БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра дискретной математики и алгоритмики

КОПОЧЕЛЬ
Валентин Викторович

СРАВНЕНИЕ ЭФФЕКТИВНОСТИ ПРЕДСКАЗАНИЙ РЕЗУЛЬТАТОВ ФУТБОЛЬНЫХ МАТЧЕЙ НА ОСНОВЕ РАЗЛИЧНЫХ МЕТОДОВ

Дипломная работа

Научный руководитель:
ст. преподаватель А.А. Буславский

Допущена к защите
«___»_____2021 г.
Зав. кафедрой дискретной математики и алгоритмики
доктор физ.-мат. наук, профессор В.М. Котов

Минск, 2021

РЕФЕРАТ

Дипломная работа, 54 страницы, 5 рисунков, 14 таблиц, 45 источников.

Ключевые слова: МАШИННОЕ ОБУЧЕНИЕ, ОЖИДАЕМЫЕ ГОЛЫ, ОЖИДАЕМЫЕ ОЧКИ, РЕЙТИНГ ЭЛО.

Объект исследования — алгоритмы машинного обучения.

Предмет исследования — применение алгоритмов машинного обучения к данным о футбольных матчах.

Цель работы — исследование возможности применения различных алгоритмов машинного обучения для предсказания исходов футбольных матчей, а также определение алгоритмов машинного обучения, построение моделей на их основе и определение входных признаков, обеспечивающих их наилучшую точность.

В процессе работы были изучены и проанализированы алгоритмы машинного обучения, которые могут быть использованы для решения задачи предсказания результата (отнесения к одному из трёх классов: победа домашней команды, ничья, поражение домашней команды) футбольных матчей. Были обучены ряд моделей, которые в качестве входных признаков использовали как обычные статистические показатели футбольного матча, такие как количество ударов по воротам и количество угловых, так и посчитанные при помощи алгоритмов машинного обучения значения ожидаемых голов, ожидаемых очков, рассчитанный по различным формулам рейтинг Эло. Было изучено влияние на предсказательную способность моделей применения методов уменьшения размерности вектора входных признаков, балансировки тренировочной выборки. Было показано, что лучшие результаты показывают модели на основе алгоритмов бустинга, использующие показатели продвинутой статистики в качестве входных признаков.

РЭФЕРАТ

Дыпломная праца, 54 старонкі, 5 малюнкаў, 14 табліц, 45 крыніц.

Ключавыя словы: МАШЫННАЕ НАВУЧАННЕ, ЧАКАНЫЯ ГАЛЫ, ЧАКАНЫЯ АЧКІ, РЭЙТЫНГ ЭЛА.

Аб'ект даследавання — алгарытмы машыннага навучання.

Прадмет даследавання — ужыванне алгарытмаў машыннага навучання да дадзеных аб футбольных матчах

Мэта працы — даследаванне магчымасці ўжывання розных алгарытмаў машыннага навучання для прадказанні зыходаў футбольных матчаў, а таксама вызначэнне алгарытмаў машыннага навучання, пабудова мадэляў на іх аснове і вызначэнне ўваходных прыкмет, якія забяспечваюць іх найлепшую дакладнасць.

У працэсе работы былі вывучаны і прааналізаваны алгарытмы машыннага навучання, якія могуць быць выкарыстаны для вырашэння задачы прадказанні выніку (аднясення да аднаго з трох класаў: перамога хатняй каманды, нічыя, параза хатняй каманды) футбольных матчаў. Былі навучаны шэраг мадэляў, якія ў якасці ўваходных прыкмет выкарыстоўвалі як звычайныя статыстычныя паказчыкі футбольнага матчу, такія як колькасць удараў па варотах і колькасць кутніх удараў, так і падлічаныя пры дапамозе алгарытмаў машыннага навучання значэння чаканых галоў, чаканых ачкоў, разлічаны па розных формулах рэйтынг Эла. Вывучаўся ўплыў на прадказальную здольнасць мадэляў прымянення метадаў памяншэння памернасці вектара ўваходных прыкмет, балансавання трэніровачнай выбаркі. Было паказана, што лепшыя вынікі паказваюць мадэлі на аснове алгарытмаў бустынга, якія выкарыстоўваюць паказчыкі прасунутай статыстыкі ў якасці ўваходных прыкмет.

ABSTRACT

Thesis, 54 pages, 5 figures, 14 tables, 45 sources

Keywords: MACHINE LEARNING, EXPECTED GOALS, EXPECTED POINTS, ELO RATING SYSTEM.

The object of study — machine learning algorithms.

The subject of the research is the application of machine learning algorithms to data on football matches.

The purpose of the work is the research of possibilities of application of various algorithms of machine learning for prediction of results of football matches, and also definition of algorithms of machine learning, construction of models on their basis and definition of input signs, providing their best accuracy.

In the process of the work machine learning algorithms were studied and analyzed, which can be used to solve the problem of predicting the results (attributed to one of the three classes: victory of home teams, draws, defeat of home teams) of football matches. A number of models were created, which used as their input signs not only the ordinary statistics of a football match, such as the number of shots on goal and the number of corrners, but also values of expected goals, expected points, that were calculated with the use of other machine learning models, and Elo rating values, calculated with different formulas . The influence of models of application of methods of reduction of size of input signs vector, balancing the classes of training samples on the predictive ability was studied. It has been shown that the best results are achieved by boosting algorithm-based models that make use of extended statistics values as input features.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Людям всегда хотелось заглянуть в будущее, особенно интересно было им предсказать те события, в которые они были бы непосредственно вовлечены. Страсть к футболу, который общепризнанно считается самым популярным спортом на планете [17], объединяет миллионы людей по всему миру, и каждого из них интересует то, как сыграет их любимая команда на следующих выходных. Больше, чем футбольных фанатов, результат будущей игры интересует только букмекеров, финансово заинтересованных в наиболее точном выставлении коэффициентов, по которым будут приниматься ставки на будущие спортивные события. Для точной оценки вероятности различных событий в матче необходимо построение как можно более точной математической модели, предсказывающей его количественные характеристики, на исследования для построения которой и будет направлена данная работа. Также такая модель может быть полезна для определения итогового положения команд в турнире, который был прерван из-за форс-мажорных обстоятельств. К примеру, в прошлом году встал вопрос о досрочном завершении национальных чемпионатов европейских стран из-за пандемии COVID-19, но из-за отсутствия признанной модели некоторые лиги приняли решение об определении результатов по текущей таблице, которые были встречены в целом негативно. Так, в Нидерландах за досрочное завершение сезона проголосовало только 16 из 34 клубов двух высших дивизионов, а «Утрехт», претендовавший на выход в Лигу Европы и лишившийся этой возможности, даже намеревался подать в суд на федерацию футбола страны [3].

Построение такой модели — весьма тяжелая задача из-за самой природы футбола. Футбольный матч имеет фиксированную длину (в отличие, например, от теннисного), в нём существует лишь один вид событий, позволяющих набирать очки, которые могут случаться любое количество раз. Тем не менее, голы в футболе забиваются очень редко и зависят от множества факторов. Например, в 10 сезонах Английской Премьер-лиги, с 2001 по 2011 года, самым частым исходом матчей для команд, играющих на своём поле, была ничья со счётом 1-1 (так завершились 11.6% матчей), также часто домашние команды выигрывали с разницей в один или два мяча (со счётом 1-0 завершились 10.92% матчей; 9.37% завершились со счётом 2-1; 8.68% завершились со счётом 2-0), играли со счётом 0-0 (8.34% матчей) или же проигрывали с разницей в один гол (0-1, 7.58%) [6]. Кроме того, на результаты матчей влияют довольно неожиданные факторы, например, было показано, что в чемпионате Германии команды в среднем зарабатывают 61.84% своих очков, играя дома, хотя следовало бы ожидать, что данное число будет ближе к 50% [23].

Целью данной работы является исследование возможности применения различных алгоритмов машинного обучения для предсказания исходов

футбольных матчей, а также определение алгоритмов машинного обучения, построение моделей на их основе и определение входных признаков, обеспечивающих их наилучшую точность. Особый интерес для исследования представляло применение для решения задачи данных так называемой продвинутой статистики и, в частности, концепции ожидаемых голов (xG) — сопоставления каждому удару числового значения от 0 до 1, которое можно воспринимать как вероятность того, что удар с данными характеристиками станет голевым, а также концепции ожидаемых очков (xP), развивающей её.

Объектом исследования являются алгоритмы машинного обучения, предметом — применение алгоритмов машинного обучения к данным о футбольных матчах, зависимость эффективности классификации матчей по результату от использованных алгоритмах, применяемых на различных типах входных данных.

Работа состоит из введения, трёх глав и заключения. Введение раскрывает цель и актуальность исследования, обозначает теоретическую и практическую значимость работы. Первая глава посвящена обзору исследований по проблематике работы. Во второй главе рассматривается процесс получения входных признаков для моделей из исходных данных, описываются концепции ожидаемых голов и очков, рейтинг Эло, а также особенности исследуемых наборов данных. В третьей главе главе описываются алгоритмы, применяемые для предсказания исходов матчей и их количественных характеристик, приводятся результат обучения моделей на их основе, оценивается их эффективность и приводятся детали реализации. В заключении подводятся итоги работы и формулируются выводы.

Глава 1. ОБЗОР ИССЛЕДОВАНИЙ

1.1 Применение моделей ожидаемых голов и рейтинга Эло

Модель ожидаемых голов представляет собой алгоритм, который сопоставляет каждому нанесённому по воротам ударам число от 0 до 1, которое можно интерпретировать как вероятность того, что данный удар станет голевым. Полученные показатели можно использовать для оценки того, какое число голов должна была в среднем забить команда в матче. Использование модели ожидаемых голов позволяет уменьшить элемент случайности, связанный с реализацией голевых моментов, и лучше оценить качество футбола как в исполнении команды в целом, так и в исполнении отдельных игроков. Первой моделью ожидаемых голов считается разработанная Брайаном Макдональдом для другого вида спорта с низкой результативностью — хоккея на льду. Предсказания лучшей из моделей Макдональда достигли корреляции с количеством реальных голов в матче равной 0.69 [31]. В 2015 году, группа под руководством Патрика Люси [24] предложила модель оценки вероятности того, что удар станет голевым, в зависимости от места, с которого он был нанесён, близости защитников к мячу, их расположения, ситуации в матче, в которой был нанесён удар (контратака, позиционная атака, штрафной удар, и.т.д.) ,а также передвижений всех игроков за десять секунд до нанесения удара. В основе модели лежала логистическая регрессия. Похожая модель была разработана под руководством Харма Игелса, но она использовала в качестве входных признаков, например, место удара, часть тела, которой он был нанесён, оценку футбольных навыков бьющего и вратаря, и применяла помимо логистической регрессии, такие алгоритмы, как AdaBoost, дерево решений и случайный лес [12]. Но наиболее известной и применимой в СМИ стала модель Майка Кейли. В попытке создать как можно меньше формул, но при этом сохранить нюансы различных видов игровых ситуаций, Майкл остановился на выделении 6 типов ударов по воротам:

- прямые удары со штрафного;
- удары после обыгрыша вратаря;
- удары головой после навеса;
- удары головой после других типов передач;
- удары другими частями тела после навеса;
- так называемые «обычные» удары — удары, нанесённые не головой и после передачи, отличной от навеса.

Пенальти и автоголы в модели Кэйли не учитываются и просто записываются как дополнительный статистический параметр. Исключение пенальти можно понять, поскольку его вероятность трудно оценить, в отрыве

от статистики предыдущих пенальти бьющего игрока. Некоторые модели приравнивают одиннадцатиметровый удар к 1,0 xG, хотя процент реализации пенальти обычно составляет 0,8±0,05 (в зависимости от лиги и сезона). Автоголы же абсолютно случайны, оценить вероятность автогола в матче просто невозможно. Кроме этих факторов, учитываются позиция пасующего и тип атаки. Мастерство вратаря и умение бьющего игрока влияют на вероятность гола, по мнению Кэйли, гораздо меньше,Для построения моделей ожидаемых голов в первую очередь важно определить, какие из характеристик удара повышают его опасность для команды-соперника. При этом, главной идеей, объединяющей большинство моделей xG является предположение о том, что «качество» удара не зависит от мастерства бьющего и вратаря, что подтверждается многочисленными исследованиями. Так, Майк Кэйли, разбивал удары одних и тех же игроков на протяжении 6 сезонов на случайные выборки, и искал корреляцию между их реализацией. Лучший результат был получен для игроков с количеством ударов больше 250 и равнялся 0.3. Подробнее о его результатах можно почитать в [35].

В моделировании результатов футбольных матчей широкое применение находят модели, использующие рейтинг Эло, изобретённый профессором физики Арпадом Имре Эло в 1978 году для оценки соревновательного уровня шахматистов [4] . Расчёт рейтинга игрока по результатам турнира основывается на сравнении числа реально набранных им очков по сравнению с ожидаемым (на основе его текущего рейтинга). Рейтинг игрока возрастает, если по результатам турнира количество набранных баллов оказывается больше ,чем ожидаемое значение, и наоборот.

