/МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

\_\_\_\_\_\_\_\_/Караваева О.В./

(подпись) (Ф.И.О)

«\_\_»\_\_\_\_\_\_\_\_\_\_202\_г.

Разработка математической модели для прогнозирования статистических показателей игроков НБА

Пояснительная записка курсового проекта по дисциплине

«Комплекс знаний бакалавра в области программного и аппаратного обеспечения вычислительной техники»

ТПЖА.09.03.01.723 ПЗ

Разработал студент группы ИВТ-31 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Кудяшев Я.Ю./

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Караваева О.В./

Консультант \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Долженкова М.Л./

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (дата)

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ \_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

Киров 2022

Реферат

Кудяшев Я.Ю. Разработка приложения «Математическая модель для прогнозирования статистических показателей игроков НБА». ТПЖА.090301.021 ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Караваева О.В.. - Киров, 2022. – ПЗ 56с, 29 рис., 1 прил.

Объект курсового проекта – аналитика спортивных результатов.

Предмет курсового проекта – прогнозирование статистических показателей игроков НБА.

Цель курсового проекта – облегчить работу аналитиков при прогнозировании и обсуждении статистики игроков НБА.

Результатом выполнения курсового проекта является математическая модель, прогнозирующая статистику игроков НБА.

Тут будет задание на КП

Содержание

[ВВЕДЕНИЕ 6](#_Toc105707652)

[1 Обзор предметной области 7](#_Toc105707653)

[1.1 Обзор аналогов 8](#_Toc105707654)

[1.2 Общие сведения 10](#_Toc105707655)

[1.2.1 Полное наименование системы и её условное обозначение 10](#_Toc105707656)

[1.2.2 Наименование разработчика системы и реквизиты заказчика 10](#_Toc105707657)

[1.2.3 Основания для создания системы 10](#_Toc105707658)

[1.2.4 Плановые сроки начала и окончания работы по созданию системы. 10](#_Toc105707659)

[1.2.5 Источник финансирования работ по созданию системы 10](#_Toc105707660)

[1.3 Назначение и цели создания системы 11](#_Toc105707661)

[1.3.1 Назначение системы 11](#_Toc105707662)

[1.3.2 Цели создания системы 11](#_Toc105707663)

[1.4 Характеристики объекта автоматизации 11](#_Toc105707664)

[1.4.1 Краткие сведения об объекте автоматизации 11](#_Toc105707665)

[1.4.2 Сведения об эксплуатации объекта автоматизации и характеристики окружающей среды 11](#_Toc105707666)

[1.5 Выводы по обзору предметной области 12](#_Toc105707667)

[2 Проектирование программного продукта 13](#_Toc105707668)

[2.1 Исследование предметной области 13](#_Toc105707669)

[2.1.1 Закономерность развития карьеры игроков 13](#_Toc105707670)

[2.1.2 Динамичный рост показателей игрока 15](#_Toc105707671)

[2.1.3 Влияние стиля игры на показатели игроков 16](#_Toc105707672)

[2.1.4 Влияние возраста на показатели игрока 18](#_Toc105707673)

[2.1.5 Анализ взаимосвязи показателей 19](#_Toc105707674)

[2.1.6 Позиционный аспект показателей игроков 20](#_Toc105707675)

[2.2 Выбор модели 23](#_Toc105707676)

[2.2.1 Линейная регрессия 23](#_Toc105707677)

[2.2.2 Random Forrest 24](#_Toc105707678)

[2.2.3 Extra Trees 25](#_Toc105707679)

[2.2.4 XGBoost 26](#_Toc105707680)

[2.3 Подготовка данных 27](#_Toc105707681)

[2.4 Параметризация модели 29](#_Toc105707682)

[2.5 Выводы по части проектирования программного продукта 30](#_Toc105707683)

[3 Реализация модели 31](#_Toc105707684)

[3.1 Устранение ненужных показателей 31](#_Toc105707685)

[3.2 Представление показаталей в каноничном для математической модели виде 32](#_Toc105707686)

[3.3 Прогнозирование результатов 32](#_Toc105707687)

[3.4 Вывод по части реализации программного продукта 35](#_Toc105707688)

[4 Тестирование программной модели 36](#_Toc105707689)

[4.1 Модель линейной регрессии 36](#_Toc105707690)

[4.2 Модель Extra Trees 38](#_Toc105707691)

[4.3 Модель Random Forest 39](#_Toc105707692)

[4.4 Модель XGBoost 41](#_Toc105707693)

[4.5 Выводы по тестовой части 42](#_Toc105707694)

[ЗАКЛЮЧЕНИЕ 43](#_Toc105707695)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 44](#_Toc105707696)

[ПРИЛОЖЕНИЕ А 45](#_Toc105707697)

ВВЕДЕНИЕ

Человек, наблюдая за последними минутами финального матча сезона, в ожидании финального свистка, неловко ловит себя на мыслях: почему именно эта команда взяла чемпионский титул в сезоне, из-за чего другая команда уступила, как этот проигрыш отразится на будущем команды и всех игроков в целом, могло ли все быть иначе. Вряд ли простой баскетбольный фанат сможет ответить на эти вопросы. Именно для решения таких вопросов и существуют спортивные аналитики, готовые, опираясь на свой жизненный опыт и сподручные средства, дать ответ на подобного рода вопросы.

Прогнозирование статистики как игроков, так и команд на следующий сезон довольно сложное дело. Любому профессиональному аналитику или менеджеру спортивного клуба для принятия решения, от которого будет зависеть будущее клуба: какого игрока в команде стоит обменять, подойдет ли новый игрок под нынешний командный стиль игры, заслуживает ли он подписания большого контракта – необходимо опираться на факты и проверенную информацию. При составлении прогнозов на предстоящий сезон аналитики сталкиваются с главной проблемой: какую брать информацию и как её правильно обработать, чтобы получить желанный результат.

Решением этой проблемы может послужить математическая модель статистических показателей игроков НБА. Подробное изучение предметной области и наличие данных, дает возможность выявить закономерность их появления и дальнейшие изменения. Таким образом, правильно составленная математическая модель может послужить мощным аргументом в решении споров, связанных с прогнозом результатов на предстоящий игровой сезон в НБА.

1 Обзор предметной области

Сегодняшний мир сложно представить без спорта и спортивных мероприятий. Спортивная индустрия насчитывает огромное количество различных видов спорта, таких как футбол, баскетбол, хоккей и другие. Данный проект посвящён баскетболу, а именно – национальной баскетбольной ассоциации. НБА – самая популярная мужская баскетбольная лига, расположенная в Северной Америке. Она бы не пользовалась такой популярностью, если бы не «звёздные» игроки и их достижения. Подсчётом статистики и дальнейшим прогнозированием будущего того или иного игрока занимаются аналитики. Однако было бы довольно сложно следить за игрой каждого из членов лиги без использования дополнительных средств.

Рассматривались различные подходы к решению проблемы, но выбор пал на создание математической модели, предназначенной для прогнозирования сезонных показателей игроков. Данная модель, опираясь на результаты игроков предыдущих лет, способна предсказывать, как действующие игроки лиги будут действовать в течение своего следующего сезона в НБА. Помимо действующих игроков каждый год в лигу приходит большое количество новичком, уровень игры которых тоже довольно сложно предугадать: кто-то может стать «новичком года», а кому-то достанется прозвище «неудачник года». Математическая модель, опираясь на схожих игроков прошлых поколений, также способна прогнозировать то, как себя проявит новичок в свой дебютный год в лиге.

На момент реализации проекта регулярный сезон НБА 2021-2022 подошёл к концу, поэтому наиболее актуальным будет являться прогноз показателей игроков на 2022-2023 игровой сезон и на плей-офф 2021-2022 г. Главным помощником и опорой при создании проекта будет выступать база данных игроков и их статистических показателей начиная с сезона 1946-1947 и заканчивая сезоном 2021-2022.

1.1 Обзор аналогов

Создаваемая математическая модель не имеет полностью идентичных аналогов. Это обусловлено тем, что каждый разработчик при создании модели, ссылается на набор показателей и ключей для анализа, свойственных только ему. Вряд ли удастся найти модель, построенную точно по таким же критериям, что и эта.

Однако в любом случае это не отменяет наличие моделей для прогнозирования, имеющих другие алгоритмы поиска и вычисления статистики. Так, к примеру, на сайте overbetting.ru представлен список игроков, которые, по их мнению, возьмут звание МВП в следующих 5 сезонах, рисунок 1. Алгоритм, которого они придерживались при составлении списка, нет в открытом доступе, поэтому нельзя с точностью заявлять, что была разработана математическая модель. Возможно, автор ссылался лишь на свои навыки и опыт.

Еще одним примером аналога может послужить уже англоязычный источник project.fivethirtyeight.com, который прогнозирует шансы команд на выход в плей-офф, финал и просто их успехи в сезоне. На рисунке 2 показан пример оформления и отображения статистики.

