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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Індивідуальний дослідницький проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інтегровані інформаційні системи»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Прогнозування результатів спортивних подій на основі зібраної статистики»

Виконав:
студент IV курсу, групи ІА-81
Климов Андрій Андрійович _____

Керівник:
Доцент каф. ІСТ, к. т. н.
Шимкович Володимир Миколайович _____

Засвідчую, що у цьому проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інтегровані інформаційні системи»

ЗАВДАННЯ

на індивідуальний дослідницький проєкт студенту
Климову Андрію Андрійовичу

1. Тема проєкту «Прогнозування результатів спортивних подій на основі зібраної статистики», керівник проєкту Шимкович Володимир Миколайович, доцент каф. ІСТ, к. т. н.

2. Термін подання студентом проєкту: 15 червня 2022 року

3. Вихідні дані до проєкту: дані про статистику футбольних клубів в матчах в 2014-2019 роках, засоби розробки.

4. Зміст пояснювальної записки: аналіз завдання, планування шляхів до розв’язання поставленої задачі, збір даних, аналіз даних, вибір моделі для навчання, навчання моделі, розроблення додатку, перевірка якості прогнозування.

5. Перелік графічного матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо): діаграма розгортання, діаграма послідовності, діаграма прецедентів, схема структурна, схеми перетворення таблиць даних.

6. Дата видачі завдання 1 грудня 2021 року

Календарний план

№ з/п	Назва етапів виконання проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз актуальності системи	31.03.2022 – 02.04.2022	
2	Аналіз існуючих рішень	02.04.2022 – 03.04.2022	
3	Розроблення схем	04.04.2022 – 05.04.2022	
4	Підготовка теоретичної бази та засобів розробки	06.04.2022 – 12.04.2022	
5	Робота з даними	12.04.2022 – 30.04.2022	
6	Застосування методів машинного навчання	31.05.2022 – 02.06.2022	
7	Розроблення додатку	03.06.2022 – 04.06.2022	
8	Перевірка якості роботи розробленої системи	05.06.2022 – 13.06.2022	

Студент Андрій КЛИМОВ

Керівник Володимир ШИМКОВИЧ

АНОТАЦІЯ

Климов А.А. Прогнозування результатів спортивних подій на основі зібраної статистики. КПІ ім. Ігоря Сікорського, Київ, 2022.

Проєкт містить 92 с. тексту, 68 рисунків, посилання на 45 літературних джерел та 5 креслеників.

Ключові слова: прогнозування, аналіз даних, data science, спорт, футбол, машинне навчання, логістична регресія, лінійна регресія, pandas, sklearn, SQL.

Об’єктом розробки є система прогнозування результатів спортивних подій на основі зібраної статистики.

Мета розробки – задоволення потреб користувача системи та надання інформації для користування за власним розсудом.

У індивідуальному дослідницькому проєкті розроблено додаток, який прогнозує ймовірності результату футбольного поєдинку за обраних команд. В основі системи лежить модель машинного навчання, а саме модель логістичної регресії. Для актуальності даних наявний скрипт, який оновлює характеристики команд.

ANNOTATION

Klymov A.A. Forecasting the results of sporting events based on collected statistics. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, 2022.

The project contains 94 pages of text, 68 figures, references to 45 literature sources, appendices: code and 5 drawings.

Key words: forecasting, data analysis, data science, sports, football, machine learning, logistic regression, linear regression, pandas, sklearn, SQL.

The object of development is a system for forecasting the results of sporting events based on collected statistics.

The purpose of development is to meet the needs of the system user and provide information for use at its discretion.

An application that has been developed predicts the probabilities of the outcome of a football match for selected teams. The system is based on the machine learning model, namely the logistic regression model. For data relevance, there is a script that updates the characteristics of the commands.

Номер рядка	Формат	Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІА81.090БАК.003 ПЗ	Пояснювальна записка	92		
6	А3	ІА81.090БАК.003 Д1	Прогнозування результатів	1		
7			спортивних подій на основі			
8			зібраної статистики.			
9			Діаграма розгортання.			
10	А3	ІА81.090БАК.003 Д2	Прогнозування результатів	1		
11			спортивних подій на основі			
12			зібраної статистики.			
13			Діаграма послідовності			
14	А3	ІА81.090БАК.003 Д3	Прогнозування результатів	1		
15			спортивних подій на основі			
16			зібраної статистики.			
17			Діаграма прецедентів			
18	А3	ІА81.090БАК.003 Э1	Прогнозування результатів	1		
19			спортивних подій на основі			
20			зібраної статистики.			
21			Схема структурна			
22	А3	ІА81.090БАК.003 Д4	Прогнозування результатів	1		
23			спортивних подій на основі			
24			зібраної статистики.			
25			Схеми перетворення			
26			таблиці даних			
27						
28						

Зм.	Арк.	№ докум.	Підпис	Дата	ІА81.090БАК.003 ТП		
Розроб.		Климов А.А.			Прогнозування результатів спортивних подій на основі зібраної статистики. Відомість проєкту	Літ.	Аркуш 1 / Аркушів 1
Перевір.		Шимкович В.М.				Т	
Затв.						КПІ ім. І. Сікорського Група ІА-81	

Пояснювальна записка
до індивідуального дослідницького проєкту
на тему: «Прогнозування результатів спортивних подій на основі зібраної статистики»

Київ – 2022 року

ЗМІСТ

ВСТУП

В сучасному світі майже кожна людина більшою чи меншою мірою слідкує за спортом. Мільйони вболівальників збираються перед екранами телевізорів та комп'ютерів під час ключових подій та з захопленням спостерігають на виступ команди чи конкретного спортсмена.

В Україні найбільш популярними видами спорту, певно, є футбол та бокс. Під час матчів збірної або бою нашого атлету фанати забувають про особисті проблеми та повністю віддають події свої увагу і емоції.

Також варто сказати, що вболівальники люблять дискутувати з собі подібними та заздалегідь прогнозувати результат тієї чи іншої спортивної події, а деякі люди навіть ставлять свої гроші в букмекерських конторах: найбільш впевнені у собі таким чином намагаються заробити гроші, а хтось просто таким чином збільшує інтерес до перегляду.

Зважаючи на інтерес українців до футболу було прийняте рішення розробити систему, яка буде прогнозувати результат спортивної події на прикладі футбольного матчу, спираючись на статистику, умови, форму та статус команд.

Аналогічні системи використовуються в букмекерських конторах. Вони мають дуже високу точність та враховують сотні факторів при аналізі. За справну роботу формувачів коефіцієнтів відповідає цілий штаб працівників: аналітики даних, інженери даних, інженери з машинного навчання, дослідники з машинного навчання. Саме через високий рівень якості подібних систем букмекерські контори в Україні кожен рік входять до найбільш прибуткових корпорацій, наприклад чистий прибуток найбільш популярного букмекеру України Parimatch в 2020-му році становив 400 мільйонів доларів [1], а вартість самої компанії в 2021 році перевалила за 1 мільярд доларів [2].

Ці надзвичайно великі цифри чітко демонструють, що більшість гравців в таких закладах не виграють грошей. Суто фізично людина не може врахувати всі відомі фактори та об'єктивно розрахувати ймовірність тієї чи іншої події.

Метою роботи є створення системи прогнозування результатів футбольних матчів засобами машинного навчання, що дозволить отримувати прогнозування на необхідну користувачеві подію.

В ході розробки будуть вирішені наступні задачі:

– аналіз даних з набору статистики футбольних матчів;

– виведення нових статистичних показників, необхідних для подальшого аналізу та використання при навчанні;

– збір додаткової необхідної інформації, якої не було в наборі даних;

– навчання нейронної мережі методами машинного навчання;

– збір актуальної статистичної інформації про футбольні клуби та приведення її до необхідного вигляду;

– перевірка працездатності навченої нейронної мережі на актуальних даних в порівнянні з аналогічними професійними системами.

Дана система дозволить відкинути всі упередження та розрахувати оцінювану ймовірність перемоги команди-господаря, нічиєї та перемоги гостьової команди спираючись на необхідні статистичні показники. Результати роботи системи напрямлені на надання інформації для задоволення власних потреб користувача.

Дипломний проєкт складається з наступних розділів: вступ, призначення та галузь застосування, огляд та аналіз аналогічних програмних систем, розроблення схем програмного додатку, підготовка до реалізації системи, робота з даними, застосування методів машинного навчання, актуальні дані, розроблення застосунку, перевірка якості роботи розробленої системи, висновки, список використаних джерел із 45 найменувань. Графічна частина включає 5 креслеників формату А3. Загальний обсяг 92 сторінки.

1 ПРИЗНАЧЕННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Призначення системи, яка прогнозує результат спортивної події може бути доволі широким: як комерційним, так і некомерційним. До першої групи відносяться продукти, які може придбати будь-яка людина і отримати доступ до прогнозів, зроблених даним сервісом. Надалі покупець такої послуги може розпоряджатися придбаною інформацією на власний розсуд: спробувати обіграти букмекерську контору, порівняти свої прогнози з прогнозами системи, використати отриману інформацію виключно з метою задоволення власного інтересу тощо. Також до комерційної галузі застосування входять безпосередньо засоби формування коефіцієнтів букмекерських контор, в основі яких якраз-таки лежить оцінка ймовірності тієї чи іншої події. Як вже було зазначено у вступі, такі системи є найбільш точними та професіональними, саме на них побудований багатомільйонний бізнес.

Некомерційними ж є штучні інтелекти аналогічні тим, що були описані на самому початку розділу: це різноманітні безкоштовні програми та сервіси, в яких за спеціальним алгоритмом оцінюється ймовірність різних результатів на спортивну подію. Разом з платними аналогами, такі системи рідко коли можуть конкурувати з професійними, адже перші включать в себе набагато менше факторів для аналізу. Високою якістю для них вважається здатність заздалегідь формувати ймовірності близькі до тих, що пропонуються букмекерською конторою. Тільки вони за умови якісного аналізу зі сторони гравця можуть теоретично допомогти заробити гроші в букмекерській конторі. Розроблювана система в жодному разі не має на меті гарантування прибутку у випадку гри на ставках, а орієнтована перш за все на задоволення інтересу вболівальників та надання інформації для розпорядження нею за власним розсудом.

2 ОГЛЯД ТА АНАЛІЗ АНАЛОГІЧНИХ ПРОГРАМНИХ СИСТЕМ

Оскільки майже всі програми та сервіси, які прогнозують результат спортивної події прив'язані до індустрії ставок на спорт, то доцільно буде оцінювати якість саме таких продуктів. Відповідно, якщо користувачі задоволені, це означає, що система якісно оцінює ймовірності в тій чи іншій спортивній події.

2.1 Soccer Match Predictor

Soccer Match Predictor – це програма, розроблена досить давно, у 2013 році. Компанія розробників NeuralBet, за їхніми словами, забезпечила додаток інноваційними алгоритмами для максимально точних прогнозів на ставки на футбол. Саме такий опис знаходиться під посиланням для завантаження програми на його офіційному сайті.

За словами розробників, на ринку програма вже 9 років. Її не оновлювали жодного разу, проте відгуки на сайті виключно позитивні. За словами користувачів на сайті програми, навіть через 9 років усе працює справно [3]. Інтерфейс програми наведений на рисунку 2.1.

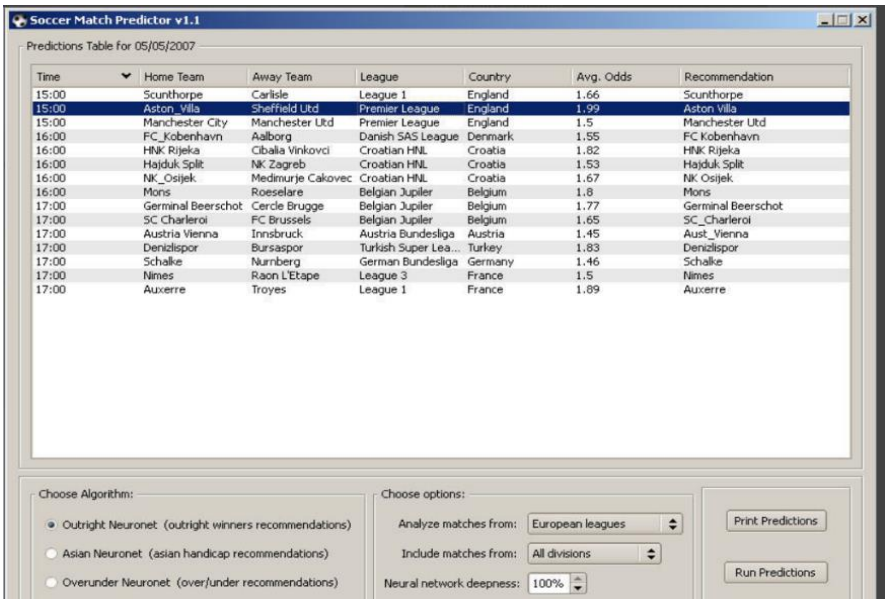


Рисунок 2.1 – Soccer Match Predictor [3]

Очевидно, що інтерфейс виглядає застаріло, хоча й інтуїтивно зрозумілим. Крім цього, станом на 2022 рік, програма працює по застарілій схемі: Soccer Match Predictor має інформаційну базу, яка складається з загальновідомих характеристик: дані про команду, гравців, рейтинг команди, забиті та пропущені голи тощо.

Проте, це ще півбіди, адже розробники займаються відвертим шахрайством: вони заявляють про те, що програма спроможна відрізняти так звані «договірні» матчі – ігри, в яких результат відомий заздалегідь. Існує купа доказів, що жодним чином аналітично не можливо визначити подібний поєдинок. Вищезазначене явище дійсно існує в спорті, але інформація про такі події зберігається в строгому секреті, також в переважній більшості країн світу для боротьби зі спортивними злочинами залучені спеціальні комітети, а також існують міжнародні організації, які мають повноваження відсторонити недоброчесних спортсменів від змагань на офіційному рівні. Саме через ці причини інформація про «договірні» матчі не доступна для звичайної людини, а самі такі поєдинки проходять на невисокому рівні та без наявності відеотрансляцій.

Окрім цього, програма є дуже дорогою, її ціна – 100 доларів. Виробники заявляють, що 95% відсотків прогнозів будуть вірними, але такі цифри не можливі.

Програма працює в автоматичному режимі: щоденно опівночі по британському часу вона оновлює прогнози на наступний день. Такий формат є доволі незручним.

Враховуючи все вищезазначене, не дивно, що справжні відгуки про програму виключно негативні. На незалежному порталі рейтинг програми становить 1.74/10 – критично низький. Розробники займаються відвертим шахрайством та оманюють користувачів.

2.2 Oncourt

«OnCourt – це програма для всіх любителів великого тенісу. База даних програми містить результати понад 1.5 мільйона тенісних матчів, зіграних найкращими тенісистами та тенісистками планети, починаючи з 1990 року. За допомогою цієї програми можна отримати безліч статистичної інформації про будь-якого гравця, турніру, історію особистих зустрічей тенісистів або окремий матч» [4] – саме такий опис фігурує на офіційному сайті розробника.

Як і попередній представник, програма виглядає застаріло, але візуально корисної інформації дійсно багато. Інтерфейс голосного меню представлено на рисунку 2.2, а на рисунку 2.3 представлений інтерфейс вибору вікна тенісисту.

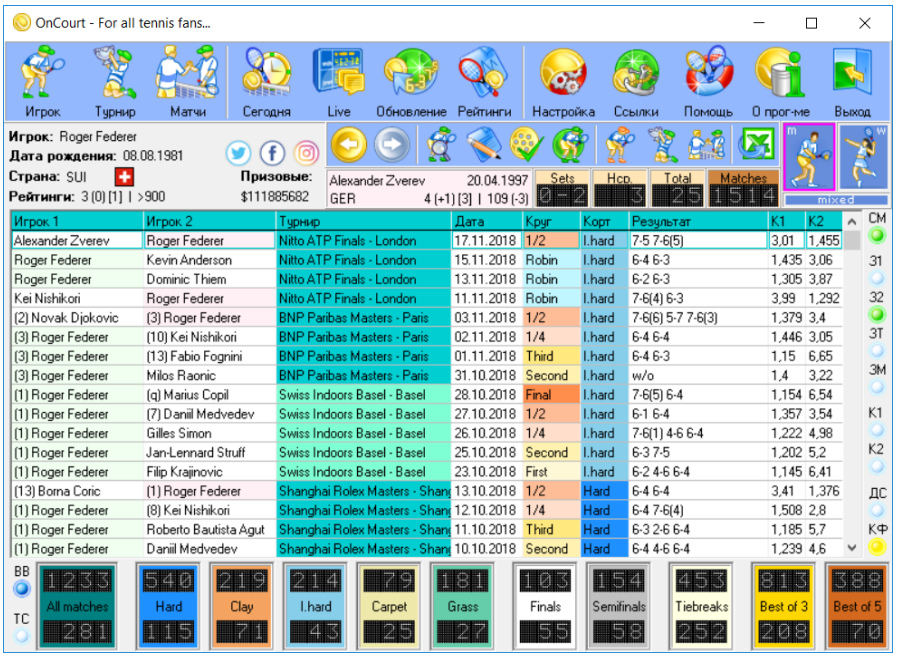


Рисунок 2.2 – Програма OnCourt – головне меню [5]

Розробник дає користуватися програмою безкоштовно напротязі 15-ти днів після загрузки, отже користувач має можливість на власному досвіді оцінити якість аналізу. Окрім того, дана програма – одна з небагатьох, яка не позиціонує себе, як прогнозист в ставках на спорт, вона нічого не гарантує, лише надає багато

необхідної для аналізу інформації. Отже з цього боку позиція виробника доброчесна.

У меню зверху можна вибрати одну з кількох вкладок. Тут зібрано матчі у прямому ефірі, список ігор за весь день, актуальні рейтинги тенісисток та тенісистів, а також історія виступів кожного з них на тривалому відрізку часу. Є дані щодо майже всіх ігор діючих професіональних спортсменів від початку їх кар'єри [6].

Крім опції перегляду інформації по конкретному тенісисту (рисунок 2.3) та історії його виступів, або очних протистоянь з іншим спортсменом, також пропонується можливість ознайомитися з математичною ймовірністю перемоги кожного з них у майбутньому поєдинку. Програма вказує ймовірність за різними параметрами, а в кінці видає загальну ймовірність результату матчу.

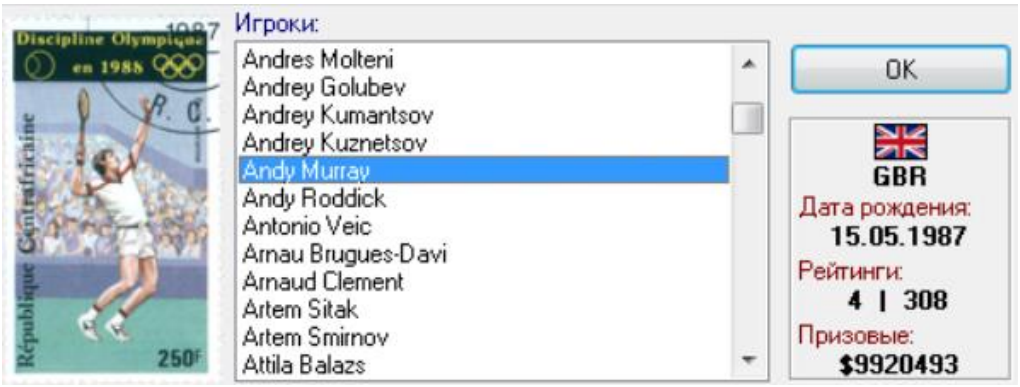


Рисунок 2.3 – Програма OnCourt – віконце вибору спортсмену [7]

Ознайомившись з відгуками на незалежних джерелах зроблено висновок, що програма якісна, містить правдиву та корисну інформацію та дійсно доволі точно розраховує ймовірність перемоги тенісисту в найближчих поєдинках.

Недоліками програми є застарілий інтерфейс та доволі висока ціна на повну версію програми. Окрім того, в нинішній час в режимі онлайн легко знайти ресурси, які містять аналогічну базу даних. Проте огляд програми ведеться з точки зору якості прогнозування спортивної події, тому ці недоліки не є важливими.

2.3 РОБОБЕТ

Це система автоматичного розрахунку переможця на підставі особистих зустрічей, останніх зустрічей, стану команд тощо. Факторів багато. При цьому в системі не враховуються коефіцієнти букмекерських контор [8].

На відміну від попередників дана система знаходиться в інтернеті і є повністю безкоштовною, що одразу заноситься в актив для неї.

Веб-сторінка РОБОБЕТ зображена на рисунку 2.4.

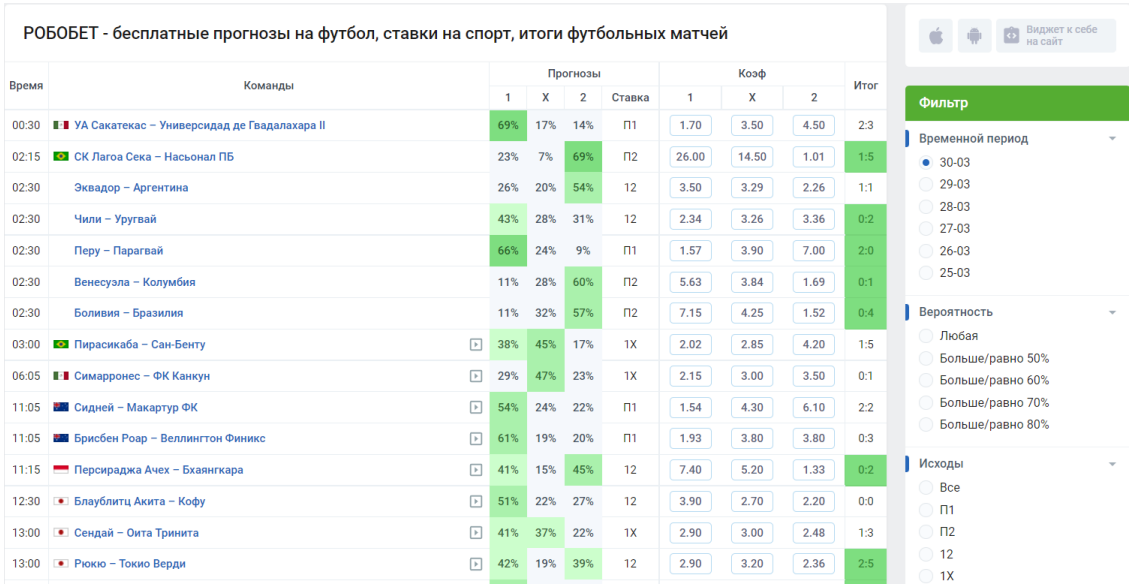


Рисунок 2.4 – РОБОБЕТ

Вона сучасна, мінімалістична, немає нічого зайвого. Розраховану ймовірність одразу можна порівняти з ймовірністю, яку дає букмекерська контора на різноманітні сценарії розвитку події, все необхідне для цього знаходиться в межах однієї веб-сторінки. Також сервіс оновлює результати футбольних матчів і чітко дає зрозуміти, чи вірний був прогноз. В цьому контексті сервіс доброчесний.

Але найголовніше – якість оцінювання ймовірностей дуже слабка. Цей штучний інтелект постійно називає нічийний результат найбільш імовірним, що в реальному спорті зустрічається тільки тоді, коли обидві команди влаштує такий

результат, а це дуже рідкісне явище. В іншому випадку, навіть за умови того, що колективи мають однакову силу, більш імовірна перемога однієї та перемога другої команди.

Окрім цього, сервіс бере відповідальність вказати користувачу найкращу ставку і робить це абсолютно неправильно. На рисунку 2.4 зображено, що сервіс оцінює перемогу бразильської команди з ймовірністю 69% і запевняє, що саме це є найкращим вибором для ставки. Але коефіцієнт на перемогу цієї команди – 1.01, тобто сума, яка буде поставлена на дану подію у випадку виграшу ставки помножиться на 1.01. Очевидно, що можливий виграш мізерний, і пропонована ймовірність надто мала, щоб вважати цю пропозицію хорошою порадою.

Відгуки на незалежних ресурсах в переважній більшості негативні, і враховуючи вищезазначені аргументи, зроблено висновок, що прогнозування, яке здійснює сервіс, не є точним.