Для двух соперников с рейтингами R_A и R_B ожидаемое количество очков, которое наберёт игрок A, оценивается по формуле:

$$E_A = \frac{1}{1 + 10^{\frac{R_B - R_A}{400}}} \tag{1.1}$$

Сумма E_B и E_A всегда считается равной единице. После матча, в котором игрок A набирает S_A очков (1 за победу, 0.5 за ничью, 0 за поражение), его рейтинг обновляется по формуле:

$$R'_A = R_A + K\left(S_A - E_A\right) \tag{1.2}$$

В 2016 Ян Ласек весьма точно предсказал результаты предстоящего чемпионата Европы с использованием метода Монте-Карло и различных модификаций рейтинга Эло, что доказывает обоснованность его использования для решения нашей задачи. [26]

1.2 Обзор алгоритмов, не реализованных в данной работе

Задача предсказания результатов матчей активно исследуется начиная с середины 20 века. Так, уже в 1956 году Майкл Морони [27] использовал распределение Пуассона и негативное биномиальное распределение для моделирования на основе предыдущих результатов команды числа забитых в матче голов. Важной вехой стала статья Иэна Хилла[19], вышедшая в 1974, в которой автор пришёл к выводу, что результаты футбольных матчей не являются абсолютно случайными и могут быть предсказаны.

В 1997 Марк Диксон и Стюарт Коулз представили модель, оценивающую количество голов, забиваемых каждой из команд, используя гипотезу о том, что они независимы и подчиняются распределению Пуассона, из недостатков которой можно выделить постоянность параметров, оценивающих качество атаки и защиты команды [11]. Также в модели использовались вычисленные при помощи метода максимального правдоподобия показатели атакующего и защитного рейтинга. Данная модель получила развитие в работе Гаварда Рю [37], моделировавшего атакующий и защитный рейтинг каждой команды при помощи процесса броуновского движения, учитывающего, например, память о прошлых поражениях, в предположении

$$P\left(x_{A,B}|\left(e_A, e_B\right)\right)=P\left(x_{A,B}|a_A-d_B-\gamma\left(a_A+d_A-a_B-d_B\right)/2\right), \quad (1.3)$$

где $x_{A,B}$ — прогнозируемое количество голов, которое забьёт команда A в матче с командой B, e=(a, d) — пара из атакующего и защитного рейтинга команды соответственно, а γ — константа психологического эффекта. Для получения предсказаний использовались множественные симуляции цепей Маркова по методу Монте-Карло. Модель использовалась для разработки стратегии ставок, которая смогла получить выигрыш в 100% после предсказания 35 матчей тестовой выборки. В начале 21 века начали исследоваться модели, которые стали предсказывать класс результата матчей без использования регрессии для предсказания числа голов. Данный переход позволил разрешить проблему того, что многие модели считали число забитых командой в матче голов независимым от того, против кого играет команда. Одной из первой таких работ [14] стало исследование 2000 года авторства Дэвида Форреста и Роберта Симмонса. В 2005 году появилась статья Джона Годдарда [18], одна из первых, в которой в качестве входных для моделей использовались признаки помимо предыдущих счетов матчей, букмекерских котировов и экспертных предсказаний: значимость матча и расстояние между городами, которые представляли команды. Также выделим статью Ника Такса 2015 года, в которой использование техник понижения размерности позволило достигнуть хороших результатов по точности. Так, после применения PCA лучшит наивный

байесовский классификатор и полносвязная нейронная сеть из рассматривавшихся показали точность классификации в 54.5%.

Стоит отметить примеры успешного применения нейронных сетей для решения данной задачи. В [7] модель на основе многослойного перцептрона, использовавшая лишь предыдущие результаты команд, смогла предсказать 5 из 6 участников Азиатской лиги чемпионов от Ирана. Также существует модель, использующая нейронные сети, сумевшая классифицировать матчи чемпионата Финляндии, отличающегося большой разницей в классе участников и, как следствие, высокой предсказуемостью по типу результата на пять категорий в зависимости от разницы в счёте. Худший результат по точности — 0.833 был показан на классе низкорезультативных поражений домашней команды [36].

Кроме того существует большое количество работ, предсказывающих результаты матчей с помощью байесовских сетей — ациклических ориентированных графов, в которых каждая вершина (узел сети) представляет n-значную переменную, дуги обозначают существование непосредственных причинно-следственных зависимостей между соединёнными переменными, а сила этих зависимостей количественно выражается в виде условных вероятностей, сопоставленных каждой из переменных [33]. В 1997 была разработана байесовская сеть, предсказывавшая результаты матчей клуба «Тоттенхэм Хотспур» в период 1995-1997 годов, использовавшая в качестве переменных наличие на поле Тедди Шерингема, Даррена Андертона и Криса Армстронга, играет ли в полузащите Клайв Уилсон, место соперника в таблице, а также место проведения игры. Модель превзошла всё прочие модели, рассмотренные в статье, такие как KNN-классификатор и достигла точности в 59.21% [22]. В 2013 году была разработана байесовская сеть для предсказаний исходов матчей футбольного клуба «Барселона», использовавшая, например, погоду и психологическое состояние игроков, которая достигла точности в 92%, но использовала лишь 20 матчей для оценки своей точности [32]. Её схема приведена на рисунке 5. Такие модели являются сложными в построении из-за необходимости экспертного знания, а также быстро устаревают (например, некоторые из игроков «Тоттенхэма» покинули его спустя два сезона после первого матча датасета сети).

В уже упоминавшемся исследовании на данных чемпионата Финляндии [36] применялись модели на основе генетических алгоритмов, достигнувшие точности в 0.770 для худшего из классов (победа домашней команды с небольшой разницей).

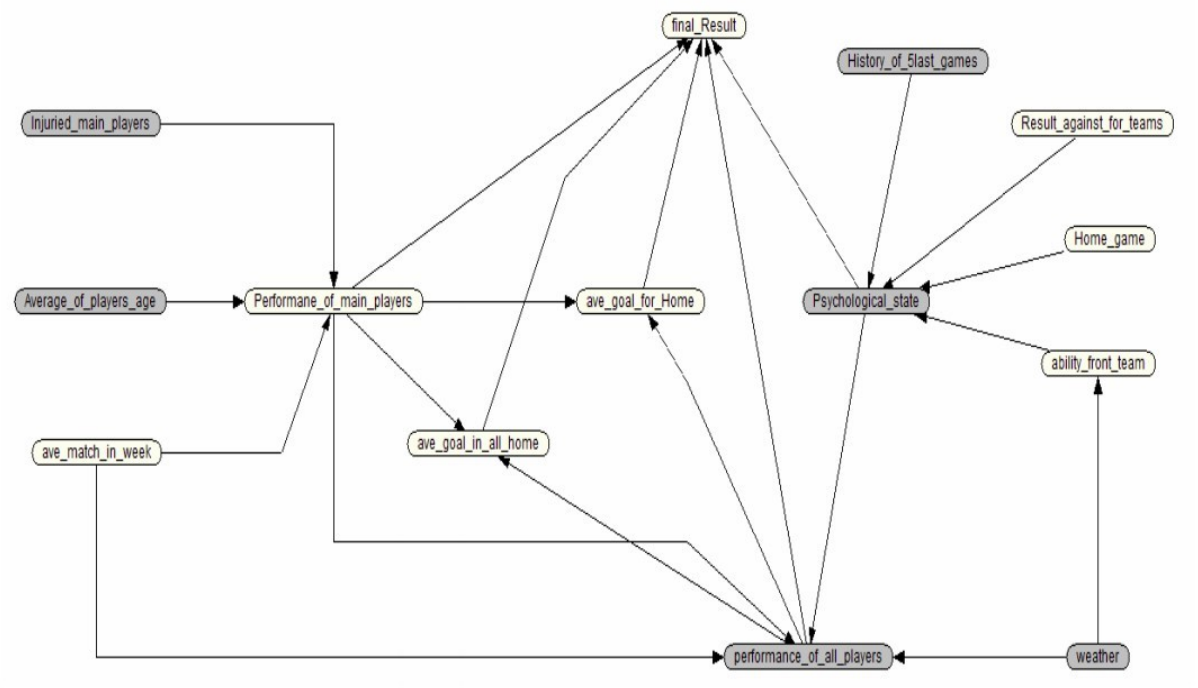

Рисунок 1.1 — Визуализация байесовской сети для предсказания матчей ФК "Барселона". Источник: [28]

Глава 2. КОНСТРУИРОВАНИЕ ПРИЗНАКОВ

2.1 Входные данные

Первоначально, для моделирования использовался датасет European Soccer Database, размещённый на Kaggle [13]. Он представляет собой базу данных SQLite, содержащую данные о более чем 28000 матчах в чемпионатах одиннадцати стран, проходивших с 2008 по 2016 года, включая букмекерские котировки на них, расстановку игроков на поле, и даже атрибуты игроков, участвующих в матче команд, в серии компьютерных симуляторов FIFA (впрочем, они в данной работе не использовались в силу своей субъективности). Также, в датасете присутствуют данные о событиях более чем 10000 матчей, таких как удары в створ ворот и мимо, количество угловых, количество совершённых фолов, в виде XML-документов, с указанием координат места на поле, где они происходили, но в большинстве случаев они неполные, в частности, для многих матчей значения соответствующих ячеек таблиц содержат лишь пустой XML-тег (например, <goal/>), поэтому данный датасет использовался для обучения моделей, использующих в качестве входных лишь простые статистические показатели, такие как результаты последних матчей играющих команд (вообще и между собой), разница забитых и пропущенных мячей, среднее количество ударов в створ, наносимых командой за матч, и так далее. В качестве источников для пополнения данной базы использовались ресурсы http://football-data.mx-api.enetscores.com/ (для счетов матчей, составов, расстановок игроков, информации о событиях матча), http://www.football-data.co.uk/ (букмекерские котировки) и http://sofifa.com/ (данные о видеоиграх FIFA).

Для предсказания результатов матчей с использование продвинутой статистики, было принято решение использовать собранными учёными университета Пизы набор данных Soccer match event dataset с сайта FigShare [40]. Он представляет собой набор файлов в формате JSON, которые являются результатами запросов к API компании Wyscout — одного из самых крупных и известных поставщиков футбольной статистики. Данные содержат информацию о матчах так называемых топ-5 лиг (первого дивизиона чемпионатов Англии, Германии, Франции, Италии и Испании) сезона 2017/2018, а также чемпионата мира-2018 и чемпионата Европы-2016. Итого, в нём доступна информация о почти двух тысячах матчей и более чем трём миллионам событий в них. Каждое событие характеризуется своим типом (пас, нарушение правил, удар, штрафной удар и.т.д.), подтипом (например, штрафные удары могут подразделяться на угловые удары, удары от ворот), массивом тегов, описывающих событие (например, тег 101 означает, что удар

стал результативным), координатами, где оно произошло на поле (в формате процента от размеров поля), временем матча, когда оно произошло, уникальными идентификаторами игрока, поучаствовавшего в событии и его команды, а также собственный идентификатор. Также существуют файл с информацией об игроках, файл с данными о соревнованиях файл с данными об угловых и файл с данными о матчах. Последний использовался в качестве источника информации о дате матча, игровой неделе, на которой он был сыгран, ID матча, соревнования, счёте по итогам первого тайма и всего матча, является матч домашним или гостевым.

2.2 Выделение данных из датасета European soccer database

Конструирование признаков — это процесс использования предметной области данных для создания признаков из исходных данных, которые будут использоваться при обучении алгоритмов машинного обучения. Конструирование признаков является фундаментом для приложений машинного обучения, а также процессом трудным и затратным.

Как уже отмечалось ранее, данные из European soccer database использовались для обучения моделей, не использующих продвинутую статистику. Так, для обучения всех моделей с использованием этой базы данных использовались в качестве входных использовались такие признаки, как разница забитых и пропущенных мячей для последних десяти матчей каждой из команд, количество побед в них, количество побед и поражений в последних трёх встречах команд между собой (если такие имелись), логические флаги, указывающие на то, в каком турнире был сыгран данный матч (один из следующего списка: английская Премьер-лига, итальянская Серия А, немецкая первая Бундеслига, французская Лига 1 или Испанская Лига BBVA) и вероятность каждого из трёх исходов матча по версии букмекерских контор «Bet365» и «Bet&win», полученные как обратная величина котировки на данный исход, делённая на сумму обратных величин всех котировок. В некоторых моделях к ним также добавлялись среднее количество угловых за матч, поданных командой и допущенных у своих работ, аналогичные показатели для ударов в створ и мимо ворот, средний процент времени владения мячом. Все вышеперечисленные показатели рассчитывались за последние десять матчей. Включение угловых в данный перечень можно объяснить не только большим количеством опасных моментов, создаваемых после них, но и тем, что они назначаются в случае ухода мяча за лицевую линию после контакта с игроком защищающейся команды, что почти всегда означает удар в створ ворот, отражённый голкипером.