Исходя из всего выше сказанного, становится хорошо понятно, что такие математические модели, для предсказания статистики игроков редко попадают в открытый доступ, и являются некой «визитной карточной» каждого аналитики, на которую они могут ссылаться при аргументировании своих решений. Главным отличием каждой математической модели в данной предметной области является то, как и какие обрабатывались данные для тренировки модели, что за особенности и тенденции развития баскетбола были учтены при создании и какие метрики были выбраны, как ключевые.

Хочется отметить: математическая модель для каждой предметной области являтся уникальной. Нельзя использовать, к примеру, математическую модель для прогнозирования погоды при предсказании статистики игроков: баскетбол имеет свои особенные метрики, показатели и закономерности, которые строились десятилетиями.

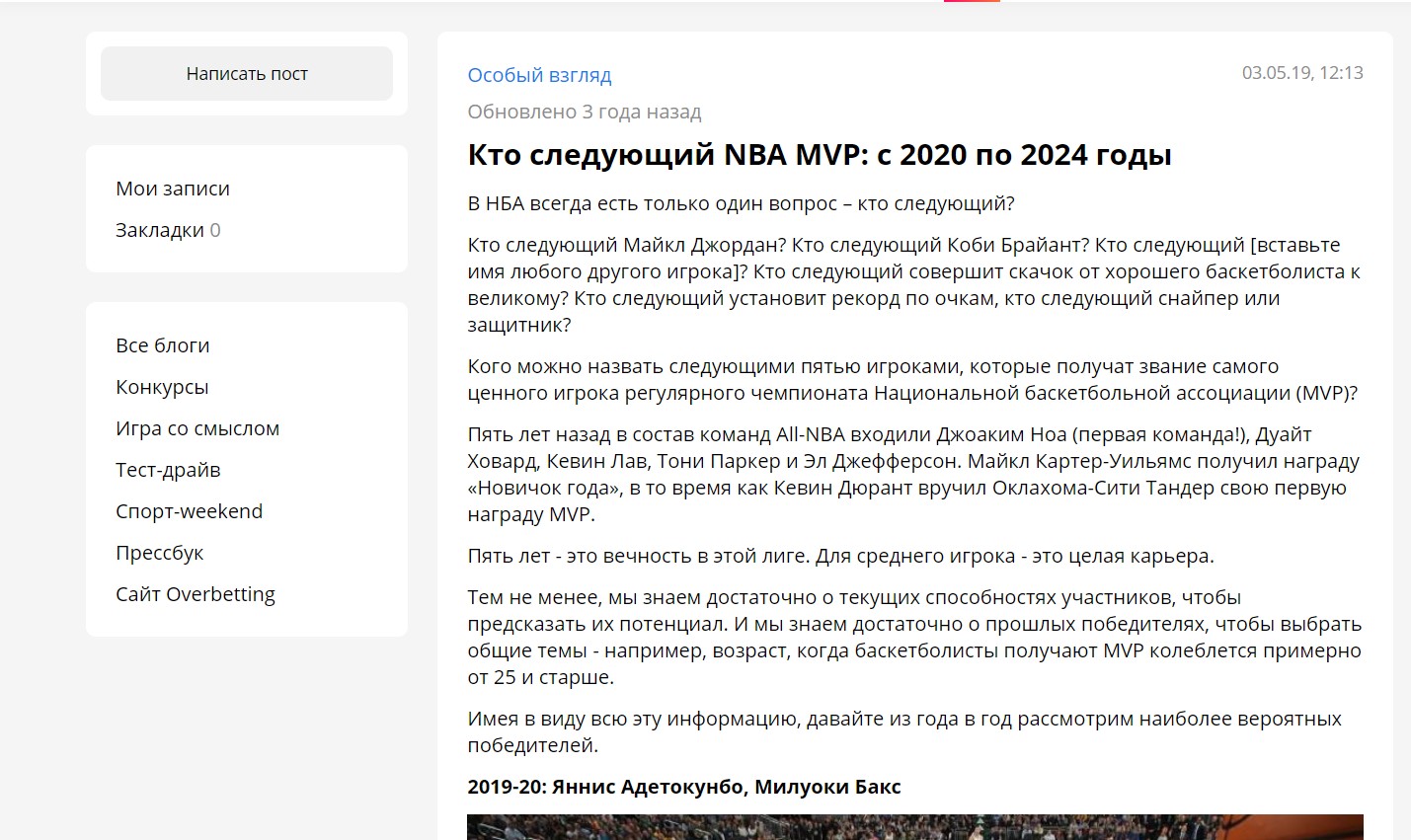


Рисунок 1 – Пример прогнозирования с сайта overbetting.ru

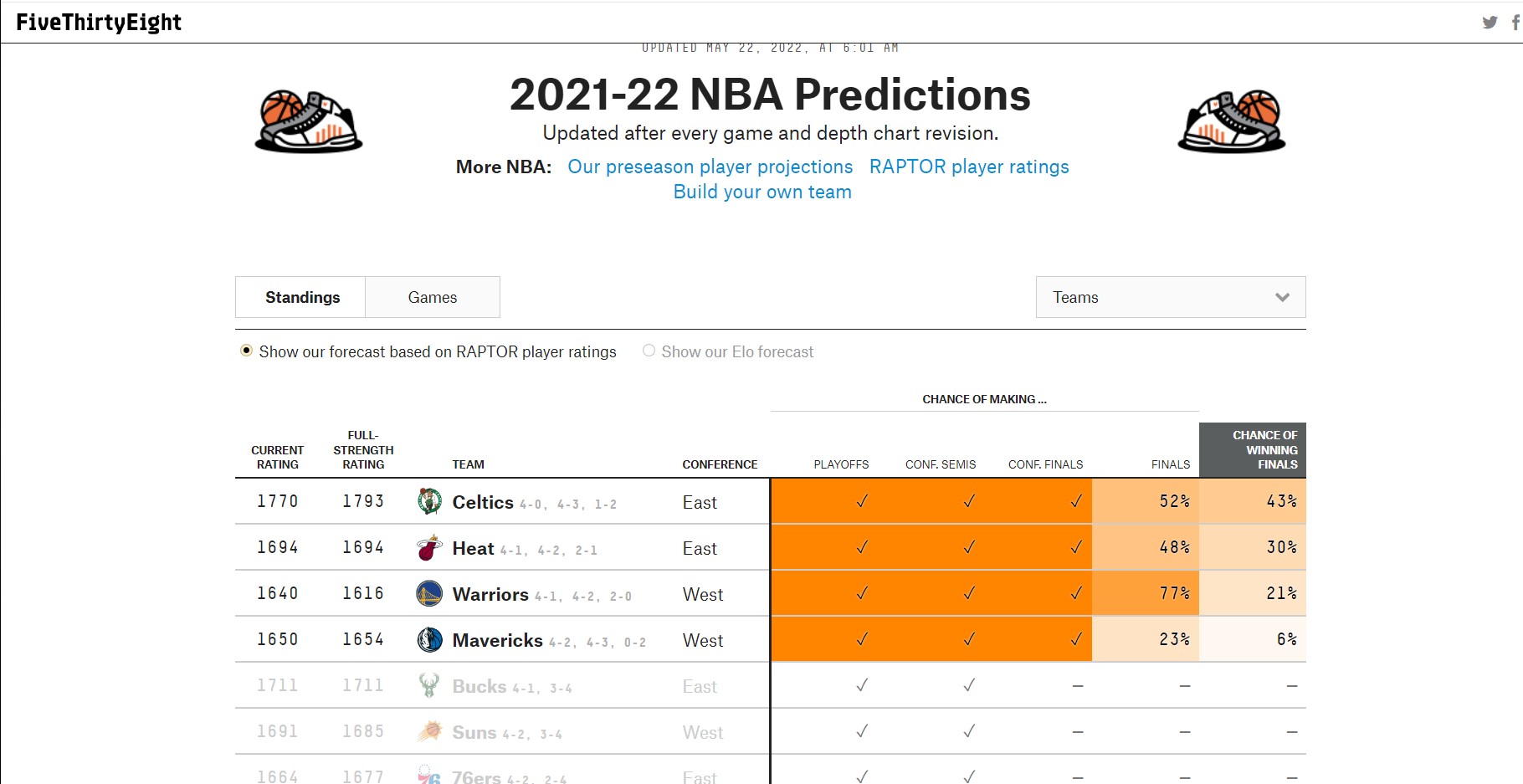


Рисунок 2 – Пример отображения прогноза с сайта project.fivethirtyeight.com

1.2 Общие сведения

В данном разделе будут представлены общие сведения о проекте и его свойствах.

1.2.1 Полное наименование системы и её условное обозначение

Математическая модель для прогнозирования статистических показателей игроков НБА.

Условное обозначение: Система прогнозирования статистики.