2.4 Професійні моделі, які формують коефіцієнти в букмекерських конторах

Саме моделі букмекерських контор будуть слугувати орієнтиром для оцінювання якості розроблюваної системи. В цих установах наявна вся доступна інформація про майбутній поєдинок. Аналіз включає в себе сотні характеристик, які не є доступними для зазначених раніше сервісів.

Як тільки важливий гравець команди отримує ушкодження, коефіцієнт на перемогу цього колективу зростає. Тільки-но в пресі з’являється інформація, що команда може підписати контракт з сильним гравцем, коефіцієнт на перемогу цієї команди в найближчому матчі трохи зменшується, а якщо команда дійсно підпише цього гравця – впаде ще нижче. У випадку, коли прогноз букмекера дає збій, гравці бачать вигідну подію і починають масово ставити гроші на якийсь результат, система перерозподіляє коефіцієнти, а отже, й ймовірності, таким чином, щоб дана подія була вже менш привабливою для гравців. Нижче наведено приклад зміни коефіцієнтів в одній з контор (рисунок 2.5).

Рисунок 2.5 – Зміни коефіцієнтів в одному матчі

Цифри кажуть самі за себе: найпопулярніші букмекерські контори заробляють сотні мільйонів та мільярди доларів. При цьому переважна більшість таких установ легальна й відносно чесна, тому прибуток складається лише з програних гравцями ставок. Для цього штучний інтелект враховує величезну кількість факторів, формує ймовірності, переводить їх у коефіцієнти, трохи знижує їх і оновлює значення на сервері – саме таким чином побудований один з найбільш прибуткових бізнесів двадцять першого століття.

Висновки до розділу 2

В даному розділі були сформовані призначення та галузі використання систем, які прогнозують результат спортивних подій на основі зібраної статистики. Окрім цього, були розглянуті найпопулярніші сервіси, які надають послугу прогнозування спортивної події.

Встановлено, що найкращі системи прогнозування належать букмекерським конторам, діяльність яких нерозривно пов’язана з ними. Ті ж аналоги, які були розроблені з метою перевершити систему букмекера в переважній більшості випадків виявились малоефективними або ж застарілими.

3 РОЗРОБЛЕННЯ СХЕМ ПРОГРАМНОГО ДОДАТКУ

3.1 Схема компонентів системи і діаграми розгортання

Для розроблюваної системи ключовим інструментом буде навчена модель машинного навчання, яка й буде формувати ймовірності того чи іншого результату футбольної події. Навчання такої моделі буде відведено в окремий скрипт, який створюватиме її за фактом свого виконання.

Разом з наявністю моделі необхідним є збір актуальних статистичних даних для використання їх на вхід системи. Необхідними даними володіє web-ресурс understat.com, який надає повноцінну та багатогранну інформацію про зіграні поєдинки. Для використання цього ресурсу необхідним є VPN (Virtual Private Network) – віртуальна приватна мережа, що забезпечує захищену мережу або захищений зв’язок за вже наявного інтернет-з’єднання [9]. За наявністю активованого VPN-сервісу, спеціальний скрипт буде проводити парсинг даних – автоматичний збір інформації з веб-сторінки, та її структурування.

Зібрана інформація буде наявна вже безпосередньо на локальному пристрої та буде зберігатися в реляційній системі управління базами даних MySQL, за це теж відповідатиме окремий скрипт.

Для розробки програми, реалізації її графічного інтерфейсу та компонування всіх окремих структур у єдину буде використано PyCharm – середовище розробки на мові програмування Python. Саме завдяки даному продукту стане можливим об'єднати модель машинного навчання, запити до бази даних в режимі реального часу та користувацький інтерфейс у єдиний проєкт, який і буде програмою, що прогнозує результат футбольної події. Користувачеві залишається лише запустити програму і обрати футбольний матч для оцінки ймовірностей.

Кінцева схема зображена на рисунку 3.1.

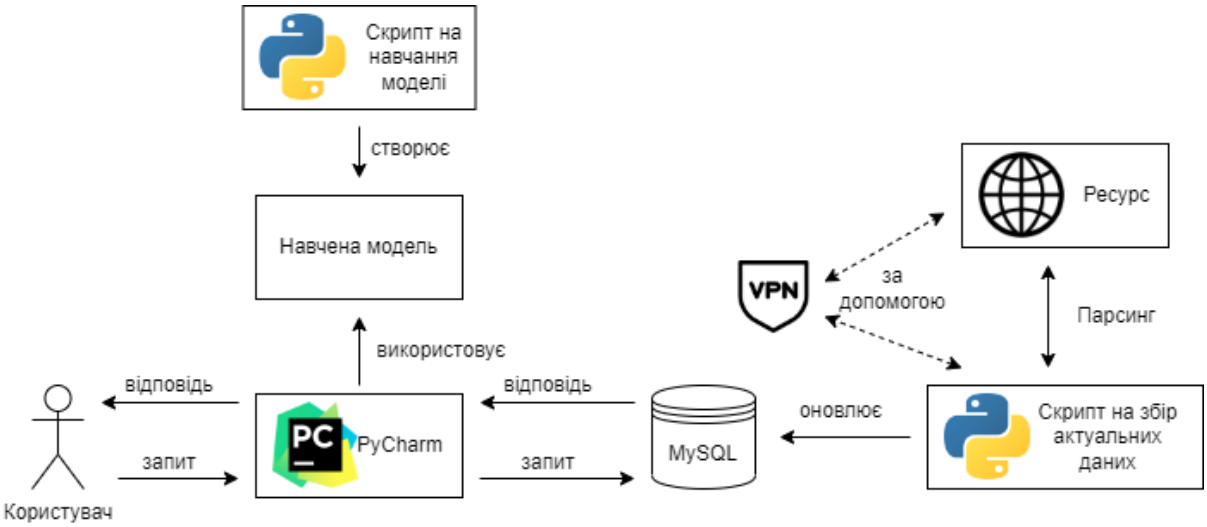


Рисунок 3.1 – Схема компонентів системи

Також згідно з описаним алгоритмом роботи системи була розроблена відповідна діаграма розгортання, яка схематично демонструє апаратні, програмні компоненти системи, а також взаємозв'язок між ними та об'єктами даних компонентів. Вона наведена в кресленнях до індивідуального дослідницького проєкту за шифром ІА.81.090БАК.003 Д1.

Дана схема є схемою UML. Універсальна мова моделювання (Unified Modelling Language або UML) — це мова позначень або побудови діаграм, призначена для визначення, візуалізації і документування моделей зорієнтованих на об'єкти систем програмного забезпечення [10].

3.2 Схема додатку

Розроблювана програма буде мати максимально простий да інтуїтивно зрозумілий для користувача функціонал. Для початку користувачу потрібно буде обрати футбольну лігу, для якої він бажає оцінити ймовірності. Після цього вміст «випадаючих» меню буде заповнено командами ліги. В лівому меню потрібно буде обрати колектив, який гратиме на своєму полі, а в правому – команду-гостя. Під час обрання команди зі списку поруч з ними буде виводитися емблема обраного колективу.

Коли обидві команди обрані, нейронна мережа автоматично розраховуватиме ймовірність перемоги господарів поля, нічиєї та гостьового колективу й виводитиме їх на екран. Нижче на рисунку 3.2 представлений орієнтовний інтерфейс додатку.

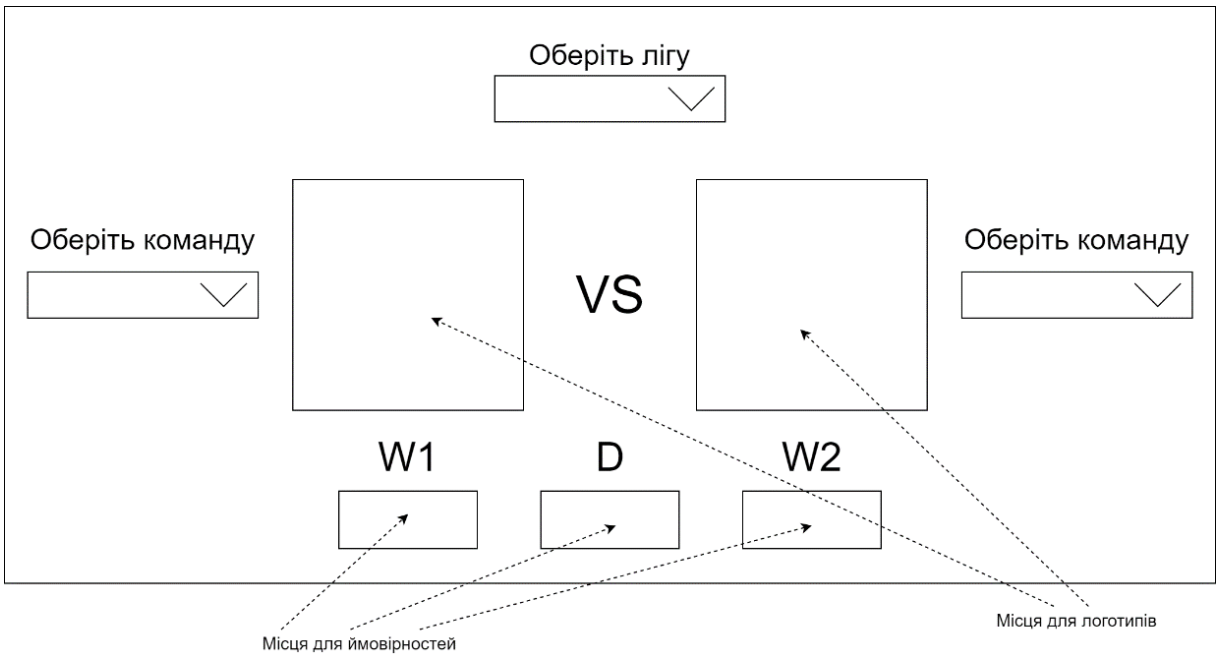


Рисунок 3.2 – Схема додатку

Основною перевагою даного інтерфейсу є його простота: всі функції знаходяться в межах одного віконця, а розрахування ймовірностей не потребує жодного додаткового натискання на кнопки або чогось іншого, що буде ускладнювати процес. На екрані у користувача буде лише чемпіонат, назви команд, ймовірності результатів події та емблем колективів – нічого зайвого.

3.3 Діаграма послідовності

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень [11].

Дана діаграма допоможе чітко зрозуміти обіг подій під час користування додатком. Для справного користування застосунком необхідна заздалегідь заповнена актуальними статистичними даними база даних з таблицею, а також навчена модель машинного навчання. За таких обставин програма буде повністю готова до застосування і перебіг її роботи буде виглядати наступним чином:

– користувач запускає програму. Під час завантаження програми, за допомогою спеціальної python-бібліотеки відбувається підключення до локального серверу MySQL, в якому знаходиться база даних з таблицею, що містить необхідні нам показники. Дане підключення дозволяє робити запити безпосередньо з застосунку до бази даних, яка буде повертати результат у вигляді python-об’єкту;

– після завантаження програми, як вже було зазначено у попередньому розділі, користувач обирає спочатку лігу, а потім команду, що є господарем поля, та команду-гостя. Кожен з цих кроків буде супроводжуватись запитом до бази даних через попередньо установлене підключення. Під час вибору команд в програмі з’являтиметься новий об’єкт, що міститиме характеристики відповідної команди;

– тоді, коли обидві команди будуть обрані, об’єкти з характеристиками перетворяться на список з характеристиками цих двох команд, і даний список піде на вхід до навченої моделі машинного навчання, яка видаватиме розраховані ймовірності;

– розраховані ймовірності передаються в текстові об’єкти, які будуть виведені на екран у відповідному для них місці;

– користувач закриває програму. Паралельно з закриттям програми відбувається завершення підключення до локального серверу MySQL.

Спираючись на запланований хід роботи була побудована відповідна діаграма послідовності, яка містить коротко описаний цикл життя програми. Вона зображена на кресленику ІА.81.090БАК.003 Д2.

На даній діаграмі послідовності наявні наступні елементи UML-нотації, які застосовуються для даного типу діаграм:

– користувач (суб’єкт) – у вигляді накресленої людини;

– об’єкти взаємодії – у вигляді прямокутників;

– вертикальні тонкі лінії – умовні проміжки часу;

– прямокутники на вертикальних лініях під об’єктами та суб’єктами, які позначають «час життя» відповідної сутності, тобто умовний проміжок часу, який об’єкт/суб’єкт перебуває в активному стані;

– суцільні стрілки, які означають виклик одним об’єктом/суб’єктом процедури іншого об’єкта суб’єкта;

– «штриховані» стрілки, які означають повідомлення з результатом. Зазвичай знаходяться після суцільних стрілок в зворотному напрямку;

– цикл – прямокутний фрагмент, що виділяє на схемі проміжок, який може повторюватись.

3.4 Діаграма прецедентів

Для того щоб представити взаємозв’язок акторів та прецедентів (варіантів використання) була побудована діаграма прецедентів. Акторами в даній діаграмі виступають об’єкти системи, не обов’язково тільки суб’єкти.

Дана схема дає представлення про функціональність системи та демонструє її складові та взаємозв’язок. Також дозволяє чітко зрозуміти, в якому порядку необхідно проводити розробку.

Діаграма прецедентів наведена на кресленику ІА.81.090БАК.003 ДЗ .

Надпис <<include>> над стрілками в даній діаграмі означає, що прецедент, від якого йде стрілка використовує прецедент до якого вона проведена, а <<extend>> означає якими варіантами використання може бути розширений прецедент до якого проведено стрілку.

3.5 Структурна схема

Структурна схема – це концептуальний інструмент моделювання, який використовується для документування різних структур, що складають таку систему, як база даних або додаток. Він показує ієрархію або структуру різних компонентів або модулів системи та показує, як вони з'єднуються та взаємодіють між собою [12].

Структурна схема зображена на кресленику ІА.81.090БАК.003 Э1.

Ланки на схемі зображують у вигляді прямокутників або умовних графічних позначень, які з'єднуються лініями взаємозв'язку. Ці лінії варто позначати стрілками для зазначення напрямку ходу процесів між ланками [13].

Висновки до розділу 3

В даному розділі були описані складові розроблюваної системи та їхній взаємозв’язок, за результатом яких була побудована відповідна схема компонентів розроблюваної системи. Окрім цього, після проведення аналізу складових та побудови структурної схеми, стали чітко окреслені та зрозумілі підсистеми, що дозволяє розбити основну задачу розробки на декілька менших, виконавши кожну з яких, буде розроблено заплановану систему.

Було прийняте рішення про зовнішній вигляд майбутньої програми Приблизний її схематичний вигляд був зображений на відповідній схемі. При його розробці головним фактором була простота інтерфейсу та мінімалізм.

Спираючись на запланований перебіг подій під час користування застосунком була накреслена діаграма послідовності, яка демонструє взаємодію між складовими системи під час її роботи.

4 ПІДГОТОВКА ДО РЕАЛІЗАЦІЇ СИСТЕМИ

Прогнозування результатів спортивних подій на основі зібраної статистики буде здійснено за допомогою різних факторів та характеристик, що впливають на ймовірність перемоги тієї чи іншої команди. Перед ще не зіграною грою аналітики, системи прогнозування, та й в принципі людина, яка на певному рівні розбирається в якомусь виді спорту можуть приблизно оцінити яка команда чи спортсмен з більшою ймовірністю переможе. Але якщо спрогнозувати результат намагається звичайна людина, то вона може це зробити це виключно приблизно, включивши в свій аналіз обмежену кількість інформації – тільки ту, яка цій особі відома. Основними перевагами розроблюваної системи є:

– наявність широкого спектру характеристик в аналізі;

– відсутність упередженості;

– можливість розрахувати специфічні характеристики математичним шляхом та методами машинного навчання.

Окрім розрахунку характеристик, методами машинного навчання буде реалізоване безпосереднє формування ймовірностей, отже саме дане явище є основним інструментом в розробці системи прогнозування.

Машинне навчання – один з методів функціонування штучного інтелекту, а саме – практичної реалізації його можливостей шляхом створення алгоритмів для виявлення закономірностей під час аналізу великих даних, та їх подальше використання для самонавчання [14].

Даний метод буває двох типів: з учителем, та без нього. Для реалізації заданої системи буде використаний перший тип, який означає, що для комп’ютера наявні приклади сукупностей значень вхідних характеристик та вихідного значення – безпосередньо вчителя.

Основна мета системи, яка навчається, — це робити узагальнення зі свого досвіду. Узагальнення в цьому контексті є здатністю машини, яка вчиться,

працювати точно на нових, не бачених прикладах/задачах після отримання досвіду навчального набору даних [15].

В ході розробки системи прогнозування методами машинного навчання будуть вирішені наступні задачі:

– задача класифікації – виявлення категорії об’єкта спираючись на його характеристики;

– поновлення регресії – прогнозування числової величини за характеристиками.

4.1 Алгоритми машинного навчання, необхідні для розробки системи.

Задача класифікації, а саме визначення прогнозування результату футбольного матчу, може бути вирішена шляхом застосування логістичної регресії або за допомогою нейронної мережі.

4.1.1 Логістична регресія

Логістична регресія – це алгоритм машинного навчання, який застосовується у випадку, коли вихідна величина є номінативною, тобто такою, що означає одну з груп, на які можна розділити відповіді на наявну проблему.

Прикладами застосування даного алгоритму є задачі класифікації, наприклад:

– за наявними в електронному листі словами зробити висновок, чи є даний лист спамом (1 – так, 0 – ні);

– за показниками аналізу крові оцінити групу крові людини (1 – перша, 2 – друга, 3 – третя, 4 – четверта).

Визначення найбільш ймовірного переможця поєдинку також є задачею класифікації, адже результат футбольного матчу можна розбити на три номінативних змінних, які позначають відповідно перемогу першої команди,

нічийний результат, та перемогу другої команди. Окрім того, що логістична регресія здатна визначати найбільш ймовірний результат, вона оцінює ймовірності, що кожна з номінативних змінних є відповіддю на питання за заданих характеристик – це і будуть необхідні для системи прогнозування ймовірності.

Для кожної з характеристик оцінюваної проблеми наявний коефіцієнт, який впливає на ймовірність підсумкового результату. Наприклад, наявність слова «Дешево» в листі збільшує ймовірність того, що лист є спамом.

Використання логістичної регресії для задачі бінарної класифікації (лише два класи) проходить за алгоритмом описаним нижче.

Введемо наступні змінні:

– множина характеристик для конкретного експерименту x = {x0 = 1, x1, x2, …, xn};

– множина коефіцієнтів θ = {θ0, θ1, θ2, …, θn};

– множина можливих відповідей y = {0, 1}.

Тоді:

$$y^* = x^T * \theta = \sum_{i=0}^{n} x_i * \theta_i \quad (4.1)$$

За формулою 4.1 отримано значення, що означає суму всіх добутків характеристик на відповідні коефіцієнти. Оскільки область значень y^* жодним чином не обмежена, для того, щоб оцінити ймовірність отримати номінативний 0 (зазвичай – негативний результат експерименту) та номінативну 1 (зазвичай – позитивний результат експерименту) треба «загнати» отримане значення в рамки від 0 до 1.

Для цього в логістичній регресії зазвичай застосовується функція сигмоїда (рисунок 4.1). Ймовірність того, що буде отримане значення номінативної одиниці розраховується за формулою 4.2.

$$P(y = 1|x; \theta) = sig(t) = \frac{1}{1+e^{-t}} \quad (4.2)$$

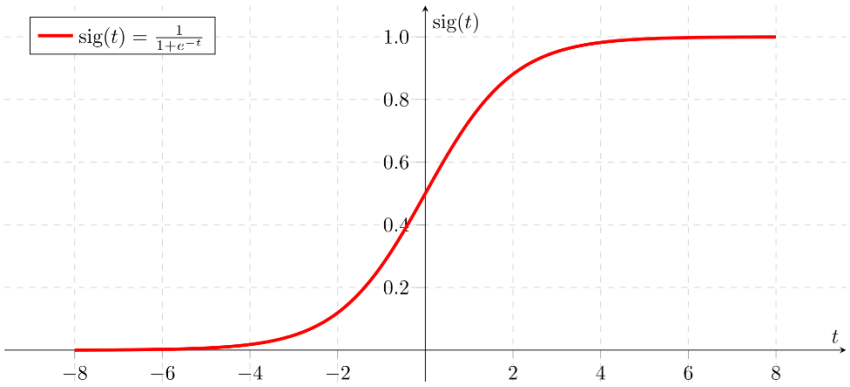


Рисунок 4.1 – Сигмоїда [16]

Область значень даної функції – (0; 1). Таким чином, після застосування даної функції на виході отримується число в проміжку від 0 до 1, яке оцінює ймовірність отримати номінативну одиницю (позитивний результат).

Враховуючи особливість даної функції зроблено висновок, що логістична регресія прогнозує позитивний результат у випадку, коли сума добутків характеристик на відповідні коефіцієнти додатна, і, як наслідок, ймовірність перевершує 0.5. В інакшому випадку більш ймовірним вважатиметься негативний результат (формула 4.3).

$$Y = \begin{cases} 1, & sig(y^*) \geq 0.5 \\ 0, & sig(y^*) < 0.5 \end{cases} \quad (4.3)$$

Для застосування множинної лінійної регресії, призначеної для задач класифікації при трьох та більшій кількості класів на виході застосовується аналогічний алгоритм n разів, де n – кількість номінативних змінних.

Кожен з цих експериментів проводиться за наступним алгоритмом: За позитивний результат (одиницю) береться чергова номінативна змінна, а за

негативний – сукупність всіх інших номінативних змінних. Тобто для того, щоб оцінити перемогу першої команди в матчі треба оцінити ймовірність

отримати 1 на виході логістичної регресії, де 0 на виході буде змінна, яка означає нічийний результат та перемогу другої команди разом. Після цього оцінюється ймовірність нічиєї, при негативному результаті, який є сукупністю перемог першої та другої команди, і, аналогічним чином оцінюється ймовірність перемоги другої команди.

На відміну від бінарної логістичної регресії, нема жодних гарантій, що ймовірність отримати один з класів буде перевершувати 0.5. Прогнозованим значенням у випадку застосування такого типу регресії буде значення з найбільшою ймовірністю порівняно з іншими (формула 4.4).

$$Y = \max_i(P(y = \mathrm{i}|x;\theta)) \tag{4.4}$$

де I = (1, 2, 3, … , n) – числове значення класу, n – кількість класів а виході.

4.1.2 Штучна нейронна мережа

Іншою моделлю, яка спроможна вирішувати завдання класифікації є штучна нейронна мережа (ШНМ). Концепція нейронних мереж базується на функціональних особливостях головного мозку.

Є певна кількість нейронів, які між собою пов'язані та взаємодіють один з одним шляхом передачі сигналів. Також є рецептори, які отримують інформацію, що надходить ззовні, та виконавчий орган, на який надходить підсумковий сигнал. За подібним принципом працюють штучні нейромережі: є кілька шарів з нейронами та зв'язку між ними (кожний зв'язок має свій ваговий коефіцієнт). По зв'язках передаються сигнали як чисельних значень, перший шар виконує собою роль рецепторів, тобто отримує набір ознак навчання, і є вихідний шар, який видає відповідь [17].

Штучні нейронні мережі застосовуються не тільки в задачах класифікації, а й в задачах регресії та кластеризації.

На відміну від логістичної регресії, для якої необхідний єдиний вектор вагових коефіцієнтів, в нейронних мережах застосовуються матриці вагів, при чому їхня кількість залежить від кількості шарів, а розмір – від кількості нейронів в обраному та наступному шарі.

На рисунку 4.2 зображено доволі просту ШНМ. Зеленим кольором зображено вхідний шар, який містить 2 нейрони, синім – скритий шар, жовтим – вихідний шар.

