \cdot (350 + 250 + 150\,] \cdot 0{,}8333 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 3812500 \text{ грн.}$$

5.5 Висновки

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту» становить 40,7 балів, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

Також термін окупності становить 0,9 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту».

ВИСНОВКИ

У під час виконання магістерської кваліфікаційної роботи було розроблено методи та засоби веб-сервісу для прогнозування футбольних матчів на основі моделей штучного інтелекту. Для розробки було використано середовище програмування Visual Studio 2019 та Visual Studio Code. Робота оформлена згідно методичних вказівок [36].

Було проведено аналіз сучасного стану прогнозування результатів футбольних матчів з використанням моделей штучного інтелекту. Відзначено, що зростання популярності спортивних ставок та аналітики спонукало до пошуку ефективних методів прогнозування, а моделі штучного інтелекту відіграють важливу роль у досягненні цієї мети.

Проаналізовано існуючі аналоги програмних додатків для прогнозування результатів футбольних матчів та зроблено висновок про доцільність розробки власного програмного рішення з покращеними характеристиками. У процесі розробки вибрано мову програмування C# та технологію ASP.NET Core для створення веб-сервісу, а також бібліотеку ML.NET для реалізації статистичного моделювання на основі аналізу часових рядів.

Основною моделлю, вибраною для прогнозування результатів матчів, стала удосконалена модель Діксона-Коулза, яка враховує індивідуальну статистику кожного гравця на відповідній позиції для підвищення точності прогнозу. Розроблено архітектуру та структуру веб-сервісу, включаючи загальні алгоритми роботи додатку, отримання, обробки та представлення статистичної інформації про турніри, команди, матчі, події в матчі, гравців та їх статистику.

Результатом дослідження стала розробка веб-ресурсу, який надає користувачам можливість отримувати статистичну інформацію про матчі та переглядати прогнози на них, забезпечуючи комбінований підхід до аналізу результатів футбольних матчів. Програмний продукт було ретельно

протестовано, проведено навчання та валідацію даних, а також доведено його повну працездатність та відповідність поставленому технічному завданню.

З огляду на проведені дослідження, комерційний потенціал розробки виявився вище середнього, що свідчить про важливість даної розробки для ринку та можливість привабити інвесторів для впровадження та виведення продукту на ринок. Таким чином, виконання магістерської кваліфікаційної роботи за темою "Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту" виявилося доцільним, сприяючи розвитку технологій прогнозування в спорті та покращенню якості аналітики для користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hucaljuk, J., & Rakipović, A. (2011). Predicting football scores using machine learning techniques. Journal of Universal Computer Science, 17(16), 2359-2379. [Електронний ресурс]. URL: http://www.jucs.org/jucs_17_16/ predicting_football_scores_using/ jucs_17_16_2359_2379_hucaljuk.pdf
2. Bunker, R., & Thabtah, F. (2019). A machine learning framework for sport result prediction. Applied Computing and Informatics, 15(1), 27-33. [Електронний ресурс]. URL: https://doi.org/10.1016/j.aci.2017.12.001
3. Karlik, B., & Tokat, S. (2011). An approach to soccer match result estimation using artificial neural networks. International Journal of Computer Science & Information Technology, 3(4), 1-11. [Електронний ресурс]. URL: http://airccse.org/journal/jcsit/0811csit01.pdf
4. Dixon, M.J., & Coles, S.G. (1997). Modelling Association Football Scores and Inefficiencies in the Football Betting Market. Journal of the Royal Statistical Society: Series C (Applied Statistics), 46(2), 265-280. [Електронний ресурс]. Режим доступу до ресурсу: https://www.jstor.org/stable/2988418
5. Ross, S.M. (2010). Introduction to Probability Models, 10th Edition. Academic Press. ISBN-13: 978-0123756862.
6. Катєльніков Д. І., Перебейнос Р. Л. Використання моделей штучного інтелекту для прогнозування результатів футбольних матчів/ Збірник матеріалів Міжнародної науково-практичної Інтернет конференції 20-21 листопада 2023 р. – Суми/Вінниця: НІКО/ КЗВО «Вінницька академія безперервної освіти», 2023. – 336 с.
7. Комп'ютерні ігри та мультимедіа як інноваційний підхід до комунікації - 2023 / Матеріали III Всеукраїнської науково-технічної конференції молодих вчених, аспірантів і студентів, Одеса, 28-29 жовтня 2023 р. - Одеса, Видавництво ОНТУ, 2023 р. – 270 с.

8. Bunker, R.P., & Thabtah, F. (2019). A Machine Learning Framework for Sport Result Prediction. Applied Computing and Informatics, 15(1), 27-33.
9. Lago-Peñas, C., & Lago-Ballesteros, J. (2011). Game Location and Team Quality Effects on Performance Profiles in Professional Soccer. Journal of Sports Science and Medicine, 10(3), 465-471. [Електронний ресурс]. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3737879/
10. World Football Elo Ratings - [Електронний ресурс]. URL: https://www.eloratings.net/
11. Scisports - [Електронний ресурс]. URL: https://www.scisports.com/
12. Kickoff.ai - [Електронний ресурс]. URL: https://kickoff.ai/
13. Wyscout - [Електронний ресурс]. URL: https://wyscout.com/
14. xG Philosophy - [Електронний ресурс]. URL: https://xgphilosophy.com/
15. Goddard, J. (2005). Regression Models for Forecasting Goals and Match Results in Association Football. International Journal of Forecasting, 21(2), 331-340.
16. Bunker, R.P., & Thabtah, F. (2019). A Machine Learning Framework for Sport Result Prediction. Applied Computing and Informatics, 15(1), 27-33.
17. Tsakonas, A., & Dounias, G. (2004). A Fuzzy Cognitive Map Approach to Differential Diagnosis of Specific Language Impairment. Artificial Intelligence in Medicine, 31(1), 57-79.
18. McHale, I.G., & Scarf, P.A. (2007). Modelling soccer matches using bivariate discrete distributions with general dependence structure. Statistica Neerlandica, 61(4), 432-445.
19. Constantinou, A.C., Fenton, N., & Neil, M. (2012). Profiting from an Inefficient Association Football Gambling Market: Prediction, Risk and Uncertainty using Bayesian Networks. Knowledge-Based Systems, 50, 60-86.
20. Chou, J.S. (2018). Predicting Results of Football Matches Using Deep Learning. International Journal of Computer Science in Sport, 17(1), 54-69.
21. Brin, S., & Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems, 30(1-7), 107-117.

[Електронний ресурс]. URL: https://www.sciencedirect.com/science/article/pii/S016975529800110X

22. Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. Nature, 521(7553), 436-444. CNN and RNN [Електронний ресурс]. URL: https://www.nature.com/articles/nature14539
23. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. [Електронний ресурс].URL: https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html
24. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). (Recurrent Neural Networks for sequence data [Електронний ресурс]. URL: https://aclanthology.org/D14-1179/
25. Tilkov, S., & Vinoski, S. (2010). Node: An exploration of the REST architectural style. IEEE Internet Computing, 14(1), 22-26. [Електронний ресурс]. URL: https://ieeexplore.ieee.org/document/5353363
26. Sedgewick, R., & Wayne, K. (2011). Algorithms (4th ed.). Addison-Wesley Professional. (Algorithm development for system scalability and optimization) [Електронний ресурс]. URL: https://www.cs.princeton.edu/~rs/AlgsDS07/Algorithms_4th.pdf
27. Документація APIfootball [Електронний ресурс]. URL: https://apifootball.com/documentation/
28. Ramaswamy, C., & Sandhu, R. (2016). Towards a multilevel secure cloud: An authorization perspective. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16) (pp. 378-389). ACM. (Contemporary authorization techniques in the context of cloud computing) [Електронний ресурс]. URL: https://dl.acm.org/doi/10.1145/2976749.2978305

29. Redmiles, E. M., Kross, S., & Mazurek, M. L. (2019). How well do my results generalize? The security external validity in empirical computer science. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP '19) (pp. 285-302). IEEE. (Exploring user registration and security practices in modern web applications [Електронний ресурс]. URL: https://ieeexplore.ieee.org/document/8835245
30. Troelsen, A., & Japikse, P. (2017). Pro C# 7: With .NET and .NET Core. Apress. (An in-depth exploration of C# 7 programming with .NET and .NET Core) [ISBN: 978-1-4842-3017-6]. [Електронний ресурс]. URL: https://www.apress.com/gp/book/9781484230176
31. Freeman, A. (2017). Pro ASP.NET Core MVC 2. Apress. (An in-depth guide to building web applications using ASP.NET Core MVC 2) [Електронний ресурс]. URL: https://www.apress.com/gp/book/9781484231494
32. Cesar, L., & Sousa, R. (2019). Machine Learning with Microsoft Technologies: Selecting the Right Architecture and Tools for Your Project. Apress. (An overview of ML.NET and other Microsoft machine learning technologies) [Електронний ресурс]. URL: https://www.apress.com/gp/book/9781484244296
33. Microsoft Visual Studio 2019 [Електронний ресурс]. URL: https://learn.microsoft.com/en-us/visualstudio/releases/2019/release-notes
34. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.
35. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця : ВНТУ, 2016. 113 с.
36. Положення про кваліфікаційну роботу на другому (вищому) рівні вищої освіти. Уклад / А.О. Семенов - Вінниця : ВНТУ, 2021. 60 с. СУЯ ВНТУ-03.02.02-П.001.01:21

ДОДАТКИ

Додаток А – Технічне завдання

112

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
д.т.н., проф. О. Н. Романюк
"19" вересня 2023 р.

Технічне завдання
на магістерську кваліфікаційну роботу «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту»
за спеціальністю
121 – Інженерія програмного забезпечення

Керівник магістерської кваліфікаційної роботи:
_____ к.т.н., доц. Д.І. Катєльніков
"19" 09 2023 р.

Виконав:
_____ студент гр. 1ПІ-22м Р.Л. Перебейнос
"19" 09 2023 р.

Вінниця – 2023 року

1. Найменування та галузь застосування

Магістерська кваліфікаційна робота: «Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту».

Галузь застосування – автоматизовані системи обробки даних.

2. Підстава для розробки.