1.2.2 Наименование разработчика системы и реквизиты заказчика

Заказчик – Lakers

Разработчик – студент группы ИВТб 3301-01-00 Кудяшев Ярослав Юрьевич

1.2.3 Основания для создания системы

Учебный план по дисциплине комплекс знаний бакалавра в области программного и аппаратного обеспечения вычислительной техники.

1.2.4 Плановые сроки начала и окончания работы по созданию системы.

- Начало работ по созданию системы – февраль 2022 г.

- Окончание работ по созданию системы – май 2022 г.

1.2.5 Источник финансирования работ по созданию системы

Собственные средства разработчика

1.3 Назначение и цели создания системы

В данном разделе будут описаны основные пути применения разработанной системы и цели, для которых она была создана.

1.3.1 Назначение системы

Математическая модель для прогнозирования статистики игроков предназначена для облегчения работы аналитиков и аргументированного подхода при прогнозировании результатов игроков.

1.3.2 Цели создания системы

Целью создания модели является:

• Облегчение работы аналитиков при прогнозировании статистики игроков НБА;

• Представление в удобной форме статистики игроков за все время существования НБА;

• Удобное представление данных для манипуляций над ними.

1.4 Характеристики объекта автоматизации

1.4.1 Краткие сведения об объекте автоматизации

Объектом автоматизации является мужская профессиональная баскетбольная лига Северной Америки, в частности, США и Канады. Входит в четверку главных профессиональных спортивных лиг Северной Америки, наряду с НХЛ, МЛБ и НФЛ.

1.4.2 Сведения об эксплуатации объекта автоматизации и характеристики окружающей среды

• Регулярный сезон в национальной баскетбольной ассоциации начинается с середины октября и заканчивается в середине весны, затем начинается плей-офф;

• Статистика сезонов будет учитываться при выявлении похожих тенденций развития уровня игры у игроков и дальнейшей опоре на эти сведения для прогноза статистики;

• НБА состоит из 30 команд, каждая из которых имеет свою статистику и своих игроков;

• Анализ игроков прошлых лет производится двумя разными подходами: для новичком и обычных игроков лиги.

1.5 Выводы по обзору предметной области

Просмотрев аналоги, выделив цели создания системы и характеристики объекта автоматизации можно прийти к определенным выводам по данной части; проект по своему алгоритму является уникальным и может стать можным инструментом в руках умелово аналитика.

2 Проектирование программного продукта

В данном разделе представлен процесс проектирования основной части проекта и алгоритмы, которые были разработаны для прогнозирования результатов игроков.

2.1 Исследование предметной области

Главной целью данного проекта является предсказание того, как игроки НБА будут вести себя в следующем году; для некоторых из них следующий год станет дебютным: важно показать и проявить себя.

2.1.1 Закономерность развития карьеры игроков

Человек, разбирающийся в НБА и решивший сделать подобный проект, должен понимать, что баскетбол, как и любой вид спорта, имеет свои закономерности, правила и тенденции развития. Лига «не стоит на месте»: вводятся новые трофеи и правила, происходят ребрендинги команд, смена владельцев. И, конечно же, главная составляющая НБА – ее игроки. Каждый год в лигу приходят сотни новичков с порога колледжей, готовых показывать себя во всей красе перед многомилионной аудиторией и доказывать, что именно они - будущее этой лиги. Кто-то проводит свой лучший сезон в карьере А для кото-то все складывается не лучшим образом и его карьера постепенно подходит к концу. Тем не менее практически любой игрок НБА рано или поздно претерпевает подобные ситуации. Даже такой великий игрок, как Майкл Джордан, не может поддерживать звездный уровень игры всю карьеру. На рисунке 3 приведена гистограмма, демонстрирующая среднее количество набиремых им очком за время, проведенное в лиге. Подобная тенденция проглядывается у многих звездных игроков и касается не только среднего количества очков за игру, но и других статистических показателей. Таким примером может послужить среднее количество подборов за игру в карьере Тима Данкана, выступающего на позиции тяжелого форварда, представленное на рисунке 4. Гистограммы реализованы при помощи matplotlib.