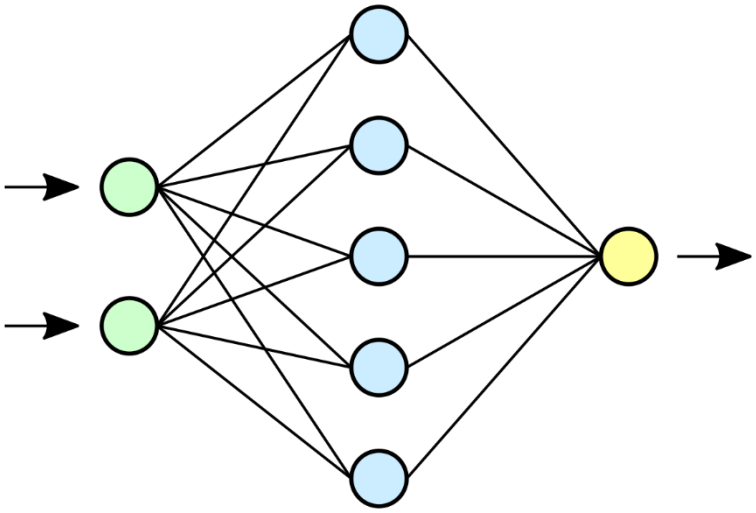


Рисунок 4.2 – Проста штучна нейронна мережа [18]

Класична нейронна мережа для задач класифікації працює наступним чином: спочатку значення на вході перемножаються на матрицю вагів, в результаті чого отримується вектор, розмір якого дорівнює розміру другого шару. Після цього, для значень цього вектора застосовується функція активації: нею може слугувати раніше розглянута функція сигмоїда, але більш популярними в скритих шарах є інші функції активації, такі як ReLU або тангенс гіперболічний. Далі алгоритм від шару до шару аналогічний, поки врешті не буде розрахований вихідний нейрон (або нейрони). Обрана для вихідного шару функція активації (часто – сигмоїда) розраховує значення ймовірності.

Формула розрахунку значень нейронів для кожного наступного шару виглядатиме таким чином (формула 4.5).

$$a_{1\,x\,n}^{(k+1)} = g(a_{1\,x\,m}^{(k)} * \theta_{m\,x\,n}^{(k)}) \quad (4.5)$$

де k = (1, 2, …, p-1), $a^{(t)}$ – вектор нейронів на t-му шарі, k – номер шару, g(t) – функція активації, $\theta^{(t)}$ – матриця вагів, p – кількість шарів.

Для останнього шару за наявності єдиного нейрона на виході отримується єдине значення $a^{(p)}$, а у випадку, коли на виході декілька – $a_{1\,x\,n}^{(p)}$, де n – це кількість нейронів на виході.

Для задач множинної класифікації застосовується як раз таки ШНМ з трьома та більше виходами. На відміну від логістичної регресії в даному випадку нема сенсу проводити експеримент декілька разів, адже на виході одержується вектор розміром 1 x n, зі значеннями ймовірності отримати кожен з наявних класів. Вихідне значення розраховується за формулою (4.6).

$$Y = \max(a_{1\,x\,n}^{(p)}) \quad (4.6)$$

4.1.3 Лінійна регресія

Лінійна регресія – це алгоритм машинного навчання, який застосовується у випадку, коли вихідна величина є кількісною, тобто такою, що означає якесь конкретне числове значення.

Прикладами застосування цього алгоритму є задачі прогнозування, наприклад, прогнозування значення ціни дому.

Для розроблюваної системи даний алгоритм буде застосований для обчислення характеристики, що буде означати форму команд за останні матчі.

З математичної точки зору лінійна регресія схожа на логістичну. В ній так само кожній характеристиці відповідає власна вага, але після одержання добутку значень характеристик на відповідні коефіцієнти ніяких додаткових функцій не застосовується (формула 4.7).

$$Y = x^T * \theta = \sum_{i=0}^{n} x_i * \theta_i \quad (4.7)$$

де x – множина коефіцієнтів, θ – множина вагів.

Оскільки прийнято брати $x_0 = 1$, то для двох характеристик вихідне значення буде $Y = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2$. Приклад такої ситуації зображено графічно (рисунок 4.3):

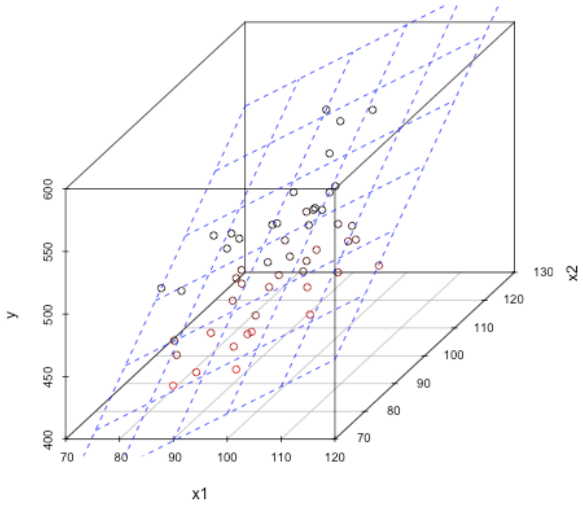


Рисунок 4.3 – Лінійна регресія з двома незалежними змінними [19]

Синя сітка на графіку зображає множину рішень $Y = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2$. Лінійну регресію з трьома та ще більшою кількістю характеристик зобразити неможливо.

Поряд з лінійною регресією обов'язково варто розглянути поліноміальну регресію. В моделі такого типу вихідне значення залежить не лише від кожної

характеристики окремо, а й від комбінацій характеристик, характеристик в степенях тощо.

На рисунку 4.4 приведений приклад ситуації, коли залежність змінної Y від єдиної незалежної змінної X нелінійна. В такому випадку крива, яка прогнозує значення буде обчислена за формулою, яка включає значення незалежної змінної в квадраті. Очевидно, що модель справа на рисунку показуватиме кращу якість для обраної ситуації.

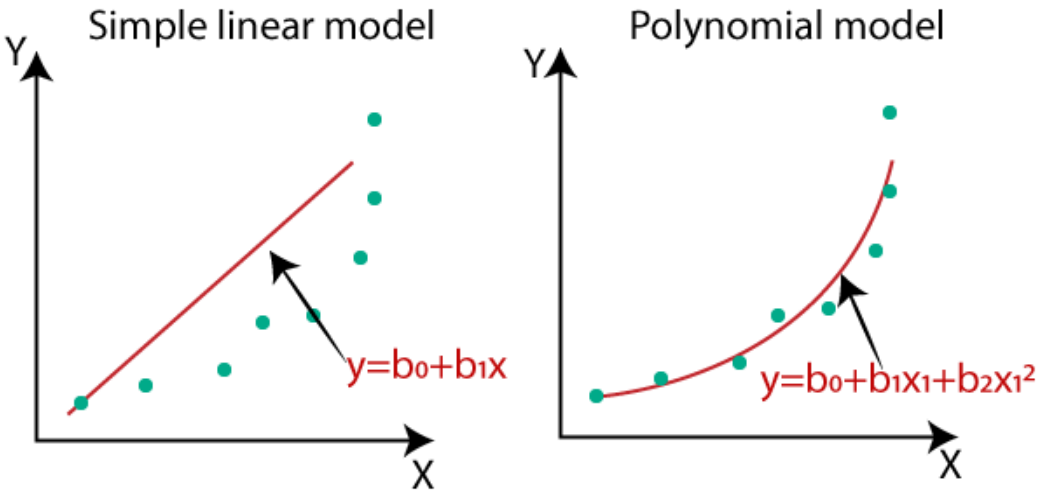


Рисунок 4.4 – Порівняння лінійної регресії та поліноміальної [20]

4.1.4 Ваги

В усіх вищезазначених алгоритмах були використані ваги – коефіцієнти, які впливають на кінцевий результат. За умови, що конкретна модель спроможна вирішити необхідне завдання, задача створення якісної моделі зводиться до задачі підбору коефіцієнтів.

Безпосередньо підбір правильних коефіцієнтів – це те, що називається навчанням моделі. Кожен крок після застосування алгоритму навчання вираховується функція втрат, яка означає якість нинішнього стану моделі.

Алгоритми навчання, так само як і функції втрат бувають різними, але головною метою їх застосування є якраз знаходження найбільш оптимальних коефіцієнтів моделі завдяки обчислювальній потужності комп’ютера.

4.2 Засоби та технології розробки

Задля досягнення поставленої мети – побудови системи прогнозування результатів спортивної події, необхідно вміти працювати з наступними засобами та технологіями.

4.2.1 Мова програмування Python

Для роботи з даними та алгоритмами машинного навчання щонайкраще підходить найбільш популярна для даних цілей мова програмування Python. Це інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом [21].

Головною її перевагою в порівнянні з іншими мовами програмування є спеціалізація на вирішенні задач, пов’язаних з роботою з даними, для чого використовуються бібліотеки numpy та pandas. Для машинного навчання наявна велика кількість бібліотек, яка дозволить обрати ледве не будь-яку модель з широким вибором алгоритмів навчання та конфігурацій. Окрім того, дана мова програмування містить й бібліотеки для парсингу, зв'язку з РСУБД, які також будуть необхідні для розроблюваної системи.

4.2.2 SQL

SQL (Structured Query Language) - мова структурованих запитів, яка була розроблена для роботи з БД, щоб отримувати/додавати/змінювати дані, мати можливість опрацьовувати великі масиви інформації та швидко отримувати

структуровану та згруповану інформацію. Існує багато СУБД, основними є: Microsoft Access, Microsoft SQL Server, MySQL, Oracle SQL, IBM DB2 SQL, PostgreSQL, SQLite [22].

Для реалізації системи прогнозування результатів спортивних подій мову SQL буде застосовано для написання запитів з додатку, які будуть змінюватись залежно від обраної користувачем ліги та команди, що дозволить утворити динамічність в системі та забезпечити чітко структуроване зберігання даних.

4.2.3 Jupyter Notebook

Jupyter Notebook – інтерактивне середовище розробки на мові програмування Python. Саме інтерактивність є головним козирем даної технології.

Головна відмінність від традиційних інструментів розробки – можливість розбити код на частини та виконувати їх окремо. Наприклад, програміст може написати одну функцію і одразу перевірити, як вона працює, не запускаючи решту фрагментів коду. Можна змінювати порядок виконання коду, або не виконувати його певну частину. Завдяки цьому Jupyter ноутбук дуже популярний в аналітиці даних та Data Science. Фахівці отримують попередні результати, будують графіки та іншу візуалізацію [23].

Для проходження усього шляху з нуля до розробки робочої моделі, та для написання скриптів, які необхідні для правильної роботи застосунку і

будуть збирати актуальні дані буде використано саме Jupyter Notebook, який

дозволить без зайвої витрати часу крок за кроком досягнути запланованого. Після цього, за потреби, код з цього середовища розробки легко конвертується в звичайний python-файл.

Саме в Jupyter Notebook буде проведена робота з даними, які будуть застосовані в подальшому при машинному навчанні.

4.2.4 PyCharm

PyCharm – це середовище розробки з повним набором засобів для ефективної розробки мовою Python. На відміну від зазначеного вище Jupyter Notebook не може змінювати порядок коду чи виконувати тільки певні його частини.

Натомість має цілу низку позитивних аспектів таких як:

– потужний і функціональний редактор коду з підсвічуванням синтаксису, авто-форматуванням та авто-відступами для мов, що підтримуються;

– проста та потужна навігація в коді;

– допомога при написанні коду, що включає автодоповнення, авто-імпорт, шаблони коду, перевірка на сумісність версії інтерпретатора мови тощо [24].

Окрім цього PyCharm дозволяє створити проєкт, який комбінуватиме всі файли та матеріали разом, а також застосувати інтерпретатор віртуального середовища, який дозволить встановити виключно необхідні бібліотеки, які залишаться в середині проєкту, таким чином забезпечуючи безпроблемний запуск застосунку на іншому комп’ютері.

Для реалізації системи PyCharm буде застосований вже безпосередньо при написанні додатку з графічним інтерфейсом.

4.2.5 MySQL

MySQL – це система управління реляційними базами даних із відкритим вихідним кодом (РСУБД) з моделлю клієнт-сервер.

РСУБД — це програмне забезпечення або служба, яка використовується для створення та керування базами даних на основі реляційної моделі. Комп'ютери, які встановлюють та запускають програмне забезпечення РСУБД, називаються

клієнтами. Коли їм необхідно отримати доступ до даних, вони підключаються до сервера РСУБД. Це система «клієнт-сервер» [25].

Як вже було зазначено раніше, MySQL є однією з декількох найбільш популярних систем управління базами даних. Жодної принциповості при виборі даної РСУБД серед інших не було.

4.2.6 Бібліотеки Python

– NumPy – це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом із великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій із цими масивами [26];

– pandas – програмна бібліотека мовою Python для маніпулювання даними та їхнього аналізу. Надає спеціальні структури даних та операції для маніпулювання числовими таблицями та тимчасовими рядами [27];

– Requests – це бібліотека HTTP для мови програмування Python. Мета проекту – зробити HTTP-запити простішими та зручнішими [28];

– Beautiful Soup – це пакет Python для аналізу документів HTML і XML. Він створює дерево аналізу для зчитаних сторінок, яке використовується для вилучення даних з HTML, що корисно для веб-парсингу [29];

– JSON (JavaScript Object Notation) – простий формат обміну даними, заснований на підмножині синтаксису JavaScript. Модуль json дозволяє кодувати та декодувати дані в зручному форматі;

– Scikit-learn (також відомий як sklearn) – це безкоштовна бібліотека машинного навчання для мови програмування Python. Вона містить різні алгоритми класифікації, регресії та кластеризації, і розроблена для взаємодії з числовими та науковими бібліотеками Python NumPy і SciPy [30];

– Joblib – це зовнішня бібліоека scikit-learn, яка дозволяє зберігати за завантажувати моделі машинного навчання;

– SQLAlchemy – це програмне забезпечення з відкритим кодом для роботи з базами даних за допомогою мови SQL. SQLAlchemy дозволяє описувати структури баз даних та способи взаємодії з ними прямо на мові Python [31];

– re – реалізує функціонал для роботи з регулярними виразами на мові Python;

– Tkinter – графічна бібліотека Python, яка призначена для створення програм із віконним інтерфейсом. Вона кросплатформена, тобто за її допомогою можна писати програми для Windows, Linux, MacOS [32];

– Math – модуль, який надає широкий математичний функціонал для роботи з числами [33].

Висновки до розділу 4

В даному розділі була проведена вся підготовча робота до реалізації системи прогнозування результатів спортивних подій на основі зібраної статистики.

Були розглянуті такі моделі машинного навчання, як логістична регресія, нейронна мережа та логістична регресія, які будуть реалізовані під час побудови системи, а саме описана суть їх роботи, визначені ситуації, в яких вони застосовуються, окреслені можливості кожної з них, наведене необхідне математичне підґрунтя для використання кожної з моделей.

Для програмної реалізації системи на мові Python були підібрані бібліотеки та модулі, які дозволять досягнути запланованої мети. Кожен з цих інструментів зробить свій внесок у розробку запланованої системи.

5 РОБОТА З ДАНИМИ

5.1 Підбір необхідних характеристик та джерела даних

Для того, щоб зрозуміти, як саме треба проводити навчання моделі, яка буде прогнозувати результат спортивної події необхідно зімітувати цю ситуацію.

Перед подією відома велика кількість характеристик, які в сукупності повинні дати оцінку ймовірностям перемоги домашньої та гостьової команди, а також нічиєї. Але для того, щоб натренувати модель якісно прогнозувати результат, необхідно провести навчання з вчителем. Оскільки метою розроблюваної системи є прогнозування, як саме завершиться матч та з якою імовірністю, то й вчителем повинна стати номінативна змінна, яка позначатиме один з трьох можливих вищезазначених результатів матчу.

Отже, дані для навчання повинні містити сукупність характеристик для домашньої та гостьової команди з якою вони підходили до поєдинку, а також результат їхньої зустрічі, який стане вчителем в моделі. Саме завдяки такому підходу, за умови достатньої кількості даних, модель в подальшому буде якісно натренована і буде здатна прогнозувати як завершиться подія. Тепер, коли необхідні вхідні характеристики для моделі чітко окреслені, необхідно приступити до пошуку даних, які задовільнять умові.

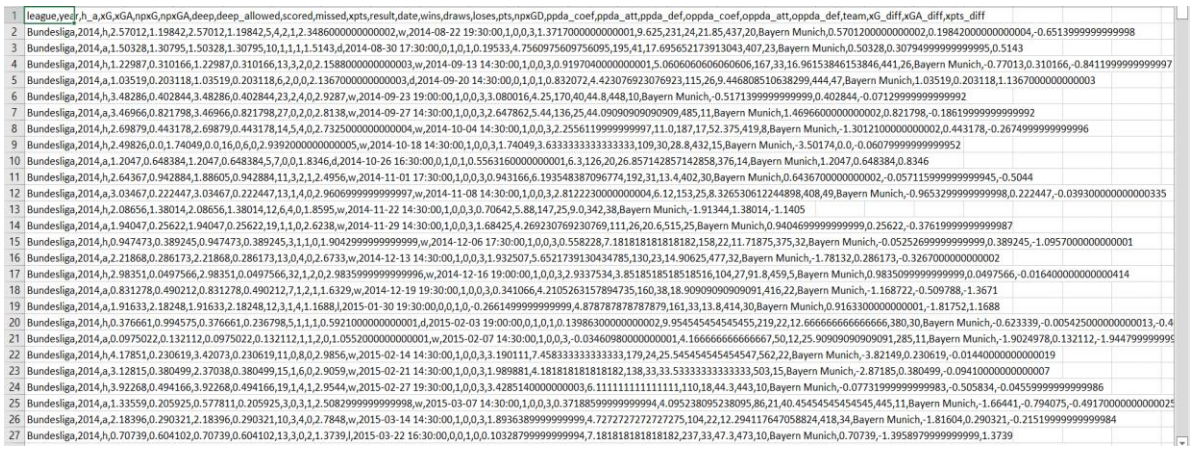
Найголовнішим показником серед аналітиків негласно вважається показник «xG» - очікувані голи. Даний показник означає на скільки забитих голів напрацював в конкретному матчі обраний колектив, тобто він відображає об'єктивну силу команди в матчі, а не просто враховує скільки команда забила голів. Цей показник вираховується уповноваженими спеціалістами з футбольної аналітики й означає суму ймовірностей результативності гольових моментів — математичну величину, яка розраховується з урахуванням параметрів потенційно гольових моментів, до яких відносяться удари по воротах та небезпечні моменти без завершальних ударів [34].

Можлива ситуація, коли команда повністю домінувала на полі, створила багато небезпеки біля воріт суперника, але врешті програла матч другій команді. І якщо в параметрах моделі включити виключно забиті голи, то така модель буде необ'єктивна, адже насправді краще грала перша команда. Саме тому показник xG є надважливим для розроблюваної системи.

Показник xG якісно розраховується ресурсом Understat, який спеціалізується на професійній футбольній статистиці. Він надає багато корисної інформації по найбільш популярним європейським чемпіонатам. Отже, було прийняте рішення використовувати інформацію саме з цього джерела.

Найбільшою платформою з даними для аналізу та для машинного навчання є сайт Kaggle. Саме там був знайдений датасет, що складається з даних ресурсу Understat з 2014 року й містить майже 22000 записів. Датасет (дослівно – набір даних) – це оброблена та структурована інформація у табличному вигляді [35].

Набір обраних даних представлений у форматі csv – comma-separated values (рисунок 5.1). В цьому форматі в першому рядку перелічені через кому назви колонок, а починаючи з другого і до кінця – записи, теж через кому, де кожне значення відповідає своїй колонці.



```
league,year,h_a,xG,xGA,npxG,npxGA,deep,deep_allowed,scored,missed,xpts,result,date,wins,draws,loses,pts,npxGD,ppda_coef,ppda_att,ppda_def,oppda_coef,oppda_att,oppda_def,team,xG_diff,xGA_diff,xpts_diff
Bundesliga,2014,h,2.57012,1.19842,2.57012,1.19842,5,4,2,1,2.3486000000000002,w,2014-08-22 19:30:00,1,0,0,3,1.3717000000000001,9.625,231,24,21.85,437,20,Bayern Munich,0.5701200000000002,0.1984200000000004,-0.6513999999999998
Bundesliga,2014,a,1.50328,1.30795,1.50328,1.30795,10,1,1,1,1.5143,d,2014-08-30 17:30:00,0,1,0,1,0.19533,4.7560975609756095,195,41,17.695652173913043,407,23,Bayern Munich,0.50328,0.3079499999999995,0.5143
Bundesliga,2014,h,1.22987,0.310166,1.22987,0.310166,13,3,2,0,2.1588000000000003,w,2014-09-13 14:30:00,1,0,0,3,0.9197040000000001,5.060606060606061,167,33,16.961538461538461,441,26,Bayern Munich,-0.77013,0.310166,-0.8411999999999997
Bundesliga,2014,a,1.03519,0.203118,1.03519,0.203118,6,2,0,0,2.1367000000000003,d,2014-09-20 14:30:00,0,1,0,1,0.832072,4.423076923076923,115,26,9.446808510638299,444,47,Bayern Munich,1.03519,0.203118,1.1367000000000003
Bundesliga,2014,h,3.48286,0.402844,3.48286,0.402844,23,2,4,0,2.9287,w,2014-09-23 19:00:00,1,0,0,3,3.080016,4.25,170,40,44.8,448,10,Bayern Munich,-0.5171399999999999,0.402844,-0.07129999999999992
Bundesliga,2014,a,3.46966,0.821798,3.46966,0.821798,27,0,2,0,2.8138,w,2014-09-27 14:30:00,1,0,0,3,2.647862,5.44,136,25,44.09090909090909,485,11,Bayern Munich,1.4696600000000002,0.821798,-0.1861999999999992
Bundesliga,2014,h,2.69879,0.443178,2.69879,0.443178,14,5,4,0,2.7325000000000004,w,2014-10-04 14:30:00,1,0,0,3,2.2556119999999997,11.0,187,17,52.375,419,8,Bayern Munich,-1.3012100000000002,0.443178,-0.2674999999999996
Bundesliga,2014,h,2.49826,0.0,1.74049,0.0,16,0,6,0,2.9392000000000005,w,2014-10-18 14:30:00,1,0,0,3,1.74049,3.6333333333333333,109,30,28.8,432,15,Bayern Munich,-3.50174,0.0,-0.06079999999999952
Bundesliga,2014,a,1.2047,0.648384,1.2047,0.648384,5,7,0,0,1.8346,d,2014-10-26 16:30:00,0,1,0,1,0.5563160000000001,6.3,126,20,26.857142857142858,376,14,Bayern Munich,1.2047,0.648384,0.8346
Bundesliga,2014,h,2.64367,0.942884,1.88605,0.942884,11,3,2,1,2.4956,w,2014-11-01 17:30:00,1,0,0,3,0.943166,6.1935483870967744,192,31,13.4,402,30,Bayern Munich,0.6436700000000002,-0.05711599999999945,-0.5044
Bundesliga,2014,a,3.03467,0.222447,3.03467,0.222447,13,1,4,0,2.9606999999999997,w,2014-11-08 14:30:00,1,0,0,3,2.8122230000000004,6.12,153,25,8.326530612244898,408,49,Bayern Munich,-0.9653299999999998,0.222447,-0.03930000000000335
Bundesliga,2014,h,2.08656,1.38014,2.08656,1.38014,12,6,4,0,1.8595,w,2014-11-22 14:30:00,1,0,0,3,0.70642,5.88,147,25,9.0,342,38,Bayern Munich,-1.91344,1.38014,-1.1405
Bundesliga,2014,a,1.94047,0.25622,1.94047,0.25622,19,1,1,0,2.6238,w,2014-11-29 14:30:00,1,0,0,3,1.68425,4.269230769230769,111,26,20.6,515,25,Bayern Munich,0.9404699999999999,0.25622,-0.3761999999999987
Bundesliga,2014,h,0.947473,0.389245,0.947473,0.389245,3,1,1,0,1.9042999999999999,w,2014-12-06 17:30:00,1,0,0,3,0.558228,7.181818181818182,158,22,11.71875,375,32,Bayern Munich,-0.05252699999999999,0.389245,-1.0957000000000001
Bundesliga,2014,a,2.21868,0.286173,2.21868,0.286173,13,0,4,0,2.6733,w,2014-12-13 14:30:00,1,0,0,3,1.932507,5.652173913043478,130,23,14.90625,477,32,Bayern Munich,-1.78132,0.286173,-0.3267000000000002
Bundesliga,2014,h,2.98351,0.0497566,2.98351,0.0497566,32,1,2,0,2.9835999999999996,w,2014-12-16 19:00:00,1,0,0,3,2.9337534,3.851851851851516,104,27,91.8,459,5,Bayern Munich,0.9835099999999999,0.0497566,-0.016400000000000414
Bundesliga,2014,a,0.831278,0.490212,0.831278,0.490212,7,1,2,1,1.6329,w,2014-12-19 19:30:00,1,0,0,3,0.341066,4.210526315789473 5,160,38,18.90909090909091,416,22,Bayern Munich,-1.168722,-0.509788,-1.3671
Bundesliga,2014,a,1.91633,2.18248,1.91633,2.18248,12,3,1,4,1.1688,l,2015-01-30 19:30:00,0,0,1,0,-0.2661499999999999,4.878787878787879,161,33,13.8,414,30,Bayern Munich,0.9163300000000001,-1.81752,1.1688
Bundesliga,2014,h,0.376661,0.994575,0.376661,0.236798,5,1,1,1,0.5921000000000001,d,2015-02-03 19:00:00,0,1,0,1,0.13986300000000002,9.954545454545455,219,22,12.666666666666666,380,30,Bayern Munich,-0.623339,-0.005425000000000013,-0.4
Bundesliga,2014,a,0.0975022,0.132112,0.0975022,0.132112,1,1,2,0,1.0552000000000001,w,2015-02-07 14:30:00,1,0,0,3,-0.03460980000000001,4.166666666666667,50,12,25.90909090909091,285,11,Bayern Munich,-1.9024978,0.132112,-1.9447999999
Bundesliga,2014,h,4.17851,0.230619,3.42073,0.230619,11,0,8,0,2.9856,w,2015-02-14 14:30:00,1,0,0,3,3.190111,7.458333333333333,179,24,25.545454545454547,562,22,Bayern Munich,-3.82149,0.230619,-0.01440000000000019
Bundesliga,2014,a,3.12815,0.380499,2.37038,0.380499,15,1,6,0,2.9059,w,2015-02-21 14:30:00,1,0,0,3,1.989881,4.181818181818182,138,33,33.53333333333333,503,15,Bayern Munich,-2.87185,0.380499,-0.09410000000000007
Bundesliga,2014,h,3.92268,0.494166,3.92268,0.494166,19,1,4,1,2.9544,w,2015-02-27 19:30:00,1,0,0,3,3.4285140000000003,6.111111111111111,110,18,44.3,443,10,Bayern Munich,-0.07731999999999983,-0.505834,-0.04559999999999986
Bundesliga,2014,a,1.33559,0.205925,0.577811,0.205925,3,0,3,1,2.5082999999999998,w,2015-03-07 14:30:00,1,0,0,3,0.37188599999999994,4.095238095238095,86,21,40.454545454545445,445,11,Bayern Munich,-1.66441,-0.794075,-0.4917000000000025
Bundesliga,2014,a,2.18396,0.290321,2.18396,0.290321,10,3,4,0,2.7848,w,2015-03-14 14:30:00,1,0,0,3,1.8936389999999999,4.727272727272725,104,22,12.294117647058824,418,34,Bayern Munich,-1.81604,0.290321,-0.2151999999999984
Bundesliga,2014,h,0.70739,0.604102,0.70739,0.604102,13,3,0,2,1.3739,l,2015-03-22 16:30:00,0,0,1,0,0.10328799999999994,7.181818181818182,237,33,47.3,473,10,Bayern Munich,0.70739,-1.3958979999999999,1.3739
```