2.3 Выделение данных из датасета Soccer match event dataset.

2.3.1 Выделение признаков, используемых в качестве входных, для моделей xG

Для построения моделей ожидаемых голов в первую очередь важно определить, какие из характеристик удара повышают его опасность для команды-соперника.. В первую очередь, было решено визуализировать содержимое файлов датасета. Для этого был построен график координат всех результативных ударов, представленный на рисунке 2.1. Из него видно, что большинство голов случаются после ударов изнутри штрафной площади из позиций с широким углом обстрела ворот. Поэтому было решено выделить в отдельные признаки расстояние в метрах от центра ворот и угол в треугольнике между штангами и бьющим, вычисляемый по формуле:

$$\alpha = \tan^{-1}\left(7.32 \times (105 - x) / \left((105 - x)^2 + (68/2 - y)^2 - (7.32/2)^2\right)\right), \quad (2.1)$$

где x и y соответствующие координаты удара (заранее переведённые из процентов), 7.32 — ширина футбольных ворот. По правилам ФИФА, длина футбольного поля может варьироваться от 90 до 120 метров, в то время как ширина — от 45 до 90. Тем не менее, в большинстве случаев для футбольных матчей на всех уровнях используется поле длиной 105 и шириной 68 метров, поэтому именно эти числа было решено использовать в качестве констант в формулах. Также, на первом этапе конструирования признаков, были созданы логистические признаки, показывающие был ли удар нанесён со штрафного, был ли удар заблокирован, а также, наносился ли удар во время контратаки. Здесь стоит пояснить, что в терминологии WyScout любой рикошет от защитника считается блокировкой удара, то есть заблокированный удар мало того, что может оказаться голевым, так ещё и «блокировка» удара может увеличить опасность удара, например, попадание мячом в стенку при штрафном может дезориентировать вратаря. Штрафные удары было решено выделить в отдельную категорию, так как многие аналитики сходятся на том, что применять к ним карты распределения xG обычных ударов некорректно, однако, прямые удары со штрафных весьма легко моделируются, поскольку, в силу своей статичной природы, определяются лишь точкой нанесения удара, а также тем, какой ногой они наносятся. Признак для выделения контратак был введён, так как при них атакующей команде противостоит малое число защитников, находящихся в неудобных для себя позициях, что, естественно, положительно сказывается на опасности атаки. Также, была добавлена информация о том, какой частью тела наносился удар (правой ногой, левой или

головой) и о том, в какую зону ворот был направлен мяч. Первоначально, информация о зоне, куда направился мяч после удара, казалась незначительной, поэтому были построены модели ожидаемых голов как использующие информацию о ней, так и нет. Также была добавлена информация о времени удара, ведь статистика показывает, что во втором тайме забивается гораздо больше мячей, чем в первом, что объясняется усталостью игроков [42].

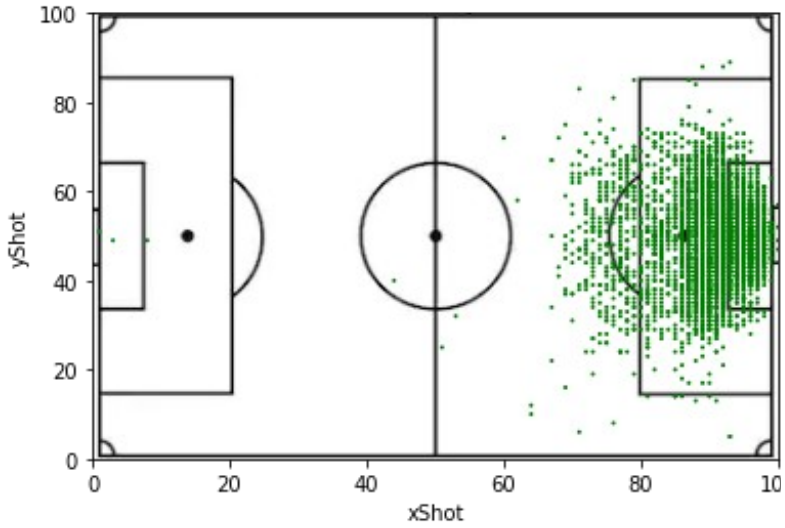

Рисунок 2.1 — Визуализация координат голевых ударов

Затем, было начато конструирование признаков, основанных на связях событий с другими файлами датасета, а также на связях с другими событиями. Так, на основе информации о наносившем удар игроке, в таблицу событий было добавлено логическое поле, с информацией о том, нанесён ли удар рабочей ногой, то есть той, с помощью которой игрок чаще совершает удары, что обычно означает их большую опасность. Помимо этого, для каждого удара было вычислено время, прошедшее с предыдущего удара той же команды в том же тайме. Это позволяет учесть «добивания», на которые вратарю гораздо тяжелее среагировать. После чего была построена таблица корреляции для оценки значимости каждого из признаков, представленная на рисунке 2.2. Как видно из неё, действительно наибольшее влияние на то, станет ли удар точным, оказывают угол удара и расстояние до ворот, поэтому дальнейшие эксперименты проводились с тремя типами моделей ожидаемых голов: где в качестве входных признаков использовались только координаты и угол удара, все признаки, а также, как было указано ранее, все признаки, кроме зоны, в которую попал удар.

Для получения значения xG для каждого удара, строились модели, решающие задачу регрессии: каждому удару, если он является голевым, присваивалось числовое значение 1, а если нет — 0. Поиск наилучшей модели

осуществлялся среди следующих семейств алгоритмов машинного обучения: обобщённые линейные модели, модели градиентного бустинга, XGBoost, случайные леса, нейронные сети. Данные алгоритмы будут описаны в следующих разделах.

	isGoal	isCounter	realEventSec	timeFromLastShot	skilledFoot	distanceToGoal	angleToGoal	isFreeKick	zone
isGoal	1.000000	0.035568	0.014354	-0.000774	-0.035266	-0.252415	0.327348	-0.030682	-0.3474
isCounter	0.035568	1.000000	0.010550	0.007906	0.040215	-0.010296	-0.038250	-0.053076	-0.0691
realEventSec	0.014354	0.010550	1.000000	0.287604	0.003324	-0.010902	0.006556	0.008413	0.0049
timeFromLastShot	-0.000774	0.007906	0.287604	1.000000	0.002790	0.003409	0.001920	0.014329	0.0095
skilledFoot	-0.035266	0.040215	0.003324	0.002790	1.000000	0.365797	-0.317671	0.150404	-0.0394
distanceToGoal	-0.252415	-0.010296	-0.010902	0.003409	0.365797	1.000000	-0.726388	0.247244	0.1260
angleToGoal	0.327348	-0.038250	0.006556	0.001920	-0.317671	-0.726388	1.000000	-0.153393	-0.1103
isFreeKick	-0.030682	-0.053076	0.008413	0.014329	0.150404	0.247244	-0.153393	1.000000	0.0474
zone	-0.347455	-0.069192	0.004920	0.009589	-0.039445	0.126017	-0.110361	0.047420	1.0000

Рисунок 2.2 — Таблица корреляции признаков, полученных на этапе Feature engineering из Soccer match event dataset

2.3.2 Рейтинг Эло

Перед применением рейтинга Эло в задаче классификации матча по результату, необходимо найти ответы на два вопроса: что следует считать очками и какое значение должен принимать K-фактор. В данной работе в качестве очков для рейтинга Эло рассматривались результат матча (принцип аналогичный шахматам), количество голов в матче (считается, что команда набирает количество очков равное доле забитых ей голов от общего числа голов в матче), количество xG, сгенерированных командой за матч (принцип набора аналогичен второму случаю). Что касается K-фактора, то его значение больше, чем может показаться. Если он слишком большой, то рейтинг будет преувеличивать значимость нескольких последних игр, а в каждой отдельно взятой игре соперники будут обмениваться слишком большим количеством очков. Если же значение K-фактора невелико, то рейтинг будет недостаточно быстро отражать изменение уровня качества игры игроков. В оригинальном рейтинге Эло, K полагалось постоянным и равным 10. Данное значение признается современными шахматными экспертами заниженным, поэтому большинство шахматных сайтов используют большее значение (например, k = 32 для Internet chess club). ФИДЕ (международная шахматная федерация) сейчас использует трёхуровневую систему, при которой K = 40 для игроков, проведших менее 30 игр, или несовершеннолетних игроков, чей рейтинг не превышает 2300 очков; K = 20 для игроков с рейтингом ниже 2400 очков; K = 10 для игроков, с рейтингом более 2400 и проведших более 30 матчей. Также

существует подход, при котором К-фактор является целочисленной функцией. В частности, исследования [30, 20] показывают, что в случае, когда в качестве очков используется доля голов команды в матче, имеет смысл использовать формулу

$$K(D)=26(D+1), \tag{2.2}$$

где через D обозначена разница между числом голов, забитых победителем и проигравшим. В данной работе рассматривались постоянные К-факторы со значениями 20 и 64, а также рейтинги Эло на основе числа забитых голов с переменным К-фактором, определяемым по формуле (2.2).

2.3.3 Подсчёт ожидаемых очков

Логичным продолжением концепции ожидаемых голов является концепция ожидаемых очков (xPoints, xP). При её использовании по итогам матчей каждой команде присваивается дробное число от 0 до 3, которое можно воспринимать как математическое ожидание количества очков набранных в матче. Обычно, в основу их расчёта ложится количество и сумма ожидаемых голов, созданных командой и допущенных около своих ворот. Оценив разницу реальных очков и xPoints, можно сделать выводы об удаче, сопутствующей команде, и понять, какие команды явно прыгают выше головы, а какие, наоборот, не реализуют свои шансы должным образом, поэтому современные аналитики часто используют данные модели при принятии решения об увольнении тренера.

Существуют различные методы подсчёта ожидаемых очков, приведём те из них, которые использовались в данной работе. Первым был рассмотрен метод, в рамках которого количество ожидаемых очков сопоставляется с разностью сумм xG ударов созданных и пропущенных командой по таблице 2.1, используемой некоторыми аналитиками, например в [44]. Среди очевидных достоинств данного подхода можно выделить быстроту и простоту подсчёта, причём это приближение показывает себя хорошо на дистанции из многих матчей.

Таблица 2.1 — Определение количества xP по разнице сгенерированных и допущенных у своих ворот xG

Разница в xG	Количество ожидаемых очков
XGDiff > 1.5	2.7
1 < XGDiff ≤ 1.5	2.3

$0.5 < XGDiff \leq 1$	2
$0 < XGDiff \leq 0.5$	1.5
$-0.5 < XGDiff \leq 0$	0.7
$-1 < XGDiff \leq -0.5$	0.5
$-1.5 < XGDiff \leq -1$	0.3
$-XGDiff \leq -1.5$	0.1

Второй подход к подсчёту ожидаемых очков основывается на гипотезе о том, что количество голов, которые будут забиты каждой из команд в матче являются случайной величиной, имеющей распределение Пуассона, обоснованность которой доказывается во многих работах, например в [11, 31]. Поэтому в качестве числа ожидаемых очков было предложено использовать утроенную вероятность того, что одна пуассоновская случайная величина с математическим ожиданием равным сумме ожидаемых голов команды, больше другой пуассоновской величины, с математическим ожиданием равным сумме ожидаемых голов команды-оппонента, которая определяется формулой:

$$P(A>B)=\sum_{k=0}^{\infty}\left(\sum_{l=k+1}^{\infty}\frac{\lambda_A^l e^{-\lambda_A}}{l!}\right)\frac{\lambda_B^k e^{-\lambda_B}}{k!}, \quad (2.3)$$

где через λ_A и λ_B обозначены соответствующие суммы xG. Значения l и k с целью упрощения вычислений ограничивались константой, значение которой равно максимально возможному числу голов в матче, которое полагалось равным 10.

Третий же подход основан на большом количестве симуляций матча с заданным набором ударов. В рамках симуляций, предполагалось, что конверсия удара с заданным xG в гол представляет собой испытание по схеме Бернулли с вероятностью успеха равной xG. Для подсчёта xP для каждого удара при помощи функции библиотеки numPy random.binomial генерировался массив нулей и единиц с размером равным числу симуляций (в рамках данной работы — 1000), после чего значения соответствующих ячеек массива для каждой команды суммировались. Таким образом определялся счёт матча по итогам симуляции. После чего ожидаемые очки команды в матче определялись как сумма утроенного процента побед команд и процента ничьих.

Следует отметить ещё один способ определения xG, не реализованный на практике в рамках данной работы, при котором предлагается использовать для оценки числа ожидаемых очков формулу полной вероятности и предсказывать вероятности, каждого из возможных счетов: от 0:0 до реализации командами

каждого из своих ударов, опять же исходя из того, что xG удара является вероятностью того, что он станет голом, описанный в [43].