Підставою для виконання магістерської кваліфікаційної роботи (МКР) є індивідуальне завдання на МКР та наказ ректора №247 від 18 вересня 2023 р. по ВНТУ про закріплення тем МКР.

3. Мета та призначення розробки.

Метою роботи є підвищення точності прогнозування результатів футбольних матчів за рахунок використання удосконаленої моделі штучного інтелекту Dixon-Coles та продуктивності обробки набору статистичних даних.

Призначення роботи – розробка методів та засобів прогнозування результатів матчів на основі футбольної статистики.

4. Вихідні дані для проведення НДР

1. Hucaljuk, J., & Rakipović, A. (2011). Predicting football scores using machine learning techniques. Journal of Universal Computer Science, 17(16), 2359-2379. [Електронний ресурс]. URL: http://www.jucs.org/jucs_17_16/ predicting_football_scores_using/ jucs_17_16_2359_2379_hucaljuk.pdf
2. Dixon, M.J., & Coles, S.G. (1997). Modelling Association Football Scores and Inefficiencies in the Football Betting Market. Journal of the Royal Statistical Society: Series C (Applied Statistics), 46(2), 265-280. [Електронний ресурс]. URL: https://www.jstor.org/stable/2988418
3. Ross, S.M. (2010). Introduction to Probability Models, 10th Edition. Academic Press. ISBN-13: 978-0123756862.

4. Constantinou, A.C., Fenton, N., & Neil, M. (2012). Profiting from an Inefficient Association Football Gambling Market: Prediction, Risk and Uncertainty using Bayesian Networks. Knowledge-Based Systems, 50, 60-86.

5. Технічні вимоги

Вхідні дані – футбольна статистика розподілена на сутності та отримувана в форматі JSON;

вихідні дані – результат прогнозування футбольних матчів у вигляді списку ймовірностей перемоги, нічиї та поразки.

6. Конструктивні вимоги.

Конструкція пристрою повинна відповідати естетичним та ергономічним вимогам, повинна бути зручною в обслуговуванні та керуванні.

Графічна та текстова документація повинна відповідати діючим стандартам України.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

– пояснювальна записка до МКР;

– технічне завдання;

– лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії і етапи розробки

№ з/п	Назва етапів магістерської кваліфікаційної Роботи	Строк виконання етапів роботи
1	Аналіз моделей нейронних мереж та вибір найбільш доцільної для поставленої задачі	15.09.2023 –30.09.2023
2	Розробка алгоритмів системи	01.10.2023 – 10.10.2023
3	Розробка методу отримання вхідних даних	11.10.2023 – 25.10.2023
4	Розробка методу прогнозування результатів на основі статистики	26.10.2021 – 15.11.2021
5	Економічна частина	22.11.2023 – 01.12.2023

10. Порядок контролю та прийняття.

Виконання етапів магістерської кваліфікаційної роботи контролюється керівником згідно з графіком виконання роботи. Прийняття магістерської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком

116

Додаток Б – Протокол перевірки на плагіат

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: **Розробка методів і програмних засобів прогнозування результатів футбольних матчів на основі моделей штучного інтелекту**

Тип роботи: кваліфікаційна робота

Підрозділ : кафедра програмного забезпечення, ФІТКІ, 1ПІ – 22м

Науковий керівник: к.т.н. доц. Катєльніков Д.І.

Unicheck	
Оригінальність	93.4%
Схожість	6.6%

Аналіз звіту подібності

■ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Черноволик Г. О.

Опис прийнятого рішення: **допустити до захисту**

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Автор роботи _____ Перебейнос Р. Л.

Керівник роботи _____ Катєльніков Д.І.