Рисунок 5.1 – Частина обраного датасету

5.2 Аналіз даних з набору

Представити обраний набір даних в структурованому виді допоможе бібліотека Pandas, яка містить необхідний функціонал по перетворенню файлів csv-формату в DataFrame. DataFrame — це структура, яка містить двовимірні дані [36] та відповідні їм колонки та індекси. Більш простими словами, це таблиця з записами, де колонками є назви параметрів, а рядками представлені записи в цю таблицю. Дана структура є «перлиною» даної бібліотеки, переважна більшість робіт з аналізу даних та машинного навчання не обходяться без її використання.

Після завантаження датасету в структурованому вигляді перші п'ять записів виглядають наступним чином (рисунок 5.2):

	league	year	h_a	xG	xGA	npxG	npxGA	deep	deep_allowed	scored	...	ppda_coef	ppda_att	ppda_def	oppda_coef	oppda_att	oppda
0	Bundesliga	2014	h	2.57012	1.198420	2.57012	1.198420	5	4	2	...	9.625000	231	24	21.850000	437	
1	Bundesliga	2014	a	1.50328	1.307950	1.50328	1.307950	10	1	1	...	4.756098	195	41	17.695652	407	
2	Bundesliga	2014	h	1.22987	0.310166	1.22987	0.310166	13	3	2	...	5.060606	167	33	16.961538	441	
3	Bundesliga	2014	a	1.03519	0.203118	1.03519	0.203118	6	2	0	...	4.423077	115	26	9.446809	444	
4	Bundesliga	2014	h	3.48286	0.402844	3.48286	0.402844	23	2	4	...	4.250000	170	40	44.800000	448	

Рисунок 5.2 – Початок набору даних в структурованому вигляді

Цей набір даних містить 29 колонок, тобто 29 характеристик команди в матчі, а рядками виступають безпосередньо записи характеристик з футбольних матчів кожної команди з п’яти найбільш популярних чемпіонатів в сезонах з 2014 по 2019 рік.

Необхідно залишити лише ті колонки, які потенційно будуть потрібними для побудови системи (рисунок 5.3).

```
Index(['league', 'year', 'h_a', 'xG', 'xGA', 'npxG', 'npxGA', 'deep',
       'deep_allowed', 'scored', 'missed', 'xpts', 'result', 'date', 'wins',
       'draws', 'loses', 'pts', 'npxGD', 'ppda_coef', 'ppda_att', 'ppda_def',
       'oppda_coef', 'oppda_att', 'oppda_def', 'team', 'xG_diff', 'xGA_diff',
       'xpts_diff'],
      dtype='object')
```

Рисунок 5.3 – Колонки набору даних

Після аналізу характеристик, опис яких надано в документації до набору даних, було прийняти рішення залишити наступні з них:

– league – ліга, в якій грає команда;

– year – рік, в якому почався сезон, в якому був зіграний обраний матч;

– h_a – номінативна характеристика, яка означає, грала команда на своєму полі, чи грала в гостях. “h” – вдома, “a” – в гостях;

– xG – очікувані голи, характеристика описана раніше;

– xGA – очікувані пропущені голи, тобто на скільки пропущених голів заслужив в конкретному матчі обраний колектив;

– scored – скільки голів забила команда в матчі;

– missed – скільки голів пропустила команда в матчі;

– xpts – очікувана кількість залікових пунктів, тобто на скільки очок награла команда;

– result – результат матчу відносно обраної команди: перемога “w”, нічия “d” чи поразка “l”;

– date – час та дата матчу;

– pts – скільки очок заробила команда за матч. 3 за перемогу, 1 за нічию, 0 за поразку;

– team – назва команди [37].

Тепер записи виглядають наступним чином (рисунок 5.4):

	league	year	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	result	date	pts	team
4333	EPL	2014	a	0.435091	1.218620	0.435091	1.218620	0	2	0.6282	l	2014-12-13 15:00:00	0	Hull
4334	EPL	2014	h	0.825885	1.453310	0.825885	1.453310	0	1	0.8314	l	2014-12-20 15:00:00	0	Hull
4335	EPL	2014	a	0.855730	1.355510	0.855730	1.355510	3	1	0.9067	w	2014-12-26 15:00:00	3	Hull
4336	EPL	2014	h	1.723140	0.180127	1.723140	0.180127	0	1	2.5995	l	2014-12-28 15:00:00	0	Hull
4337	EPL	2014	h	1.956820	0.957698	1.956820	0.957698	2	0	2.1050	w	2015-01-01 15:00:00	3	Hull
4338	EPL	2014	a	0.395844	0.784159	0.395844	0.784159	0	1	0.8781	l	2015-01-10 15:00:00	0	Hull
4339	EPL	2014	a	0.423719	2.686290	0.423719	2.686290	0	3	0.1374	l	2015-01-18 13:30:00	0	Hull
4340	EPL	2014	h	0.989377	0.509260	0.989377	0.509260	0	3	1.7966	l	2015-01-31 12:45:00	0	Hull
4341	EPL	2014	a	1.133480	0.992526	1.133480	0.992526	1	1	1.5181	d	2015-02-07 15:00:00	1	Hull
4342	EPL	2014	h	0.985975	0.423817	0.985975	0.423817	2	0	1.8925	w	2015-02-10 19:45:00	3	Hull

Рисунок 5.4 – Десять записів з набору після очищення

Окрім цього, необхідно оцінити якість набор даних, а саме дізнатись, чи правильну він надає інформацію. Перш за все, була проведена перевірка на наявність порожніх комірок, їх у наборі не виявилось (рисунок 5.5).

```
df[df.isna().any(axis=1)]
```

league	year	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	result	date	pts	team

Рисунок 5.5 – Перевірка на порожні комірки

Тепер потрібно перевірити дані на правдивість. Для цього було обрано 10 довільних записів, та знайдені відповідні матчі на ресурсі Understat (рисунок 5.6).

	league	year	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	result	date	pts	team
6392	EPL	2017	a	1.476030	1.948500	1.476030	1.948500	0	3	1.0388	l	2018-01-15 20:00:00	0	Stoke
15794	Ligue_1	2017	a	0.292445	1.727830	0.292445	0.967734	0	4	0.2534	l	2017-08-05 19:00:00	0	Strasbourg
17254	Serie_A	2014	a	4.454210	0.368621	2.931610	0.368621	3	1	2.9849	w	2015-04-04 14:00:00	3	Lazio
5397	EPL	2016	h	0.473979	1.224590	0.473979	1.224590	0	1	0.7037	l	2016-12-14 23:45:00	0	Sunderland
2382	Bundesliga	2017	h	0.490076	1.568010	0.490076	1.568010	0	0	0.5545	d	2017-09-09 14:30:00	1	Freiburg
7786	EPL	2019	h	1.535970	1.931920	1.535970	1.931920	2	3	1.0710	l	2019-11-02 15:00:00	0	West Ham
20418	Serie_A	2018	a	0.585585	1.798670	0.585585	1.798670	0	1	0.4730	l	2018-08-26 21:30:00	0	Sampdoria
9478	La_liga	2015	a	0.630978	1.871150	0.630978	1.871150	0	0	0.4845	d	2016-04-04 22:30:00	1	Sporting Gijon
8331	La_liga	2014	h	1.082560	1.927390	1.082560	1.927390	0	2	0.4768	l	2015-02-21 21:00:00	0	Deportivo La Coruna
11566	La_liga	2018	a	1.061840	0.243937	1.061840	0.243937	2	1	2.1612	w	2019-03-29 20:00:00	3	Athletic Club

Рисунок 5.6 – Довільні 10 записів з набору

Всі 10 записів виявились правдивими, на сайті дійсно міститься вказана інформація по обраних матчах. Приклад одного з матчів наведено на рисунку 5.7.

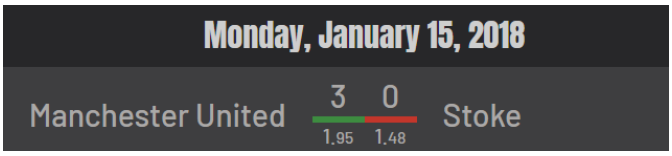


Рисунок 5.7 – Приклад першого з обраних довільним чином матчів

Останнім фактором, який розвіяв сумніви стосовно якості даних й дозволив спокійно використовувати обраний датасет стала відсутність негативних

коментарів на сайті Kaggle. При цьому в наявності чимала кількість дискусій, проєктів з даного набору даних та «подяк» автору.

5.3 Парсинг актуальної інформації

Збір актуальної інформації з ресурсу Understat необхідний з двох причин:

– формування поточних характеристик у команд для застосування їх у навченій моделі;

– доповнення даних до готового набору і, відповідно, збільшення якості навчання моделі в перспективі.

Інформація на сайті Understat представлена у вигляді JSON-об'єктів, які містяться в одному з тегів <script> – складовій веб-сторінки.

JSON (англ. JavaScript Object Notation) — текстовий формат обміну даними, заснований на JavaScript. Але при цьому формат не залежить від JS і може використовуватися у будь-якій мові програмування [38]. Приклад JSON-файлу зображено на рисунку 5.8.

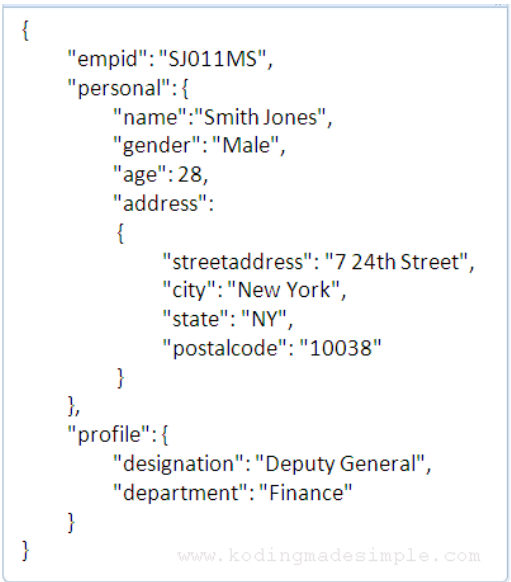


Рисунок 5.8 – Приклад JSON-об’єкту [39]

Перш за все, необхідно дістатися до складових в тегах <script> на сторінці кожної з ліг. Для цього застосовується спеціальний метод бібліотеки BeautifulSoup, який дозволяє отримувати дані, які знаходяться у тегах.

Після знаходження складових необхідного тегу, було отримане наступне (рисунок 5.9):

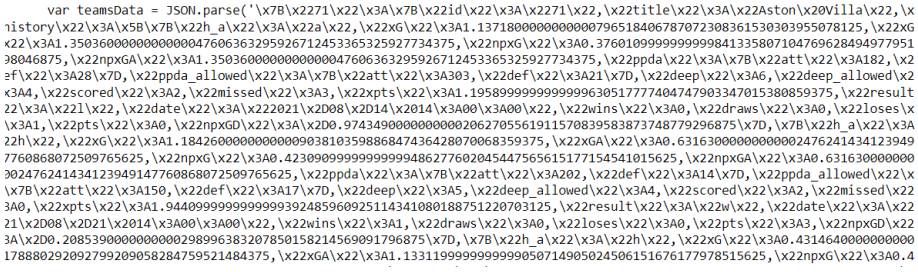


Рисунок 5.9 – Складова тегу

Даний текст представляє собою вказівку на мові JavaScript трансформувати аргумент функції parse в об'єкт teamsData, який необхідний для відображення інформації на сайті. Засоби Python теж дозволяють зчитати даний JSON-об’єкт. Для цього необхідно конвертувати цей об’єкт у вигляді строки за стандартом utf-8, а після цього, отриманий об'єкт декодувати за стандартом Unicode.

Стандарт utf-8 дозволяє компактніше зберігати і передавати символи Юнікоду, використовуючи від 1 до 4 байт, саме тому почакові дані спочатку треба було перевести в «чистий» utf, а вже після цього перевести його в Unicode – стандарт, який включає в себе майже всі знаки на різних мовах світу.

Після застосування інструкцій отримано наступне (рисунок 5.10):

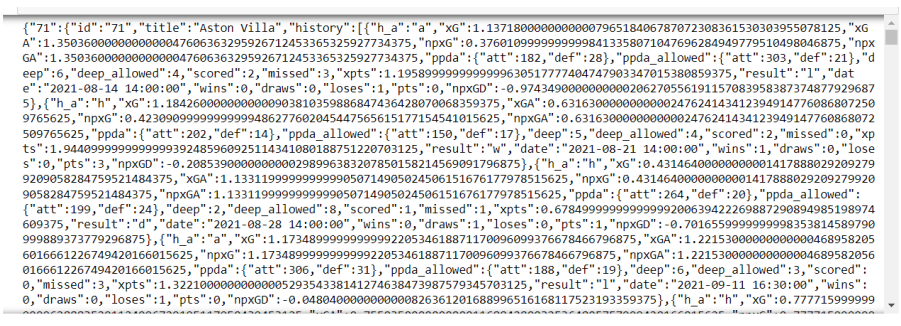


Рисунок 5.10 – JSON-об’єкт після кодування/декодування

Вищезазначені маніпуляції дозволили позбутися службових символів та представити об’єкт у вигляді, сприйнятному для людського ока.

Тепер, методом Python-модулю json даний об’єкт перетворюється на словник – структуру даних Python (рисунок 5.11).

```
Ввод [33]: data['EPL']

Out[33]: {'71': {'id': '71',
                 'title': 'Aston Villa',
                 'history': [{'h_a': 'a',
                   'xG': 1.13718,
                   'xGA': 1.35036,
                   'npxG': 0.376011,
                   'npxGA': 1.35036,
                   'ppda': {'att': 182, 'def': 28},
                   'ppda_allowed': {'att': 303, 'def': 21},
                   'deep': 6,
                   'deep_allowed': 4,
                   'scored': 2,
                   'missed': 3,
                   'xpts': 1.1959,
                   'result': 'l',
                   'date': '2021-08-14 14:00:00',
                   'wins': 0,
```

Рисунок 5.11 – Кінцевий результат перетворення JSON-об’єкта

Таким чином, будо отримано словник для кожної ліги, який в ключі ‘history’ містить список зіграних поєдинків з характеристиками. Це ті самі характеристики, які містяться в готовому наборі даних, отже, залишається лише представити отриману інформацію в тому самому вигляді, в якому знаходяться дані з Kaggle.

За допомогою вбудованих засобів Python і бібліотеки pandas було перетворено словники з інформацією про ліги в єдиний DataFrame (рисунок 5.12).

	league	year	team	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	date	result	pts
0	EPL	2021	Aston Villa	a	1.13718	1.35036	0.376011	1.35036	2	3	1.1959	2021-08-14 14:00:00	l	0
1	EPL	2021	Aston Villa	h	1.18426	0.63163	0.423091	0.63163	2	0	1.9441	2021-08-21 14:00:00	w	3
2	EPL	2021	Aston Villa	h	0.431464	1.13312	0.431464	1.13312	1	1	0.6785	2021-08-28 14:00:00	d	1
3	EPL	2021	Aston Villa	a	1.17349	1.22153	1.17349	1.22153	0	3	1.3221	2021-09-11 16:30:00	l	0
4	EPL	2021	Aston Villa	h	0.777716	0.755035	0.777716	0.755035	3	0	1.3358	2021-09-18 16:30:00	w	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3647	Ligue_1	2021	Clermont Foot	h	1.74002	1.3986	1.74002	1.3986	2	2	1.6446	2022-04-24 13:00:00	d	1
3648	Ligue_1	2021	Clermont Foot	a	0.993885	1.31508	0.993885	1.31508	0	2	0.9587	2022-05-01 13:00:00	l	0
3649	Ligue_1	2021	Clermont Foot	h	1.81186	1.11037	1.05177	1.11037	2	1	2.0541	2022-05-08 13:00:00	w	3
3650	Ligue_1	2021	Clermont Foot	a	0.4006	0.635695	0.4006	0.635695	0	1	1.0348	2022-05-14 19:00:00	l	0
3651	Ligue_1	2021	Clermont Foot	h	2.31455	2.47046	1.41774	2.47046	1	2	1.308	2022-05-21 19:00:00	l	0

3652 rows × 14 columns

Рисунок 5.12 – Отриманий набір даних

Тепер необхідно об’єднати отриманий DataFrame із вже існуючим набором даних з датасету. Для цього необхідно зробити конкатенацію цих двох таблиць з даними і, оскільки стовпці в них мають однакові назви, загальний вигляд не зміниться (рисунок 5.13).

	league	year	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	result	date	pts	team
0	Bundesliga	2014	h	2.57012	1.19842	2.57012	1.19842	2	1	2.3486	w	2014-08-22 19:30:00	3	Bayern Munich
1	Bundesliga	2014	a	1.50328	1.30795	1.50328	1.30795	1	1	1.5143	d	2014-08-30 17:30:00	1	Bayern Munich
2	Bundesliga	2014	h	1.22987	0.310166	1.22987	0.310166	2	0	2.1588	w	2014-09-13 14:30:00	3	Bayern Munich
3	Bundesliga	2014	a	1.03519	0.203118	1.03519	0.203118	0	0	2.1367	d	2014-09-20 14:30:00	1	Bayern Munich
4	Bundesliga	2014	h	3.48286	0.402844	3.48286	0.402844	4	0	2.9287	w	2014-09-23 19:00:00	3	Bayern Munich
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
25347	Ligue_1	2021	h	1.74002	1.3986	1.74002	1.3986	2	2	1.6446	d	2022-04-24 13:00:00	1	Clermont Foot
25348	Ligue_1	2021	a	0.993885	1.31508	0.993885	1.31508	0	2	0.9587	l	2022-05-01 13:00:00	0	Clermont Foot
25349	Ligue_1	2021	h	1.81186	1.11037	1.05177	1.11037	2	1	2.0541	w	2022-05-08 13:00:00	3	Clermont Foot

Рисунок 5.13 – Об’єднаний набір даних

5.4 Виведення первинних характеристик

Перед виведенням нових характеристик обов'язково треба пам'ятати про головний принцип: для моделі необхідна статистика команди безпосередньо перед матчем, а отже всі показники повинні бути розраховані саме на цей момент часу, а не за сезон загалом.

На першому етапі необхідно зрозуміти які змінні можуть бути виведеними з наявних даних. В ході аналізу для початку заплановане виведення наступних змінних:

– середня кількість xG за весь сезон до обраного поєдинку;
– середня кількість xGA за весь сезон до обраного поєдинку;
– середня кількість забитих голів за весь сезон до обраного поєдинку;
– середня кількість пропущених голів до обраного поєдинку;
– середня кількість xG за всі домашні матчі;
– середня кількість xG за останні 5 матчів;
– середня кількість xGA за останні 5 матчів.

З великою ймовірністю деякі з цих змінних увійдуть в фінальну модель, а також допоможуть у подальшій роботі.

Середня кількість xG та xGA за сезон на момент проведення матчу будуть представлені середнім арифметичним даних показників за всі ігри сезону до обраного матчу, та будуть показувати наскільки добре команда грає у атаці та захисті напротязі всього сезону.

Незважаючи на те, що xG та xGA відображають набагато більш об'єктивну інформацію про стан колективів, аніж забиті та пропущені м'ячі, не варто повністю викреслювати ці фактори, оскільки доволі часто деякі команди мають неймовірно хорошу реалізацію моментів, що сприяє більшій кількості забитих м'ячів, ніж показник очікуваних голів, при цьому цілком виправдано чекати доволі велику кількість забитих м'ячів в цьому випадку. Так само й навпаки, команда може утворювати багато моментів, але забивати явно менше, аніж заслуговує, і це не буде випадковістю.

Середня кількість xG за всі домашні матчі продемонструє наскільки важливим для обраного колективу є фактор домашнього поля, адже деякі команди демонструють набагато кращу гру при «своїх» трибунах, аніж в гостьових поєдинках, інші ж не є залежними від цього фактору. Вираховуватиметься взяттям середнього арифметичного забитих голів в усіх домашніх матчах до обраного. Планується зробити даний показник нормалізованим для того щоб позначати наскільки «домашньою» є командою, а не просто отримати додаткове доволі схожий на показник xG значення.

Середня кількість xG та xGA за 5 матчів напрямлена на те, щоб дізнатися форму команди в останніх матчах. Дані показники є доволі неточними, оскільки не враховують рівень суперників в останніх матчах, але за їхньою участю пізніше буде виведено більш об'єктивний показник гри команд в нещодавніх поєдинках.