Глава 3. ОБУЧЕНИЕ МОДЕЛЕЙ КЛАССИФИКАЦИИ ФУТБОЛЬНЫХ МАТЧЕЙ. СРАВНЕНИЕ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

3.1 Алгоритмы, использовавшиеся для классификации данных из Soccer match event dataset.

В данном разделе будут описаны различные по своей природе алгоритмы машинного обучения, которые использовались для решения задачи классификаций матчей по результату. Простейшим из этих методов является метод KNN (K ближайших соседей). Суть его состоит в вычислении расстояния до каждого из объектов тренировочной выборки, выделения k ближайших из них и отнесения объекта к тому классу, к которому относятся большинство из соседей. Из достоинств данного метода следует отметить отсутствие необходимости обучения в принципе, а также его эффективность на датасетах большого размера, но сам процесс классификации требует большого количества вычислений. В рамках работы, рассматривались модели с 3, 5 и 10 соседями, рассматривались как одинаковые веса для соседей при принятии решения о принадлежности к классу, так и учитывающие расстояние от объекта, в качестве расстояния использовалась метрика Минковского.

Далее были рассмотрены ансамбли деревьев решений. Для построения деревьев использовался алгоритм CART, на каждой своей итерации осуществляющий разбиение по единственному рассматриваемому признаку таким образом, чтобы максимизировать функцию оценки качества разбиения (чаще всего энтропию Шеннона или коэффициент Джини), после чего осуществляет процесс отсечения поддеревьев, о котором подробнее можно прочитать в [9]. В основе работы использованных ансамблей лежит понятие бутстрэпа — метода исследования распределения статистик вероятностных распределений, основанный на многократной генерации выборок на базе имеющейся выборки. В случае если для каждой из сгенерированных с помощью бутстрэпа выборок построим свой собственный классификатор-решающее дерево, а в качестве ответа будем выдавать усреднённый ответ всех деревьев, получим ансамбль называемый бэггингом. Бэггинг позволяет снизить дисперсию обучаемого классификатора и предотвращает переобучение. Эффективность бэггинга достигается благодаря тому, что базовые алгоритмы, обученные по различным подвыборкам, получаются достаточно различными, и их ошибки взаимно компенсируются при голосовании, а также за счёт того, что объекты-выбросы могут не попадать в некоторые обучающие подвыборки.

Доказательством эффективности бэггинга может служить, например, рисунок 3.1, взятый из документации библиотеки scikit.

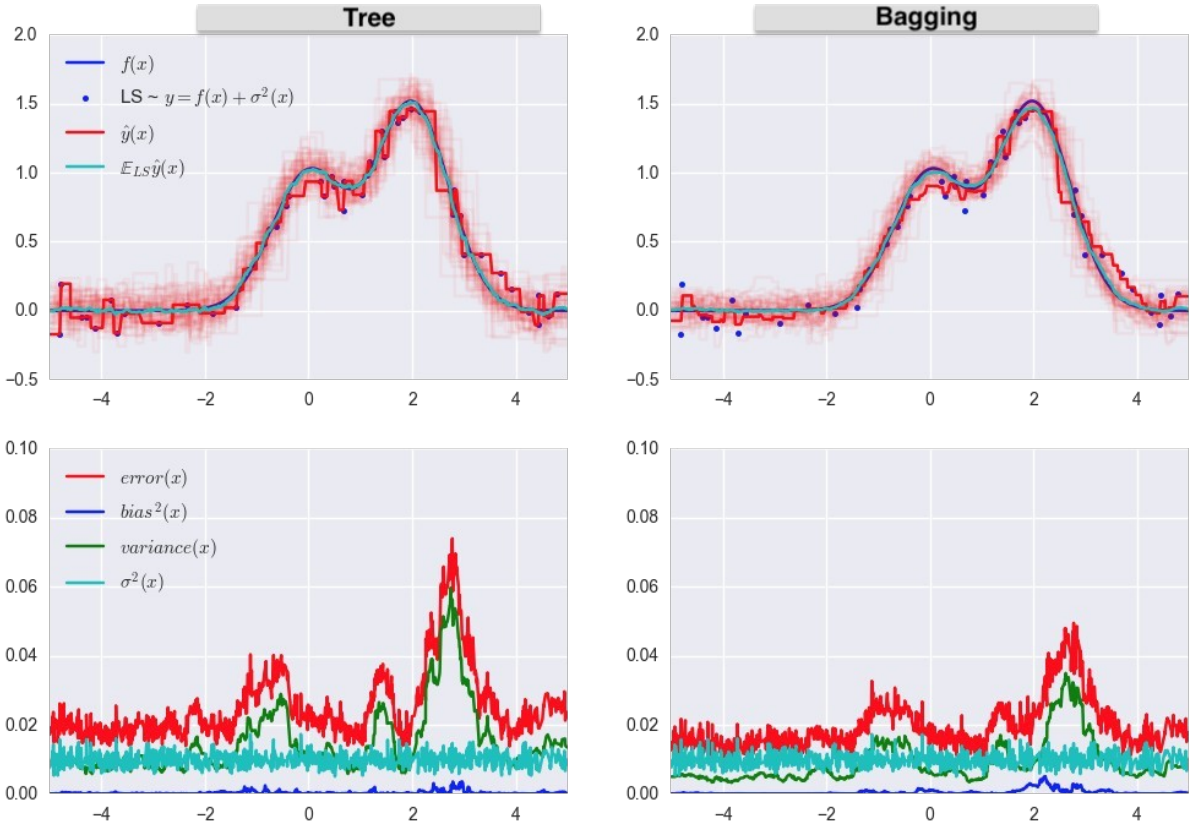

Рисунок 3.1 — Сравнение ошибок бэггинга и одиночного дерева решений. Источник: [39]

Идеи бэггинга развиваются в классификаторе «случайный лес». Решающие деревья являются хорошим семейством базовых классификаторов для бэггинга, поскольку они достаточно сложны и могут достигать нулевой ошибки на любой выборке. Отличием от обычного бэггинга на основе решающих деревьев является то, что при каждом новом разбиении сначала выбирается m случайных признаков из n исходных, и оптимальное разделение выборки ищется только среди них, что позволяет улучшить робастность модели, а также сгенерировать большее число различных деревьев, предсказания с помощью которых будут давать различные результаты [39]. Лучшие результаты обучения моделей-классификаторов матчей по результату, основанных на применении вышеописанных алгоритмов приведены в таблице 3.1. Как из неё видно, лучше всего себя в задаче классификации показывает Random Forest, бэггинг показывает приемлимые результаты в задачах регрессии.

Таблица 3.1 — Результаты, полученные с применением ансамблей решающих деревьев и KNN

Дополнительные входные признаки	Алгоритм	Точность классификатора на тестовой выборке
-	KNN	0.4585
	Random Forest	0.5123
	Bagging	0.4695
Количество угловых в прошедших матчах	KNN	0.4616
	Random Forest	0.5229
	Bagging	0.4604
Количество угловых в прошедших матчах	KNN	0.4638
	Random Forest	0.5168
	Bagging	0.4638
Количество угловых и ударов в прошедших матчах	KNN	0.4536
	Random Forest	0.5131
	Bagging	0.4627
Количество ударов, процент владения мячом в предыдущих матчах	KNN	0.4642
	Random Forest	0.5157
	Bagging	0.4589
Количество угловых, ударов, процент владения мячом в предыдущих матчах	KNN	0.4646
	Random Forest	0.5127
	Bagging	0.4642

Следующим рассмотренным типом классификаторов стали так называемые наивные байесовские классификаторы. Пусть, y — переменная, означающая принадлежность объекта к определённому классу, x i — его признаки. Тогда, применив теорему Байеса, получим:

$$P(y \vee x_1, x_2, \dots, x_n) = \frac{P(y)P(x_1, x_2, \dots, x_n \vee y)}{P(x_1, x_2, \dots, x_n)} \quad (3.1)$$

Вычисление второй вероятности в числителе при больших n оказывается весьма трудоёмким, поэтому используют «наивное» предположение о том, что

x_1, x_2, …, x_n условно независимы относительно y, которое позволяет упростить выражение выше:

$$P(y \vee x_1, x_2, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i \vee y)}{P(x_1, x_2, \ldots, x_n)} \tag{3.2}$$

Вопреки наивному виду и упрощённым условиям (которые почти никогда не верны), наивные байесовские классификаторы часто хорошо себя показывают во многих сложных задачах. Достоинством наивного байесовского классификатора является малое количество данных для обучения, необходимых для оценки параметров, требуемых для классификации [5]. Так как знаменатель выражения выше является константой, будем считать, что объект относится к тому классу, подстановка значения которого максимизирует числитель по y. Обычно для оценки параметров модели P(y) и $P(x_i|y)$ используют метод апостериорного максимума, в частности P(y) считается равной относительной частоте класса y в тренировочной выборке. Различные наивные байесовские классификаторы отличаются, в основном, предположениями о распределении величин $P(x_i|y)$. В данной работе использовалось предположение о нормальном распределении данных величин. Помимо этого, был исследован метод опорных векторов (SVM), чьей целью является построение разделяющей гиперплоскости, уравнение которой имеет вид:

$$\langle w, x \rangle = x_0 \tag{3.3}$$

Постановка задачи для SVM такая: необходимо найти гиперплоскость, разделяющую объекты класса и не принадлежащие ему. Предположим, что объекты являются линейно разделимыми, то есть существуют такие значения параметров w и w_0, при которых функционал числа ошибок

$$Q(w, w0) = \sum_{i=1}^{l} \left[y_i \left(\langle w, x_i \rangle - w_0 \right) < 0 \right] \tag{3.4}$$

принимает нулевое значение. Тогда разделяющая гиперплоскость не единственна, поскольку существуют и другие положения разделяющей гиперплоскости, реализующие то же самое разбиение выборки. Идея метода заключается в том, чтобы разумным образом распорядиться этой свободой выбора. Потребуем, чтобы разделяющая гиперплоскость максимально далеко отстояла от ближайших к ней точек обоих классов. Первоначально данный принцип классификации возник из эвристических соображений: вполне естественно полагать, что максимизация зазора (margin) между классами должна способствовать более уверенной классификации. Было показано, что

величина зазора обратно пропорциональна норме, что сводит задачу к решению оптимизационной:

$$\begin{cases} \|w\|^2 \to min \\ y_i\left(\langle w, x_i\rangle - w_0\right) \geq 1, \end{cases} \tag{3.5}$$

где l — число объектов в обучающей выборке. По теореме Куна-Таккера эта задача эквивалентна двойственной задаче поиска седловой точки функции Лагранжа [1]. Обобщим данный метод на случай линейной неразделимости. Для этого, позволим алгоритму допускать ошибки на обучающих объектах, но при этом постараемся, чтобы ошибок было поменьше. Введём набор дополнительных переменных $\xi_i > 0$, характеризующих величину ошибки на объектах x_i, i =1, ..., l. Для этого в задаче (10) смягчим ограничения-неравенства, и одновременно введём в минимизируемый функционал штраф за суммарную ошибку:

$$\begin{cases} 0.5\|w\|^2 + C\sum_{i=1}^{l} \xi_i \to min \\ y_i\left(\langle w, x_i\rangle - w_0\right) \geq 1 - \xi_i \\ \xi_i \geq 0, i = 1, \ldots, l \end{cases} \tag{3.6}$$

Существует ещё один подход к решению проблемы линейной неразделимости. Это переход от исходного пространства признаковых описаний объектов X к новому пространству H с помощью некоторого преобразования ψ: X → H. Если пространство H имеет достаточно высокую размерность, то можно надеяться, что в нём выборка окажется линейно разделимой. Пространство H называют спрямляющим. Функция K : X × X → R называется ядром, если она представима в виде $K(x,x') = \langle\psi(x),\psi(x')\rangle$ при некотором отображении ψ: X → H, где H — пространство со скалярным произведением. Существует несколько «стандартных» ядер, которые при ближайшем рассмотрении приводят к уже известным алгоритмам: например, к полиномиальным разделяющим поверхностям, потенциальным функциям (RBF-сетям), сигмоидным. Все вышеперечисленные ядра были использованы в данной работе. В таблице 3.2. представим результаты применения наивных байесовских классификаторов и SVM. Из них видно, что SVM показывает результаты гораздо лучшие, чем у наивных байесовских классификаторов, причём SVM, использующие в качестве ядер RBF-сети, показывают себя лучше всех прочих SVM.