Додаток В – Лістинг програми

```
using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Linq;
using MathNet.Numerics;
using MathNet.Numerics.LinearAlgebra;
using MathNet.Numerics.Optimization;

public class FootballMatchData
{
   public string HomeTeam;
   public string AwayTeam;
   public int HomeGoals;
   public int AwayGoals;
   public double HomeAvgGoals;
   public double AwayAvgGoals;
   public int DaysSinceLastMatch;
}

public static class DixonColesModel
{
   public static double[] TrainModel(List<FootballMatchData> data, double timeDecay = 365.0)
   {
      var teams = data.Select(x => x.HomeTeam).Union(data.Select(x =>
x.AwayTeam)).Distinct().ToArray();

      var initialParams = new double[teams.Length * 4 + 1];
      for (int i = 0; i < teams.Length * 4; i++)
      {
         initialParams[i] = 0.0;
      }
      initialParams[^1] = 0.1;

      var objective = ObjectiveFunction.Value((params) => DixonColesLikelihood(params, data, teams,
timeDecay));
      var solver = new BfgsMinimizer();
      var result = solver.FindMinimum(objective, initialParams);

      return result.MinimizingPoint;
   }

   private static double DixonColesLikelihood(double[] parameters, List<FootballMatchData> data,
string[] teams, double timeDecay)
   {
      var numTeams = teams.Length;
      var alphaHome = Vector<double>.Build.Dense(numTeams, i => Math.Exp(parameters[i]));
      var alphaAway = Vector<double>.Build.Dense(numTeams, i => Math.Exp(parameters[i +
numTeams]));
      var betaHome = Vector<double>.Build.Dense(numTeams, i => Math.Exp(parameters[i +
numTeams * 2]));
```

```
            var betaAway = Vector<double>.Build.Dense(numTeams, i => Math.Exp(parameters[i +
numTeams * 3]));
            var gamma = parameters[^1];

            var lambdaHome = alphaHome.ToColumnMatrix() * betaAway.ToRowMatrix();
            var lambdaAway = alphaAway.ToColumnMatrix() * betaHome.ToRowMatrix();
            lambdaHome =
lambdaHome.PointwiseMultiply(Matrix<double>.Build.DenseOfRowMajor(numTeams, numTeams,
data.Select(x => x.HomeAvgGoals).ToArray()));
            lambdaAway =
lambdaAway.PointwiseMultiply(Matrix<double>.Build.DenseOfRowMajor(numTeams, numTeams,
data.Select(x => x.AwayAvgGoals).ToArray()));

            double likelihood = 0;
            foreach (var match in data)
            {
               int homeIndex = Array.IndexOf(teams, match.HomeTeam);
               int awayIndex = Array.IndexOf(teams, match.AwayTeam);

               double decay = Math.Exp(-gamma * match.DaysSinceLastMatch / timeDecay);
               double poisson = DixonColesPoisson(match.HomeGoals, match.AwayGoals,
lambdaHome[homeIndex, awayIndex], lambdaAway[homeIndex, awayIndex], decay);
               likelihood += Math.Log(poisson);
            }

            return -likelihood;
         }

         private static double DixonColesPoisson(int homeGoals, int awayGoals, double lambdaHome, double
lambdaAway, double decay)
         {
            if (homeGoals == 0 && awayGoals == 0)
            {
               return Math.Exp(-decay * lambdaHome * lambdaAway) * (1 - Math.Exp(-decay));
            }
            else if (homeGoals == 0 && awayGoals == 1)
            {
               return lambdaAway * Math.Exp(-decay * lambdaHome * lambdaAway);
            }     else if (homeGoals == 1 && awayGoals == 0)
         {
            return lambdaHome * Math.Exp(-decay * lambdaHome * lambdaAway);
         }
         else
         {
            return Math.Pow(lambdaHome, homeGoals) * Math.Pow(lambdaAway, awayGoals) * Math.Exp(-
lambdaHome * lambdaAway);
         }
      }
      private static List<FootballMatchData> LoadAndPreprocessData(string path)
      data;
      }

private static async Task<List<FootballMatchData>> LoadAndPreprocessData()
{
   string apiUrl = "https://api.football.com/football-data";
   string apiKey = "apiKey"; /
```

```
    using var httpClient = new HttpClient();
     httpClient.DefaultRequestHeaders.Add("api-key", apiKey);

    var response = await httpClient.GetAsync(apiUrl);
    if (!response.IsSuccessStatusCode)
    {
        throw new Exception("Failed to fetch data from the API");
    }

    var json = await response.Content.ReadAsStringAsync();
    var rawMatches = JsonConvert.DeserializeObject<List<RawFootballMatchData>>(json);

    var preprocessedData = PreprocessData(rawMatches);
    return preprocessedData;
}
private static List<FootballMatchData> PreprocessData(List<RawFootballMatchData> rawData)
{
    var homeMatches = rawData.GroupBy(x => x.HomeTeam);
    var awayMatches = rawData.GroupBy(x => x.AwayTeam);

    var homeGoals = homeMatches.ToDictionary(x => x.Key, x => x.Sum(m => m.HomeGoals));
    var awayGoals = awayMatches.ToDictionary(x => x.Key, x => x.Sum(m => m.AwayGoals));
    var homeConcededGoals = homeMatches.ToDictionary(x => x.Key, x => x.Sum(m => m.AwayGoals));
    var awayConcededGoals = awayMatches.ToDictionary(x => x.Key, x => x.Sum(m => m.HomeGoals));

    var totalMatches = rawData.Count;
    var totalHomeGoals = rawData.Sum(x => x.HomeGoals);
    var totalAwayGoals = rawData.Sum(x => x.AwayGoals);

    var avgHomeGoals = totalHomeGoals / (double)totalMatches;
    var avgAwayGoals = totalAwayGoals / (double)totalMatches;

    var preprocessedData = rawData.Select(x =>
    {
        var homeAttackStrength = (homeGoals[x.HomeTeam] / (double)homeMatches.Count()) /
avgHomeGoals;
        var awayAttackStrength = (awayGoals[x.AwayTeam] / (double)awayMatches.Count()) /
avgAwayGoals;
        var homeDefenseStrength = (homeConcededGoals[x.HomeTeam] / (double)homeMatches.Count()) /
avgAwayGoals;
        var awayDefenseStrength = (awayConcededGoals[x.AwayTeam] / (double)awayMatches.Count()) /
avgHomeGoals;

        return new FootballMatchData
        {
            HomeTeam = x.HomeTeam,
            AwayTeam = x.AwayTeam,
            HomeGoals = x.HomeGoals,
            AwayGoals = x.AwayGoals,
            HomeAvgGoals = homeAttackStrength * awayDefenseStrength * avgHomeGoals,
            AwayAvgGoals = awayAttackStrength * homeDefenseStrength * avgAwayGoals,
            DaysSinceLastMatch = x.DaysSinceLastMatch
        };
    }).ToList();
```

```
    return preprocessedData;
}
private static void Predict ()
{
  var mlContext = new MLContext();
       var data = mlContext.Data.LoadFromTextFile<FootballMatchData>("./football_data.csv",
separatorChar: ',');

       var pipeline = mlContext.Transforms.Concatenate("Features", "HomeGoals", "AwayGoals")
          .Append(mlContext.Transforms.NormalizeMinMax("Features"))
          .Append(mlContext.Transforms.Regression.PoissonRegression("HomeGoals", "Features"))
          .Append(mlContext.Transforms.CopyColumns("Score", "Score.PoissonRegression"));

       var model = pipeline.Fit(data);

       var predictionEngine = mlContext.Model.CreatePredictionEngine<FootballMatchData,
FootballMatchPrediction>(model);

       var matchData = new FootballMatchData
       {
          HomeTeam = "TeamA",
          AwayTeam = "TeamB",
          HomeGoals = 0,
          AwayGoals = 0
       };

       var prediction = predictionEngine.Predict(matchData);
       Console.WriteLine($"Predicted scores: Home {prediction.PredictedScores[0]}, Away
{prediction.PredictedScores[1]}");
}
  public interface IRepositoryApi<T,K>
       {
          Task<T> GetEntityAsync(string path, HttpClient client);
          Task<List<T>> GetListEntityAsync(string path, HttpClient client);
          List<T> SaveAllToDateBase(List<T> elements);
          IQueryable<T> GetEntityItems();
          T GetEntityItemById(K id);
          Task SaveEntity(T entity);
    }
    public interface IRequest
    {
       string CreateRequest();
       string CreateRequest(string id);
    }
public class EventApi : IRepositoryApi<Event, string>
    {
       AppDBContext _context;
       public EventApi(AppDBContext context)
       {
          _context = context;
       }

       public async Task<Event> GetEntityAsync(string path, HttpClient client)
       {
          Event country = null;
          List<Event> countries = new List<Event>();
```

```
    string json;
    HttpResponseMessage response = await client.GetAsync(path);
    if (response.IsSuccessStatusCode)
    {
        json = await response.Content.ReadAsStringAsync();

        countries = JsonSerializer.Deserialize<List<Event>>(json);
      //  country = countries.FirstOrDefault(p => p.Id == "43");
    }
    return country;
}

public Event GetEntityItemById(string id)
{
    throw new NotImplementedException();
}

public IQueryable<Event> GetEntityItems()
{
    throw new NotImplementedException();
}

public async Task<List<Event>> GetListEntityAsync(string path, HttpClient client)
{
    string json;

    List<Event> events = new List<Event>();
    HttpResponseMessage response = await client.GetAsync(path);
    if (response.IsSuccessStatusCode)
    {
        json = await response.Content.ReadAsStringAsync();
        try
        {
            events = JsonSerializer.Deserialize<List<Event>>(json);
        }
        catch
        {
            Event ev = new Event();
            ev.league_id = "0";
            events.Add(ev);
            return events;
        }

    }
    return events;
}

public List<Event> SaveAllToDateBase(List<Event> elements)
{
    throw new NotImplementedException();
}

public Task SaveEntity(Event entity)
{
    throw new NotImplementedException();
}
```

```
    }
public class StandingApi : IRepositoryApi<Standing, int>
    {
        AppDBContext _context;
        public StandingApi(AppDBContext context)
        {
            _context = context;
        }
        public Task<Standing> GetEntityAsync(string path, HttpClient client)
        {
            throw new NotImplementedException();
        }

        public  Standing GetEntityItemById(int id)
        {
            return _context.Standings.FirstOrDefault(p => p.Id == id);
        }

        public IQueryable<Standing> GetEntityItems()
        {
            return _context.Standings;
        }

        public async Task<List<Standing>> GetListEntityAsync(string path, HttpClient client)
        {
            string json;
            List<Standing> standings = new List<Standing>();

            string eror = "{\"error\":404,\"message\":\"Standing not found!\"}";

            HttpResponseMessage response = await client.GetAsync(path);
            if (response.IsSuccessStatusCode)
            {
                json = await response.Content.ReadAsStringAsync();
                if (json == eror)
                {
                    standings = null;
                    return standings;
                }
                standings = JsonSerializer.Deserialize<List<Standing>>(json);
            }
            return standings;
        }
        public List<Standing> SaveAllToDateBase(List<Standing> standings)
        {
            List<string> str1 = new List<string>();
            List<string> str2 = new List<string>();
            List<Standing> standAdd = new List<Standing>();

            foreach (Standing t in standings)
            {
                str1.Add(t.Team_id);
            }
            foreach (Standing t in _context.Standings.Where(
                p => p.Competition_Id == standings[0].Competition_Id))
            {
```

```
            str2.Add(t.Team_id);
        }
        str1 = str1.Except(str2).ToList();
        if (str1.Count == 0)
        {
            return (standAdd);
        }
        foreach (Standing t in standings)
        {
            foreach (string s in str1)
            {
                if (t.Team_id == s)
                {
                    standAdd.Add(t);
                }
            }
        }
        return (standAdd);
    }
    public async Task SaveEntity(Standing entity)
    {
        _context.Standings.Add(entity);
        await _context.SaveChangesAsync();
    }
  }
public class TeamApiController : Controller
  {
    private static HttpClient client = new HttpClient();

    static List<Player> players = new List<Player>();
    static List<Team> teams = new List<Team>();
    static string competitionID;
    private readonly DataManager dataManager;
    public TeamApiController(DataManager dataManager)
    {

        this.dataManager = dataManager;
    }
    public async Task<IActionResult> Index()
    {
        List<Team> teamsforone = new List<Team>();
        string defolt =
"https://apiv2.apifootball.com?action=get_teams&APIkey=345ebb1d5cb902a9de9280564d2c2467de4d55af8
3f5ee1b7b25c4591ebb078e&team_id=2611";
        teamsforone.Add(await dataManager.TeamRepositoryApi.GetEntityAsync(defolt, client));
        return View(teams.ToList());
    }
    public async Task<IActionResult> ViewTemsByLeagID(string id)
    {
        if (id == null)
        {
            return NotFound();
        }
        competitionID = id;
```

```
            string defolt =
"https://apiv2.apifootball.com?action=get_teams&APIkey=345ebb1d5cb902a9de9280564d2c2467de4d55af8
3f5ee1b7b25c4591ebb078e&league_id=";
            defolt += id;
            ViewBag.Id = id;
            teams = await dataManager.TeamRepositoryApi.GetListEntityAsync(defolt, client);
            return View(teams.ToList());
        }
        public async Task<IActionResult> SaveToDateBase(string id)
        {
            var team = dataManager.TeamRepositoryApi.GetEntityItemById(id);
            ViewBag.Id = competitionID;
            if (team == null)
            {
                foreach (Team c in teams)
                {
                    if (c.Id == id)
                    {
                        team = c;
                    }
                }
                await dataManager.TeamRepositoryApi.SaveEntity(team);
                ViewData["Answer"] = "Збережено";
                return View(team);
            }
            ViewData["Answer"] = "Команда вже була збережена в базі даних";
            return View(team);
        }
        public async Task<IActionResult> SaveAllToDateBase()
        {
            List<Team> teamAdd;
            ViewData["Answer"] = "Збережено";
            ViewBag.Id = competitionID;
            teamAdd = dataManager.TeamRepositoryApi.SaveAllToDateBase(teams);
            if(teamAdd.Count() == 0)
            {
                ViewData["Answer"] = "Команда вже була збережена в базі даних";
                return View();
            }
            foreach (Team t in teamAdd)
            {
                await dataManager.TeamRepositoryApi.SaveEntity(t);
            }
            return View();
        }
public class DixonColesLikelihoodTransformer : ITransformer
{
    public class Options
    {
        public float Lambda { get; set; } = 0.001f;
    }

    public readonly string[] InputColumnNames;
    public readonly string[] OutputColumnNames;
    public readonly Options Options;
```

```
    public DixonColesLikelihoodTransformer(MLContext context, string[] inputColumnNames, string[]
outputColumnNames, Options options)
    {
        InputColumnNames = inputColumnNames;
        OutputColumnNames = outputColumnNames;
        Options = options;
    }

    public bool IsRowToRowMapper => false;

    public SchemaShape GetOutputSchema(SchemaShape inputSchema)
    {
        var columns = new SchemaShape.Column[OutputColumnNames.Length];
        for (int i = 0; i < OutputColumnNames.Length; i++)
        {
            columns[i] = new SchemaShape.Column(OutputColumnNames[i],
SchemaShape.Column.VectorKind.Scalar, NumberDataViewType.Single, false);
        }
        return new SchemaShape(columns);
    }

    public IRowToRowMapper GetRowToRowMapper(SchemaShape inputSchema)
    {
        throw new NotImplementedException("DixonColesLikelihoodTransformer should not be used as a
row-to-row mapper.");
    }

    public DataViewRowTransform[] GetRowTransforms()
    {
        return new DataViewRowTransform[]
        {
            CalculateDixonColesLikelihood
        };
    }

}

public class CompetitionRepositoryBd : IRepositoryBd<Competition, string>
    {
        AppDBContext _context;
        public CompetitionRepositoryBd(AppDBContext context)
        {
            _context = context;
        }
        public bool EntityExists(string id)
        {
            return _context.Competitions.Any(e => e.Id == id);
        }

        public async Task<Competition> GetEntityItems(string id)
        {
            return await _context.Competitions.Include(c => c.Country)
                .FirstOrDefaultAsync(m => m.Id == id);
        }

        public async Task<List<Competition>> GetEntityListItemsByKey()
```

```
        {
            return await _context.Competitions.Include(c => c.Country).ToListAsync();
        }
        public async Task<List<Competition>> GetEntityListItems()
        {
            return await _context.Competitions.ToListAsync();
        }

        public async Task RemoveEntity(Competition entity)
        {
            _context.Competitions.Remove(entity);
            await _context.SaveChangesAsync();
        }

        public async Task SaveEntity(Competition entity)
        {
            _context.Competitions.Add(entity);
            await _context.SaveChangesAsync();
        }

        public async Task UpdateEntity(Competition entity)
        {
            _context.Competitions.Update(entity);
            await _context.SaveChangesAsync();
        }

        public IQueryable<Competition> GetEntityNoAsyncListItems()
        {
            return _context.Competitions;
        }
    }
namespace FootballMVC.Data.BdData.Repositories
{
    public class StandingRepositoryBd : IRepositoryBd<Standing, int>
    {
        AppDBContext _context;
        public StandingRepositoryBd(AppDBContext context)
        {
            _context = context;
        }

        public bool EntityExists(int id)
        {
            return _context.Players.Any(e => e.Id == id);
        }

        public async Task<Standing> GetEntityItems(int id)
        {
            return await _context.Standings
                  .FirstOrDefaultAsync(m => m.Id == id);
        }

        public async Task<List<Standing>> GetEntityListItems()
        {
            return await _context.Standings.ToListAsync();
        }
```

```
        public Task<List<Standing>> GetEntityListItemsByKey()