Для виведення характеристик команди на момент гри треба знати як грав цей колектив в обраному сезоні до цього. Функціонал pandas дозволяє групувати записи за унікальними значеннями в певних колонках і розрахувати агрегатні

функції для отриманих комбінацій унікальних значень в обраних колонках. Агрегатні функції - це такі функції, яка виконують обчислення на наборі значень та виводять єдине значення. Наприклад, функція sum() порахує суму набору значень, функція mean() - середнє арифметичне, а count() - кількість значень - всі дані функції є агрегатними. Отже, щоб отримати дані про кожну команду за певний сезон необхідно згрупувати записи по рокам, а потім по командам (рисунок 5.14).

```
Ввод [22]: list(df.groupby(['year', 'team']).groups.keys())

Out[22]: [(2014, 'AC Milan'),
          (2014, 'Almeria'),
          (2014, 'Arsenal'),
          (2014, 'Aston Villa'),
          (2014, 'Atalanta'),
          (2014, 'Athletic Club'),
          (2014, 'Atletico Madrid'),
          (2014, 'Augsburg'),
          (2014, 'Barcelona'),
```

Рисунок 5.14 – Приклади отриманих груп, за якими відбувається групування

Тепер для кожної з цих груп є можливість застосувати агрегатні функції на їх стовпці, які містять числові значення. Наприклад, можна вирахувати середнє значення очікуваних забитих голів для кожної команди в кожному сезоні (рисунок 5.15).

```
df.groupby(['year', 'team'])['xG'].mean()

year  team
2014  AC Milan          1.238751
      Almeria           1.017630
      Arsenal           1.836910
      Aston Villa       0.871123
```

Рисунок 5.15 – Середній xG по командам

Але необхідно вирахувати середнє значення саме на момент гри команди, тобто знайти середнє арифметичне всіх ігор до обраної. Для цього разом з groupby() треба використати метод apply(), який застосує бажану функцію для кожної таблиці даних, отриманої з головної після розбиття по групах. Функція,

що застосовується, повинна мати аргумент – DataFrame, який відповідає одній з тих самих таблиць даних, а також повинна повертати цю саму таблицю.

Для того, щоб пройти по кожному рядку таблиці треба застосувати метод iterrows(), який дозволяє отримати кожний індекс і кожну строку під час проходження по всім ним. Знаючи початковий індекс для обраної таблиці, а також індекс строки під час застосування вищезазначеного метода, розраховується і записується в новий стовпець середнє арифметичне на даному проміжку. Функція, яка це робить наведена на рисунку 5.16, а результат зображено на рисунку 5.17.

```
def xgmean(frame):
    means = pd.Series(index=frame.index, dtype=float)
    prev_index = 0
    for index, row in frame.iterrows():
        if index != frame.index[0]:
            means[index] = frame.loc[frame.index[0]:prev_index]['xG'].mean()
        prev_index = index
    frame['xGmean'] = means
    return frame

df = df.groupby(['year','team']).apply(xgmean)
df
```

Рисунок 5.16 – Розрахунок середнього значення xG перед кожним матчем

	league	year	h_a	xG	xGA	npxG	npxGA	scored	missed	xpts	result	date	pts	team	xGmean
0	Bundesliga	2014	h	2.57012	1.198420	2.57012	1.198420	2	1	2.3486	w	2014-08-22 19:30:00	3	Bayern Munich	NaN
1	Bundesliga	2014	a	1.50328	1.307950	1.50328	1.307950	1	1	1.5143	d	2014-08-30 17:30:00	1	Bayern Munich	2.570120
2	Bundesliga	2014	h	1.22987	0.310166	1.22987	0.310166	2	0	2.1588	w	2014-09-13 14:30:00	3	Bayern Munich	2.036700
3	Bundesliga	2014	a	1.03519	0.203118	1.03519	0.203118	0	0	2.1367	d	2014-09-20 14:30:00	1	Bayern Munich	1.767757
4	Bundesliga	2014	h	3.48286	0.402844	3.48286	0.402844	4	0	2.9287	w	2014-09-23 19:00:00	3	Bayern Munich	1.584615
5	Bundesliga	2014	a	3.46966	0.821798	3.46966	0.821798	2	0	2.8138	w	2014-09-27 14:30:00	3	Bayern Munich	1.964264
6	Bundesliga	2014	h	2.69879	0.443178	2.69879	0.443178	4	0	2.7325	w	2014-10-04 14:30:00	3	Bayern Munich	2.215163
7	Bundesliga	2014	h	2.49826	0.000000	1.74049	0.000000	6	0	2.9392	w	2014-10-18 14:30:00	3	Bayern Munich	2.284253
8	Bundesliga	2014	a	1.20470	0.648384	1.20470	0.648384	0	0	1.8346	d	2014-10-26 16:30:00	1	Bayern Munich	2.311004
9	Bundesliga	2014	h	2.64367	0.942884	1.88605	0.942884	2	1	2.4956	w	2014-11-01 17:30:00	3	Bayern Munich	2.188081

Рисунок 5.17 – Отриманий результат

У функції для розрахунку середнього значення фігурує Series-об'єкт, який представляє собою одновимірну структуру даних з індексом та значенням. Series може стати колонкою в двовимірній таблиці, окільки за своєю сутністю об’єкт такого типу є датафреймом з однією колонкою.

Більшість інших значень розраховуються аналогічним чином, лише трохи відрізняються функції в залежності від того, що дана функція знаходить, наприклад, під час пошуку середнього xG виключно для домашніх поєдинків стояв відповідний фільтр, а під час пошуку очікуваних голів за останні п'ять поєдинків ті записи, які фігурують до обраного на 6 і більше індексів назад ігноруються.

Також метод apply() може застосовуватися в датафреймі без застосування groupby, а також приймати аргументом лямбда-вираз – невелику функцію без назви (анонімну) [40] та зі специфічним синтаксисом. Наприклад, під час переведення значень стовпця h_a, який містить буквенну номінативну змінну, що позначає, де команда грала матч: "h" – вдома, "a" – на виїзді в відповідні числові значення: 1 – вдома, 0 – на виїзді застосується лямбда-вираз, зображений на рисунку 5.18:

```
df['h_a'] = df['h_a'].apply(lambda x: 1 if x == 'h' else 0)
```

Рисунок 5.18 – Застосування лямбда-функції

X в контексті даної функції означає кожний елемент обраного стовпця.

Окрім цього, лямбда-вираз слугує аргументом методу apply у тому випадку, коли виконувана функція приймає додаткові компоненти. В реалізованій системі це функція, яка порахує середній xG та xGA за 5 поєдинків (рисунок 5.19):

```
df = df.groupby(['year','team']).apply(lambda x: xgmeancount(x, 5))
df = df.groupby(['year','team']).apply(lambda x: xgAmeancount(x, 5))
```

Рисунок 5.19 – Застосування лямбда-функції (2)

Невідома x в цій ситуації означає таблицю з даними, яка передається першим аргументом в відповідну функцію, а другим компонентом є число за яку кількість матчів треба шукати середні значення. В цьому випадку не відбувається анонімності, оскільки повинна бути ініціалізована функція, яка задає інструкції по

вираховуванню необхідних значень, тим не менше лямбда-вираз допоміг передати додатковий аргумент, що неможливо зробити без лямбди. Одна з таких функцій наведена на рисунку 5.20.

```
def xgmeancount(frame, count):
    column_name = 'xGmean'+str(count)
    means = pd.Series(index=frame.index, dtype=float)
    prev_index = 0
    for index, row in frame.iterrows():
        if index >= frame.index[count]:
            means[index] = frame.loc[(index-count):prev_index]['xG'].mean()
        prev_index = index
    frame[column_name] = means
    return frame
```

Рисунок 5.20 – Функція з інструкціями, що містить не один аргумент

Отже, після написання необхідних функцій та розрахунку необхідних на цьому кроці змінних частина таблиці виглядає наступним чином (рисунок 5.21):

	league	year	h_a	team	xGmean	xGAmean	ScoredMean	MissedMean	xGMeanHome	xGmean5	xGAMean5
0	Bundesliga	2014	1	Bayern Munich	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Bundesliga	2014	0	Bayern Munich	2.570120	1.198420	2.000000	1.000000	2.570120	NaN	NaN
2	Bundesliga	2014	1	Bayern Munich	2.036700	1.253185	1.500000	1.000000	2.570120	NaN	NaN
3	Bundesliga	2014	0	Bayern Munich	1.767757	0.938845	1.666667	0.666667	1.899995	NaN	NaN
4	Bundesliga	2014	1	Bayern Munich	1.584615	0.754914	1.250000	0.500000	1.899995	NaN	NaN
5	Bundesliga	2014	0	Bayern Munich	1.964264	0.684500	1.800000	0.400000	2.427617	1.964264	0.684500
6	Bundesliga	2014	1	Bayern Munich	2.215163	0.707383	1.833333	0.333333	2.427617	2.144172	0.609175

Рисунок 5.21 – Розраховані первинні характеристики

Розраховані на цьому кроці характеристики розташувались в рядках зіграних поєдинків і вони означають показники, з якими команда підходила до кожного матчу.

Ще одним дуже важливим показником є вартість футбольного клубу, яка складається з вартості гравців. Цілком очевидно, що більш майстерні футболісти коштують дорожче за менш обдарованих колег, тому очевидно, що чим більша ціна команди, тим більша ймовірність перемоги цього колективу.

Ані набір даних, ані ресурс Understat не надають інформації про вартість футбольних клубів, тому є необхідність звернутись до додаткового ресурсу transfermarkt, який спеціалізується на футбольному ринку. На цьому сайті наявна

ціна великої кількості клубів за різні роки. На рисунку 5.22 зображені дані з цього порталу про англійські клуби, в тому числі про їхню ціну.

CLUBS - PREMIER LEAGUE 21/22

club	Squad	ø age	Foreigners	ø market value	Total market value
Manchester City	22	27.8	15	€43.60m	€959.30m
Liverpool FC	33	26.5	22	€28.11m	€927.50m
Chelsea FC	31	26.8	21	€28.05m	€869.40m
Manchester United	31	27.2	18	€24.98m	€774.25m
Tottenham Hotspur	23	25.6	15	€25.75m	€592.25m
Arsenal FC	21	24.8	14	€24.81m	€521.00m
Leicester City	28	27.0	18	€18.39m	€514.80m
Aston Villa	28	26.0	13	€16.59m	€464.40m
Everton FC	28	27.4	14	€15.10m	€422.75m
Wolverhampton Wanderers	28	25.7	22	€12.98m	€363.50m
West Ham United	26	28.5	17	€13.44m	€349.45m
Leeds United	28	24.3	16	€10.50m	€294.10m
Newcastle United	30	28.3	16	€9.68m	€290.40m
Brighton & Hove Albion	29	24.7	20	€9.57m	€277.40m
Southampton FC	26	26.8	14	€9.59m	€249.25m
Brentford FC	29	25.5	23	€8.17m	€236.90m
Crystal Palace	26	27.5	14	€9.07m	€235.75m
Norwich City	37	25.1	23	€4.15m	€153.50m
Watford FC	34	27.4	29	€4.38m	€148.75m
Burnley FC	26	29.5	12	€5.50m	€143.03m
	564	27.1 Years	356	€15.58m	€8.79bn

Рисунок 5.22 – Ціна клубів на Transfermarkt

Для того, щоб кожній команді задати відповідну ціну необхідно згрупувати таблицю за роком та командою (Рисунок 6.14). Даний список потрібно зберегти у вигляді змінної Python. Тепер, спираючись на рік та назву команди, необхідно створити список цін команд таким чином, щоб позиція року та назви співпадала з позицією ціни команди в обраному році. Для більшої зручності було прийняте рішення групувати додатково за назвою чемпіонату. Результат неведений на рисунку 5.23.

Ресурс Transfermarkt пропонує загальну ціну составу та середню вартість одного гравця. Між даними двома характеристиками була обрана друга, оскільки вона відображає більш об’єктивну силу гравців основної команди.

```
Ввод [46]: list(df.groupby(['year', 'league', 'team']).groups.keys())

Out[46]: [(2014, 'Bundesliga', 'Augsburg'),
          (2014, 'Bundesliga', 'Bayer Leverkusen'),
          (2014, 'Bundesliga', 'Bayern Munich'),
          (2014, 'Bundesliga', 'Borussia Dortmund'),
          (2014, 'Bundesliga', 'Borussia M.Gladbach'),
          (2014, 'Bundesliga', 'Eintracht Frankfurt'),
          (2014, 'Bundesliga', 'FC Cologne'),
          (2014, 'Bundesliga', 'Freiburg'),
          (2014, 'Bundesliga', 'Hamburger SV'),
          (2014, 'Bundesliga', 'Hannover 96'),
          (2014, 'Bundesliga', 'Hertha Berlin'),
          (2014, 'Bundesliga', 'Hoffenheim'),
          (2014, 'Bundesliga', 'Mainz 05'),
          (2014, 'Bundesliga', 'Paderborn'),
          (2014, 'Bundesliga', 'Schalke 04'),
          (2014, 'Bundesliga', 'VfB Stuttgart'),
          (2014, 'Bundesliga', 'Werder Bremen'),
          (2014, 'Bundesliga', 'Wolfsburg'),
          (2014, 'EPL', 'Arsenal'),

Ввод [ ]: overall_costs = [1.43, 6.82, 17.43, 8.64, 4.84, 2.5, 1.88, 1.91, 2.29, 2.19, 2.2, 4.06, 2.44, 1.02, 5.89, 2.9, 1.88, 7.37, 9.88,
                           1.95, 7.31, 17.53, 11.77, 6.42, 0.838, 2.19, 2.75, 1.81, 1.56, 2.32, 3.01, 1.23, 2.18, 7.04, 2.39, 1.66, 7.73,
                           1.78, 6.8, 17.05, 12.6, 4.91, 0.666, 2.07, 3.87, 2.41, 2.12, 2.93, 4.2, 1.21, 2.48, 5.34, 5.57, 2.18, 4.72, 12.9
                           3.04, 13.24, 21.36, 16.41, 5.52, 3.5, 2.41, 2.79, 1.84, 3.1, 4.11, 5.91, 2.98, 13.33, 9.46, 3.96, 3.25, 4.35, 15
                           3.48, 13.85, 24.53, 18.92, 8.81, 6.67, 2.79, 3.59, 2.47, 5.77, 8.85, 4.56, 1.88, 14.09, 5.67, 4.23, 5.09, 6.37,
                           3.05, 13.76, 20.46, 17.68, 7.49, 5.97, 2.81, 1.94, 4.09, 5.17, 6.11, 3.47, 0.903, 15.01, 6.71, 1.21,3.75, 5.46,
                           2.54, 3.49, 16.09, 30.94, 1.71, 18.07, 8.47, 6.22, 2.94, 5.86, 1.21, 3.1, 7.05, 4.88, 17.91, 2.61, 5.21, 8.45,
```

Рисунок 5.23 – Формування ціни клубів по сезонам

Тепер необхідно сформувати словник, ключами якого стануть сукупності року, ліги та команди, а значеннями – відповідна ціна (рисунок 5.24).

```
Ввод [55]: dict_cost = dict(zip(list(df.groupby(['year', 'league', 'team']).groups.keys()), overall_costs))
           dict_cost

Out[55]: {(2014, 'Bundesliga', 'Augsburg'): 1.43,
          (2014, 'Bundesliga', 'Bayer Leverkusen'): 6.82,
          (2014, 'Bundesliga', 'Bayern Munich'): 17.43,
          (2014, 'Bundesliga', 'Borussia Dortmund'): 8.64,
          (2014, 'Bundesliga', 'Borussia M.Gladbach'): 4.84,
          (2014, 'Bundesliga', 'Eintracht Frankfurt'): 2.5,
          (2014, 'Bundesliga', 'FC Cologne'): 1.88,
          (2014, 'Bundesliga', 'Freiburg'): 1.91,
          (2014, 'Bundesliga', 'Hamburger SV'): 2.29,
          (2014, 'Bundesliga', 'Hannover 96'): 2.19,
          (2014, 'Bundesliga', 'Hertha Berlin'): 2.2,
          (2014, 'Bundesliga', 'Hoffenheim'): 4.06,
          (2014, 'Bundesliga', 'Mainz 05'): 2.44,
          (2014, 'Bundesliga', 'Paderborn'): 1.02,
```

Рисунок 5.24 – Рік, команда та їх вартість

Задля того, щоб додати ціну команди в таблицю, необхідно, перш за все, оновити індекс таблиці, який буде складатися з комбінації (рік, ліга, команда), так само, як ключі в словнику. Тепер, пройшовши циклом по ньому, знаходяться відповіді записи в датафреймі, оскільки кожен ключ за своєю суттю є індексом таблиці. Врешті-решт для кожного індекса необхідно вписати відповідне значення ціни. Результат додаванні ціни в таблицю наведено нижчу на рисунку 5.25.

			overall_cost
year	league	team	
2014	Bundesliga	Bayern Munich	17.43
		Bayern Munich	17.43
		Bayern Munich	17.43
		Bayern Munich	17.43
		Bayern Munich	17.43
...	...	...	...
2021	Ligue_1	Clermont Foot	1.70
		Clermont Foot	1.70
		Clermont Foot	1.70
		Clermont Foot	1.70
		Clermont Foot	1.70

Рисунок 5.25 – Ціна клубів в таблиці

5.5 Формування футбольних матчів

Для навчання моделі, як вже було зазначено раніше, необхідно мати характеристики домашньої команди та гостьової, але обраний набір даних містить інформацію в матчах лише відносно однієї команди. Отже, за один реальний матч в таблиці з'являються дві записи. Метою пункту є об'єднання кожного з таких двох записів в один єдиний, що містить інформацію про характеристики домашньої та гостьової команди.

Оскільки набір даних містить інформацію про лігу, дату та час зустрічі, то можна знайти команди з однієї країни, які грали поєдинок одночасно. Але таких підхід не включає ситуацію, коли декілька поєдинків розпочалося одночасно, тому необхідно додатково створити перевірку.

Щоб гарантувати, що суперника для обраної команди знайдено правильно, треба звернутися до наявних даних: такі показники як "xG" та "xGA", кількість забитих та кількість пропущених голів у випадку записів двох команд, які грали один матч приймають протилежні значення. Тобто якщо для одного з колективів голи будуть забитими, то для другого – пропущеними. Аналогічно з очікуваними голами: що для першої команди буде "xG", те для другої "xGA", та навпаки. Таким чином знаходиться суперник для будь-якої команди.

Для покращення продуктивності варто згрупувати таблицю по рокам та лігам, а потім застосувати для отриманих груп функцію для пошуку опоненту, зображену на рисунку 5.26:

```
def opponent(frame):
    away_teams = pd.Series(index=frame.index, dtype = object)
    for index, row in frame.iterrows():
        xGcurrent = row['npxG']
        xGAcurrent = row['npxGA']
        scoredcurrent = row['scored']
        missedcurrent = row['missed']
        date = row['date']
        away_team = frame[(frame['npxGA'] == xGcurrent) &
                          (frame['npxG'] == xGAcurrent) &
                          (frame['missed'] == scoredcurrent) &
                          (frame['scored'] == missedcurrent) &
                          (frame['date'] == date)]['team'].item()
        away_teams[index] = away_team
    frame['opponent'] = away_teams
    return frame
```

Рисунок 5.26 – Функція знаходження опонента

Після застосування функції отримано назву опоненту для команд (рисунок 5.27):

	team	opponent
0	Bayern Munich	Wolfsburg
1	Bayern Munich	Schalke 04
2	Bayern Munich	VfB Stuttgart
3	Bayern Munich	Hamburger SV
4	Bayern Munich	Paderborn
...	...	...
25347	Clermont Foot	Angers
25348	Clermont Foot	Brest
25349	Clermont Foot	Montpellier

Рисунок 5.27 – Опонент кожної команди

Після знаходження назв суперників необхідно додати до таблиці з даними характеристики знайденої команди-опонента в матчах. Для того, щоб зробити це, необхідно для кожного запису визначити команду та її суперника та знайти в таблиці запис, де основною командою буде саме суперник, а обрана в рядку основна команда - суперником. Таким чином отримано доступ до характеристик

команди-опонента для кожного запису. Схема додавання нових колонок до таблиці зображена на кресленику ІА.81.090БАК.003 Д4 з лівого боку.

На рисунку зверху продемонстрована схема додавання характеристик до рядків. Однаковим кольором позначені команди, які грали один матч, а в таблиці міститься два записи: кожний відносно однієї з команд. Знайшовши ось таку пару записів, залишається лише додати характеристики команди-опонента, розташувавши їх в відповідних нових колонках.

Після застосування інструкцій вище частина таблиці виглядатиме так, як наведено на рисунку 5.28:

	team	opponent	xGmean	ScoredMean	xGmean5	xGmeanOpp	ScoredMeanOpp	xGmean5Opp
0	Bayern Munich	Wolfsburg	NaN	NaN	NaN	NaN	NaN	NaN
1	Bayern Munich	Schalke 04	2.570120	2.000000	NaN	0.956665	1.0	NaN
2	Bayern Munich	VfB Stuttgart	2.036700	1.500000	NaN	0.578042	0.5	NaN
3	Bayern Munich	Hamburger SV	1.767757	1.666667	NaN	1.010187	0.0	NaN
4	Bayern Munich	Paderborn	1.584615	1.250000	NaN	1.275367	1.75	NaN
5	Bayern Munich	FC Cologne	1.964264	1.800000	1.964264	0.690802	0.4	0.690802
6	Bayern Munich	Hannover 96	2.215163	1.833333	2.144172	0.96334	0.833333	0.92005

Рисунок 5.28 – Характеристики гостьової команди в таблиці

5.6 Нормалізація змінних

Нормалізація – це процес переведення даних до певного діапазону, наприклад від 0 до 1 або від -1 до +1. Нормалізація потрібна, коли є великі відмінності в діапазонах різних ознак [41].

Також нормалізація має позитивний вплив на якість моделі машинного навчання в тому випадку, коли характеристики, в яких вона буде проведена, позначають скоріш відносне значення, а не кількісне.

Для обраної моделі було прийняте рішення нормалізувати дві характеристики: ціну команди та середнє очікуване значення забитих голів на домашньому полі (xGMeanHome).

Розроблювана система буде спеціалізуватися на прогнозуванні результатів поєдинків по лігам. Оскільки найдорожча команда чемпіонату може коштувати в

десятки разів більше, ніж найдешевша, корисним кроком буде нормалізувати даний критерій в інтервалі від 0 до 1, після чого характеристика по суті буде означати відсоток, який складає ціна обраної команди від найдорожчої команди турніру її країни.

Для того, щоб реалізувати цей вид нормалізації необхідно поділити ціну обраної команди на найдорожчу команду чемпіонату в обраному році. Результат зміни характеристик наведений на рисунку 5.29 нижче.

	team	overall_cost
0	Bayern Munich	17.43
34	Hamburger SV	2.29
68	Bayer Leverkusen	6.82
102	Hoffenheim	4.06
136	Augsburg	1.43
170	Hertha Berlin	2.20
204	Werder Bremen	1.88
238	Schalke 04	5.89
272	Mainz 05	2.44
306	Hannover 96	2.19
340	Borussia Dortmund	8.64

	team	overall_cost
0	Bayern Munich	1.000000
34	Hamburger SV	0.131383
68	Bayer Leverkusen	0.391279
102	Hoffenheim	0.232932
136	Augsburg	0.082042
170	Hertha Berlin	0.126219
204	Werder Bremen	0.107860
238	Schalke 04	0.337923
272	Mainz 05	0.139989
306	Hannover 96	0.125645
340	Borussia Dortmund	0.495697

Рисунок 5.29 – Нормалізація ціни клубу (було – стало)

У випадку характеристики середнього значення очікуваних забитих голів є необхідність перетворити кількісну характеристику на відносну. Сама по собі в базовому вигляді дана характеристика сильно корелює з критерієм середньої кількості очікуваних голів за сезон (xGMeanHome), що не є добре, оскільки може утворити труднощі при навчанні.

Перетворивши показник на відносний, він буде означати наскільки краще чи гірше команда грає на своєму полі відносно інших команд набору даних. В такому випадку ця характеристика дійсно відображатиме наскільки команда грає вдома краще, ніж в гостях. Тоді, коли значення буде позитивне, це означатиме, що обраний колектив сильно залежить від домашнього стадіону. У випадку значення близького до нуля – команда залежить від домашнього поля не більше, ніж інші в

середньому, а у випадку від'ємного – для команди не є сильно важливим домашній стадіон, а іноді навіть означитиме те, що в гостях команда грає краще, ніж вдома, що є доволі рідкісним явищем в футболі.