Таблица 3.2 — Результаты, полученные с применением наивных байесовских классификаторов и SVM

Дополнительные входные признаки	Алгоритм	Точность классификатора на тестовой выборке
-	Наивный байесовский классификатор	0.4059
	SVM (RBF-ядро)	0.5330
	SVM (полиномиальное ядро)	0.5153
	SVM (сигмоидное ядро)	0.4631
Количество угловых в прошедших матчах	Наивный байесовский классификатор	0.4097
	SVM (RBF-ядро)	0.5131
	SVM (полиномиальное ядро)	0.5115
	SVM (сигмоидное ядро)	0.4629
Количество угловых в прошедших матчах	Наивный байесовский классификатор	0.4135
	SVM (RBF-ядро)	0.5119
	SVM (полиномиальное ядро)	0.5085
	SVM (сигмоидное ядро)	0.4650
Количество угловых и ударов в прошедших матчах	Наивный байесовский классификатор	0.4135
	SVM (RBF-ядро)	0.5119
	SVM (полиномиальное ядро)	0.5017
	SVM (сигмоидное ядро)	0.4688
Количество ударов, процент владения мячом в предыдущих матчах	Наивный байесовский классификатор	0.4165
	SVM (RBF-ядро)	0.5112
	SVM (полиномиальное ядро)	0.5104

	SVM (сигмоидное ядро)	0.4881
Количество угловых, ударов, процент владения мячом в предыдущих матчах	Наивный байесовский классификатор	0.4169
	SVM (RBF-ядро)	0.5081
	SVM (полиномиальное ядро)	0.5085
	SVM (сигмоидное ядро)	0.4877

Последним рассмотренным классом алгоритмов машинного обучения для первого датасета стали алгоритмы на основе бустинга. Методы бустинга работают в том же духе, что и методы бэггинга: мы создаем семейство моделей, которые объединяются, чтобы получить сильного ученика, который лучше работает. Однако, в отличие от бэггинга, модели обучаются последовательно на различных данных: те объекты, на которых предыдущие классификаторы в последовательности допускали ошибки, чаще попадают в обучающую выборку. Таким образом, алгоритмы на основе принципа бустинга во время обучения пытаются создать новый классификатор, который будет лучше себя показывать на объектах, на которых текущий ансамбль ошибается [28]. Идея адаптивного бустинга, исторически первого алгоритма, использующего данный принцип, состоит в следующем: сперва строится линейная комбинация простых моделей, с помощью изменения весов для входных данных, а затем модель (обычно, дерево решений) строится на основе раннее не верных предсказаний, соответствующим объектам которых присваиваются большие веса. Изначально веса объектов задаются равными, в сумме дающими единицу, а затем используется следующее правило обновления:

$$w_j^{[t]} = w_j^{[t-1]} e^{-0.5 \ln \frac{1-\epsilon_t}{\epsilon_t} b_t(x_i)}, \tag{3.7}$$

где ε_t — ошибка на тренировочной выборке t-ого обучаемого слабого алгоритма, $b_t(x_i)$ — его предсказание на i-ом объекте. Таким образом, адаптивный бустинг обновляет веса объектов на каждой итерации. Веса хорошо классифицированных объектов уменьшаются относительно весов неправильно классифицированных объектов. После вычисления (12) происходит нормализация весов: чтобы удостоверится в том, сумма весов модели оставалась равной единице, каждый $w_j^{[t]}$ делиться на сумму всех весов, полученных на данной итерации. Данный алгоритм описан в [38].

Несмотря на хорошие результаты во многих задачах, в других, содержащих сильные выбросы во входных данных задачах, адаптивный бустинг склонен к переобучению. Для решения данной проблемы, было разработано его обобщение, известное под названием градиентный бустинг.

При нём модель ансамбля, которую мы пытаемся построить, также представляет собой взвешенную сумму слабых учеников. Градиентный бустинг сводит задачу к градиентному спуску: на каждой итерации новый ученик обучается с учётом антиградиента текущей функции ошибки текущей модели ансамбля. Стоит отметить, что антиградиент функции потерь является функцией, которая на практике может оцениваться только для объектов в обучающей выборке: эти оценки называются псевдо-остатками, прикреплёнными к каждому объекту:

$$r_{\text{im}} = \frac{-\partial L\left(y_i, F_{m-1}\left(x_i\right)\right)}{\partial F\left(x_i\right)} \tag{3.8}$$

После того, как псевдо-остатки для всех объектов обучающей выборки подсчитаны, новый алгоритм обучается на датасете (x_i, r_{im}) вместо исходного (x_i, y_i). После чего, полученная «слабая» модель h_m добавляется к линейной комбинацией ансамбля с множителем γ_m, являющимся решением оптимизационной задачи:

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L\left(y_i, F_{m-1}\left(x_i\right) + \gamma\, h_m\left(x_i\right)\right) \tag{3.9}$$

Более подробно алгоритм описан в [15]. В таблице 3.3 представим результаты применения алгоритмов, основанных на применении бустинга для классификации матчей по данным датасета Soccer match event dataset. Как видим, полученные модели в среднем показывают более высокую точность, чем все рассмотренные до этого.

Таблица 3.3 — Результаты, полученные с применением бустинга

Дополнительные входные признаки	Алгоритм	Точность классификатора на тестовой выборке
-	AdaBoost	0.5240
	Градиентный бустинг	0.5330
Количество угловых в прошедших матчах	AdaBoost	0.5229
	Градиентный бустинг	0.5206
Количество угловых в прошедших матчах	AdaBoost	0.5222
	Градиентный бустинг	0.5282
Количество угловых и	AdaBoost	0.5214

ударов в прошедших матчах	Градиентный бустинг	0.5282
Количество ударов, процент владения мячом в предыдущих матчах	AdaBoost	0.5218
	Градиентный бустинг	0.5328
Количество угловых, ударов, процент владения мячом в предыдущих матчах	AdaBoost	0.5237
	Градиентный бустинг	0.5343

Для повышения эффективности моделей, были рассмотрены методы понижения размерности. Под уменьшением размерности (англ. dimensionality reduction) в машинном обучении подразумевается уменьшение числа признаков набора данных. Наличие в нем признаков избыточных, неинформативных или слабо информативных может понизить эффективность модели, а после такого преобразования она упрощается, и соответственно уменьшается размер набора данных в памяти и ускоряется работа алгоритмов ML на нем [29]. В данной задаче широкое применение нашли алгоритмы обучения без учителя, не требующие разметки данных. В ходе работы, проводился поиск алгоритма с наилучшей предсказывающей возможностью среди использовавших уменьшение размерности до числа компонент равному 1/5, 2/5, 3/5 и 4/5 от числа исходных признаков.

Самым часто используемым алгоритмом для понижения размерности, является PCA (Principal Component Analysis). Основной идеей этого метода является поиск такой гиперплоскости, на которую при ортогональной проекции всех признаков максимизируется дисперсия. Пусть имеется n числовых признаков $f_j(x),j=1,...,n$. Объекты обучающей выборки будем отождествлять с их признаковыми описаниями: $x_i \equiv (f_1(x_i),...,f_n(x_i)),i=1,...,l$. Рассмотрим матрицу F, строки которой соответствуют признаковым описаниям обучающих объектов. Обозначим через $z_i=(g_1(x_i),...,g_m(x_i))$ признаковые описания тех же объектов в новом пространстве $Z=R^m$ меньшей размерности, $m<n$. Потребуем, чтобы исходные признаковые описания можно было восстановить по новым описаниям с помощью некоторого линейного преобразования, определяемого матрицей $U=(u_{js})_{n\times m}$. То есть, чтобы выполнялось равенство:

$$x' = zU^T \tag{3.10}$$

Восстановленное описание x’ не обязано в точности совпадать с исходным описанием x, но их отличие на объектах обучающей выборки должно быть как можно меньше при выбранной размерности m. Будем искать одновременно и матрицу новых признаковых описаний G, и матрицу линейного преобразования U, при которых суммарная невязка восстановленных описаний

минимальна по евклидовой норме. Оказывается, что минимум невязки достигается, когда столбцы матрицы U есть собственные векторы F^TF, соответствующие m максимальным собственным значениям. При этом G=FU, а матрицы U и G ортогональны. Подробнее об алгоритме PCA можно почитать в [34, 41].

Далее рассмотрим преобразование ICA (independent component analysis). Матрицу данных X будем считать линейной комбинацией независимых негауссовских компонент, то есть X = AS, где S — столбцы-независимые компоненты. Будем использовать алгоритм FastICA, предложенный в [21]. Суть его состоит в следующем:

1. Для лучшей сходимости к матрице X предварительно применяется преобразование отбеливания
2. Выбирается случайный вектор w(0), норма которого равна 1, переменной k присваивается значение 1
3. Значение вектора w обновляется по правилу

$$w(k)=E\left(X\left(w(k-1)^T x\right)^3\right)-3\,w(k-1) \tag{3.11}$$

4. Вектор w(k) делится на свою норму
5. Если значение $|w(k)^Tw(k-1)|$ недостаточно близкое к 1, то значение переменной k увеличивается на 1 и алгоритм переходит к шагу 3. Иначе, — w(k) считается одним из столбцов матрицы S.

Алгоритм имеет кубическую сходимость, поэтому обычно для получения результата достаточно 5-10 итераций. Для нахождения n независимых компонент, алгоритм прогоняется n раз, причём, чтобы удостовериться, что на каждой итерации оценивается новая компонента, начиная с некоторой итерации k перед нормализацией вектора, а так же сразу после инициализации w(0) рекомендуется отнимать от него $BB^Tw(k)$, где B — матрица, состоящая из уже найденных на предыдущих итерациях столбцов. Применение PCA и FastICA даёт существенное улучшение точности предсказаний для раннее полученных моделей, что иллюстрирует таблица 3.4.

Таблица 3.4 — Сравнение лучших результатов предсказаний до и после применения методов уменьшения размерности

Классификатор	Дополнительные признаки , использовавшиеся для обучения модели	Алгоритм уменьшения размерности	Изменение точности

Random Forest	Количество угловых, ударов, процент владения в предыдущих матчах	FastICA	0.5286 (+0.0072)
AdaBoost	Количество ударов, процент владения в предыдущих матчах	PCA	0.5354 (+0.0136)
Наивный байесовский классификатор	Количество угловых, ударов, процент владения в предыдущих матчах	PCA	0.5176 (+0.1007)
KNN	Количество угловых в предыдущих матчах	FastICA	0.5017 (+0.0401)
SVM(RBF-ядро)	Количество угловых в предыдущих матчах	FastICA	0.5381 (+0.0175)
Градиентный бустинг	Количество угловых, ударов, процент владения в предыдущих матчах	FastICA	0.5275 (-0.0068)
Бэггинг	Количество угловых в предыдущих матчах	FastICA	0.5055 (+0.0451)
SVM (полиномиальное ядро)	Количество угловых в предыдущих матчах	FastICA	0.5222 (+0.0107)
SVM (сигмоидное ядро)	Количество угловых, ударов, процент владения в предыдущих матчах	FastICA	0.5407 (+0.0530)

Для реализации всех алгоритмов в данном разделе применялась библиотека scikit-learn. Для Random Forest — объект класса RandomForestClassifier с 200 деревьями; для AdaBoost — AdaBoostClassifier с тем же количеством слабых алгоритмов; для наивного байесовского классификатора — GaussianNB; для SVM — svm.SVC с различными значениями параметра kernel, отвечающего за тип ядра;для градиентного бустинга и бэггинга — GradientBoostingClassifier и BaggingClassifier, соответственно. Поиск лучших моделей производился посредством перебора параметров из заданного диапазона с использованием объекта класса model_selection.GridSearchCV.

3.2 Алгоритмы, использовавшиеся для классификации данных из Soccer match event dataset.