        {
            throw new NotImplementedException();
        }

        public IQueryable<Standing> GetEntityNoAsyncListItems()
        {
            return _context.Standings;
        }

        public async Task RemoveEntity(Standing entity)
        {
            _context.Standings.Remove(entity);
            await _context.SaveChangesAsync();
        }

        public async Task SaveEntity(Standing entity)
        {
            _context.Standings.Add(entity);
            await _context.SaveChangesAsync();
        }

        public async Task UpdateEntity(Standing entity)
        {
            _context.Standings.Update(entity);
            await _context.SaveChangesAsync();
        }
    }
}

@model IEnumerable<FootballMVC.Models.Entities.Competition>

@{
    ViewData["Title"] = "CountryLeags";
}

<h1>Турніри</h1>

<a asp-action="SaveAllToDateBase" asp-route-id="@Model.FirstOrDefault().CountryId">Зберегти всі
турніри</a>
<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Id)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Name)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.CountryId)
            </th>
            <th></th>
        </tr>
    </thead>
```

```
    <tbody>
        @foreach (var item in Model)
        {
            <tr>
                <td>
                    @Html.DisplayFor(modelItem => item.Id)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Name)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.CountryId)
                </td>
                <td>
                    <a asp-controller="StandingApi" asp-action="ViewStandingByLeagID" asp-route-
id="@item.Id">Турнірна таблиця</a>
                <td>
                </td>
                <td>
                    <a asp-controller="TeamApi" asp-action="ViewTemsByLeagID" asp-route-
id="@item.Id">Команди</a>
                </td>
                <td>
                    <a asp-action="SaveToDateBase" asp-route-id="@item.Id">Зберегти</a>
                </td>
            </tr>

        }
    </tbody>
</table>
<div>
    <a asp-controller="CountryApi" asp-action="Index">Назад</a>
</div>

@model IEnumerable<FootballMVC.Models.Entities.Competition>

@{
    ViewData["Title"] = "Index";
}

<h1>Index</h1>

<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Id)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Name)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.CountryId)
            </th>
            <th></th>
        </tr>
```

```
    </thead>
    <tbody>
        @foreach (var item in Model)
        {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Id)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.CountryId)
            </td>
        </tr>
        }
    </tbody>
</table>
<!DOCTYPE HTML>

<html>
<head>
    <title>Football Statistics</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no" />
    <link rel="stylesheet" href="~/css/main.css" />
    <link rel="stylesheet" href="~/css/onlymy.css" />
</head>
<body>
    <div id="page-wrapper">

        <!-- Header -->
        @await Html.PartialAsync("_Header");

        <!-- Main -->
        <div id="main">
            <div class="container">
                <div class="row main-row">
                    <div class="col-4 col-12-medium">
                        <section>
                            <h2>         Популярні Турніри</h2>
                            <ul class="category-list-competitions clearfix">
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="148">АПЛ</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="468">Ла Ліга</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="262">Серія А</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="195">Бундесліга</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="176">Ліга 1</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="523">УПЛ</a></li>
                                <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="589">Ліга Чемпіонів</a></li>
```

```
                    <li class="category-item"> <a asp-controller="Main" asp-action="CompetitionsInfo"
asp-route-id="590">Ліга Європи</a></li>
                  </ul>
                </section>

                <section>
                  @*<h2>How about some links?</h2>*@
                  <div>
                    <div class="row">
                      <div class="col-6 col-12-small">

                      </div>
                      <div class="col-6 col-12-small">

                      </div>
                    </div>
                  </div>
                </section>

              </div>
              <div class="col-8 col-12-medium imp-medium">
                <section>
                  @RenderBody()
                </section>
              </div>
            </div>
          </div>
        </div>

        <!-- Footer -->
        @await Html.PartialAsync("_Footer");

    </div>

    <!-- Scripts -->
    <script src="~/js/jquery.min.js"></script>
    <script src="~/js/browser.min.js"></script>
    <script src="~/js/breakpoints.min.js"></script>
    <script src="~/js/util.js"></script>
    <script src="~/js/main.js"></script>

</body>
</html>
<div id="header-wrapper">
    <div class="container">
        <div class="row">
            <div class="col-12">

                <header id="header">
                    <h1><a href="index.html" id="logo">Football Statistics</a></h1>
                    <nav id="nav">
                        <a asp-controller="Main" asp-action="GetAllCountries"> Країни</a>
                        <a href="onecolumn.html">Турніри</a>
                        <a href="onecolumn.html">Матчі Live</a>
                        <a href="onecolumn.html">Команда Сезону</a>
```

```
                <a href="onecolumn.html">Пошук</a>
                <a href="threecolumn.html">Реєстрація</a>
            </nav>
        </header>

    </div>
  </div>
</div>

@model ErrorViewModel
@{
  ViewData["Title"] = "Error";
}

<h1 class="text-danger">Error.</h1>
<h2 class="text-danger">An error occurred while processing your request.</h2>

@if (Model.ShowRequestId)
{
  <p>
    <strong>Request ID:</strong> <code>@Model.RequestId</code>
  </p>
}

<h3>Development Mode</h3>
<p>
  Swapping to <strong>Development</strong> environment will display more detailed information about
the error that occurred.
</p>
<p>
  <strong>The Development environment shouldn't be enabled for deployed applications.</strong>
  It can result in displaying sensitive information from exceptions to end users.
  For local debugging, enable the <strong>Development</strong> environment by setting the
<strong>ASPNETCORE_ENVIRONMENT</strong> environment variable to
<strong>Development</strong>
  and restarting the app.
</p>

@model IEnumerable<FootballMVC.Models.Entities.Standing>

@{
  ViewData["Title"] = "ViewStandingByLeagID";
}

<h1>Турнірна таблиця </h1>

<a asp-action="SaveAllToDateBase" asp-route-id="@Model.FirstOrDefault().Competition_Id">Зберегти
турнірну таблицю</a>
<table class="table">
  <thead>
    <tr>
      @*<th>
          @Html.DisplayNameFor(model => model.Competition_Id)
        </th>*@
      <th>
```

```
            @Html.DisplayNameFor(model => model.LeaguePosition)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.Team_Name)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.League_payed)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.League_W)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.League_D)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.League_L)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.League_PTS)
          </th>
          <th></th>
        </tr>
      </thead>
      <tbody>
        @foreach (var item in Model)
        {
        <tr>
          @*<td>
          @Html.DisplayFor(modelItem => item.Competition_Id)
        </td>*@
          <td>
            @Html.DisplayFor(modelItem => item.LeaguePosition)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.Team_Name)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.League_payed)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.League_W)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.League_D)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.League_L)
          </td>
          <td>
            @Html.DisplayFor(modelItem => item.League_PTS)
          </td>
        </tr>
        }
      </tbody>
    </table>
    @model FootballMVC.Models.Entities.Standing
```

```
@{
    ViewData["Title"] = "Create";
}

<h1>Create</h1>

<h4>Standing</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Create">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Team_Name" class="control-label"></label>
                <input asp-for="Team_Name" class="form-control" />
                <span asp-validation-for="Team_Name" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="LeaguePosition" class="control-label"></label>
                <input asp-for="LeaguePosition" class="form-control" />
                <span asp-validation-for="LeaguePosition" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_payed" class="control-label"></label>
                <input asp-for="League_payed" class="form-control" />
                <span asp-validation-for="League_payed" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_W" class="control-label"></label>
                <input asp-for="League_W" class="form-control" />
                <span asp-validation-for="League_W" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_D" class="control-label"></label>
                <input asp-for="League_D" class="form-control" />
                <span asp-validation-for="League_D" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_L" class="control-label"></label>
                <input asp-for="League_L" class="form-control" />
                <span asp-validation-for="League_L" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_PTS" class="control-label"></label>
                <input asp-for="League_PTS" class="form-control" />
                <span asp-validation-for="League_PTS" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Competition_Id" class="control-label"></label>
                <input asp-for="Competition_Id" class="form-control" />
                <span asp-validation-for="Competition_Id" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Team_id" class="control-label"></label>
                <input asp-for="Team_id" class="form-control" />
```

```
            <span asp-validation-for="Team_id" class="text-danger"></span>
        </div>
        <div class="form-group">
            <input type="submit" value="Create" class="btn btn-primary" />
        </div>
    </form>
  </div>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>
@model FootballMVC.Models.Entities.Standing

@{
  ViewData["Title"] = "Delete";
}

<h1>Delete</h1>

<h3>Are you sure you want to delete this?</h3>
<div>
  <h4>Standing</h4>
  <hr />
  <dl class="row">
    <dt class = "col-sm-2">
      @Html.DisplayNameFor(model => model.Team_Name)
    </dt>
    <dd class = "col-sm-10">
      @Html.DisplayFor(model => model.Team_Name)
    </dd>
    <dt class = "col-sm-2">
      @Html.DisplayNameFor(model => model.LeaguePosition)
    </dt>
    <dd class = "col-sm-10">
      @Html.DisplayFor(model => model.LeaguePosition)
    </dd>
    <dt class = "col-sm-2">
      @Html.DisplayNameFor(model => model.League_payed)
    </dt>
    <dd class = "col-sm-10">
      @Html.DisplayFor(model => model.League_payed)
    </dd>
    <dt class = "col-sm-2">
      @Html.DisplayNameFor(model => model.League_W)
    </dt>
    <dd class = "col-sm-10">
      @Html.DisplayFor(model => model.League_W)
    </dd>
    <dt class = "col-sm-2">
      @Html.DisplayNameFor(model => model.League_D)
    </dt>
    <dd class = "col-sm-10">
      @Html.DisplayFor(model => model.League_D)
    </dd>
    <dt class = "col-sm-2">
```

```
            @Html.DisplayNameFor(model => model.League_L)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_L)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.League_PTS)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_PTS)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Competition_Id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Competition_Id)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Team_id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Team_id)
        </dd>
    </dl>

    <form asp-action="Delete">
        <input type="hidden" asp-for="Id" />
        <input type="submit" value="Delete" class="btn btn-danger" /> |
        <a asp-action="Index">Back to List</a>
    </form>
</div>
@model FootballMVC.Models.Entities.Standing

@{
    ViewData["Title"] = "Details";
}

<h1>Details</h1>

<div>
    <h4>Standing</h4>
    <hr />
    <dl class="row">
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Team_Name)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Team_Name)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.LeaguePosition)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.LeaguePosition)
        </dd>
        <dt class = "col-sm-2">
```

```
            @Html.DisplayNameFor(model => model.League_payed)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_payed)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.League_W)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_W)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.League_D)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_D)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.League_L)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_L)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.League_PTS)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.League_PTS)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Competition_Id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Competition_Id)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Team_id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Team_id)
        </dd>
    </dl>
</div>
<div>
    <a asp-action="Edit" asp-route-id="@Model.Id">Edit</a> |
    <a asp-action="Index">Back to List</a>
</div>
@model FootballMVC.Models.Entities.Standing

@{
    ViewData["Title"] = "Edit";
}

<h1>Edit</h1>

<h4>Standing</h4>
```

```
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Edit">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <input type="hidden" asp-for="Id" />
            <div class="form-group">
                <label asp-for="Team_Name" class="control-label"></label>
                <input asp-for="Team_Name" class="form-control" />
                <span asp-validation-for="Team_Name" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="LeaguePosition" class="control-label"></label>
                <input asp-for="LeaguePosition" class="form-control" />
                <span asp-validation-for="LeaguePosition" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_payed" class="control-label"></label>
                <input asp-for="League_payed" class="form-control" />
                <span asp-validation-for="League_payed" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_W" class="control-label"></label>
                <input asp-for="League_W" class="form-control" />
                <span asp-validation-for="League_W" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_D" class="control-label"></label>
                <input asp-for="League_D" class="form-control" />
                <span asp-validation-for="League_D" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_L" class="control-label"></label>
                <input asp-for="League_L" class="form-control" />
                <span asp-validation-for="League_L" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="League_PTS" class="control-label"></label>
                <input asp-for="League_PTS" class="form-control" />
                <span asp-validation-for="League_PTS" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Competition_Id" class="control-label"></label>
                <input asp-for="Competition_Id" class="form-control" />
                <span asp-validation-for="Competition_Id" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Team_id" class="control-label"></label>
                <input asp-for="Team_id" class="form-control" />
                <span asp-validation-for="Team_id" class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Save" class="btn btn-primary" />
            </div>
        </form>
    </div>
```

```
</div>

<div>
    <a asp-action="Index">Back to List</a>
</div>
@model IEnumerable<FootballMVC.Models.Entities.Standing>

@{
    ViewData["Title"] = "Index";
}

<h1>Index</h1>

<p>
    <a asp-action="Create">Create New</a>
</p>
<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Team_Name)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.LeaguePosition)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.League_payed)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.League_W)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.League_D)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.League_L)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.League_PTS)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Competition_Id)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Team_id)
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model) {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Team_Name)
            </td>
            <td>
```

```
            @Html.DisplayFor(modelItem => item.LeaguePosition)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.League_payed)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.League_W)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.League_D)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.League_L)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.League_PTS)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.Competition_Id)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.Team_id)
        </td>
        <td>
            <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> |
            <a asp-action="Details" asp-route-id="@item.Id">Details</a> |
            <a asp-action="Delete" asp-route-id="@item.Id">Delete</a>
        </td>
    </tr>
}
  </tbody>
</table>
```