Щоб зробити подібне перетворення необхідно перш за все порахувати різницю між xGMeanHome (середнім значенням очікуваних забитих перед матчем в домашніх поєдинках) та xGMean (середнім значенням очікуваних забитих перед матчем), щоб дізнатися наскільки важливим є для команди домашнє поле. Очевидно, що у випадку, коли різниця буде доволі чутлива, тоді це буде позначати, що команда сильно залежить від свого стадіону. Після того, щоб перетворити характеристику на відносну необхідно використати наступну формулу 5.1:

$$Xnorm = \frac{(X - Xmean)}{Xstd} \tag{5.1}$$

де X – поточне значення в вибірці, Xmean – середнє значення вибірки, Xstd – середньоквадратичне відхилення вибірки, Xnorm – отримане нормалізоване значення.

Таке перетворення зробить 0 середнім значенням нормалізованої вибірки. Тобто нуль стане певним середнім значенням, а відхилення в позитивний чи негативний бік позначатимуть наскільки ненормалізоване значення було більше або менше за середнє у відносних одиницях.

Результат перетворення даної характеристики виглядає наступним чином (рисунок 5.30):

	team	xGmean	xGMeanHome
2980	RasenBallsport Leipzig	1.972032	2.309854
3363	Borussia Dortmund	1.926033	2.234857
18239	Sampdoria	1.105761	1.097472
20833	Torino	1.110196	1.365778
6431	Newcastle United	1.042626	1.015339
5053	Liverpool	1.335999	1.539948
13992	Monaco	1.608899	1.445737
299	Mainz 05	1.292698	1.263345

	team	xGmean	xGMeanHome
2980	RasenBallsport Leipzig	1.972032	0.767902
3363	Borussia Dortmund	1.926033	0.638396
18239	Sampdoria	1.105761	-0.777833
20833	Torino	1.110196	0.400617
6431	Newcastle United	1.042626	-0.862685
5053	Liverpool	1.335999	0.170025
13992	Monaco	1.608899	-1.469504
299	Mainz 05	1.292698	-0.871905

Рисунок 5.30 – Нормалізація показнику xGMeanHome (було – стало)

На випадковій вибірці записів на рисунку зверху помітно, що чим більша різниця між якістю гри вдома та загалом, тим більший отриманий нормалізований показник. Також на даних прикладах помітно, що тепер кореляція між показниками близька до нуля, на відміну від того, як було до цього.

Після нормалізації було прийняте рішення про перейменування даного показника, оскільки тепер він не має нічого спільного зі своєю старою назвою. Тепер показник називається «homeDependence», що приблизно перекладається як залежність від дому (в випадку футболу – свого стадіону).

Характеристики, які були виведені в ході попереднього підпункту (показники команди-опоненту) нормалізуються аналогічним чином.

Висновки до розділу 5

В ході даного розділу була проведена робота з даними.

Було знайдене джерело даних, яке надало необхідні для розроблюваної системи статистичні показники.

Також був налаштований механізм парсингу для збору актуальної статистичної інформації, яка збільшила набір даних.

Під час роботи з даними були виведені додаткові характеристики, які будуть складовими розроблюваної моделі прогнозування, а записи в таблиці були перетворені таким чином, що тепер вони позначають записи футбольних матчів.

6 ЗАСТОСУВАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ

6.1 Виведення додаткових характеристик методами машинного навчання

В одному з попередніх кроків була вирахувана характеристика очікуваних голів за останні 5 матчів, яка за задумкою повинна позначати форму команди, тобто якщо вона вище за очікувані голи за весь сезон, тоді це означає, що команда в гарній формі та грає матчі краще, ніж зазвичай і навпаки: у випадку, коли дана характеристика менша за середній xG, тоді форма колективу вважається поганою.

Але вищезазначений підхід не враховує важливий фактор – силу суперників команди в останніх матчах. Якщо серед п'яти останніх команд-опонентів було декілька команд, які є одними з найкращих в своїх лігах, то цілком природньо, що кількість очікуваних забитих голів буде меншою, ніж аналогічний показник за весь сезон, і в такому випадку зовсім не обов'язково робити висновок, що обрана команда має погану ігрову форму. Так само цілком очевидно, що з більш слабкими командами очікуваних і забитих голів буде більше.

Тому для розрахунку форми команди на основі інформації за останні матчі був розроблений наступний алгоритм:

– на основі статистичних даних щодо обраного колективу в останніх п’яти матчах, а також враховуючи розраховані на попередніх кроках показники команди суперника і фактор, на якому стадіоні грала обрана команда в кожному з останніх матчах розрахувати методами лінійної регресії кількість очікуваних голів в наступному матчі для обраної команди;

– розрахувати різницю між показником, виведеним в попередньому кроці, і реальним показником очікуваних забитих голів, розрахованим спеціалістами на основі моментів, утворених командою;

– розрахувати середнє значення між різницями значень з попереднього кроку. Якщо воно врешті виявиться додатнім, то це означатиме, що команда перебуває в хорошій формі і створювала в середньому більше моментів, ніж зазвичай, і навпаки.

Отже, методи машинного навчання будуть вперше застосовані саме на даному етапі. На цей момент для кожного матчу наявні як реальні вихідні характеристики, так і статистичні показники перед матчем. Саме другі будуть вхідними даними для моделі машинного навчання, а вчителем стане реальний показник очікуваних забитих голів в цьому матчі.

Для того, щоб навчання проходило якісно, необхідно перш за все відкинути рядки з порожніми комірками, а саме перші п’ять матчів кожної команди, адже для них характеристика середньої кількості очікуваних голів за п’ять поєдинків не вирахувана, а інші статистичні характеристики ще могли не відображати реальної сили колективу, бо за дуже малу кількість матчів показники надто сильно залежать від кожного з матчів, тому можуть погано впливати на об’єктивність характеристик.

Бібліотека sklearn повністю спроможна навчити модель спираючись на задані характеристики. Для прогнозування показника очікуваних голів для обох команд, які грали матч були обрані наступні показники:

- середнє значення очікуваних забитих та пропущених голів за сезон;
- середнє значення забитих голів за сезон;
- середнє значення пропущених голів за сезон;
- "домашність" команди;
- середнє значення очікуваних голів за 5 матчів;
- середнє значення очікуваних пропущених голів за 5 матчів;
- ціна команди;
- фактор поля.

Вчителем, тобто змінною, яку модель буде в подальшому прогнозувати, виступить показник очікуваних забитих голів для кожного матчу. Після навчання моделі отримано python-змінну, яка приймає аргументом список значень, кожне з яких відповідає характеристиці, використаній при навчанні.

У випадку, коли модель машинного навчання прогнозує конкретне числове значення або вирішує проблему класифікації рекомендується застосовувати

підхід, при якому частина даних відводиться для навчання, а інша частина - для перевірки якості навченої моделі. Також в таких випадках вираховують функцію втрат, яка використовується для кількісної оцінки втрати між реальним значенням та прогнозованим на етапі навчання у вигляді єдиного дійсного числа [42].

Вищезазначені підходи для підрахунку якості системи використовуються в системах, де значення вчителя є непорушними істинами, тобто набір показників-аргументів призводить до певного конкретного значення прогнозованої величини. Але у випадку футбольних матчів будь-який з показників якості не буде відображати реальної картини, оскільки результат футбольного матчу - це не закономірність, яка формується на статистичних характеристиках, бо результат може бути будь-яким, можна лише припустити, який буде більш імовірним, а який менш.

Припустимо, в вибірку для перевірки якості моделі потрапила велика кількість сенсаційних результатів, тобто таких, коли реальна кількість очікуваних голів для обраної команди сильно відрізняється від прогнозованої для цього матчу. Модель, яка працює справно, цілком обумовлено, спираючись на статистику, прогнозує очікуване значення, яке сильно відрізняється від реального. Під час підрахунку функції, яка оцінює якість моделі, це матиме сильний негативний вплив. У футболі подібні ситуації зустрічаються постійно, тому немає жодного сенсу застосовувати функції втрат або інші показники, які оцінюють якість роботи системи та розбивати дані на тренувальну та тестову групу.

Враховуючи сказане вище, найбільш оптимальним рішенням буде навчання та прогнозування на одних і тих самих даних, що при доволі великій кількості записів та правильності даних забезпечить адекватно працюючу модель, яка буде якісно прогнозувати значення заданої вчителем характеристики.

Тепер характеристики з кожного запису підставляються в отриману після навчання модель, яка прогнозує значення очікуваних забитих голів за наявних статистичних даних. Це прогнозоване значення записується в окрему колонку

датафрейму. Таким чином, тепер кожен запис в таблиці містить як реальний показник xG, так і прогнозований (рисунок 6.1).

	team	h_a	opponent	xG	xG_predicted
9567	Celta Vigo	0	Eibar	1.082340	1.385450
9568	Celta Vigo	1	Getafe	1.489020	1.729756
9569	Celta Vigo	0	Villarreal	1.624080	1.198043
9570	Celta Vigo	1	Real Madrid	1.576380	1.315135
9571	Celta Vigo	0	Real Sociedad	1.438860	1.206226
9572	Celta Vigo	1	Valencia	1.631700	1.588831
9573	Celta Vigo	0	Deportivo La Coruna	0.886052	1.256995
9574	Celta Vigo	1	Sporting Gijon	1.143760	1.851502

Рисунок 6.1 – Реальне та прогнозоване значення в таблиці

Спираючись на реальні футбольні матчі модель навчилася оцінювати прогнозоване значення очікуваних голів за наявних характеристик, і якщо реальне значення виявилося вищим, то команда перевершила очікування, і навпаки, за меншого значення, колектив зіграв гірше, ніж планувалося. Таким чином показником, який буде відповідати за форму команди в атаці стане середнє арифметичне між різницями прогнозованого та реального значення за останні п'ять поєдинків. Чим більш позитивний цей показник, тим краща форма колективу, і в зворотному напрямку так само.

Абсолютно аналогічний алгоритм необхідно застосувати для оцінювання форми команди в грі в захисті. Для цього треба використати ті самі характеристики-аргументи, але вчителем буде виступати показник очікуваних пропущених голів в поєдинку.

Після розрахунку показників форми команд в атаці та захисті частина таблиці виглядає наступним чином (рисунок 6.2):

	team	h_a	opponent	xG	xGA	xG_pred	xGA_pred	forma_xg5	forma_xgA5	forma_xg5Opp	forma_xgA5Opp
9567	Celta Vigo	0	Eibar	1.082340	1.209380	1.385450	1.336974	NaN	NaN	NaN	NaN
9568	Celta Vigo	1	Getafe	1.489020	1.320920	1.729756	0.875341	NaN	NaN	NaN	NaN
9569	Celta Vigo	0	Villarreal	1.624080	0.995361	1.198043	1.484729	NaN	NaN	NaN	NaN
9570	Celta Vigo	1	Real Madrid	1.576380	1.961600	1.315135	2.182658	NaN	NaN	NaN	NaN
9571	Celta Vigo	0	Real Sociedad	1.438860	1.135160	1.206226	1.619855	NaN	NaN	NaN	NaN
9572	Celta Vigo	1	Valencia	1.631700	1.482140	1.588831	1.371361	0.075214	-0.175427	-0.137735	0.655036
9573	Celta Vigo	0	Deportivo La Coruna	0.886052	1.028480	1.256995	1.288257	0.144410	-0.127752	-0.255487	-0.294896

Рисунок 6.2 – Розраховані характеристики форми для команд в матчі

Тепер, розраховані всі необхідні характеристики для моделі прогнозування результатів спортивних подій.

6.2 Позбавлення від зайвих записів в таблиці

В реальному футболі дві команди зустрічаються між собою двічі за сезон: на полі однієї команди, та іншої. Відповідно, в таблицю додається чотири записи, бо вона з самого початку була сформована для кожної команди в кожній зустрічі.

Оскільки метою обробки даних є саме створення характеристик домашньої та гостьової команди перед поєдинком, необхідно залишити виключно записи сформовані для команди-господаря, адже інформація про один і той самий матч в таблиці повторюється двічі. На рисунку 6.3 наведено таку пару записів.

	team	h_a	opponent	xG	xGA	xG_pred	xGA_pred	forma_xg5	forma_xgA5	forma_xg5Opp	forma_xgA5Opp
9543	Levante	0	Celta Vigo	1.42901	2.15710	0.831554	1.614555	-0.529235	-0.201327	-0.286670	-0.114360
9581	Celta Vigo	1	Levante	2.15710	1.42901	1.614555	0.831554	-0.286670	-0.114360	-0.529235	-0.201327

Рисунок 6.3 – Приклад дублікатів в таблиці

Ті значення, що в одному записі розташовуються в комірках для команди ‘team’, в другому записі знаходяться в відповідних комірках для команди-опоненту. Один з таких записів треба видалити.

Загальна схема позбавлення від зайвих строк таблиці зображена на кресленику ІА.81.090БАК.003 Д5 з правого боку. Дана схема має в основі ідею

залишити виключно записи, в яких характеристика h_a (на своєму чи на чужому стадіоні грала команда, відносно якої робився запис) дорівнює одиниці, тим самим половина записів з таблиці, які є дублікатами, видаляться.

Оскільки кількість даних для навчання і без того доволі велика, нема жодної необхідності жертвувати якістю заради збереження рядків таблиці, тому видалення зайвих записів буде позитивно впливати на якість і швидкість навчання моделі.

Також є необхідність позбавитись від записів, що містять порожні комірки. Це перші п’ять записів для кожної команди, оскільки показники форми в атаці та захисті потребують п’ять записів, щоб бути розрахованими.

Крім цього даний крок збільшить якість моделі: оскільки п’ять перших матчів кожної команди вже були видалені раніше, видалення ще п’ятьох призведе до того, що статистика для кожної команди буде вираховуватись з одинадцятого матчу сезону. В цьому випадку майже повністю нівелюється вплив окремих матчів, тому статистичні показники, які відображають середнє арифметичне показників за весь сезон будуть ще більш об’єктивно відображати реальну силу команд. За умови великою кількості записів позбавлення від менш точних виключно позитивно впливає на якість моделі.

Вигляд таблиці після видалення зайвих записів зображено нижче на рисунку 6.4.

	year	league	team	h_a	xG	xGA	npxG	npxGA	scored	missed	...	xGAMean5Opp	overall_costOpp	xGA_pred	xG_pred
11	2014	Bundesliga	Bayern Munich	1	2.086560	1.380140	2.086560	1.380140	4	0	...	1.422836	0.482630	0.664947	2.412662
13	2014	Bundesliga	Bayern Munich	1	0.947473	0.389245	0.947473	0.389245	1	0	...	0.701499	0.625523	0.830185	2.086186
15	2014	Bundesliga	Bayern Munich	1	2.983510	0.049757	2.983510	0.049757	2	0	...	1.516719	0.331030	0.419141	2.390274
18	2014	Bundesliga	Bayern Munich	1	0.376661	0.994575	0.376661	0.236798	1	1	...	1.103484	0.581312	0.678026	2.181785
20	2014	Bundesliga	Bayern Munich	1	4.178510	0.230619	3.420730	0.230619	8	0	...	1.421746	0.362467	0.450834	2.245885
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
25343	2021	Ligue_1	Clermont Foot	1	0.918981	1.955860	0.918981	1.195760	2	3	...	1.377722	0.389717	1.145048	1.265138
25344	2021	Ligue_1	Clermont Foot	1	1.028580	3.573750	1.028580	2.813660	1	6	...	1.189045	1.000000	2.044429	0.814685
25347	2021	Ligue_1	Clermont Foot	1	1.740020	1.398600	1.740020	1.398600	2	2	...	1.572254	0.322836	1.139256	1.387749

Рисунок 6.4 – Таблиця після видалення зайвих записів

6.3 Навчання моделі, яка прогнозує ймовірності результату матчу

В ході розробки були сформовані записи футбольних матчів п'яти найбільш популярних європейських чемпіонатів. Кожен з записів містить для обох команд статистичні дані, а також показники, які означають ціну, рівень залежності команд від свого стадіону, атакуючу та захисну форму колективів. Разом з цим в записах наявні реальні показники з даного матчу, серед яких є важливим результат зустрічі, який і буде вчителем для моделі.

Оскільки записи в наборі даних робилися відносно кожної команди в кожному матчі, то після зберігання записів виключно з "домашніх" поєдинків позначки в стовпці результату трактуються наступним чином:

– w - перемога домашньої команди;

– d – нічия;

– l - програш в домашньому матчі, тобто перемога гостьової команди.

Задача прогнозування переможця зустрічі - це задача класифікації, тому необхідно перетворити записи стовпця з результатом у номінативний числовий вигляд. Перемога домашньої команди стане одиницею, нічия - двійкою, перемога гостьової – трійкою (рисунок 6.5):

	team	opponent	result
4544	Villarreal	Getafe	1
4545	Villarreal	Espanyol	3
4546	Villarreal	Osasuna	1
4547	Villarreal	Levante	1
4548	Villarreal	Leganes	3
4549	Villarreal	Mallorca	1
4550	Villarreal	Sevilla	2

Рисунок 6.5 – Результат матчу після перетворення

Як вже було зазначено раніше, задачі класифікації в подібних системах часто вирішуються за допомогою логістичної регресії або штучної нейронної мережі.

В дослідженні 2020 року на тему застосування алгоритмів машинного навчання в прогнозуванні футбольних матчів були порівняні чотири різних види моделі, серед яких найкращі показники продемонструвала логістична регресія [43]. В статті порівнюватися чотири види алгоритмів: випадковий ліс (Random Forest), k-найближчих сусідів (k-nearest neighbors), безпосередньо логістична регресія та метод опорних векторів (Support Vector Machine).

Результати перевірки якості кожного з алгоритмів наведені нижче на рисунку 6.6.

Model	Accuracy
Random Forest	44.78% (4.10%)
KNN	45.65% (4.01%)
Logistic Regression	49.77% (4.02%)
SVM	47.15% (4.11%)

Рисунок 6.6 – Види алгоритмів та їх якість в дослідженні [43]

В іншому дослідженні на цю тему був використаний дещо інший набір алгоритмів: логістична регресія, випадковий ліс, нейронна мережа та LSTM - довга короткочасна пам'ять (одна з архітектур штучних нейронних мереж). Найбільшу точність на тестовій вибірці продемонстрували саме нейронні мережі (рисунок 6.7).

Model	Train Accuracy	Test Accuracy
Logistic Regression	0.6095693779904306	0.5368421052631579
Forrest Classifier	0.7853269537480064	0.40669856459330145
Neural Network	0.614673	0.56363636
LSTM	0.594736	0.582456

Рисунок 6.7 – Результати іншого дослідження [44]

Отже, прийняте рішення про побудову як логістичної регресії, так і штучної нейронної мережі. В подальшому, після перевірки працездатності кожної з них, буде обрана більш підходяща модель. Для подальшого використання при навчанні були обрані наступні вхідні характеристики для домашьої та гостьової команди:

– xGmean – середня кількість очікуваних забитих голів за сезон;
– xGAmean – середня кількість очікуваних пропущених голів за сезон;
– homeDependence – показник залежності від свого стадіону;
– overall_cost – показник ціни;
– forma_xG5 – форма в атаці за останні 5 матчів;
– forma_xGA5 – форма в захисті за останні 5 матчів.

Саме такий вибір характеристик обумовлений перш за все повною або відносної незалежністю однієї характеристики від іншої. В задачах машинного навчання необхідно уникати сильно корелюючих між собою показників. Таблиц кореляції характеристик системи зображена на рисунку 6.8.

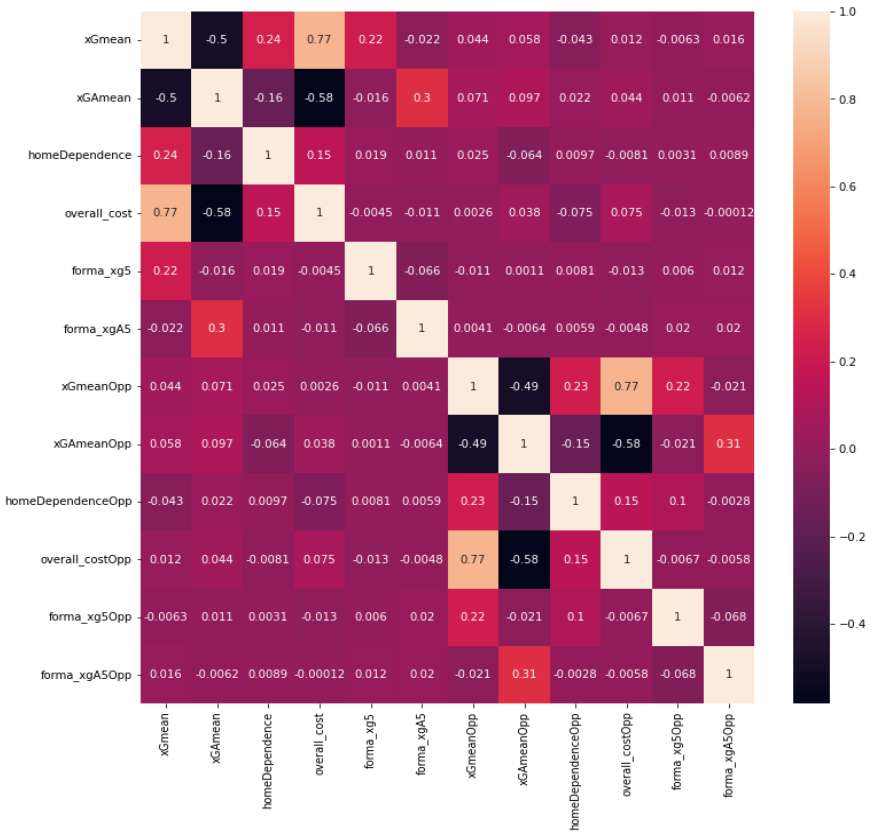


Рисунок 6.8 – Таблиця кореляції показників

Більшість характеристик жодним чином не пов'язана між собою, а ті з них, які мають високу кореляцію, не є напряму взаємопов'язаними.

Для оцінювання якості роботи моделі необхідно довільним чином розбити набір даних на дві частини: тренувальну – на записах якої буде відбуватися навчання, і тестову – на даних якої буде відбуватися оцінювання якості прогнозів моделі. Зазвичай 80% даних використовуються для тренування і 20% для тестування.

Механізм реалізації логістичної регресії засобами бібліотеки sklearn аналогічний реалізації лінійної регресії, тобто необхідно обрати характеристики на вхід та вчителя. Єдиною відмінністю слугує той факт, що вчителем в даному випадку є номінативна змінна. Використавши тренувальний набір для навчання оцінюється якість прогнозування на тестовій вибірці (рисунок 6.9).

```
model_def = LogisticRegression(max_iter=10000)
model_def.fit(X_train, Y_train)
model_def.score(X_test, Y_test)
```

```
0.5382113821138211
```

Рисунок 6.9 – Якість логістичної регресії

Даний показник означає, що близько 53,8% прогнозованих результатів поєдинків співпали з реальним результатом зустрічі. Цей відсоток є найпростішим показником якості системи.

Для розробки штучної нейронної мережі окрім подання даних на вхід і задання вчителя необхідно обрати топологію мережі, а саме кількість скритих шарів, а також кількість нейронів в кожному з них. Оптимальні значення цих показників зазвичай знаходяться експериментальним шляхом, адже аналітичним чином людині неможливо визначити найкращу кількість шарів та нейронів для вирішення конкретної задачі.

6.4 Вибір моделі прогнозування

Для того, щоб остаточно обрати модель, яка буде здійснювати прогнозування, необхідно декілька разів зробити наступне:

– довільним чином розділити дані на тренувальні та тестові;

– навчити логістичну регресію на тренувальних даних;

– оцінити якість роботи логістичної регресії на тестових даних;

– навчити моделі штучних нейронних мереж різних топологій на тренувальних даних;

– оцінити якість роботи моделей нейромереж на тестових даних;

– порівняти отримані показники якості для логістичної регресії та штучних нейронних мереж.

Завдяки такому підходу будуть використані різні топології на різних наборах даних для навчання та тестування, таким чином майже унеможливлюється упередженість для конкретного набору даних, натомість певні закономірності стануть помітними.

Для формування різних топологій була написана функція, яка на виході видає різні комбінації кортежів – python-об’єктів, які схожі на список, але є незмінними. Синтаксис sklearn зчитує інформацію про кількість шарів та нейронів саме з них. За кількість шарів відповідає кількість чисел в кортежі, а самі числа означають кількість нейронів на скритому шарі.