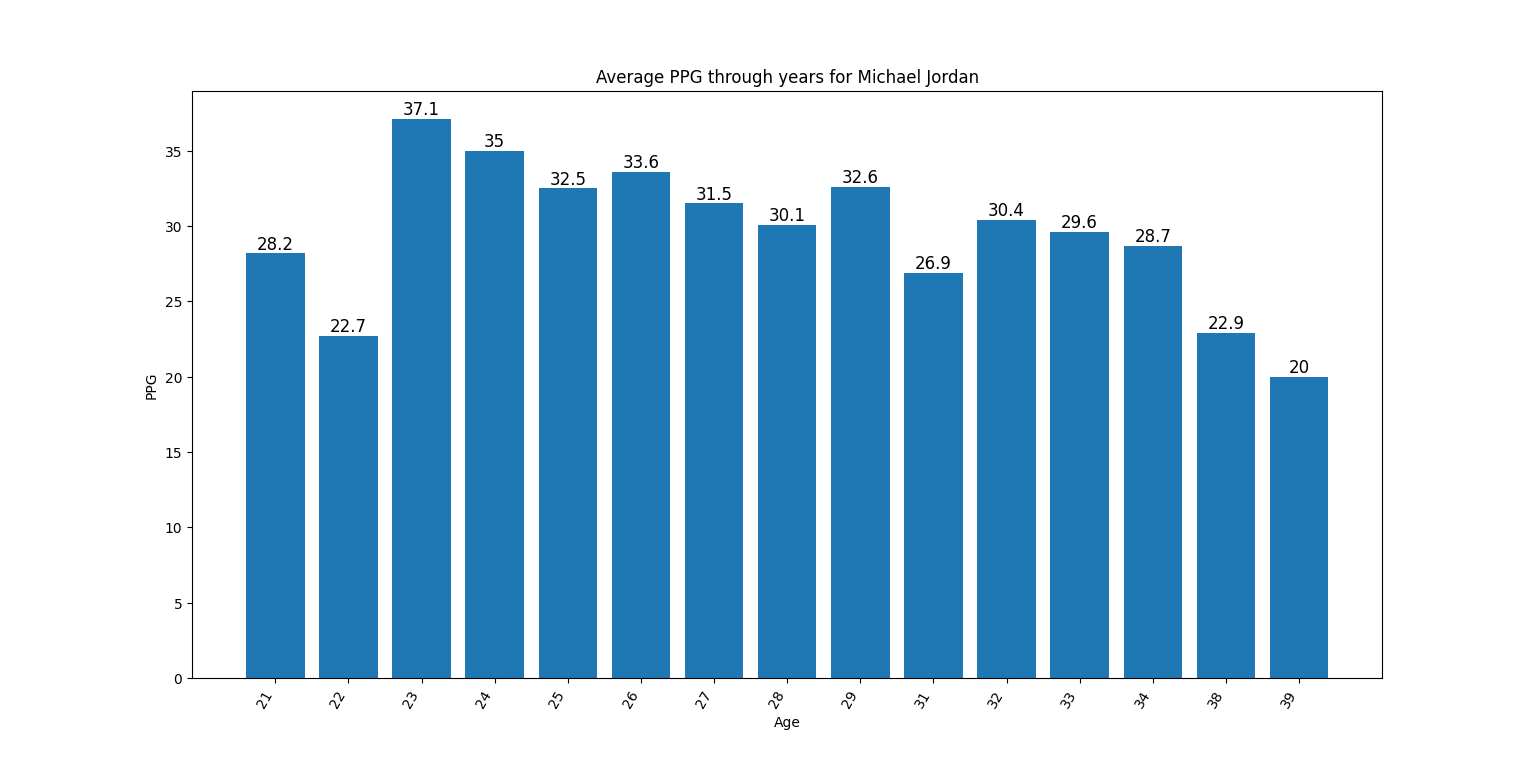


Рисунок 3 – PPG за карьеру для Майкла Джордана

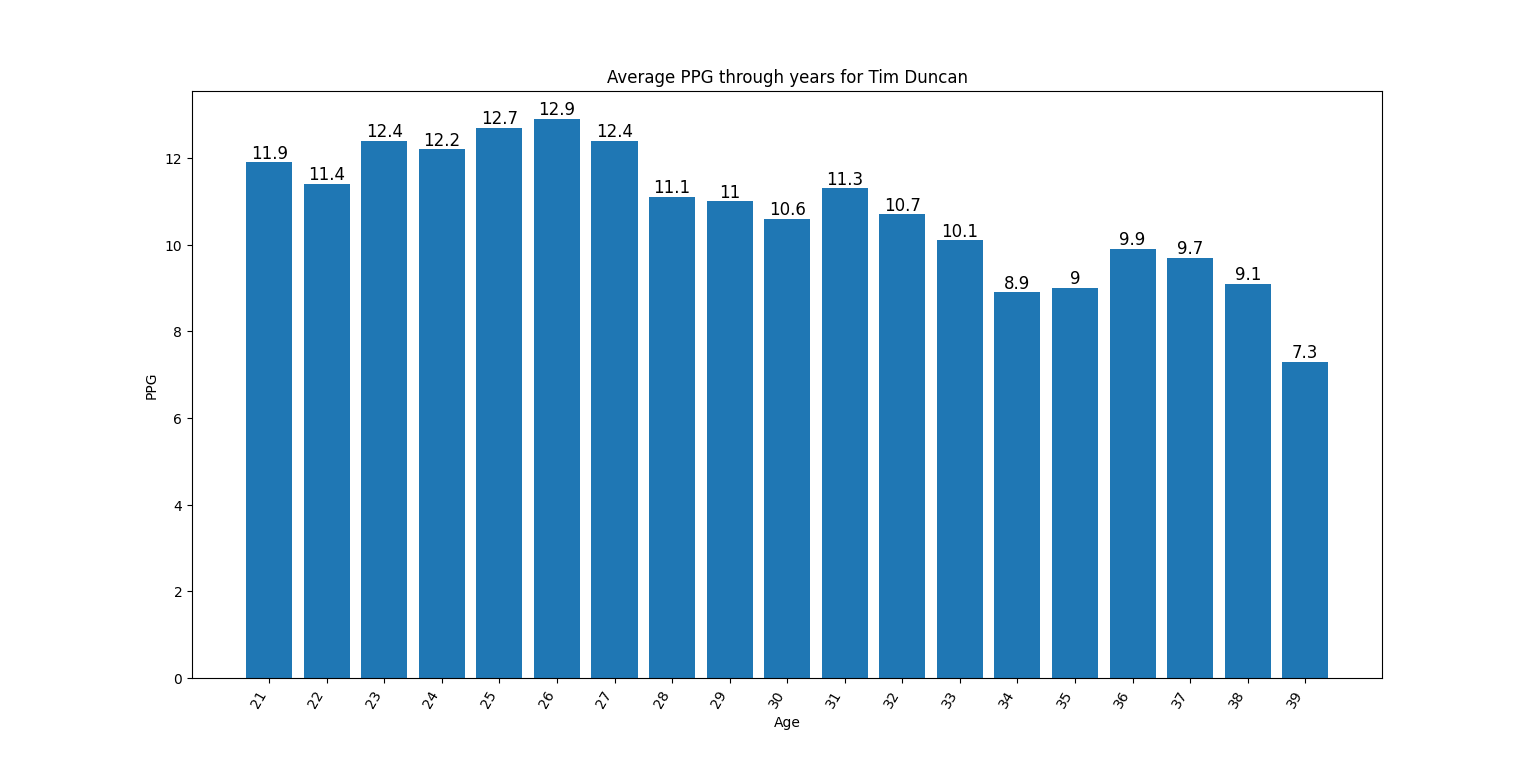


Рисунок 4 – RPG за карьеру Тима Данкана

2.1.2 Динамичный рост показателей игрока

Ещё одним неоспоримым фактом является то, что всегда будут игроки, готовые проявить себя тогда, когда никто от них не ждет никаких феноменальных результатов. В последние годы в лиге наблюдается тенденция роста звёздных игроков. Интересно, что они вырастают из игроков, которые раньше были ролевыми и не претендовали ни на какие награды. Отличными примером служит Джимми Батлер и его карьерный рост: от ролевого игрока, которому с трудом могли доверить мяч, до игрока звёздного уровня. На рисунке 5 можно проследить рост его среднего количества очком в среднем за карьеру.

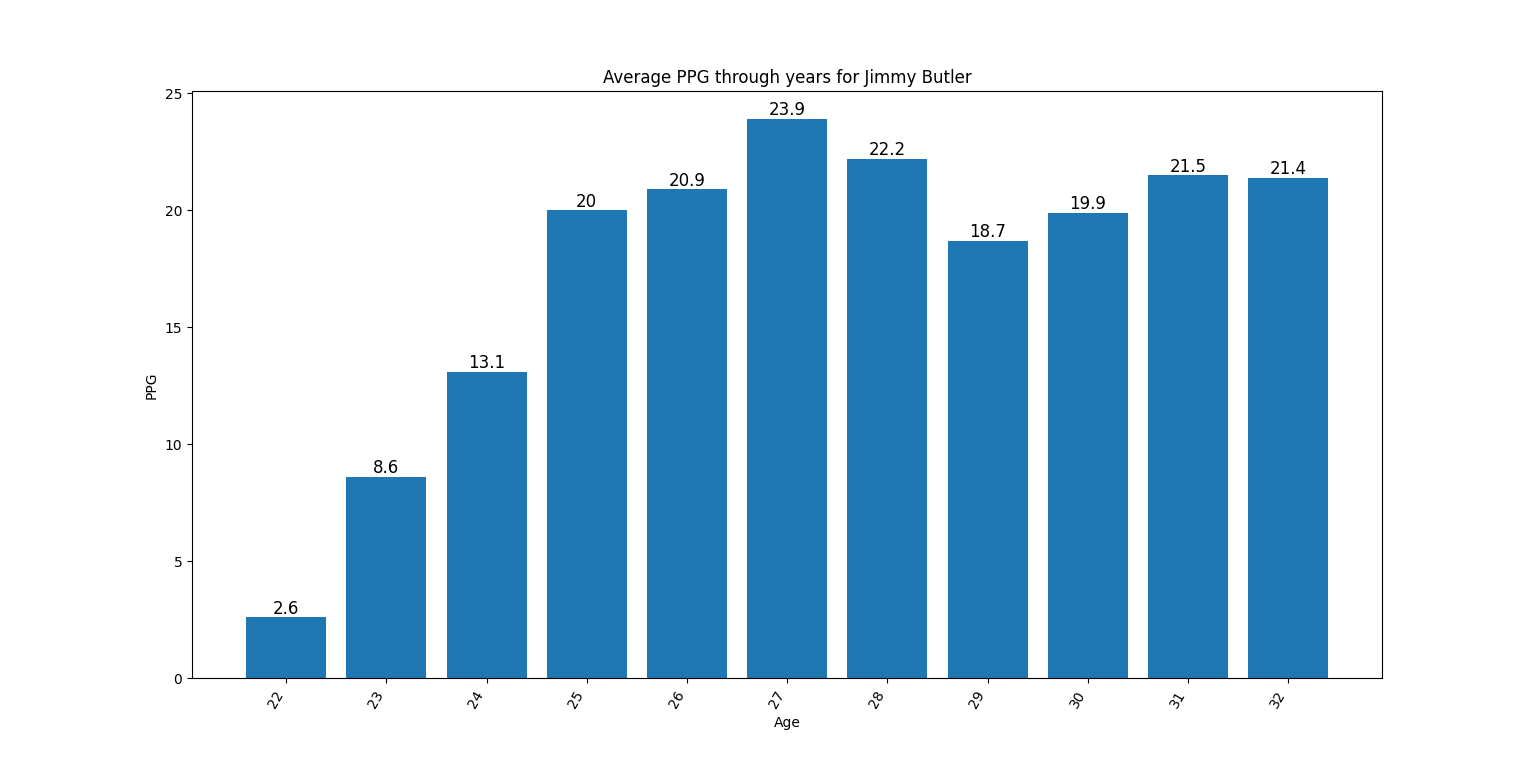


Рисунок 5 – PPG за карьеру для Джимми Батлера

2.1.3 Влияние стиля игры на показатели игроков

Рассматривая статистику игроков НБА, неловко ловишь себя на мысли: проглядывается ли тенденция роста показателей, за исключением PPG. Бесспорно, можно сказать, что - да, но не так значительно, как PPG. Это в большей степени обусловлено тем, что такие показатели, как RPG, BPG и SPG напрямую зависят от физических данных игроков и вряд ли они будуг сильно подвержены изменчивости.

Как уже было сказано ранее, баскетбол развивается и не стоит на месте. Ярким примером и одним из наиболее популярных трендов последних лет является плавное изменение стиля игры команд и игроков: с привычного 2-очкового баскетбола, который господствовал на протяжении многих десятилетий, команды переходят к 3-очковому стилю игры. При помощи диаграмм незакрытой области, приведённых на рисунках 6 и 7, можно в явном виде увидеть разницу в выброшенных 2-очковых и 3-очковых бросках за годы первого десятилетия 21 века в сравнении со вторым. Всё это напрямую связано со статистикой игроков. Так, постепенный переход к 3-очковому баскетболу, привел к увеличению количества набираемых в среднем очков за игру для игроков. Немного статистики: в 2010 году в лиге было 7 игроков, набирающих более 20 очков за игру, а в 2021 их число возросло до 27 игроков. Именно такие тенденции роста в различных показателях должна отслеживать математическая модель.