ДОДАТОК Г.

Додаток Г – Ілюстративна частина

РОЗРОБКА МЕТОДІВ І ПРОГРАМНИХ ЗАСОБІВ ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ФУТБОЛЬНИХ МАТЧІВ НА ОСНОВІ МОДЕЛЕЙ ШТУЧНОГО ІНТЕЛЕКТУ

Магістерська кваліфікаційна робота

«РОЗРОБКА МЕТОДІВ І ПРОГРАМНИХ ЗАСОБІВ ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ФУТБОЛЬНИХ МАТЧІВ НА ОСНОВІ МОДЕЛЕЙ ШТУЧНОГО ІНТЕЛЕКТУ»

Виконав: студент 2 курсу
Групи 1ПІ-22м Перебейнос Р.Л.
Керівник: к.т.н., доцент каф. ПЗ Катєльніков Д.І.

Рисунок Г.1 – Назва роботи

Метою роботи є підвищення точності прогнозування результатів футбольних матчів за рахунок використання удосконаленої моделі штучного інтелекту Dixon-Coles.

Об'єкт дослідження – це процес прогнозування результатів футбольних матчів на основі накопиченої статистики.

Предмет дослідження – є методи та засоби прогнозування результатів матчів на основі футбольної статистики та моделей штучного інтелекту.