Для експерименту було сформовано кортежі, які задають від одного до трьох скритих шарів з різною кількістю нейронів. Частина з них зображена на рисунку 6.10.

```
(8, 8, 12),
(8, 8, 14),
(8, 8, 16),
(8, 10, 2),
(8, 10, 4),
(8, 10, 6),
```

Рисунок 6.10 – Приклад сформованих кортежів

Загалом вийшло 662 топології для розгляду.

Показники якості прогнозування для кожної топології будуть записуватися в словник, ключем якого стане кортеж відповідної топології, а значенням – точність прогнозування. Після запису до словника його буде відсортовано за значенням, таким чином отримано найкращі та найгірші топології для тестового набору даних (рисунок 6.11).

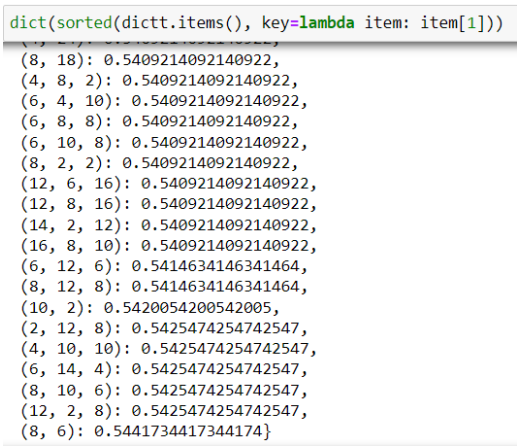


Рисунок 6.11 – Показники з найкращими топологіями

Після застосування раніше описаного алгоритму декілька разів зроблені наступні висновки:

– в більшості найгірші показники точності демонструють моделі з двома нейронами на останньому скритому шарі;

– в залежності від розбиття даних на тренувальні та тестові, одні й ті самі топології можуть демонструвати різний результат. Ті топології, які були кращими під час певного розбиття, під час іншого експерименту були в середині відсортованого списку, пропустивши перед собою сотні інших топологій;

– показники якості логістичної регресії завжди входили в найкращі 10% записів;

– різниця між показниками якості доволі невелика. Одні з найгірших топологій демонструють на 3-4% гірші показники, ніж найкращі.

Зважаючи на результати, було прийняте рішення залишити модель логістичної регресії для прогнозування результатів футбольних матчів. Головна її перевага для розроблюваної системи – стабільно високі показники якості під час кожного з експериментів, на відміну від нейронної мережі, для якої так і не вдалося підібрати оптимальну топологію.

Висновки до розділу 6

В ході даного розділу було описане практичне застосування методів машинного навчання для розроблюваної системи.

Завдяки алгоритму лінійної регресії за допомогою характеристик, сформованих в попередньому розділі, вдалося вивести додаткову характеристику, яка об’єктивним чином оцінює рівень гри команд в останніх поєдинках, що дозволяє оцінити форму, з якою команди підходять до зустрічі.

Після остаточного формування характеристик таблиця з даними була приведена до необхідного вигляду і використана вже безпосередньо в навчанні моделі, яка за наявних статистичних та додатково виведених показників прогнозує, як закінчиться зустріч та з якою ймовірністю.

Врешті-решт було розглянуто два алгоритми, які спроможні вирішити задачу класифікації, до якої належить задача прогнозування результату футбольного матчу. Між логістичною регресією та штучною нейронною мережею було обрано перший варіант, оскільки саме регресія продемонструвала більш стабільну та в переважній більшості більш якісну роботу.

7 АКТУАЛЬНІ ДАНІ

7.1 Збір актуальних даних

Актуальна інформація, зчитана з ресурсу Understat вже була використана разом з готовими даними для навчання моделі.

Оскільки збір актуальної інформації повинен бути відведений в окремий скрипт, то необхідно буде саме повторити кроки, які описані в попередніх розділах для отримання усіх необхідних характеристик. Але є дві речі, про які треба подбати.

По-перше, про виведення змінної, яка позначає наскільки команда залежна від домашнього поля. Необхідно нормалізувати різницю прогнозованих і очікуваних голів саме використовуючи середнє арифметичне і стандартне відхилення вибірки. Для цього необхідно зберегти таблицю з різницями прогнозованих та реальних очікуваних забитих та пропущених голів, яка містить всі дані (рисунок 7.1). Врешті, завантаживши її до скрипту збору актуальних даних, отримано доступ до середнього арифметичного та стандартного відхилення вибірки, відносно якої відбувалася нормалізація змінних, які приймали участь у навчанні моделі, тобто нормалізація буде виконана правильно відносно навченої моделі машинного навчання (рисунок 7.2).

```
df.to_excel("absolutely_all_matches_with_costs_and_range_diplom.xlsx", index=False)
```

Рисунок 7.1 – Зберігання таблиці з усіма даними для навчання

```
all_matches = pd.read_excel("absolutely_all_matches_with_costs_and_range.xlsx")

xGMeanHomeDiff_mean = all_matches['xGMeanHomeDiff'].mean()
xGMeanHomeDiff_std = all_matches['xGMeanHomeDiff'].std()
```

```
df['xGMeanHome'] = (df['xGMeanHome'] - xGMeanHomeDiff_mean)/xGMeanHomeDiff_std
df['xGMeanHomeOpp'] = (df['xGMeanHomeOpp'] - xGMeanHomeDiff_mean)/xGMeanHomeDiff_std
```

Рисунок 7.2 – Завантаження збереженої таблиці та використання її значень для правильного виведення показників

По-друге, для виведення змінних очікуваних забитих та пропущених голів за наявних характеристик необхідно використати моделі, які були навчені на всіх даних, які приймали участь у навчанні. Для цього необхідно використати бібліотеку joblib, яка дозволяє зберігати моделі машинного навчання у вигляді файлів, а згодом завантажувати їх з файлів. Причина, за якою був введений даний крок, така сама як для попередньої ситуації: характеристики, які приймали участь в навчанні моделі, яка прогнозує результат поєдинку, були виведені саме за допомогою цих моделей лінійної регресії, отже для того, щоб виведені дані були правильними для моделі, яка прогнозує результат, характеристики треба виводити якраз за допомогою них, а не перевчати моделі прогнозування xG та xGA виключно на нових даних. Збереження та завантаження з використання моделі представлене на рисунках 7.3 і 7.4 відповідно.

```
Ввод [58]: import joblib
           joblib.dump(model_xG_predicted, 'XG.sav')
           joblib.dump(model_xGA_predicted, 'XGA.sav')
```

Рисунок 7.3 – Збереження моделей, отриманих на всіх даних

```
: import joblib
  xg_pred = joblib.load('XG.sav')
  xgA_pred = joblib.load('XGA.sav')

: def predicted_xg_xga(row):
      row['xGA_pred'] = xgA_pred.predict([row[['xGmean', 'xGAmean', 'ScoredMean', 'MissedMean', 'homeDependence', 'xGmean5', 'xGAMe
                          'xGmeanOpp', 'xGAmeanOpp',  'ScoredMeanOpp', 'MissedMeanOpp', 'homeDependenceOpp', 'xGmean5Opp', 'xGAMean5Opp
      row['xG_pred'] = xg_pred.predict([row[['xGmean', 'xGAmean', 'ScoredMean', 'MissedMean', 'homeDependence', 'xGmean5', 'xGAMean
                         'xGmeanOpp', 'xGAmeanOpp',  'ScoredMeanOpp', 'MissedMeanOpp', 'homeDependenceOpp', 'xGmean5Opp', 'xGAMean5Opp
      return row

  df = df.apply(predicted_xg_xga, axis = 1)
```

Рисунок 7.4 – Завантаження моделей та виведення характеристик для актуальних даних

7.2 Виведення необхідних характеристик для актуальних даних

На даний момент в записах таблиці фігурують значення очікуваних забитих та пропущених голів в зіграних матчах, а також дані по різним характеристикам, з якими команди підходили до цього матчу. Для того, щоб прогнозувати результат

футбольного поєдинку, необхідно вивести характеристики, які використовуються в раніше виведеній остаточній моделі прогнозування.

Для виведення середнього значення очікуваних забитих та пропущених голів необхідно для кожної команди вивести середнє арифметичне реальних показників xG та xGA за сезон.

Показник відносної ціни команд наявний в таблиці з актуальними даними по матчам, зіграних в цьому сезоні, тому необхідно просто її додати для кожної команди.

Для виведення характеристик форми команд в атаці та захисті необхідно за останні п'ять поєдинків розрахувати середнє значення різниць прогнозованих та реальних показників очікуваних забитих та пропущених голів.

Для виведення характеристики, яка позначає наскільки команда залежна від свого стадіону необхідно вивести нормалізоване значення різниці очікуваних забитих голів на своєму полі та очікуваних забитих голів загалом.

Таким чином виводяться всі необхідні для прогнозування характеристики відносно кожної команди. Саме ці показники, що зображені на рисунку 7.5, були використані в моделі прогнозування результатів.

	league	team	xGMean	xGAMean	overall_cost	formaXG	formaXGA	xGMeanHome	homeDependence
0	Bundesliga	Arminia Bielefeld	0.800717	2.048987	0.286521	-0.177728	0.708361	0.815073	-0.676705
1	Bundesliga	Augsburg	1.145607	1.912418	0.335856	-0.197232	0.430187	1.417288	0.472514
2	Bundesliga	Bayer Leverkusen	1.950365	1.323975	0.721137	0.266367	-0.158301	1.911733	-0.913344
3	Bundesliga	Bayern Munich	2.938398	1.135578	1.000000	-0.221245	0.559547	2.898941	-0.917031
4	Bundesliga	Bochum	1.317112	1.711131	0.235092	0.057111	0.773365	1.460064	-0.102390
...	...	...	...	...	...	...	...	...	...
93	Serie_A	Spezia	1.096025	1.830215	0.321730	0.631683	0.432308	1.185475	-0.341333
94	Serie_A	Torino	1.414566	1.081907	0.585606	0.335405	0.188427	1.663069	0.369003
95	Serie_A	Udinese	1.504490	1.421373	0.424815	0.763348	0.785585	1.560882	-0.488971
96	Serie_A	Venezia	0.951533	2.035812	0.351799	0.116417	0.438413	1.192276	0.334347
97	Serie_A	Verona	1.483829	1.369457	0.466714	-0.046437	0.571774	1.418941	-1.030607

Рисунок 7.5 – Виведені характеристики для кожної команди в актуальному сезоні

7.3 Запис актуальних характеристик для кожної команди в SQL-таблицю

Для того, щоб під час використання майбутнього застосунку отримати швидкий доступ до статистичних даних команд необхідно використовувати SQL-таблицю в обраній РСУБД. Зберігання даних в такому форматі позитивно вплине на швидкодію застосунку, а функціонал Python дозволяє швидко перетворювати результати SQL-запитів в об'єкти даної мови програмування.

Актуальні дані на цей момент часу представлені у вигляді таблиці, де характеристикам однієї команди відповідає один запис в датафреймі. У вигляді SQL-таблиці представлення буде таким самим. Оскільки додаток буде містити візуальну складову, а саме емблеми команд, необхідно додати до таблиці з даними посилання на картинку, щоб обравши строку з бажаною командою була одразу отримана її емблема.

Для реалізації задуманого було прийняте рішення пронумерувати строки датафрейму починаючи з одиниці і відвести ці значення в окрему колонку з назвою ‘id’. Після цього, зважаючи на номер, який отримано для кожної команди, утворено посилання на картинку, яка повинна містити емблему заданої команди. Результат перетворень зображено на рисунку 7.6.

	league	team	xGMean	xGAMean	overall_cost	homeDependence	formaXG	formaXGA	location	id
0	Bundesliga	Arminia Bielefeld	0.800717	2.048987	0.286521	-0.676705	-0.177728	0.708361	logos/resized/1.png	1
1	Bundesliga	Augsburg	1.145607	1.912418	0.335856	0.472514	-0.197232	0.430187	logos/resized/2.png	2
2	Bundesliga	Bayer Leverkusen	1.950365	1.323975	0.721137	-0.913344	0.266367	-0.158301	logos/resized/3.png	3
3	Bundesliga	Bayern Munich	2.938398	1.135578	1.000000	-0.917031	-0.221245	0.559547	logos/resized/4.png	4
4	Bundesliga	Bochum	1.317112	1.711131	0.235092	-0.102390	0.057111	0.773365	logos/resized/5.png	5
...	...	...	...	...	...	...	...	...	...	...
93	Serie_A	Spezia	1.096025	1.830215	0.321730	-0.341333	0.631683	0.432308	logos/resized/94.png	94
94	Serie_A	Torino	1.414566	1.081907	0.585606	0.369003	0.335405	0.188427	logos/resized/95.png	95
95	Serie_A	Udinese	1.504490	1.421373	0.424815	-0.488971	0.763348	0.785585	logos/resized/96.png	96
96	Serie_A	Venezia	0.951533	2.035812	0.351799	0.334347	0.116417	0.438413	logos/resized/97.png	97
97	Serie_A	Verona	1.483829	1.369457	0.466714	-1.030607	-0.046437	0.571774	logos/resized/98.png	98

Рисунок 7.6 – Датафрейм з актуальними даними та посиланням на картинку

Тепер таблиця з даними містить всю необхідну інформацію, яку необхідно передати в SQL-таблицю.

Бібліотеки sqlalchemy та pandas вдвох дозволяють передати датафрейм в базу даних й утворити відповідну таблицю в ній. За допомогою sqlalchemy створюється об’єкт, який утворює з’єднання з базою даних в обраній РСУБД, а pandas містить для об’єкту датафрейму метод, який перетворює його на таблицю в базі даних.