3.2.1 Алгоритмы регрессии, использовавшиеся при построении моделей xG

В данном разделе опишем алгоритмы machine learning, использовавшиеся для построения моделей xG, ставящих каждому удару числовое значение от 0 до 1 и не упоминавшиеся ранее. Этот список состоит из следующих семейств алгоритмов машинного обучения: обобщённые линейные модели, XGBoost, экстремальные случайные леса, нейронные сети. Обобщённая линейная модель (ОЛМ) представляет собой гибкое обобщение классической линейной регрессии, которое позволяет использовать переменные реакции, имеющие модели распределения ошибок, отличные от нормального распределения. Суть линейной регрессии состоит в том, что искомая зависимость представляется в виде:

$$y = Xw + \epsilon, \tag{3.12}$$

где w — вектор параметров модели, X — матрица наблюдений. Тогда, согласно теореме Гаусса-Маркова, для w оценка метода наименьших квадратов будет оптимальной в классе линейных несмещённых оценок, поэтому в качестве w для модели будем рассматривать

$$\hat{w} = \left(X^T X\right)^{-1} X^T Y \tag{3.13}$$

Примером обобщённой модели линейной регрессии является логистическая регрессия, позволяющая использовать модель регрессии для решения задач классификации. Пусть w^TX задаёт плоскость, разделяющую объекты принадлежащие классу и не принадлежащие ему. Тогда вероятность принадлежности объекта к классу будет равна сигмоиде от w^TX, объект будем относить к классу, если значение больше порогового значения, а для получения w необходимо решить оптимизационную задачу:

$$\sum_{i=1}^{l} \ln\left(1+e^{-y_i w^T x_i}\right) \to min \tag{3.14}$$

ОЛМ состоит из трёх компонентов:

- Функция плотности $f(y;\theta,\phi)$ из семейства схожих случайных величин с параметрами θ и ϕ. В данной работе использовались, например, такие семейства распределений, как нормальное, Пуассона, гамма. С полным списком можно ознакомиться в документации библиотеки H2O [16]
- Линейная модель η: $\eta=X\beta$, где X — матрица данных.
- Функция связи g: $E(y)=\mu=g^{-1}(\eta)$, которая связывает отклик μ со значениями линейной модели η. Единственные ограничения, накладываемые на неё —монотонность и дифференцируемость. Благодаря данной функции, обеспечивается то, что отклик принадлежит к заданному диапазону значений (например, от 0 до 1 в случае логистической регрессии)

Экстремальный градиентный бустинг (XGBoost) — это одна из реализаций алгоритма градиентного бустинга на деревьях решений, которая отличается использованием производных второго порядка для улучшения сходимости и применением L1- и L2- регуляризаций для штрафования слишком сложных моделей, что помогает избежать переобучения. Также, алгоритм использует собственный метод определения оптимальных точек разделения решающих деревья и заполняет пропущенные в датасете значения в зависимости от значения функции потерь, что упрощает работу с разреженными данными, о чём можно подробнее прочитать в [10]. Также стоит отметить то, что XGBoost строит деревья решений параллельно, что существенно улучшает производительность алгоритма. Также отметим ещё одну модификацию раннее рассмотренных ансамблей деревьев. Помимо Random Forest, в данном разделе рассматривался также Extreme Random Forest (экстремальный случайный лес), который отличается тем, что при построении точек разделения деревьев используются не оптимальные разделения выборок, а лучшие среди некого случайного их подмножества, а также тем, что все деревья строятся на одной выборке, то есть не применяется принцип бэггинга.

Из нейронных сетей рассматривался только многослойный перцептрон — класс искусственных нейронных сетей прямого распространения, состоящих как минимум из трех слоёв: входного, скрытого и выходного. За исключением входных, все нейроны используют нелинейную функцию активации (например, tanh, ReLU, maxout). При обучении нейронных сетей применялась техника повышения точности dropout: при обучении заданная доля входных признаков и нейронов скрытых слоёв сети удаляются, что помогает избежать переобучения. Также, применялись L1- и L2-регуляризации, ограничивающие, соответственно, абсолютные значения весов нейронов и сумму их квадратов, тем самым опять же предотвращая переобучение. Для поиска глобального минимума функции ошибки при обучении использовалась модификация стохастического градиентного спуска ADADELTA, разработанная Google и учёными Нью-Йоркского университета, описанная в [45].

Также, на последнем этапе обучения, конструировались два ансамбля из индивидуальных моделей. Для формирования ансамблей использовался принцип смешивания. В отличие от бэггинга и бустинга, цель обучения ансамбля при помощи смешивания состоит в том, чтобы объединить в один алгоритм множество алгоритмов различной природы с слиьной предсказывающей способностью. При смешивании обучающую выборку делят на две части: на первой обучают базовые алгоритмы, затем получают их ответы на второй части и на тестовой выборке. Затем, ответы обученных алгоритмов добавляют к датасету в качестве новых признаков и обучают метаалгоритм на полученных данных. Обоснованность данного подхода доказывается, например, в [25]. В качестве метаалгоритма применялась ОЛМ с L1- или L2-регуляризацией (определялась после процедуры перебора гиперпараметров). В процессе обучения генерировалось два ансамбля: один на основе всех полученных моделей, второй — на основе лучших моделей из каждого семейства алгоритмов.

Результаты обучения пяти лучших (по абсолютной средней ошибке на тестовой выборке) моделей xG с соответствующим набором входных данных и временем обучения, также ставших лучшими в своём семействе, приведены в таблице 3.5. Она показывает, что модели, использующие информацию о направлении удара, а также модели только с позицией удара, обучавшиеся в течении долгого времени, показывают себя лучше моделей, при обучении которых используются все сгенерированные признаки, кроме зоны. Также стоит отметить, что ОЛМ показывают себя сильно хуже остальных рассматриваемых алгоритмов и ни разу не попали в пятёрку лучших моделей.

Таблица 3.5 — Оценка точности моделей ожидаемых голов

Набор входных признаков	Тип модели	Время обучения	Абсолютная средняя ошибка на тестовой выборке
Все, кроме зоны	Ансамбль всех моделей	Не более 10 минут	0.1544
	Ансамбль лучших моделей из каждого семейства		0.1610
	Нейронная сеть		0.1676
	XGBoost		0.1614
	Ансамбль всех моделей	Не ограничено	0.1635
	Ансамбль лучших моделей из каждого семейства		0.1843
	Градиентный бустинг		0.1610
Все	Ансамбль всех моделей	Не более 10 минут	0.1085
	Ансамбль лучших моделей из каждого семейства		0.1097
	Нейронная сеть		0.1050
	XGBoost		0.1158
	Ансамбль всех моделей	Не ограничено	0.1156
	Ансамбль лучших моделей из каждого семейства		0.1061
	Градиентный бустинг		0.1047
Только дистанция до ворот и угол	Ансамбль всех моделей	Не более 10 минут	0.1777
	Ансамбль лучших моделей из каждого семейства		0.1619
	Нейронная сеть		0.1618
	Экстремальный случайный лес		0.1609
	Ансамбль всех моделей	Не ограничено	0.1148

	Ансамбль лучших моделей из каждого семейства		0.1142
	Градиентный бустинг		0.1141

Для некоторых моделей, H2O предоставляет информацию о том, как тот или иной входной признак на результат предсказания. В частности, для моделей на основе решающих деревьев для её определения используются данные о количестве разделений, в которое вошёл признак, и о том, как выборка с его участием изменила квадратичную ошибку всего ансамбля. График с этими данными для лучшей модели приведён на рисунке 3.2. Из него видно, что из признаков наибольшее значение, как и ожидалось из данных о корреляции из второй главы, имеют точка, с которой нанесён удар, и зона, куда был направлен мяч.

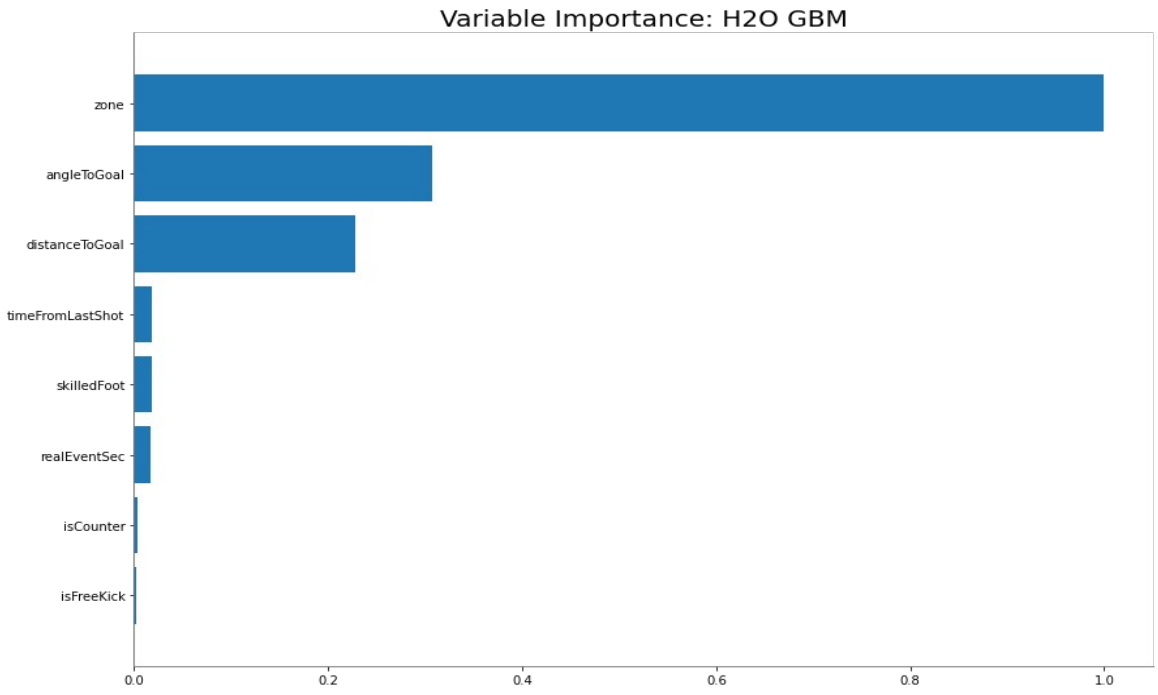

Рисунок 3.2 — Данные о значимости каждого из входных признаков лучшей модели

3.2.2 Результаты обучения классификаторов матчей по результату.

По аналогии с European soccer database, были проведены эксперименты по обучению моделей, не использующих в качестве входных признаков данные продвинутой статистики. Путём парсинга JSON-файлов, были выделены

данные о событиях из матчей предыдущих туров всего турнира, а также матчей последних пяти туров с участием команды. Были подсчитаны среднее количество ударов, наносимых командой, среднее количество ударов, допускаемое командами по своим воротам, аналогичные числа для угловых и голов. Также было подсчитано количество очков, заработанных командами перед началом матча.

Как и в European soccer database, была построена таблица корреляции признаков между собой. Сильной корреляции между выделенными признаками и результатом не наблюдается, но наиболее сильно коррелируют с ним количество созданных моментов домашней командой во всех предыдущих матчах (0.0617) и в последних пяти турах (0.0583), а также количество угловых ударов, исполненных домашней командой в последних пяти матчах (0.0532). Перед использованием в процессе обучения ранее полученных моделей xG, сперва были проведены эксперименты с моделями, использующими в качестве входных только признаки, описанные в предыдущем абзаце. Здесь и далее под ошибкой на классе X будем подразумевать отношение числа объектов в выборке, принадлежащих к классу X, но не отнесённых к нему алгоритмом классификации, к общему числу объектов класса X в выборке. Под классами «D», «L», «W» будем понимать ничьи, победы гостевой команды и победы домашней команды соответственно. Лучшие результаты обучения приведены в таблице 3.6. Из неё видно, что, весьма ожидаемо, наибольшие трудности алгоритмы испытывают в определении ничейных матчей. Относительно неплохо с этим справляются только решения на основе нейронных сетей, но они показывают низкие цифры по точности классификации побед команды-хозяина. Также отметим, что большее время обучения для таких моделей скорее приводит к переобучению, чем к повышению точности.

Таблица 3.6 — Оценка точности моделей предсказания результата матча без использования ожидаемых голов

Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Градиентный бустинг	Не более 10 минут	0.9016	0.4722	0.2857	0.5173
XGBoost		0.7705	0.5139	0.4671	0.5328
Градиентный бустинг	Не ограничено	0.8514	0.5844	0.2720	0.4948
XGBoost		0.8514	0.5974	0.3015	0.4774
Нейронная сеть		0.6486	0.7143	0.5000	0.4042

Далее было проведено обучение различных моделей с использованием сумм xG, сгенерированных командой и допущенных у своих ворот за весь турнир, а также за 5 последних матчей. Результаты обучения десяти лучших по точности на всех классах приведены в таблице 3.7. Из неё видно, что обобщённые линейные модели всё ещё плохо себя показывают, в отличие от моделей на основе градиентного бустинга и нейронных сетей. Также никуда не исчезли проблемы с предсказыванием ничьих (у лучшей модели по общей точности вообще отсутствуют верно предсказанные ничьи), однако использование «продвинутой статистики» существенно уменьшает ошибку на классе домашних и гостевых побед. Также стоит отметить, что в таблице отсутствуют модели, которые бы использовали в качестве входных признаков все, кроме зоны, куда пришёлся удар.

Таблица 3.7 — Оценка точности моделей предсказания результата матча с использованием ожидаемых голов

Входные признаки модели xG, время её обучения	Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Все, не более 10 минут	XGBoost	Не более 10 минут	0.9333	0.4946	0.1238	0.5542
Все, не более 10 минут	Нейронная сеть	Не ограничено	0.8542	0.4699	0.2583	0.5578
Все, не ограничено	Нейронная сеть	Не ограничено	0.9215	0.4615	0.2051	0.5650
Только дистанция и угол, не ограничено	Градиентный бустинг	Не более 10 минут	0.9310	0.4471	0.1826	0.5506
Только дистанция и угол, не ограничено	XGBoost	Не более 10 минут	0.9483	0.4353	0.1730	0.5546

Только дистанция и угол, не более 10 минут	Нейронная сеть	Не ограничено	1	0.3493	0.2066	0.5703
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.8387	0.4219	0.2276	0.5703
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9677	0.4375	0.1870	0.5542
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9355	0.4531	0.1951	0.5542

Далее приведём результаты обучения десяти лучших моделей с использованием xP, подсчитанного при помощи таблицы 2.1 на основе разницы сгенерированных и допущенных у своих ворот xG. Как видно из таблицы 3.8, введение данного признака повысило лучшую точность до 0.6015, причём все десять лучших результатов были достигнуты на моделях XGBoost.