Рисунок Г.2 – Мета роботи

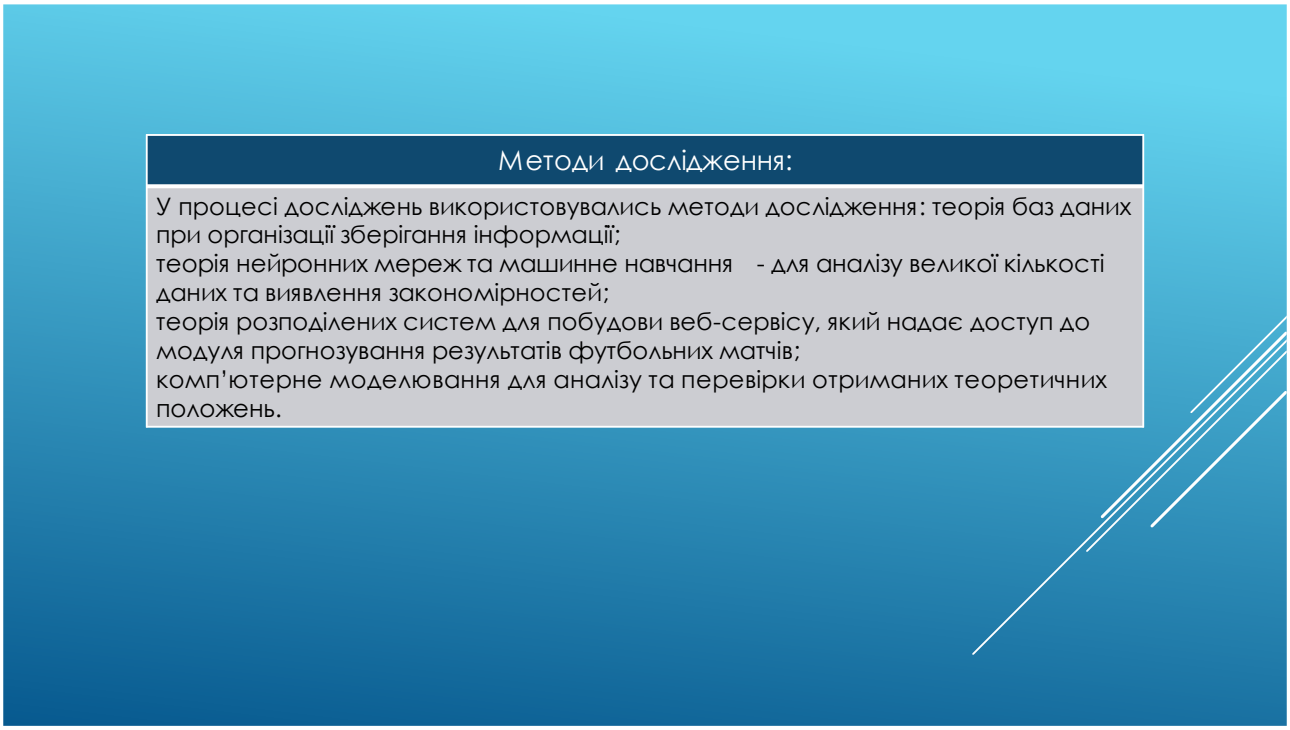


Рисунок Г.3 – Методи дослідження

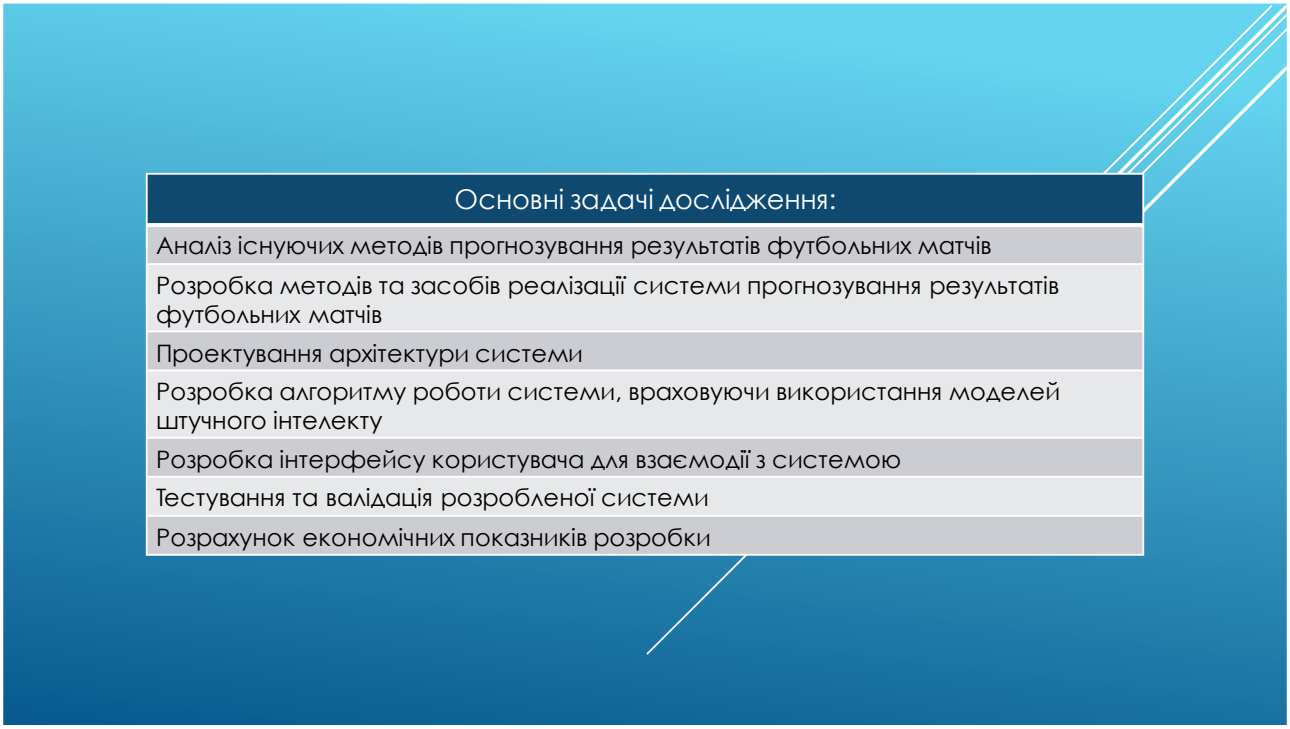


Рисунок Г.4 – Задачі дослідження

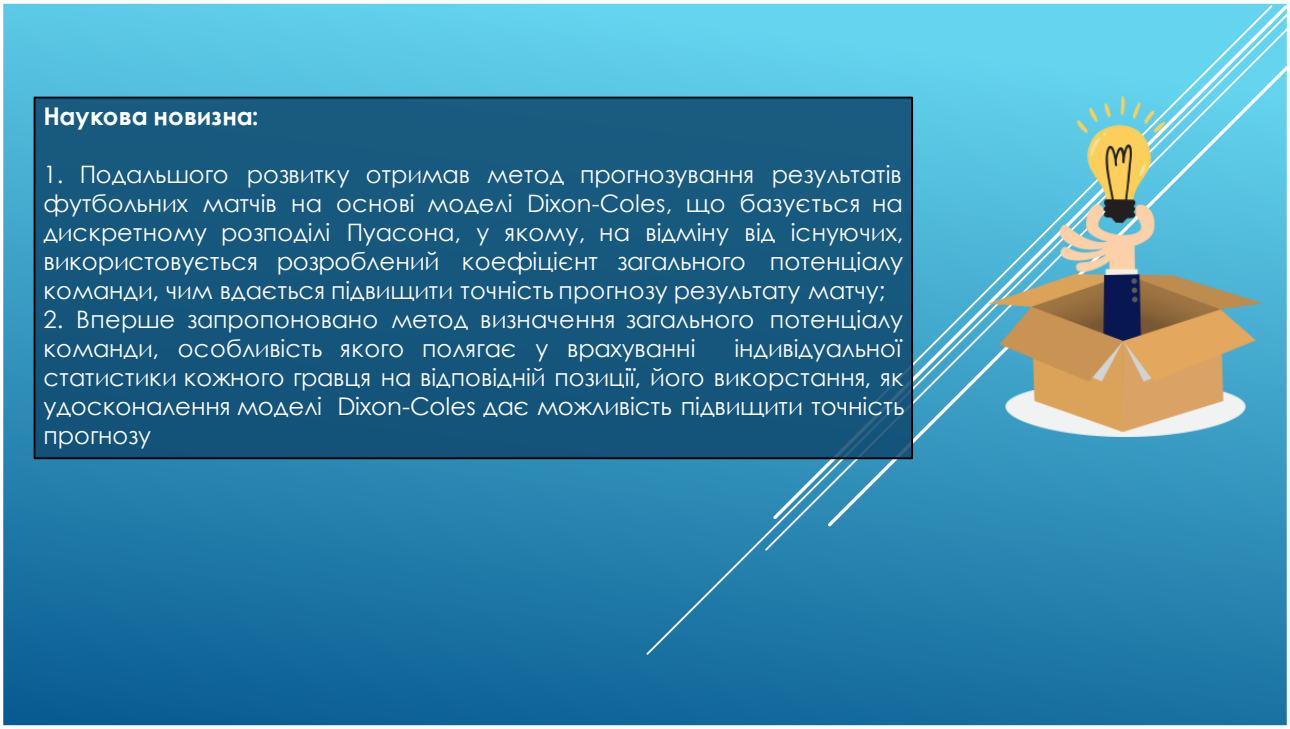


Рисунок Г.5 – Наукова новизна

Рисунок Г.6 – Практична цінність

Рисунок Г.7 – Апробація результатів роботи

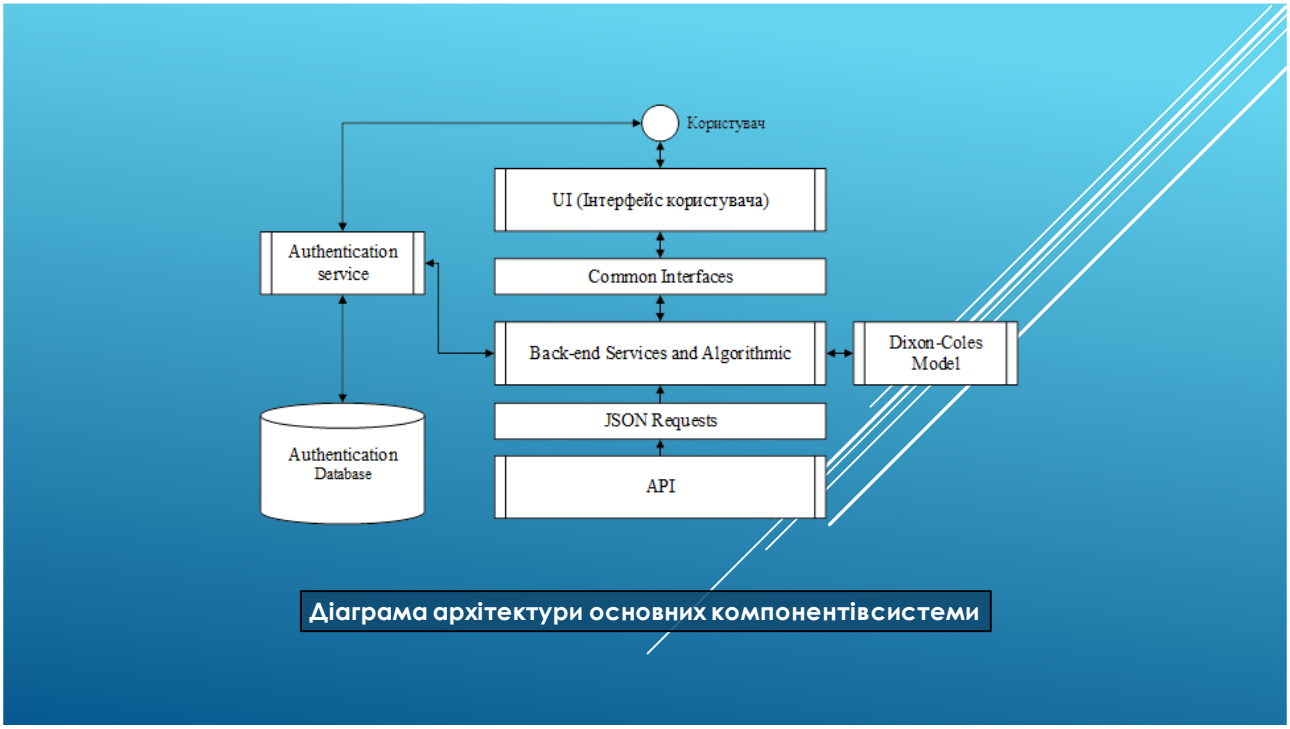


Рисунок Г.8 – Архітектура системи