Врешті в базі даних було отримано таблицю, що зображена на рисунку 7.7.

```
227 •  select * from actual_data_of_top5_leagues;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

league	team	xGMean	xGAMean	overall_cost	homeDependence	formaXG	formaXGA	location	id
Bundesliga	Arminia Bielefeld	0.8007172647058823	2.048986823529412	0.2865211618921424	-0.6767046782699062	-0.1777282605959237	0.7083606425272839	logos/resized/1.png	1
Bundesliga	Augsburg	1.1456073235294117	1.9124179264705883	0.3358555727394473	0.4725140806781 86	-0.19723159961898679	0.4301866893915018	logos/resized/2.png	2
Bundesliga	Bayer Leverkusen	1.9503645588235292	1.3239753823529412	0.7211371469732141	-0.9133443458299356	0.26636721875785146	-0.15830142015620333	logos/resized/3.png	3
Bundesliga	Bayern Munich	2.938397794117647	1.135577588235294	1	-0.9170309136647867	-0.2212445538385198	0.5595474376418113	logos/resized/4.png	4
Bundesliga	Bochum	1.3171116470588236	1.7111308823529412	0.2350920269822311	-0.10239045015069888	0.05711092207651007	0.7733646147941741	logos/resized/5.png	5
Bundesliga	Borussia Dortmund	1.9307729117647057	1.4021567352941178	0.7642209192670522	0.6653507651925746	0.12286691423332212	0.3253604932614411	logos/resized/6.png	6
Bundesliga	Borussia M.Gladbach	1.7624125	1.785645294117647	0.5232166435699435	0.3449821619771373	0.37587100789663874	0.35736993399776307	logos/resized/7.png	7
Bundesliga	Eintracht Frankfurt	1.2250006470588235	1.5994641470588236	0.4483684420850661	-0.4700095408439567	-0.39080673493237494	-0.20598113310945224	logos/resized/8.png	8
Bundesliga	FC Cologne	1.5382384411764705	1.5615422941176471	0.3082573996425538	-0.3634262752465018	0.52416025159094 64	0.24755104119957175	logos/resized/9.png	9
Bundesliga	Freiburg	1.6979038529411765	1.443667794117647	0.4351997661507414	0.4053869161567784	0.5219449860275868	0.2611139949214877	logos/resized/10.png	10
Bundesliga	Greuther Fuerth	0.9010518382352942	1.7807169411764705	0.1977573029565704	0.19213914281101596	-0.27978128487273807	-0.6561791877300523	logos/resized/11.png	11

Рисунок 7.7 – Таблиця в РСУБД MySQL

Таким чином дана SQL-таблиця є повною копією відповідного датафрейму з актуальними даними для всіх команд п’яти найбільш відомих футбольних чемпіонатів Європи.

Висновки до розділу 7

В ході даного розділу з ресурсу Understat був проведений парсинг актуальних даних, які містять всі необхідні характеристики для подальшого використання в розроблюваній системі.

Після зчитування інформації були виведені всі актуальні показники, які використовуються в моделі машинного навчання, що прогнозує результат спортивної події.

Таблиця, яка містить всі необхідні характеристики врешті була перетворена у відповідну SQL-таблицю.

8 РОЗРОБЛЕННЯ ЗАСТОСУНКУ

8.1 Підготовка емблем колективів

Для якісної графічної реалізації необхідно виводити емблеми обраних команд у головному вікні програми. Шлях до картинок, які будуть відображати логотипи кожної команди вже був заздалегідь доданий до SQL-таблиці з актуальною інформацію.

З головної програми за допомогою SQL-запитів будуть отримані дані про кожну з команд, в тому числі й шлях до картинки. Таким чином необхідно в кореневій папці проєкту створити папку з логотипами, шлях до якої співпадає з тим, що записаний в таблиці в базі даних, після чого додати туди картинки емблем, назва яких співпадатиме з записами.

Для кожної команди в інтернеті було знайдено та завантажено логотип в форматі png. З метою якісного відображення картинок в ході роботи програми існує необхідність змінити розмір кожного зображення у пікселях. Після перетворення картинок до розміру 200x200 пікселів вони були збережені в відведену для них папку (рисунок 8.1).

Рисунок 8.1 – Файли зображень з логотипами

8.2 Розроблення графічного інтерфейсу

Як вже було зазначено раніше, для графічної реалізації додатку буде використано бібліотеку Tkinter. Вона доволі легка у застосуванні порівняно з іншими фреймворками. Це робить її привабливою для створення додатків із графічним інтерфейсом на Python, особливо для програм, де не потрібні складні конструкції, а головним пріоритетом є швидке створення чогось функціонального та кросплатформеного [45].

Метод grid() для об’єктів бібліотеки працює шляхом розбиття вікна або рамки на рядки та стовпці. Для того, щоб вказати розташування об’єкту візуалізації, необхідно визвати саме цей метод та в його аргументах передати рядок та стовпець, де необхідно розташувати обраний об’єкт. Загалом в головному вікні програми будуть використані наступні об’єкти:

– надпис;

– випадаючий список;

– фрейм.

На прикладі на рисунку 8.2 зображене розбиття на рядки та колони для кожного елемента системі.

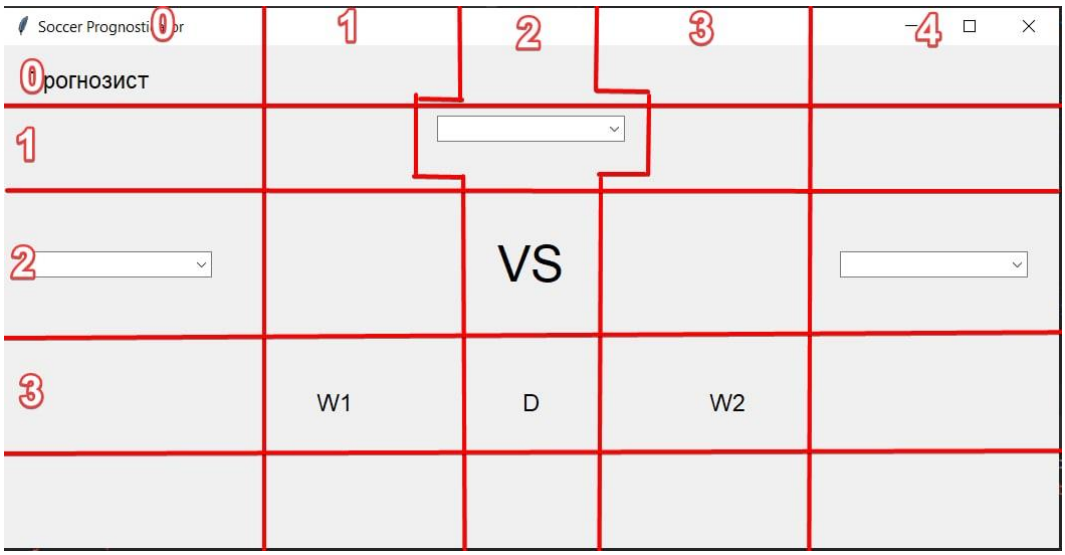


Рисунок 8.2 – Базова графічна реалізація програми

8.3 Розроблення функціоналу програми

Оскільки головною метою програми є прогнозування результату футбольного поєдинку за обраних двох команд, то необхідно до застосунку додати модель логістичної регресії, яка вже була розроблена в ході роботи. Бібліотека joblib долволяє як зберігати, так і завантажувати моделі, тому, використавши функціонал бібліотеки отримано той самий python-об’єкт, який був збережений в ході навчання моделі. Цей об’єкт приймає аргументом список характеристик та на виході видає прогнозованого переможця поєдинку та, за необхідністю, ймовірності кожного результату.

Показники кожної команди будуть отримані з SQL-таблиці, тому, як і раніше, необхідно встановити з'єднання з відповідною базою даних за допомогою sqlalchemy. Функціонал даного модуля дозволяє виконувати SQL-запити безпосередньо під час роботи програми.

Робота програми починається з вибору чемпіонату в верхньому випадаючому меню. Для того, щоб заповнити його, необхідно отримати перелік всіх ліг. Це робиться за допомогою SQL-запиту, який виводить унікальні значення стовпця таблиці. Застосувавши його для колонки ліг через метод бібліотеки sqlalchemy, отримано об’єкт який перетворюється в список. Даний список вже може бути використано як аргумент випадаючого меню (рисунок 8.3).

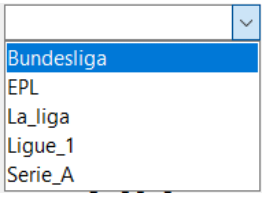


Рисунок 8.3 – Випадаюче меню з лігами

Заповнення відповідного меню лігами відбувається автоматично під час запуску застосунку, таким чином користувач одразу отримує доступ до нього.

Після того, як користувач обирає лігу, необхідно заповнити випадаючі меню відведені для команд обраного чемпіонату. Для того, щоб зробити це в автоматичному режимі необхідно використати callback-функцію. Це функція, яка буде викликана у відповідь на якусь подію. У випадку даної ситуації подією буде зміна компонента в випадаючому меню з чемпіонатами. Як тільки-но буде змінено обраний компонент, задана функція виконається.

Під час вибору ліги, обране значення буде збережено в текстову змінну. Одразу після цього буде виконано callback-функцію, змістом якої буде SQL-запит в базу даних, що видає команди, ліга яких дорівнює значенню, записаному в текстовій змінній, тобто обраному чемпіонату. Врешті список колективів буде передано, як аргумент двох випадаючих меню, відведених під команди. Приклад одного з них за обраної ліги наведено на рисунку 8.4.

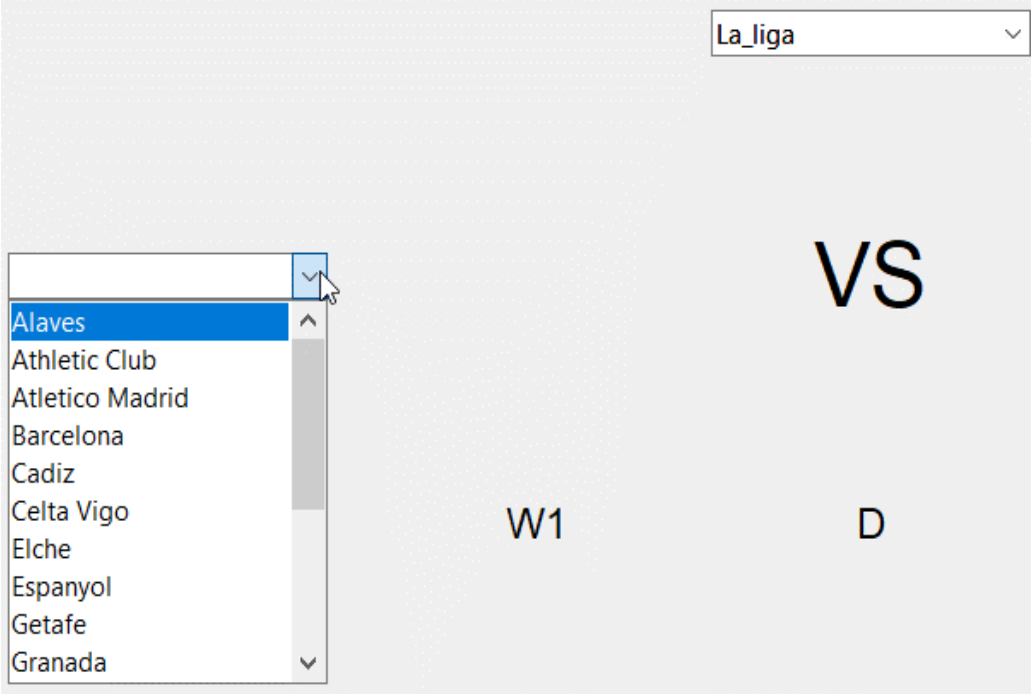


Рисунок 8.4 – Випадаюче меню з командами обраної ліги

Вибір команд повинен супроводжуватись виведенням емблеми обраних колективів. Для кожного з випадаючих меню з командами також написана callback-функція, яка виконується під час зміни компоненту меню. Дані функції,

так само, як і в попередніх випадках, представляють собою SQL-запит, які видають шлях до картинки з логотипом обраної команди. Тепер у головному вікні програми з’являється картинка, обрана за отриманим шляхом (рисунок 8.5).

Рисунок 8.5 – Виведення емблеми після вибору команди

Окрім вищезазначеного функціоналу дані callback-функції реалізують основну задачу застосунку: прогнозування результату події за обраних команд. Кожна з цих функцій після вибору команди робить SQL-запит з метою отримати характеристики обраної команди з якими вона підходить до футбольного матчу. Результат запиту перетворюється у список Python.

Оскільки бажана реалізація повинна автоматично розраховувати ймовірності перемоги кожної з обраних команд, необхідно зробити даний підрахунок одразу після вибору другої команди. Так як команди обираються в довільному порядку, то доречно для кожної з callback-функцій зробити перевірку, чи обрана друга команда. У випадку позитивної відповіді, списки з характеристиками обраних команд об'єднуються в єдиний список показників домашньої та гостьової команди, який і є аргументом моделі логістичної регресії, що прогнозує переможця за наявними характеристиками.

Врешті-решт, модель прогнозує переможця зустрічі за отриманим списком. Для того, щоб вивести ймовірності одержати кожну з можливих номінативних змінних на виході моделі, тобто отримати ймовірності кожного з трьох результатів зустрічі, необхідно використати спеціальний метод моделі. Отримані

значення ймовірності множаться на 100 з метою отримати значення у відсотках, і записуються в текстову змінну, які виводяться в призначених для цього місцях.

Остаточний результат роботи програми зображено нижче на рисунку 8.6.

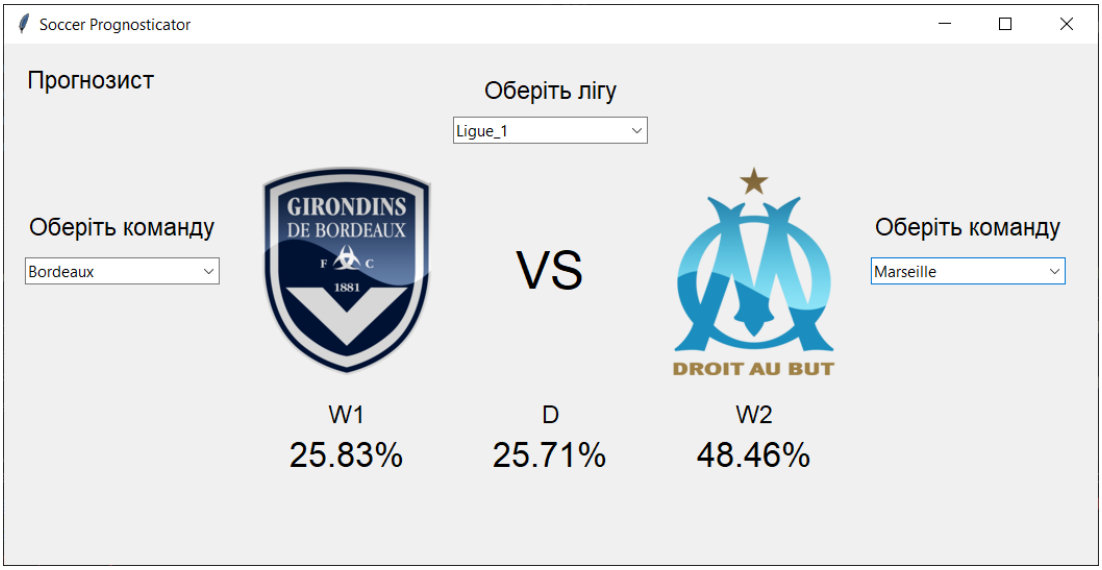


Рисунок 8.6 – Результат роботи

Висновки до розділу 8

В ході даного розділу був описаний процес розробки програми для реалізації системи прогнозування результатів спортивних подій на основі зібраної статистики.

Для покращення візуальної складової були завантажені та змінені зображення з емблемами футбольних клубів, які наявні в розроблюваній системі.

В результаті реалізації графічного інтерфейсу за допомогою бібліотеки Tkinter вдалося досягнути бажаного зовнішнього виду програми, відштовхуючись від якого був реалізований необхідний функціонал.

9 ПЕРЕВІРКА ЯКОСТІ РОБОТИ РОЗРОБЛЕНОЇ СИСТЕМИ

9.1 Запис реальних та прогнозованих ймовірностей

Для того, щоб перевірити якість прогнозування результатів футбольних матчів необхідно порівняти отримані ймовірності з коефіцієнтами, які надають букмекерські контори, адже саме вони надають найбільш об’єктивну інформацію про можливі результати майбутнього матчу.

Щоб отримати інформацію про коефіцієнти було використано ресурс Oddsportal, який містить історію зміни коефіцієнтів на велику кількість матчів в чималій кількості букмекерських контор (рисунок 9.1).

Рисунок 9.1 – Історія зміни коефіцієнтів

Починаючи з квітня 2022 року для матчів п’яти найбільш відомих чемпіонатів Європи були виписані стартові, кінцеві, найбільші та найменші відносно перемоги домашньої коефіцієнти, а також прогнозовані розробленою системою ймовірності перемоги команд.

Для того, щоб перевести коефіцієнти, які видає букмекерська контора, в ймовірності необхідно скористатися формулою 9.1.

$$p(res = i) = \frac{1}{k(res=i)} / \left(\frac{1}{k(res=w1)} + \frac{1}{k(res=d)} + \frac{1}{k(res=w2)}\right), i = \{w1, d, w2\} \quad (9.1)$$

де i – можливий результат матчу, k (res = i) – коефіцієнт на один з результатів матчу, w1, d, w2 – перемога домашньої команди, нічия та перемога гостьової команди відповідно.

Після застосування формули для кожного з поєдинків отримано набори ймовірностей на кожну з п’ятьох раніше зазначених ситуацій. На прикладі двох матчів отримані записи зображені на рисунку 9.2.

	MU		Norwich									
start	1,25	6,75	13,00	80,00%	14,81%	7,69%	102,51%	2,51%	78,04%	14,45%	7,50%
end	1,29	6,45	10,50	77,52%	15,50%	9,52%	102,55%	2,55%	75,59%	15,12%	9,29%
min	1,24	6,80	14,00	80,65%	14,71%	7,14%	102,49%	2,49%	78,68%	14,35%	6,97%
max	1,32	5,90	10,25	75,76%	16,95%	9,76%	102,46%	2,46%	73,94%	16,54%	9,52%
									76,52%	16,44%	7,04%
	WHU		Burnley								
start	1,63	4,15	5,85	61,35%	24,10%	17,09%	102,54%	2,54%	59,83%	23,50%	16,67%
end	1,83	3,88	4,55	54,64%	25,77%	21,98%	102,40%	2,40%	53,37%	25,17%	21,46%
min	1,63	4,15	5,85	61,35%	24,10%	17,09%	102,54%	2,54%	59,83%	23,50%	16,67%
max	1,86	3,85	4,40	53,76%	25,97%	22,73%	102,46%	2,46%	52,47%	25,35%	22,18%
									61,62%	23,08%	15,30%

Рисунок 9.2 – Ймовірності, оцінені букмекером на різних етапах та розробленою системою

– перший рядок – ймовірності, отримані з початкових коефіцієнтів;

– другий рядок – ймовірності, отримані з кінцевих коефіцієнтів;

– третій рядок – ймовірності за найменшого коефіцієнта на перемогу домашньої команди;

– четвертий рядок – ймовірності за найбільшого коефіцієнта на перемогу домашньої команди;

– п'ятий рядок – прогнозовані розробленою системою ймовірності.

9.2 Порівняння реальних та прогнозованих значень

На протязі декількох тижнів проводився моніторинг коефіцієнтів букмекерських контор на зіграні футбольні матчі. Значення були переведені в ймовірності та записані та записані в таблицю у вигляді представленому на рисунку 9.2.

Вдалося зібрати інформацію про 143 гри в п’яти найбільш відомих чемпіонатах Європи.

Після аналізу реальних та прогнозованих значень на прикладі ймовірності перемоги домашньої команди отримано наступне:

– 37% прогнозованих значень ймовірності потрапило в діапазон між реальним мінімальним та максимальним значенням даного критерія в обраній для аналізу букмекерській конторі;

– 72% прогнозованих значень ймовірності потрапило в діапазон між реальним мінімальним та максимальним значенням даного критерія +-3% в обраній для аналізу букмекерській конторі;

– 4 з 143 прогнозованих значення мають відхилення більш ніж 10% від найменшого чи найбільшого реального показника даного критерія;

– середня різниця між прогнозованою ймовірністю та середнім арифметичним між реальним максимальним та мінімальним значенням ймовірності перемоги домашньої команди становить 1,64%;

– середня різниця між прогнозованою ймовірністю та початковим значенням ймовірності перемоги домашньої команди становить 1,62%;

– середня різниця між прогнозованою ймовірністю та кінцевим значенням ймовірності перемоги домашньої команди становить 1,33%;

– ті ймовірності, які відхилилися більше ніж на 7% від діапазону в більшості випадків стосуються матчів, де одна з команд за декілька днів до реального поєдинку зіграла інший матч. Розроблена система не враховує даний фактор.

Також була побудована гістограма між різницею прогнозованого значення ймовірності перемоги домашньої команди та середнім між реальним мінімальним та максимальним значенням. Вона наведена на рисунку 9.3.

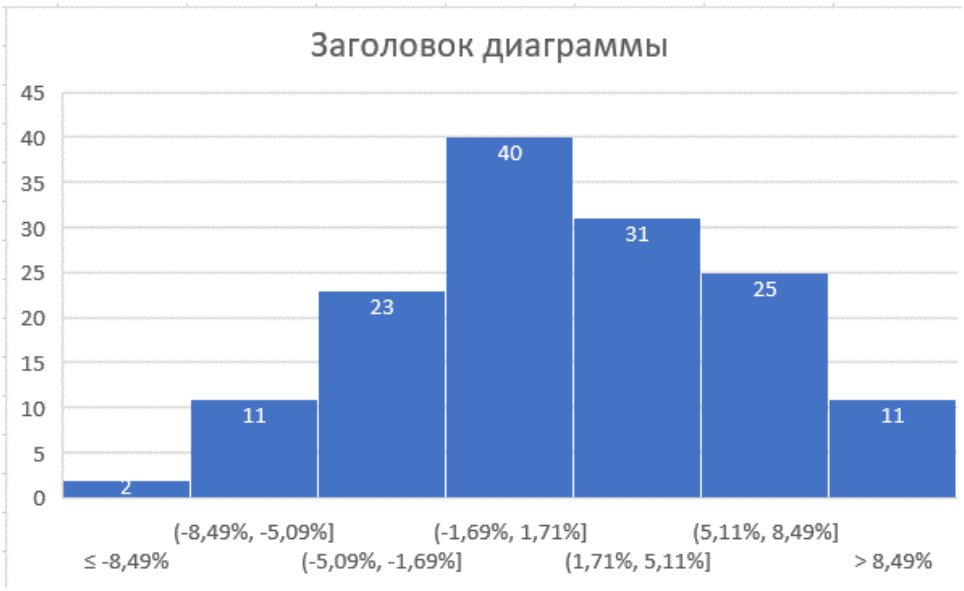


Рисунок 9.3 – Гістограма відхилень від середнього прогнозованого значення

Прогнозування розробленою системою відбувається доволі якісно. Простежується тенденція, що прогнозовані ймовірності ближчі до кінцевих значень коефіцієнтів букмекера, ніж до початкових. Також помітно, що система прогнозування зазвичай трохи переоцінює ймовірність перемоги домашньої команди, ніж це робить букмекер, в середньому на 1,64%.

В реальному футболі чималий вплив мають інші фактори, які тяжко, або навіть неможливо, включити в модель прогнозування, наприклад: мотивація команд, стартовий склад, дискваліфікації, погода, кількість днів після останнього поєдинку в команд (багато колективів приймають участь в міжнародних змаганнях), стиль гри тощо. Букмекери, в свою чергу, спроможні врахувати дані фактори, оскільки їхні системи є професійними, а також вони враховують людський фактор.

Оскільки модель машинного навчання, яка прогнозує результат поєдинку враховує виключно статистичні та відносні характеристики, результат розробки вважається успішним.

Висновки до розділу 9

В ході даного розділу було проведено порівняння ймовірностей, які пропонує розроблена система з тими значеннями, які пропонують букмекерські контори. Для цього були зібрані показники ймовірностей за деякий період час часу.

В ході аналізу отриманих значень було доведено високу якість роботи системи, яка була розроблена в ході даного індивідуального дослідницького проєкту. Показники, які виводилися нею, були близькими до коефіцієнтів, які формують професійні системи.

ВИСНОВКИ

В ході даної індивідуальної дослідницької роботи була розроблена система прогнозування результатів спортивної події.

Першочерговою метою було створити систему, яка буде конкурентоспроможною для професійних аналогів - тих, які використовують букмекерські контори, і завдяки яким дана індустрія є однією з найбільших прибуткових в світі. Перевершити професійні системи, які враховують тисячі характеристик, що впливають на формування коефіцієнтів за умови обмеженості даних неможливо.

Система прогнозування спортивної події була реалізована для футбольних матчів п'яти найбільш відомих чемпіонатів Європи: англійського, німецького, іспанського, італійського та французького. Для початку був знайдений набір даних, який містить показники, що дозволять системі претендувати на високу якість. Це показники, які відображають об'єктивну силу команд в обраному матчі. В подальшому в процесі роботи з даними були виведені додаткові характеристики, в тому числі методами машинного навчання. Всі показники є повністю або майже незалежними один від одного, майже не корелювали між собою, але при цьому несуть великий вплив на прогнозування результату майбутнього поєдинку.

Надалі з даних були сформовані безпосередньо футбольні матчі, що містять всі необхідні характеристики для моделі, яка буде прогнозувати результат. В ході аналізу було прийняте рішення вирішувати ключову задачу розроблюваної системи через алгоритм логістичної регресії. Модель була навчена на наявних даних та збережена для подальшого використання в системі.

Завдяки збору актуальної інформації з того самого ресурсу, з якого був взятий першочерговий набір даних, реалізується механізм самонавчання моделі логістичної регресії, що прогнозує результат, а також в ході аналізу зібраної

інформації безпосередньо отримуються всі необхідні для даної моделі актуальні характеристики команд обраних п'яти чемпіонатів.

Після отримання необхідної інформації про поточний стан клубів, таблиця з відповідними даними була збережена в РСУБД, щоб пізніше отримати до неї доступ.

Для зручного користування системою був розроблений додаток, що містить інтуїтивно зрозумілий інтерфейс: необхідно вибрати чемпіонат та пару команд, щоб на екрані вивелися ймовірності кожного з можливих результатів поєдинку. Прогнозування відбувається завдяки завантаженню раніше навченої моделі машинного навчання, а інформація про команди дістається з SQL-запитів.

З метою перевірки якості роботи системи прогнозування спортивних подій на основі зібраної статистики було проведене порівняння ймовірностей, що видає розроблена система та ймовірностей, що видають букмекерські контори в різні етапи часу. Результат перевірки показав, що зроблена під час написання даної індивідуальної дослідницької роботи система продемонструвала хорошу якість, а ймовірності, яка вона виводить близькі до тих, що формуються в найбільших букмекерських конторах професійними аналогами.

Для оновлення актуальної інформації про команди необхідно запустити відповідний скрипт, який зчитає дані з веб-ресурсу, виведе необхідні характеристики та збереже їх у відповідну SQL-таблицю, до якої має доступ модель машинного навчання з застосунку.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Засновник Parimatch правильно обрав наступників. | Велика історія компанії. URL: https://forbes.ua/ru/company/zasnovnik-parimatch-pravilno-obrav-nastupnikiv-yogo-donka-ta-sin-druga-peretvorili-bukmekersku-merezhu-v-internet-kholding-na-550-mln-velika-istoriya-kompanii-14122021-2963 (дата звернення: 03.04.2022).
2. Компания Parimatch стоит уже $1 млрд, вот как они строят бренд. URL: https://mc.today/makgregor-akterishhe-tajson-nakurennyj-filosof-kompaniya-parimatch-stoit-uzhe-1-mlrd-vot-kak-oni-stroyat-brend/ (дата звернення: 03.04.2022).
3. Отзывы о Soccer Match Predictor. URL: https://kapper-ratings.ru/soccer-match-predictor/ (дата звернення: 03.04.2022).
4. OnCourt. URL: http://www.oncourt.info/index_rus.html (дата звернення: 03.04.2022).
5. Скриншоты URL: http://www.oncourt.info/s.html (дата звернення: 03.04.2022).
6. Программа Oncourt. URL: https://betonmobile.com.ua/ru/oncourt (дата звернення: 03.04.2022).
7. Oncourt. URL: https://oncourt.ru.uptodown.com/windows (дата звернення: 03.04.2022).
8. Стратегия с помощью автосистемы Робобет. URL: https://pro-prognoz.ru/strategiya-s-pomoshhyu-avto-sistemy-robobet.html (дата звернення: 03.04.2022).
9. Що таке VPN і як активувати його безкоштовно за 10 хвилин. URL: https://nachasi.com/cards/2017/08/07/vpn-faq/ (дата звернення: 27.05.2022).
10. Розділ 2. Основи UML. URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html (дата звернення: 27.05.2022).

11. Діаграми послідовності (Sequence diagram) для бізнес-варіантів використання. URL: https://studwood.net/1048452/informatika/diagrami_poslidovnosti_sequence_diagram_biznes_variantiv_vikoristannya (дата звернення: 27.05.2022).

12. Що таке структурна схема? - визначення з техопедії - Розвиток – 2022. URL: https://uk.theastrologypage.com/structure-diagram (дата звернення: 27.05.2022).

13. Як відрізнити структурну схему від функціональної. URL: https://ukrpublic.com/navchannia/yak-vidrizniti-strukturnu-skhemu-vid-funktsionalnoji.html (дата звернення: 27.05.2022).

14. Як працює machine learning та його застосування на практиці. URL: https://uaspectr.com/2020/12/09/yak-pratsyuye-machine-learning/ (дата звернення: 29.05.2022).

15. Задачі машинного навчання. URL: https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B5_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%8F#%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D1%96_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%8F (дата звернення: 29.06.2022).

16. Sigmoidfunktion. URL: https://de.wikipedia.org/wiki/Sigmoidfunktion (дата звернення: 29.06.2022).

17. Machine Learning – не только нейронки. URL: https://www.simbirsoft.com/blog/machine-learning-ne-tolko-neyronki/ (дата звернення: 29.05.2022).

18. Нейронная сеть. URL: https://ru.m.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C (дата звернення: 29.05.2022).

19. Machine Learning Series: Linear Regression. URL: https://www.vertica.com/blog/machine-learning-series-linear-regression/ (дата звернення: 29.05.2022).

20. ML Polynomial Regression. URL: https://www.javatpoint.com/machine-learning-polynomial-regression (дата звернення: 29.05.2022).

21. Python. URL: https://uk.wikipedia.org/wiki/Python (дата звернення: 29.05.2022).

22. SQL-Урок 1. Мова SQL. Основні поняття. URL: http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-sql1-%D0%BC%D0%BE%D0%B2%D0%B0-%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%96-%D0%BF%D0%BE%D0%BD%D1%8F%D1%82%D1%82%D1%8F_ua (дата звернення: 30.05.2022).

23. Что такое Jupyter-ноутбук и как его использовать. URL: https://highload.today/jupyter-notebook/ (дата звернення: 30.05.2022).

24. JetBrains PyCharm. URL: https://itpro.ua/product/jetbrains-pycharm/?tab=description (дата звернення: 30.05.2022).

25. Что Такое MySQL: Объяснение MySQL Для Начинающих. URL: https://www.hostinger.com.ua/rukovodstva/shto-takoje-mysql/ (дата звернення: 30.05.2022).

26. NumPy, часть 1: начало работы. URL: https://pythonworld.ru/numpy/1.html (дата звернення: 30.05.2022).

27. Pandas. URL: https://ru.wikipedia.org/wiki/Pandas (дата звернення: 30.05.2022).

28. Requests (software). URL: https://en.wikipedia.org/wiki/Requests_(software) (дата звернення: 30.05.2022).

29. Beautiful Soup (HTML parser). URL: https://en.wikipedia.org/wiki/Beautiful_Soup_(HTML_parser) (дата звернення: 30.05.2022).

30. Scikit-learn. URL: https://en.wikipedia.org/wiki/Scikit-learn (дата звернення: 30.05.2022).

31. SQLAlchemy. URL: https://ru.wikibooks.org/wiki/SQLAlchemy (дата звернення: 30.05.2022).

32. Tkinter в Python: руководство по использованию. URL: https://highload.today/tkinter-v-python/ (дата звернення: 30.05.2022).

33. Модуль math. URL:https://pythonworld.ru/moduli/modul-math.html (дата звернення: 30.05.2022).

34. Ожидаемые голы. URL: https://ru.wikipedia.org/wiki/%D0%9E%D0%B6%D0%B8%D0%B4%D0%B0%D0%B5%D0%BC%D1%8B%D0%B5_%D0%B3%D0%BE%D0%BB%D1%8B (дата звернення: 04.06.2022).

35. Отберем то, что нужно Data Mining: как сформировать датасет для машинного обучения. URL: https://www.bigdataschool.ru/blog/dataset-data-preparation.html (дата звернення: 04.06.2022).

36. The Pandas DataFrame: Make Working With Data Delightful. URL: https://realpython.com/pandas-dataframe/ (дата звернення: 04.06.2022).

37. Football Data: Expected Goals and Other Metrics. URL: https://www.kaggle.com/datasets/slehkyi/extended-football-stats-for-european-leagues-xg (дата звернення: 04.06.2022).

38. Что такое JSON. URL: https://habr.com/ru/post/554274/ (дата звернення: 06.06.2022).

39. How to Insert JSON Data into MySQL using PHP. URL: https://www.kodingmadesimple.com/2014/12/how-to-insert-json-data-into-mysql-php.html (дата звернення: 06.06.2022)

40. Python Anonymous/Lambda Function. URL: https://www.programiz.com/python-programming/anonymous-function (дата звернення: 08.06.2022).

41. Peshawa Jammal Muhammad Ali, Rezhna Hassan Faraj, Data Normalization and Standardization: A Technical Report, Koya University, 2014, 1-2 с. (дата звернення: 08.06.2022).

42. Cost Function is No Rocket Science! URL: https://www.analyticsvidhya.com/blog/2021/02/cost-function-is-no-rocket-science/ (дата звернення: 10.06.2022).

43. Machine Learning Algorithms for Football Predictions. URL:https://towardsdatascience.com/machine-learning-algorithms-for-football-prediction-using-statistics-from-brazilian-championship-51b7d4ea0bc8 (дата звернення: 10.06.2022).

44. Ahmed Amr Awadallah, Raghav Khandelwal, Football Match Prediction using Deep Learning (Recurrent Neural Network), Stanford University, 2020, 3-4 с. (дата звернення: 10.06.2022).

45. Python GUI Programming With Tkinter. URL: https://realpython.com/python-gui-tkinter/ (дата звернення: 10.06.2022).