Таблица 3.8 — Оценка точности моделей предсказания результата матча с использованием ожидаемых очков, посчитанных при помощи таблицы 2.1

Входные признаки модели xG, время её обучения	Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9818	0.4342	0.1500	0.6015

Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9825	0.4286	0.1610	0.5714
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9474	0.5065	0.1694	0.5515
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9649	0.4415	0.1864	0.5595
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9649	0.4415	0.1441	0.5793
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9455	0.4342	0.1715	0.5977
Все, не ограничено	XGBoost	Не более 10 минут	0.9322	0.3835	0.2171	0.5747
Все, не ограничено	XGBoost	Не более 10 минут	0.9322	0.3973	0.1938	0.5823
Все, не ограничено	XGBoost	Не более 10 минут	0.9800	0.4884	0.1870	0.5598
Все, не ограничено	XGBoost	Не более 10 минут	0.9200	0.4651	0.2439	0.5521

Далее приведём результаты моделей, использующих в качестве входных данных xP, посчитанные по формуле (2.3) в предположении, что число голов есть случайная величина, имеющая Пуассоновское распределение. Как видно из таблицы ниже, результаты не сильно отличаются от полученных при использовании моделей, использующих xG без ожидаемых очков.

Таблица 3.9 — Оценка точности моделей предсказания результата матча с использованием ожидаемых очков, посчитанных при помощи формулы 2.3

Входные признаки модели xG, время её обучения	Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.8596	0.4578	0.2131	0.5687
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9355	0.4931	0.1864	0.5415
Все, не ограничено	XGBoost	Не ограничено	0.8788	0.4286	0.2174	0.5491
Только дистанция и угол, не ограничено	Градиентный бустинг	Не ограничено	0.8000	0.4418	0.2222	0.5604
Только дистанция и угол, не ограничено	XGBoost	Не ограничено	0.9608	0.5617	0.1875	0.5476
Только дистанция и угол, не ограничено	Градиентный бустинг	Не ограничено	0.9412	0.4932	0.2188	0.5555
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9677	0.4189	0.2240	0.5441
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9516	0.4459	0.1920	0.5555

Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.8870	0.3918	0.2720	0.5479
Только дистанция и угол, не более 10 минут	Нейронная сеть	Не более 10 минут	1.0000	0.4658	0.1923	0.5455

Наконец, приведём результаты моделей, использующих в качестве входных данных xP, посчитанные при помощи симуляций серии испытаний Бернулли. Больших различий с числами из предыдущей таблицы в ней не наблюдается.

Таблица 3.10 — Оценка точности моделей предсказания результата матча с использованием ожидаемых очков, посчитанных при помощи симуляций серии испытаний Бернулли

Входные признаки модели xG, время её обучения	Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Все, не ограничено	XGBoost	Не более 10 минут	0.9194	0.4933	0.1826	0.5437
Все, не ограничено	XGBoost	Не более 10 минут	0.9259	0.4719	0.2037	0.5458
Все, не ограничено	Градиентный бустинг	Не более 10 минут	0.9444	0.4494	0.1852	0.5578
Все, кроме зоны, не ограничено	Градиентный бустинг	Не более 10 минут	0.9516	0.4203	0.2441	0.5388
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9677	0.4348	0.2047	0.5504

Все, кроме зоны, не ограничено	Нейронная сеть	Не более 10 минут	1.000	0.4348	0.2047	0.5426
Все, не ограничено	XGBoost	Не более 10 минут	0.9444	0.4831	0.1944	0.5419
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.8393	0.5895	0.1681	0.54444
Все, не ограничено	XGBoost	Не более 10 минут	0.9722	0.4706	0.1062	0.5494
Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.9107	0.4947	0.1681	0.5630

Наконец, было выдвинуто предположение о том, что большая ошибка на классе ничьих может быть вызвана несбалансированностью данных, ведь ничьих в датасете всего 454, в то время как побед домашней команды аж 822. Известно, что классификаторы, построенные на основе выборки, в которой репрезентативность классов несбалансирована, имеют в процессе практического использования склонность с большей вероятностью относить новые наблюдения к классам, представленным большим числом обучающих примеров [2]. Для решения этой проблемы, использовались встроенные средства пакета H2O. При задании опции balance_classes, система будет или уменьшать количество представителей большего класса в обучающей выборке или уменьшать количество представителей большего. Тем не менее, результаты предсказания такого классификатора после обучения корректируются с помощью монотонного преобразования, позволяющего учесть соотношение классов в исходной выборке [8]. Результаты обучения пяти лучших моделей с использованием балансировки классов приведены в таблице 3.11.

Таблица 3.11 — Оценка точности моделей предсказания результата матча с использованием балансировки классов в обучающей выборке

Входные признаки модели xG, время её обучения	Тип модели	Время обучения	Ошибка на классе “D”	Ошибка на классе “L”	Ошибка на классе “W”	Точность на всех классах
Все, кроме зоны, не ограничено	Градиентный бустинг	Не более 10 минут	0.9516	0.4247	0.1570	0.5742
Все, кроме зоны, не ограничено	Градиентный бустинг	Не более 10 минут	0.9355	0.4931	0.1322	0.5703
Все	XGBoost	Не более 10 минут	0.8750	0.4699	0.2566	0.5536
Только дистанция и угол, не ограничено	XGBoost	Не более 10 минут	0.8806	0.4286	0.1680	0.5800
Только дистанция и угол, не ограничено	XGBoost	Не более 10 минут	0.9403	0.4524	0.1680	0.5580
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	1.0000	0.5176	0.1339	0.5571
Только дистанция и угол, не более 10 минут	Градиентный бустинг	Не более 10 минут	0.9322	0.5412	0.0944	0.5830
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9830	0.5294	0.1417	0.5535

Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9467	0.4265	0.1500	0.5513
Только дистанция и угол, не более 10 минут	XGBoost	Не более 10 минут	0.9091	0.4203	0.2205	0.5534

Как видно из таблицы, балансировка классов не уменьшает ошибку на классе ничьих, из чего можно сделать вывод о том, что большая ошибка классификации на нём вызвана не несбалансированностью выборки или переобучением моделей, а самой природой ничьих. Дабы подтвердить это предположение были проведены опыты по классификации матчей на ничейные и неничейные. Результаты приведём в таблице 3.12. Из неё видно, что действительно отделение ничьих от побед и поражений домашней команды не представляется возможным. Отметим, что наилучших показателей достигли модели на основе нейронных сетей.

Таблица 3.12 — Оценка точности моделей классификации матчей на ничейные и нет

Входные признаки модели xG, время её обучения	Тип модели-классификатора	Время обучения	Точность
Все, не ограничено	Нейронная сеть	Не ограничено	0.5793
Все, не ограничено	Нейронная сеть	Не более 10 минут	0.5659
Все, не более 10 минут	Нейронная сеть	Не ограничено	0.5795
Все, не ограничено	Нейронная сеть	Не ограничено	0.5848
Только координаты и зона удара, не более 10 минут	Градиентный бустинг	Не более 10 минут	0.5709

Далее были проведены по влиянию использования рейтинга Эло в качестве входного параметра для моделей-классификаторов матчей. Напомним, что использовались следующие разновидности рейтинга Эло:

1. Очки рейтинга Эло начисляются аналогично шахматам (1 очко за победу, 0 очков за поражение, 0.5 за ничью), k = 64

2. Очки рейтинга Эло начисляются аналогично шахматам (далее будем называть такой тип рейтинга «шахматным»), k = 20
3. Очки рейтинга Эло начисляются в зависимости от доли голов, забитых командой в матче (далее будем называть такой тип рейтинга «футбольным»), k = 64.
4. Очки рейтинга Эло начисляются в зависимости от доли голов, забитых командой в матче, k = 26(D+1), где D — модуль разницы забитых командами голов.
5. Очки рейтинга Эло начисляются в зависимости от доли xG, созданных командой в матче, k = 64.

В таблице 3.13 приведём результаты двух лучших по точности моделей для каждого из типов рейтинга Эло.

Таблица 3.13 — Оценка точности моделей предсказания результата матча с использованием балансировки классов в обучающей выборке

Тип рейтинга Эло	Входные признаки модели xG, время её обучения	Тип модели-классификатора	Время обучения	Точность
«Шахматный», k=64	-	Градиентный бустинг	Не ограничено	0.5778
«Шахматный», k=64	Все, кроме зоны, не более 10 минут	Градиентный бустинг	Не более 10 минут	0.5588
«Шахматный», k=20	Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.5904
«Шахматный», k=20	Все, кроме зоны, не ограничено	XGBoost	Не более 10 минут	0.5900
«Футбольный», k=64	Все, кроме координат удара, не ограничено	Градиентный бустинг	Не более 10 минут	0.5685
«Футбольный», k=64	Все, кроме координат удара, не ограничено	Градиентный бустинг	Не более 10 минут	0.5856

«Футбольный», k=26(D+1)	Все, кроме зоны, не более 10 минут	XGBoost	Не более 10 минут	0.5809
«Футбольный», k=26(D+1)	Все, не более 10 минут	XGBoost	Не более 10 минут	0.5907
По доле xG	Все, не ограничено	Градиентный бустинг	Не более 10 минут	0.5654
По доле xG	Все, не ограничено	XGBoost	Не более 10 минут	0.5599

Как видно, использование рейтинга Эло оказывает положительное влияние на точность моделей, в том числе, не использующих продвинутую статистику. Лучше всего себя показывают «футбольный» рейтинг с непостоянным k и «шахматный» с низким значением k.

Для реализации и оценки точности алгоритмов машинного обучения, описанных в данном разделе, применялся фреймворк H2O. Применялась библиотека matplotlib (для визуализации данных), pandas и numpy(для обработки табличных данных), для построения моделей применялся фреймворк H2O и, в частности, алгоритм автоматического подбора гиперпараметров AutoML, являющийся его частью.

ЗАКЛЮЧЕНИЕ

В процессе работы были изучены и проанализированы алгоритмы машинного обучения, которые могут быть использованы для решения задачи предсказания результата (отнесения к одному из трёх классов: победа домашней команды, ничья, поражение домашней команды) футбольных матчей. В рамках работы был произведён поиск данных, необходимых для обучения алгоритмов, были проанализированы XML- и JSON-файлы с подробной статистикой матча из датасетов European soccer database и Soccer match event dataset соответственно, включающие, например, данные об угловых, ударах, нанесённых командами, координатах футбольного поля, где они произошли, владении мячом, букмекерских котировках на матчи, а также об игроках в них участвующих. На языке программирования Python с применением библиотек scikit-learn и h2o были реализованы модели, в основе которых лежат алгоритмы KNN; обобщённые линейные модели, модели градиентного бустинга (и в частности XGBoost), многослойные полносвязные нейронные сети, метод опорных векторов, наивные байесовские классификаторы; лучшие представители каждого из семейств алгоритмов объединялись в ансамбли.

Была изучена литература на тему моделей ожидаемых голов (xG), и был обучен ряд моделей, присваивающих удару число от 0 до 1, которое можно интерпретировать в качестве вероятности того, что удар станет голевым. Для построения данных регрессионных моделей были использованы обобщённые линейные модели, модели градиентного бустинга (и в частности XGBoost), многослойные полносвязные нейронные сети, случайные и экстремальные случайные леса. Были рассмотрены модели xG, использующие различные характеристики удара. Было показано, что наибольшую ценность для таких моделей представляют данные о зоне ворот, куда пришёлся удар, а также дистанции до ворот и «угле обстрела». Более того, была изучена и применена развивающая концепцию ожидаемых голов концепция ожидаемых очков (xPoints, xP), по которой каждому из матчей сыгранных командой присваивается число между 0 и 3, которое можно воспринимать как оценку выступления команды в нём на основе данных о xG, то, сколько очков команда заслужила заработать по итогам матча. Были использованы несколько подходов для подсчёта xPoints: вычисление по формуле, где в качестве аргумента выступает разность xG заработанных командой и допущенных ей около своих работ, вычисление вероятности того, что пуассоновская случайная величина с математическим ожиданием равным сумме xG созданных командой больше аналогичной суммы для команды соперника, рассматривание реализации голевого момента как случайной величины Бернулли с математическим ожиданием равным xG удара и подсчёт xP с помощью симуляции серий испытаний для каждого удара обеих команд. Было показано, что использование

xP в качестве входного признака оказывает положительное влияние на точность модели. По итогам работы были построены множество моделей, точность на тестовой выборке которых превысила точность лучшей модели, не использующей их в качестве входных признаков (0.5407), что обосновывает применение xG и xP для предсказания исхода матча. Лучший результат из достигнутых, 0.6015, показан алгоритмом XGBoost, использовавшим в качестве входных данных информацию о созданных в матчах до этого ожидаемых голах, для подсчёта которых использовались только дистанция до ворот и угол, и об ожидаемых очках, рассчитанных по таблице разностей xG. Кроме того, в качестве выводов, можно отметить то, что подавляющее большинство лучших моделей-классификаторов использовали алгоритмы градиентного бустинга и, в частности, XGBoost.