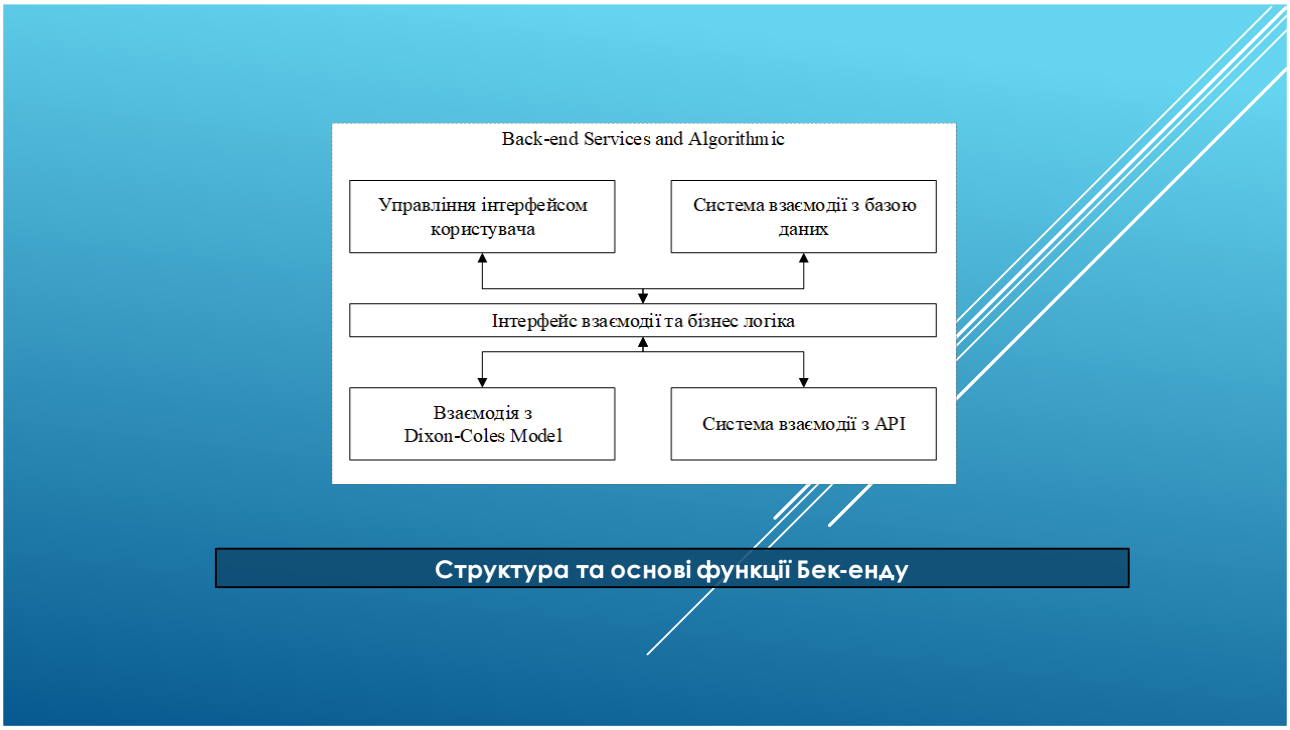


Рисунок Г.9 – Структура та основні функції бек-едну

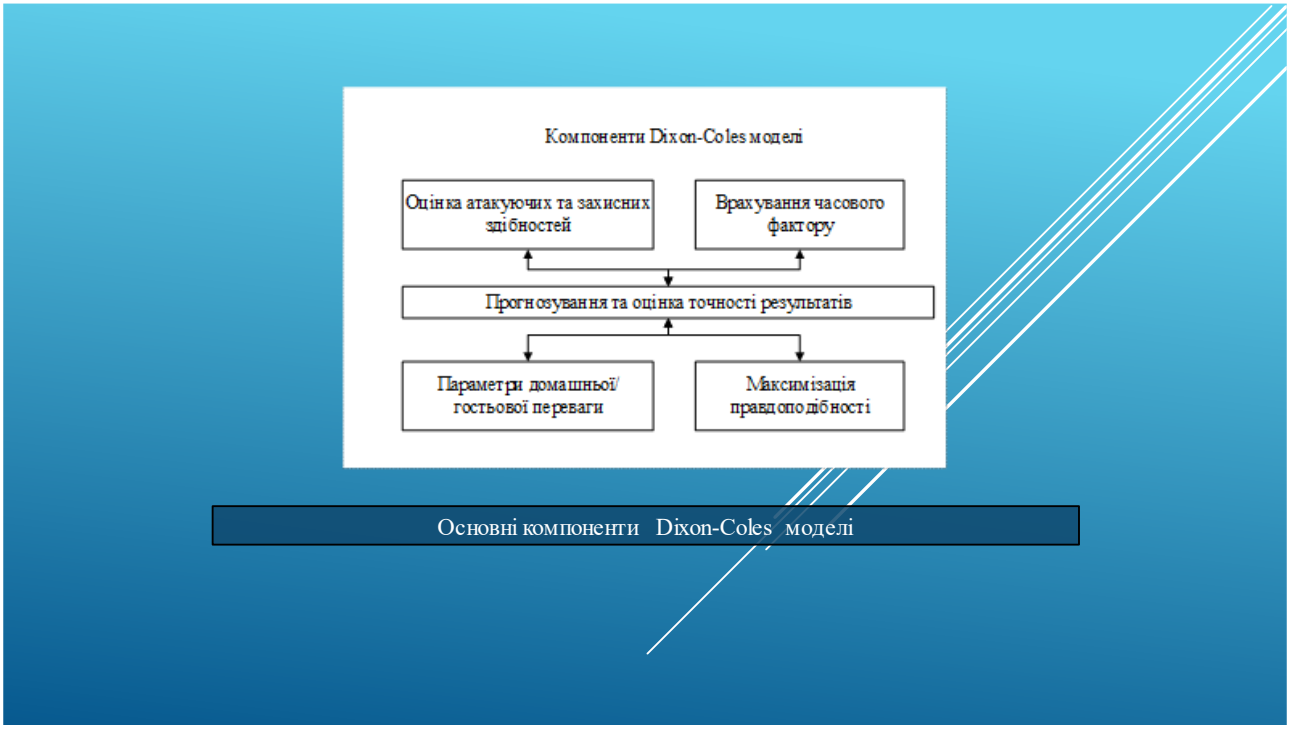


Рисунок Г.10 – Основні компоненти Dixon-Coles моделі

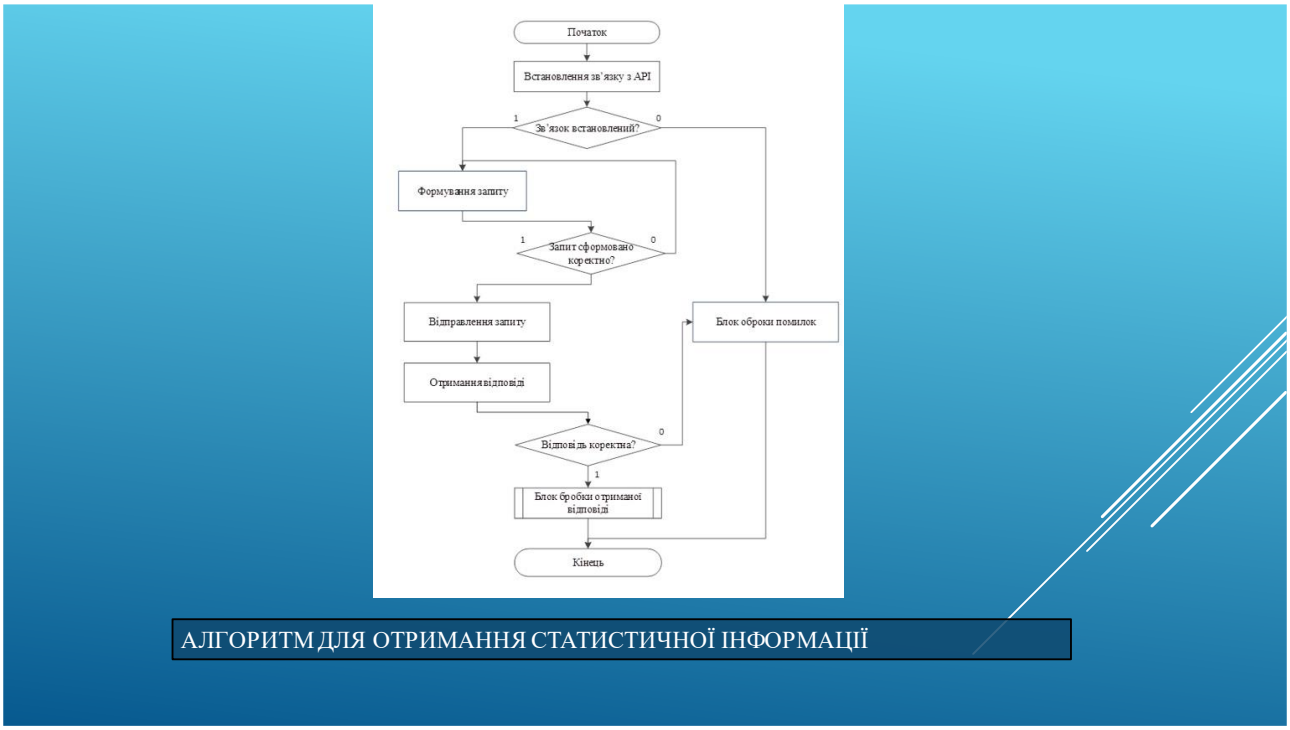


Рисунок Г.11 – Алгоритм отримання статистичної інформаціїї

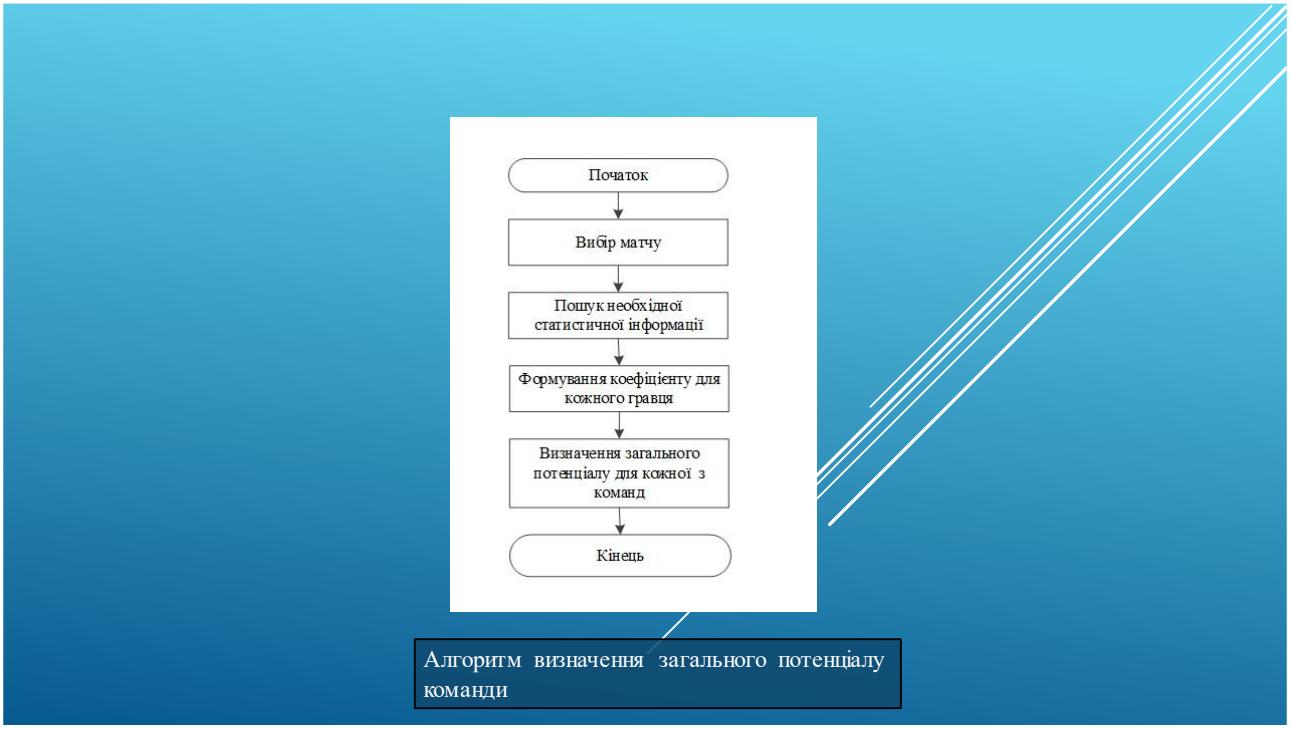


Рисунок Г.12 – Алгоритм визначення загального потенціалу команди