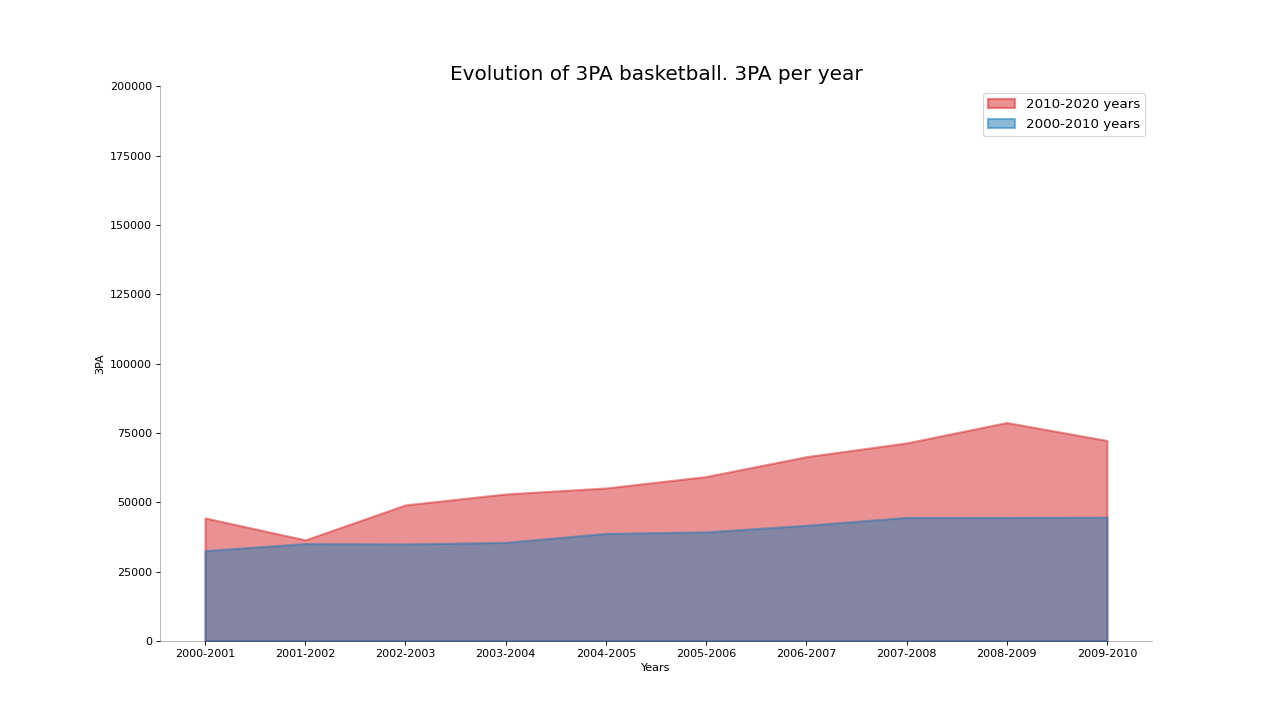


Рисунок 6 – Сравнение 3PA для первого и второго десятилетия 21 века

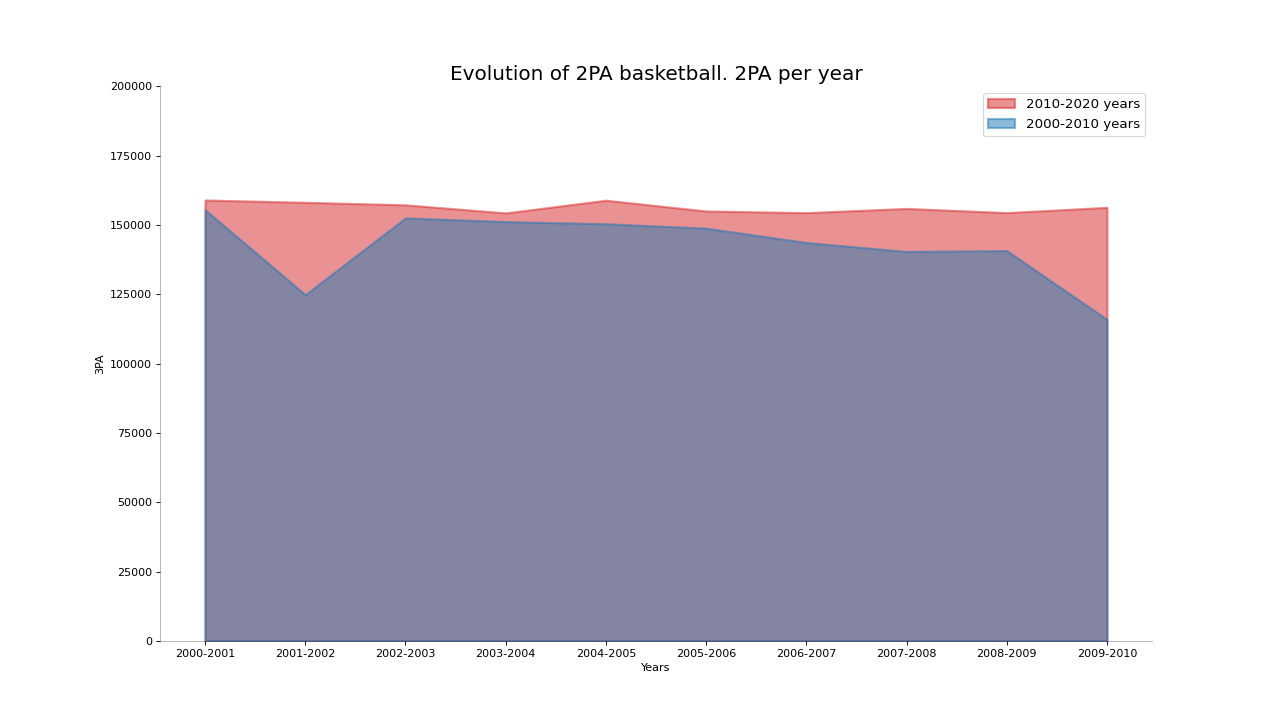


Рисунок 7 – Сравнение 2PA для первого и второго десятилетия 21 века

2.1.4 Влияние возраста на показатели игрока

Одним из наиболее важных факторов успешной игры является возраст игрока. От возраста зависит не только количество набираемых игроков очков, но и в том числе, его роль в команде. У каждого игрока есть свой возрастной пик, после прохождения которого, его статистика, как правило, плавно уменьшается. При помощи boxplot графика удалось визуализировать средний показатель очков, набираемых игроками в различный период их карьеры. Так на рисунках 8 и 9 приведены графики boxplot, визуализирующие эти показатели. Благодаря ним становится хорошо понятно: сколько в среднем очков за матч набирали игроки определенного возраста в 2009-2010 и 2020-2021 сезонах. Видно, что количество игроков, набирющих более 20 очков выше в 2020-2021 сезоне, и средний показатель PPG в среднем выше. Этот график очень полезен, можно изначально предположить: сколько игроков и кто из них будет набирать более определенного числа очков в среднем в следующем сезоне. Кроме того, можно в явном виде увидеть, сколько игроков каждого возраста присутствует в лиге.