Также, изучалось применение в качестве одного из признаков предсказывающих моделей рейтинга Эло. Рассматривались несколько его типов в зависимости от типа начисляемых очков и коэффициента k, обуславливающего степень влияния последних результатов на рейтинг команды. Использовались рейтинги Эло, в которых команды получали очки в зависимости от результата матча, аналогично шахматному рейтингу Эло: 0 за поражение, 0.5 за ничью и 1 за победу; в которых очки определялись в зависимости от доли голов, забитых командой; в зависимости от доли xG, сгенерированных командой. Применялись низкое значение k = 20, завышенное k = 64 и переменное, определяемое разницей голов в матче. Полученные результаты показали, что применение рейтинга Эло позволяет достигнуть высокой точности даже моделям, не использующим продвинутую статистику (xP и xG). Так, одна из моделей на основе градиентного бустинга при помощи использования «шахматной» формулы рейтинга Эло с k=64 достигла точности в 0.5778, в то время как лучший результат по точности таких моделей до этого был 0.5407. Наибольшую точность среди моделей, использовавших рейтинг Эло, показала модель на основе XGBoost, одним из входных признаков которой был рейтинг Эло на основе доли голов с непостоянным k, — 0.5907

Для улучшения точности моделей с целью исключения менее информативных признаков были применены алгоритмы уменьшения размерности PCA и FastICA, которые в большинстве случаев повысили точность лучших представителей каждого из рассматривавшихся семейств алгоритмов. Например, точность лучшей модели на основе наивного байесовского классификатора повысилась с 0.4169 до 0.5176.

После анализа полученных результатов стало понятно, что основным барьером, ограничивающим точность получаемых моделей, являются высокие, близкие к 100% ошибки при классификации матчей с ничейным исходом. Было выдвинуто предположение, что проблема может заключаться в меньшем по отношению к другим классам количестве ничьих в датасете. Было принято решение провести эксперименты с балансировкой классов (уравниванием количества их представителей в выборке) при обучении, однако это они не оказали никакого влияния на точность предсказания ничьих, из чего был сделан

вывод о том, что большая ошибка вызвана не несбалансированностью выборки или переобучением моделей, а самой природой ничьих. Этот вывод был подтверждён после проведения экспериментов по обучению моделей для решения более простой задачи — классификации на ничейные и нет. Стоит отметить, что лучше других себя показывали модели на основе многослойного перцептрона, лучшая из которых достигла точности 0.5848.

Также был произведён обзор научной литературы о применении алгоритмов ML, не реализовавшихся в рамках данной работы, для решения задачи классификации матчей по исходам. Наибольший интерес представляют байесовские сети, порой достигающие крайне высокой точности в предсказании результата матчей одной конкретной команды в отдельно взятом сезоне, однако такие модели являются сложными в построении из-за необходимости экспертного знания и быстро устаревают.

Код на языке Python, использовавшийся для обучения моделей, доступен по ссылкам https://bit.ly/3gDHbj8 и https://bit.ly/3f0efD0.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Воронцов, К. В. Лекции по методу опорных векторов/ К. В. Воронцов // http://www.ccas.ru/voron/download/SVM.pdf [Электронный ресурс]. — 2007. — Режим доступа: http://tka4.org/materials/lib/Articles-Books/Speech%20Recognition/from%20Nickolas/SVM.pdf. — Дата доступа: 07.12.2020.
2. Паклин, Н. Б. Построение классификаторов на несбалансированных выборках на примере кредитного скоринга / Н. Б. Паклин, С. В. Уланов, С. В. Царьков // Искусственный интеллект. — 2010. — № 3. — С. 528–534.
3. Петров, Д. 3 из 10 главных лиг Европы завершили сезон. кто еще надеется доиграть и когда? / Петров Д. // Матч ТВ [Электронный ресурс]. — 2020. — Режим доступа: https://bit.ly/2W3xAsg. — Дата доступа: 01.12.2020.c
4. Руткевич, В. Н. Оценка работы в вузе по рейтинговой системе / В. Н. Руткевич // Роль университетского образования и науки в современном обществе : материалы междунар. науч. конф., Минск, 26–27 февр. 2019 г. / Белорус. гос. ун-т ; редкол.: А. Д. Король (пред.) [и др.]. — Минск : БГУ, 2019. — С. 396-400.
5. Симонова, С. И. Интеллектуальный анализ данных для задач CRM / С. И. Симонова // International Journal of Open Information Technologies. — 2015. — №2.
6. Anderson, C. and Sally, D. The numbers game. / C. Anderson. — New York, 2013
7. Arabzad, S., Tayebi Araghi, M., Sadi-Nezhad, S., Ghofrani, N. Football Match Results Prediction Using Artificial Neural Networks; The Case of Iran Pro League. / S. Arabzad // Journal of Applied Research on Industrial Engineering. — 2014. — Vol.1, №3. — P. 159-179
8. balance_classes // H2O Documentation [Электронный ресурс]. — 2021. — Режим доступа: https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/algo-params/balance_classes.html. — Дата доступа: 09.02.2021.
9. Breiman L. [и др.]. Classification And Regression Trees / L. Breiman — Routledge, 2017.
10. Chen T., Guestrin C. XGBoost / T. Chen // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. — ACM. — 2016.
11. Dixon, M. J., & Coles, S. G Modelling Association Football Scores and Inefficiencies in the Football Betting Market / M. J. Dixon // Journal of the Royal Statistical Society: Series C (Applied Statistics). — 1997. — Vol.46 №2. — P. 265-280

12. Eggels, H. ; van Elk, R. ; Pechenizkiy, M. Explaining soccer match outcomes with goal scoring opportunities predictive analytics. / H. Eggels // Proceedings of the Workshop on Machine Learning and Data Mining for Sports Analytics 2016 co-located with the 2016 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2016). — Riva del Garda. — 2016
13. European Soccer Database // Kaggle [Электронный ресурс]. — 2016. — Режим доступа: https://www.kaggle.com/hugomathien/soccer/version/10 — Дата доступа: 01.12.2020.
14. Forrest, D., Simmons, R. Forecasting sport: the behaviour and performance of football tipsters / D. Forrest // International Journal of Forecasting. — 2000. — Vol. 3, №16. — P. 317–331.
15. Friedman, J. Greedy Function Approximation: A Gradient Boosting Machine/ Friedman, J. // The Annals of Statistics. — 2001. — Vol.29 №5
16. Generallized Linear Model (GLM) // H2O Documentation [Электронный ресурс]. — 2021. — Режим доступа: https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/algo-params/balance_classes.html. — Дата доступа: 09.02.2021.
17. Giulianotti, R. Football // R. Giulianotti // The Wiley-Blackwell Encyclopedia of Globalization. — 2012.
18. Goddard, J. Regression models for forecasting goals and match results in association football / J. Goddard // International Journal of Forecasting. — 2005. — Vol. 2, №2. — P. 331–340.
19. Hill, I. D. Association Football and Statistical Inference / I. D. Hill // Applied Statistics. — 1974. — Vol. 2, № 23. — P. 203.
20. Hvattum L.M., Arntzen H. Using ELO ratings for match result prediction in association football / L. M. Hvattum // International Journal of Forecasting. Elsevier BV. — 2010. — Vol. 26, № 3. — P. 460–470.
21. Hyvärinen, A., Oja, E. A Fast Fixed-Point Algorithm for Independent Component Analysis / A. Hyvärinen // Neural Computation. — 1997. — Vol.9 №7. — P. 1483–1492.
22. Joseph A., Fenton N. E., Neil M. Predicting football results using Bayesian nets and other machine learning techniques / A. Joseph // Knowledge-Based Systems. — 2006. — Vol. 7, № 19. — P. 544–553.
23. Lago-Peñas C. Gómez-Ruano, M., Megías-Navarro, D., & Pollard, R. Home advantage in football: Examining the effect of scoring first on match outcome in the five major European leagues / C. Lago-Peñas // International Journal of Performance Analysis in Sport. — 2016. — Vol. 16, № 2. — P. 411–421.
24. Lucey, P., Bialkowski A., Monfort, M., Carr, P., Matthews, I. Quality vs Quantity: Improved Shot Prediction in Soccer using Strategic Features from Spatiotemporal Data / P.Lucey // Proceedings of MIT Sloan Sports Analytics Conference. — Boston. — 2015

25. Laan M. J. van der, Polley E. C., Hubbard A. E. Super Learner / M. J. van der Laan // Statistical Applications in Genetics and Molecular Biology. — Vol 6, № 1. — 2017
26. Lasek, J. Euro 2016 Predictions Using Team Rating Systems / J. Lasek // Proceedings of MIT ECML/PKDD. — Riva del Garda. — 2016.
27. Moroney, M. J. Facts from figures / M. J. Moroney. — 3rd ed. — London, 1956. — P. 16
28. Opitz D., Maclin R. Popular Ensemble Methods: An Empirical Study / D. Optitz // Journal of Artificial Intelligence Research. — №11. — P. 169–198
29. Sahu B., Dehuri S., Jagadev A. A Study on the Relevance of Feature Selection Methods in Microarray Data / B. Sahu // Open Bioinforma. J. Bentham Science Publishers Ltd. — 2018. — Vol. 11, № 1. — P. 117–139.
30. Sannemo, J. Lindholm, S. Comparing the Predictive Power of Past Results Between Soccer Leagues / J. Sannemo; KTH Royal Institute of Technology. — Stockholm, 2016
31. Macdonald, B. An Expected Goals Model for Evaluating NHL Teams and Players / B.Macdonald // Proceedings of MIT Sloan Sports Analytics Conference. — Boston. — 2012
32. F. Owramipur, P. Eskandarian, and F. Sadat Mozneb F Football Result Prediction with Bayesian Network in Spanish League-Barcelona Team / F. Owramipur // International Journal of Computer Theory and Engineering. — 2013. — Vol. 5, №5
33. Pearl, J. Probabilistic Reasoning in Expert Systems: Networks of Plausible Inference / J. Pearl // Morgan Kaufmann — San Francisco, 1988. — 552 p.
34. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space / K. Pearson // The London, Edinburgh, and Dublin Philosophcal Magazine and Journal of Science. — 1901. — Vol. 2, № 11. — P. 559–572.
35. Premier League Projections and New Expected Goals// Cartilage Free Captain, a Tottenham Hotspur community [Электронный ресурс]. — 2015. — Режим доступа: https://bit.ly/3uZwVIo — Дата доступа: 09.12.2020.
36. Rotshtein, A.P., Posner, M. & Rakityanskaya, A.B. Football Predictions Based on a Fuzzy Model with Genetic and Neural Tuning. / A. P. Rotshtein // Cybern Syst Anal — 2005. — Vol.41. — P. 619–630
37. Rue, H., Salvesen, O. Prediction and Retrospective Analysis of Soccer Matches in a League // J Royal Statistical Soc. — 2000. — Vol. 49 № 3. — P. 399–418
38. Schapire, R. E. The Boosting Approach to Machine Learning: An Overview / R. E. Schapire// Nonlinear Estimation and Classification. — New York, 2003. — P. 149–171.
39. Single estimator versus bagging: bias-variance decomposition // scikit-learn [Электронный ресурс]. — 2021. — Режим доступа: https://scikit-learn.org/stable/auto_examples/ensemble/plot_bias_variance.htm. — Дата доступа: 07.12.2020.

40. Soccer match event dataset // Figshare [Электронный ресурс]. — 2020. — Режим доступа: https://bit.ly/3wkGYIn — Дата доступа: 14.03.2021.
41. Sylvester, J. J., On the reduction of a bilinear quantic of the nth order to the form of a sum of n products by a double orthogonal substitution / J. J. Silvester // Messenger of Mathematics. — 1889. — Vol. 19. — P. 42-46
42. When are the most goals scored in football? // BettingWell [Электронный ресурс]. — 2020. — Режим доступа: https://www.bettingwell.com/sports-betting-guide/football-bettors-guide/when-are-most-goals-scored-football. — Дата доступа: 08.02.2021.
43. xFootball: Что такое xG и как превратить xG в xPoints // Карриковедение [Электронный ресурс]. — 2016. — Режим доступа: https://carrick.ru/blogs/xfootball/. — Дата доступа: 15.11.2020.
44. xGunners: How to calculate xPoints // SB Nation [Электронный ресурс]. — 2017. — Режим доступа: https://theshortfuse.sbnation.com/2017/11/15/16655916/how-to-calculate-xpoints-analysis-stats-xg. — Дата доступа: 10.02.2021.
45. Zeiler M.D. ADADELTA: An Adaptive Learning Rate Method // CoRR. 2012. (abs/1212.5701).