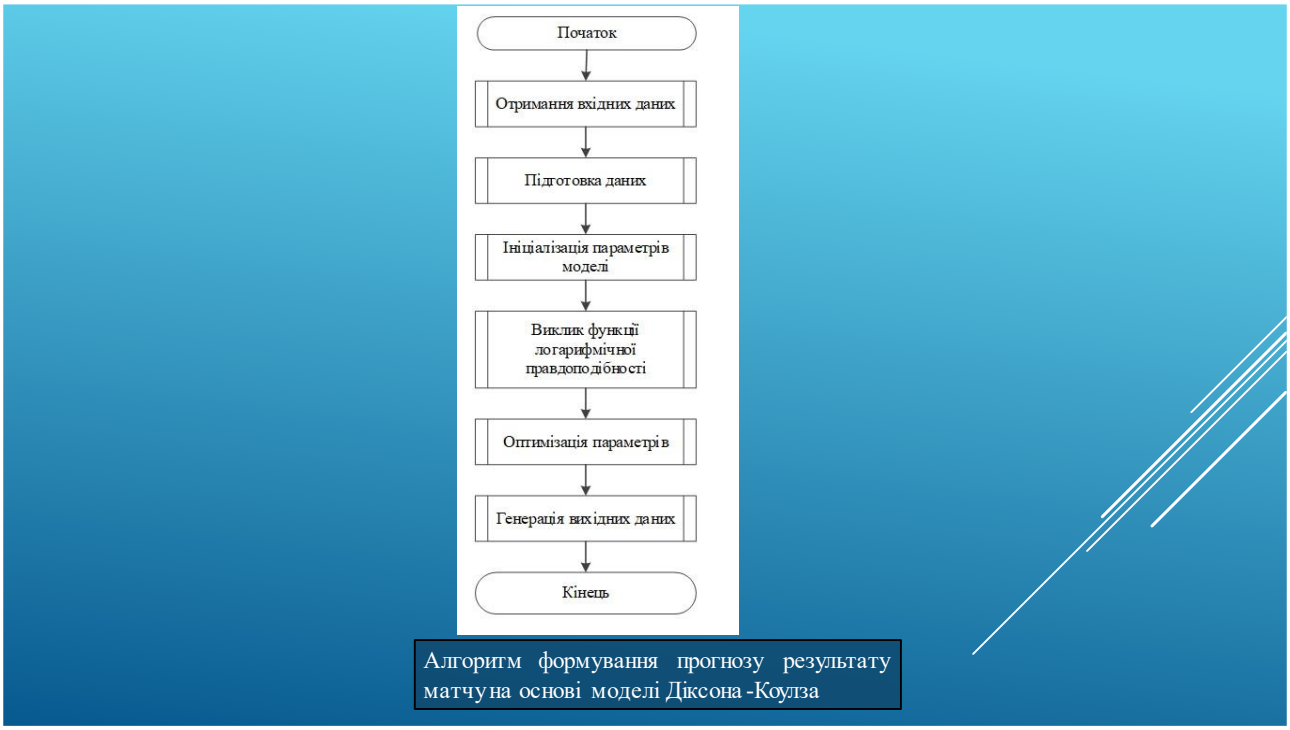


Рисунок Г.13 – Алгоритм формування прогнозу результату матчу на основі моделі Діксона-Коулза

Рисунок Г.14 – Використовувані технології

Рисунок Г.15 – Результати прогнозування

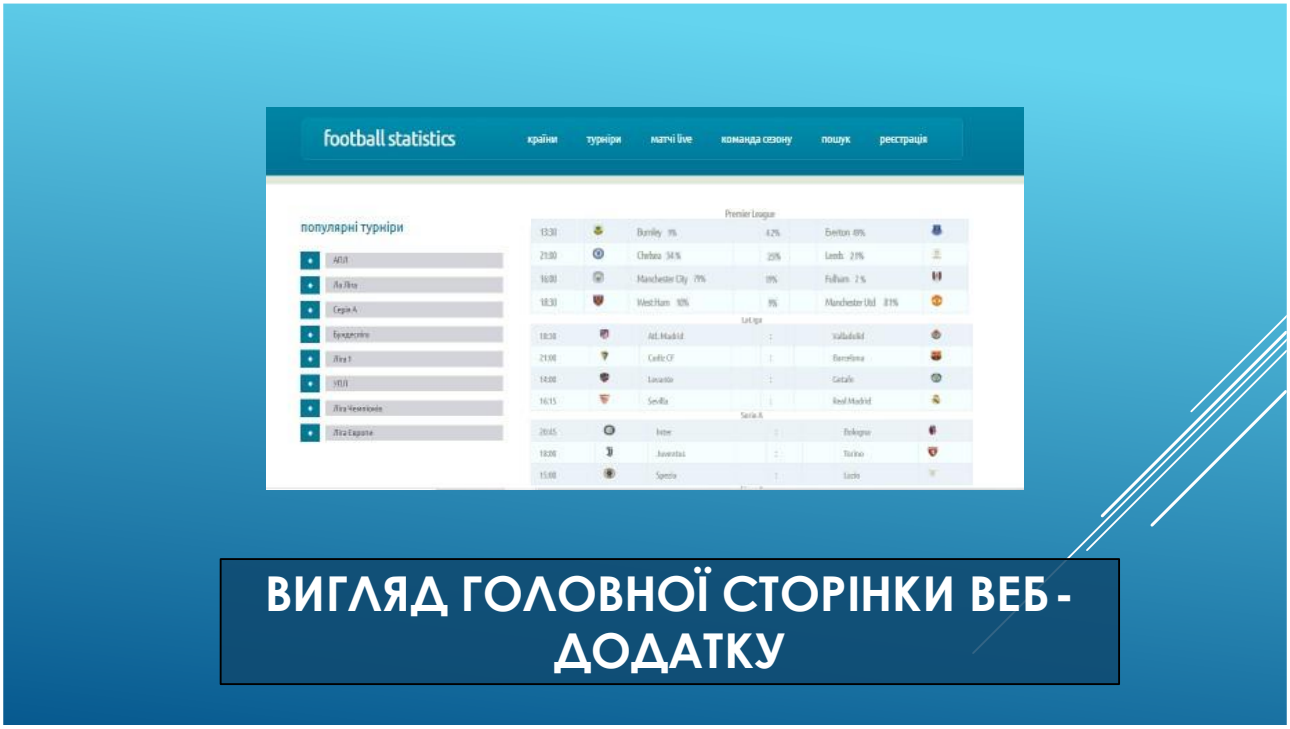


Рисунок Г.16 – Вигляд веб-додатку

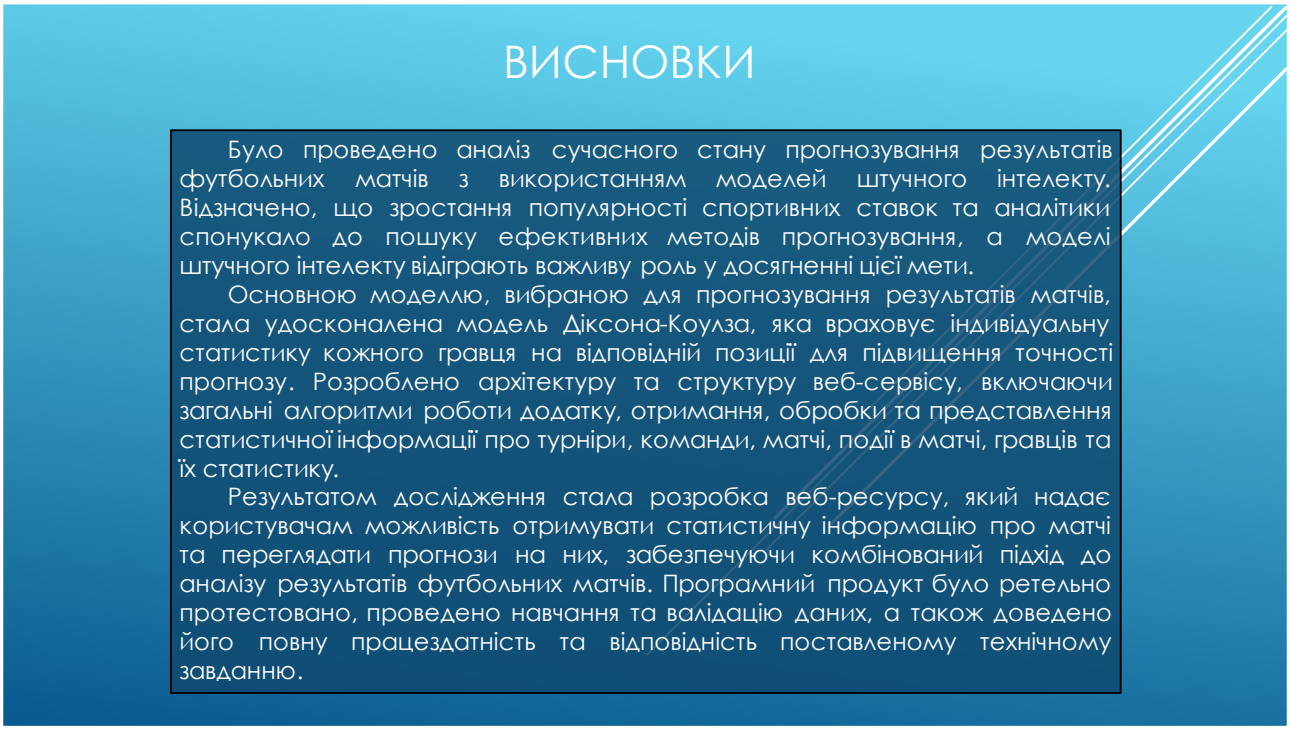


Рисунок Г.17 – Висновки