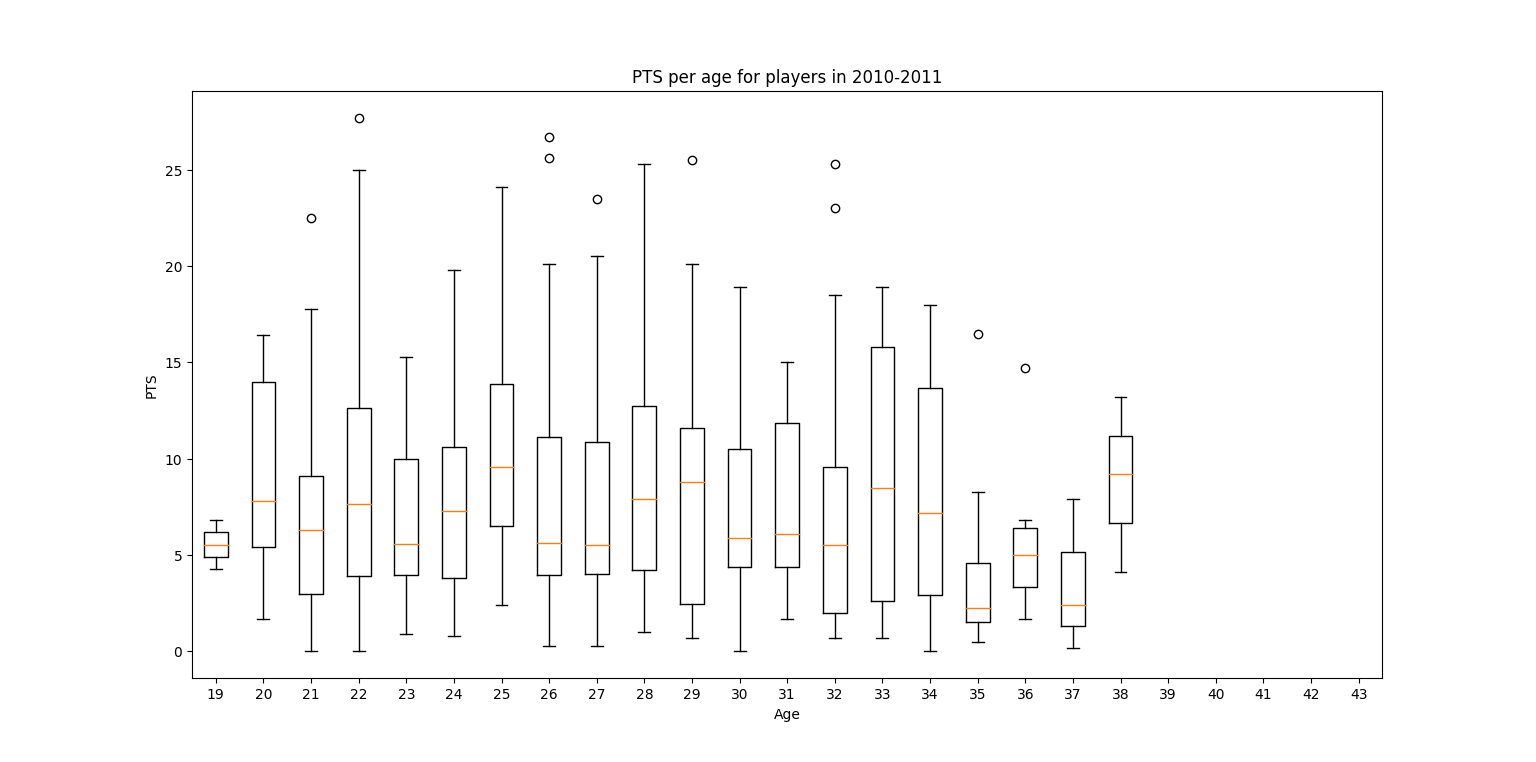


Рисунок 8 – Среднее количество набираемых очков для игроков каждого возраста за 2009-2010 сезон НБА

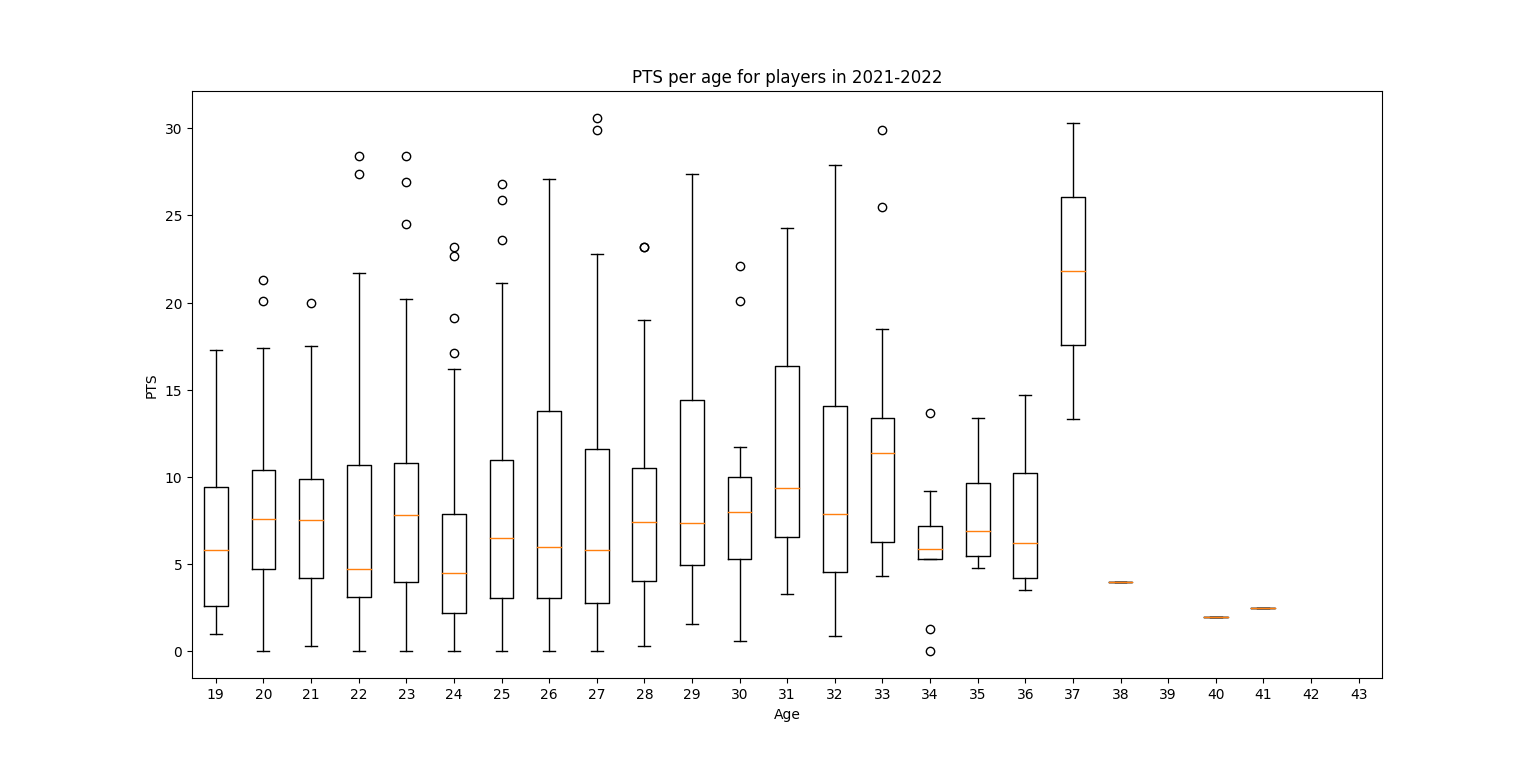


Рисунок 9 – Среднее количество набираемых очков для каждого возраста за 2020-2021 сезон НБА

2.1.5 Анализ взаимосвязи показателей

После проведения небольших экспериментов и анализов с фиктивными переменными, было принято решение перейти к модели корреляционных матриц, которые помогут понять и увидеть взаимосвязи между входными и целевыми переменными. В качестве примера была взята карьера суперзвезды нынешнего НБА Кевина Дюранта. На рисунке 10 приведена кореляционная матрица. Как видно на диаграмме, кажется, что существует сильная корреляция между подборами в атаке и возрастом игрока.

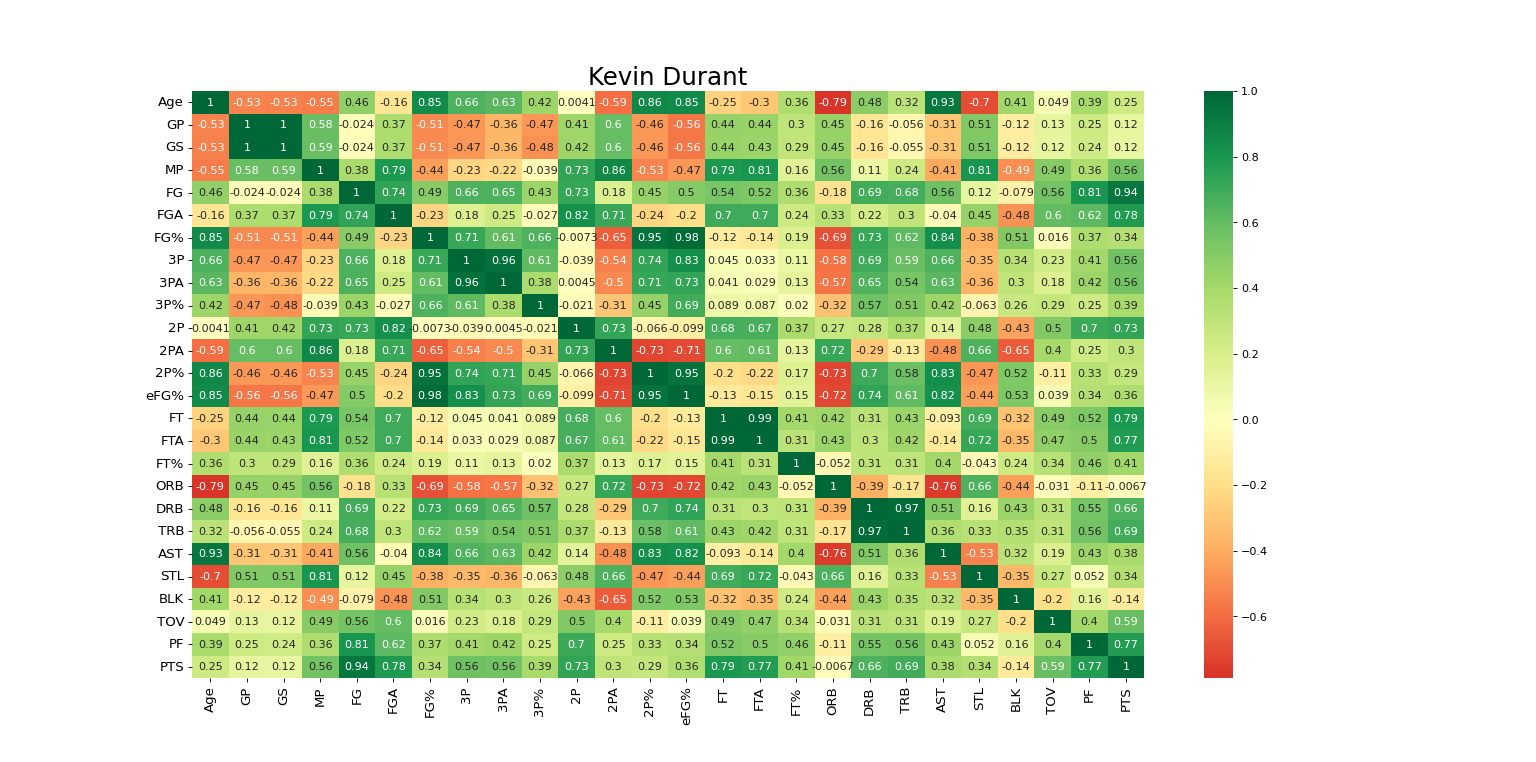


Рисунок 10 – Корреляционная матрица за карьеру Кевина Дюранта

2.1.6 Позиционный аспект показателей игроков

Немаловажным фактором при прогнозировании статистики является позиция игрока. В баскетболе выделяют 5 основных позиций: разыгрывающий, защитник, лёгкий форвард, тяжелый форвард и центровой. Каждая из позиций обладает своими свойствами и зависимостями. Так разыгрывающие и защитник обычно менее габаритны и их основной задачей, помимо набора очков, являются перехваты мяча у соперника и передачи игрокам в своей команде. Основная задача центровых – собирать подборы и делать успешные блокшоты за игру. Форварды являются универсальными игроками на площадке: они могут успешно справляться с основными задачами как центровых, так и защитников. На рисунке 11 приведен пример точечного графика, отображающего среднее количество подборов за игру в сезоне 2021-2022 для всех позиций. Хорошо заметны различия в цифрах: центровые занимают доминирующие позиции по этому показателю. В качестве ещё одного примера приведён рисунок 12, на котором в виде точечного графика того же сезона отображена зависимость проведенного на площадке времени и среднего количества передач за игру для каждой из позиции. Как и следовало ожидать, защитники занимают лидирующие позиции по данному показателю, в то время как центровые довольно слабо показывают себя по передачам. Хочется отмететить, наблюдая за активностью разных позиций за последние годы игры, удалось понять, что большинство игроков и лиге постепенно становятся универсальными: центровые могут спокойно заменить форвардов, а форварды, в свою очередь, разыгрывающих и наоборот.

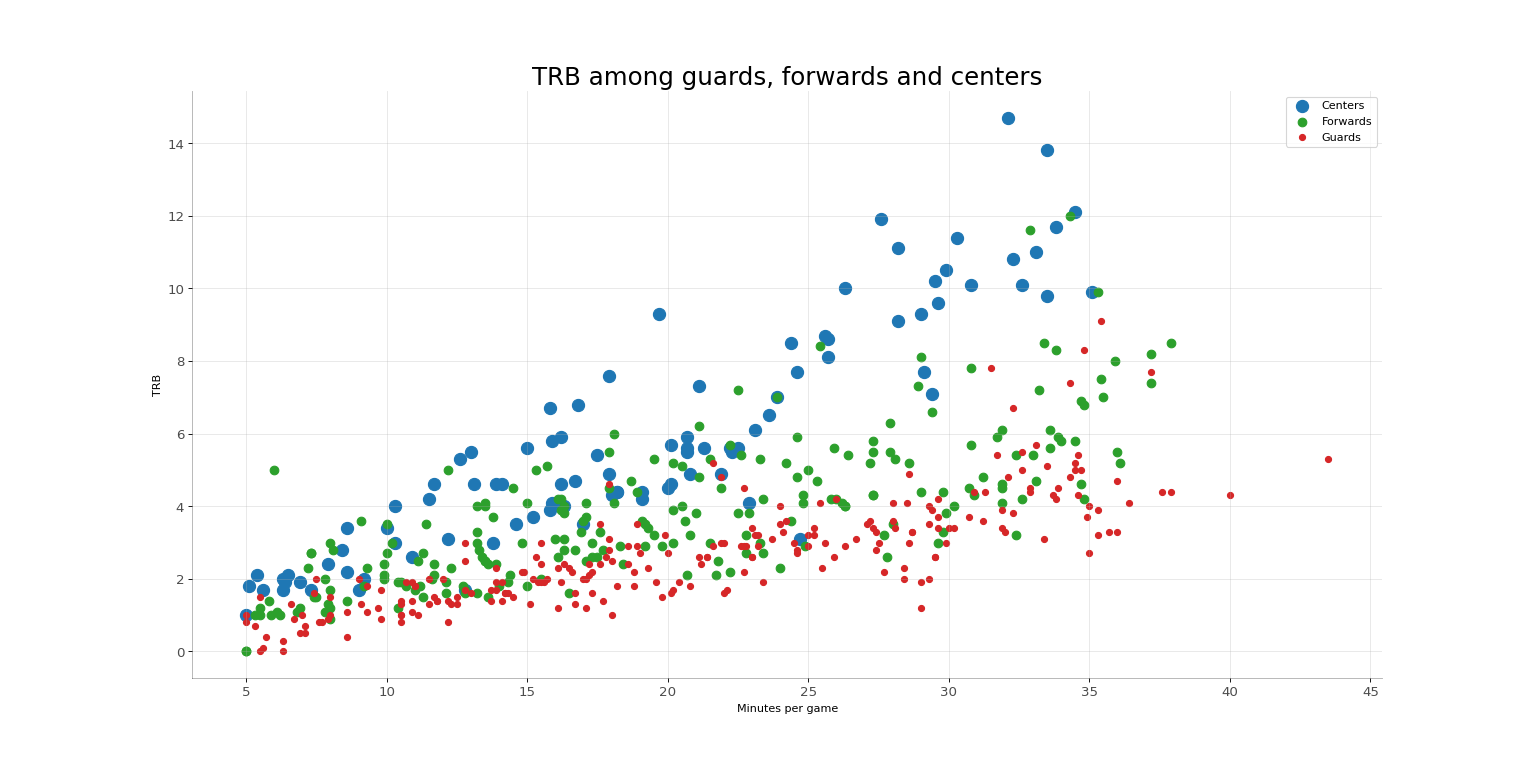


Рисунок 11 – Среднее количество подборов за игру для всех игровых позиций

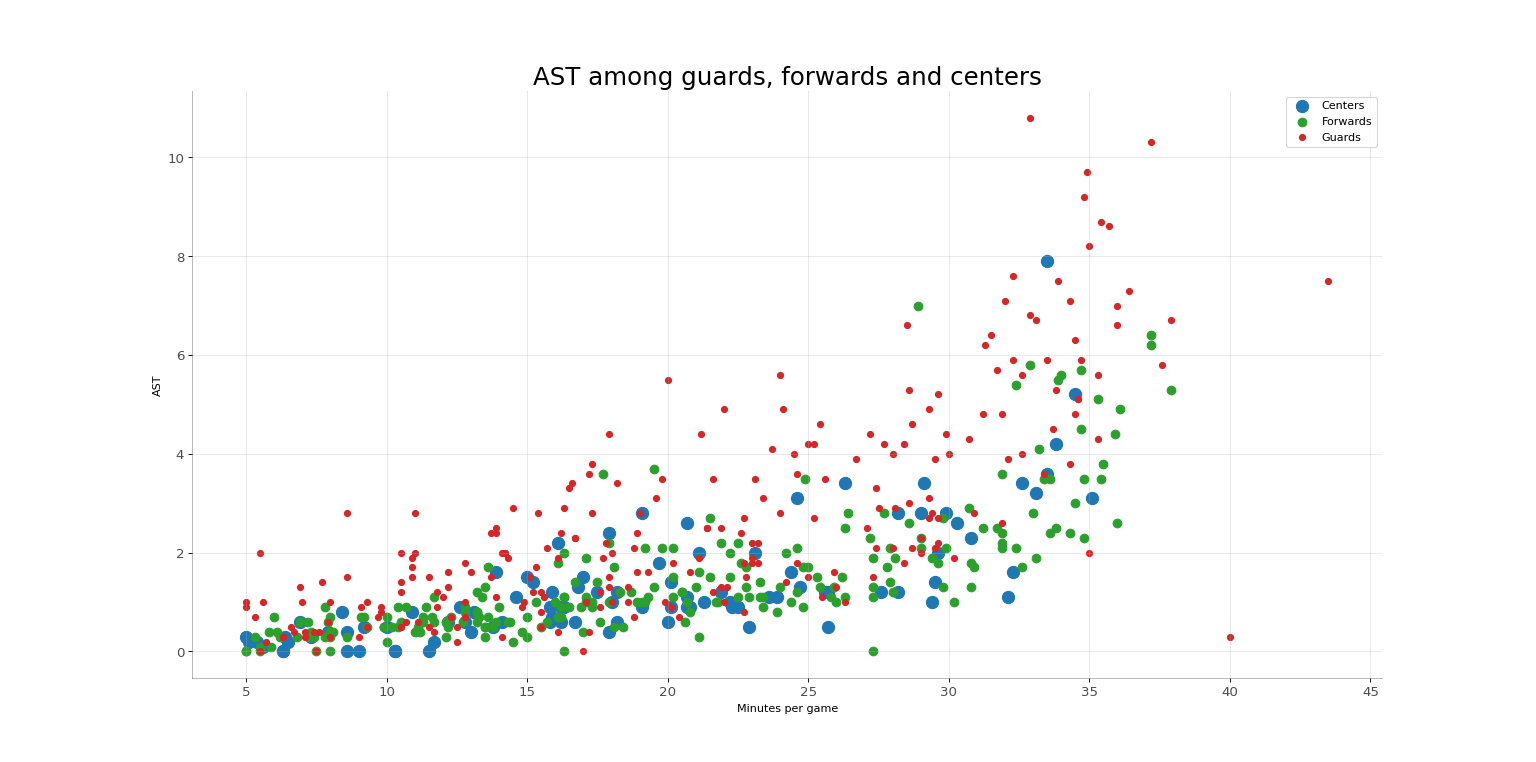


Рисунок 12 – Среднее количество передач за игру для всех позиций

Помимо аспектов игры, приведенных выше, были рассмотрено ещё несколько тенденций развития лиги на данный момент. Стоит всерьез задуматься о вкладе игроков из Европы в лигу за последние годы. Об их большом влиянии говорит хотя бы то, что за последние 5 лет награду MVP завоевало больше иностранных игроков, чем за все годы существования лиги в сумме. Помимо этого интересного факта также нельзя не отметить наличие претенедентов на одну позицию в команде. Чем больше игроков одной позиции в команде, тем меньше минут будет получать каждый из них.

Рассмотрев основные зависимости и тенденции развития современного баскетбола в НБА можно плавно переходить к выбору математической модели и её построению.

2.2 Выбор модели

В ходе эксперимента было запущено несколько наиболее популярных и подходящих алгоритмов, используемых при построении математических моделей.

2.2.1 Линейная регрессия

В качестве первой модели для анализа была рассмотрена модель линейной регрессии. Она необходима для установления базовых критериев. Этот алгоритм работает, пытаясь нарисовать прямую линию через все точки, полученные в ходе анализа. Уравнение для этой линии рассчитывается по методу наименьших квадратов, где целью является минимизация суммы квадратов ошибок. Пример отображения линейной регрессии и её рассчета приведен на рисунке 13.

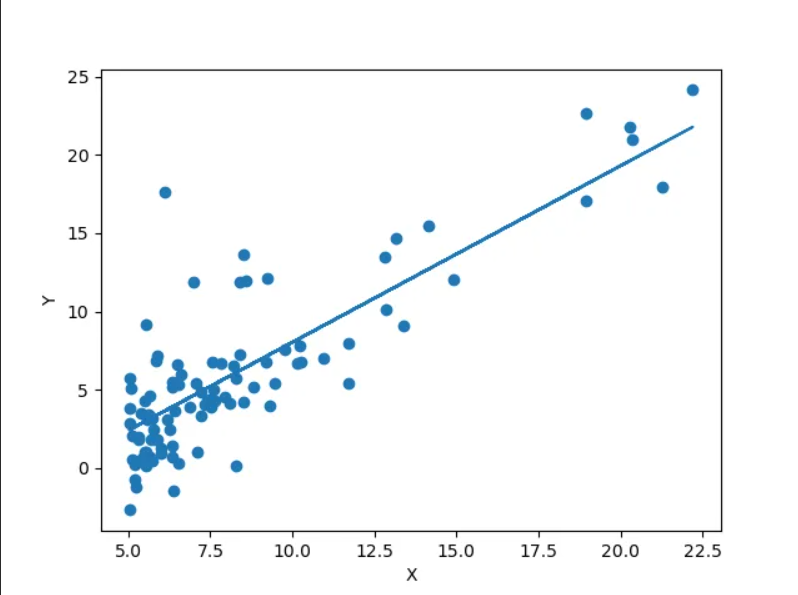


Рисунок 13 – Пример отображения линейной регрессии

2.2.2 Random Forest

В качестве первого основного алгоритма было принято решение использовать регрессов Random Forest (Случайный Лес). Этот алгоритм работает путём случайного извлечения различных подмножеств из исходного обучающего набора данных посредством процесса выбора данных, которые находятся на пересечении N случайных входных объектов и М случайных столбцов. Затем алгоритм базового дерева решений запускается на всех этих различных подмножествах. После того, как все деревья созданы, прогноз элемента в наборе тестов рассчитывается путём взятия среднего значения результатов, полученных при прохождении входных объектов через каждое дерево решений. Пример отображения данных и структуры в Random Forest представлен на рисунке 14.

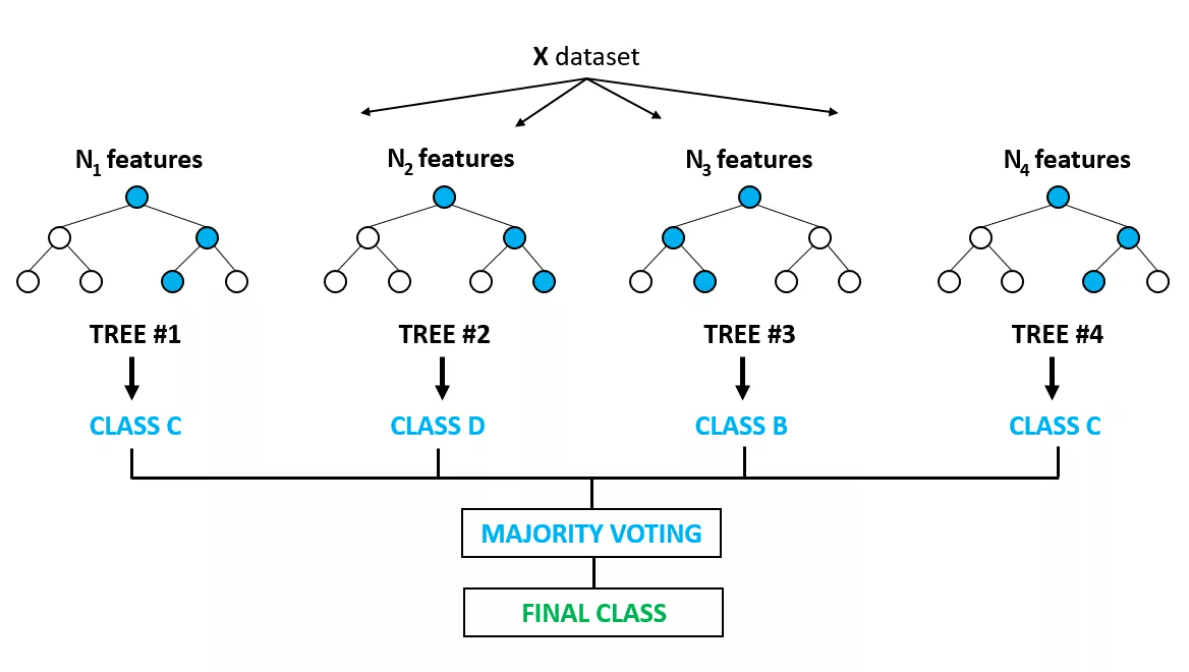


Рисунок 14 – Пример отображения Random Forest

2.2.3 Extra Trees

Следующим алгоритмом, на котором было решено протестировать алгоритм был регрессор Extra Trees, который по своим действиям и представлению напоминает Random Forest. Как и в случае с Random Forest, Extra Trees запускает алгоритм дерева решений для различных случайных подмножеств, сгенерированных из обучающего набора данных, для создания прогнозов. Самым большим различием между этими двумя методами является то, как дерево решений запускается на подмножествах. Алгоритм Random Forest использует традиционный подход дерева решений, в котором признак и значение, используемые в точке разделения, определяются на основе информации, полученной на этом этапе. Алгоритм Extra Trees же использует немного другой принцип: функция и значение в дереве решений, используемые в точке разделения, выбираются случаным способом, о чем и говорит название метода. Пример отображения структуры Extra Trees приведен на рисунке 15.

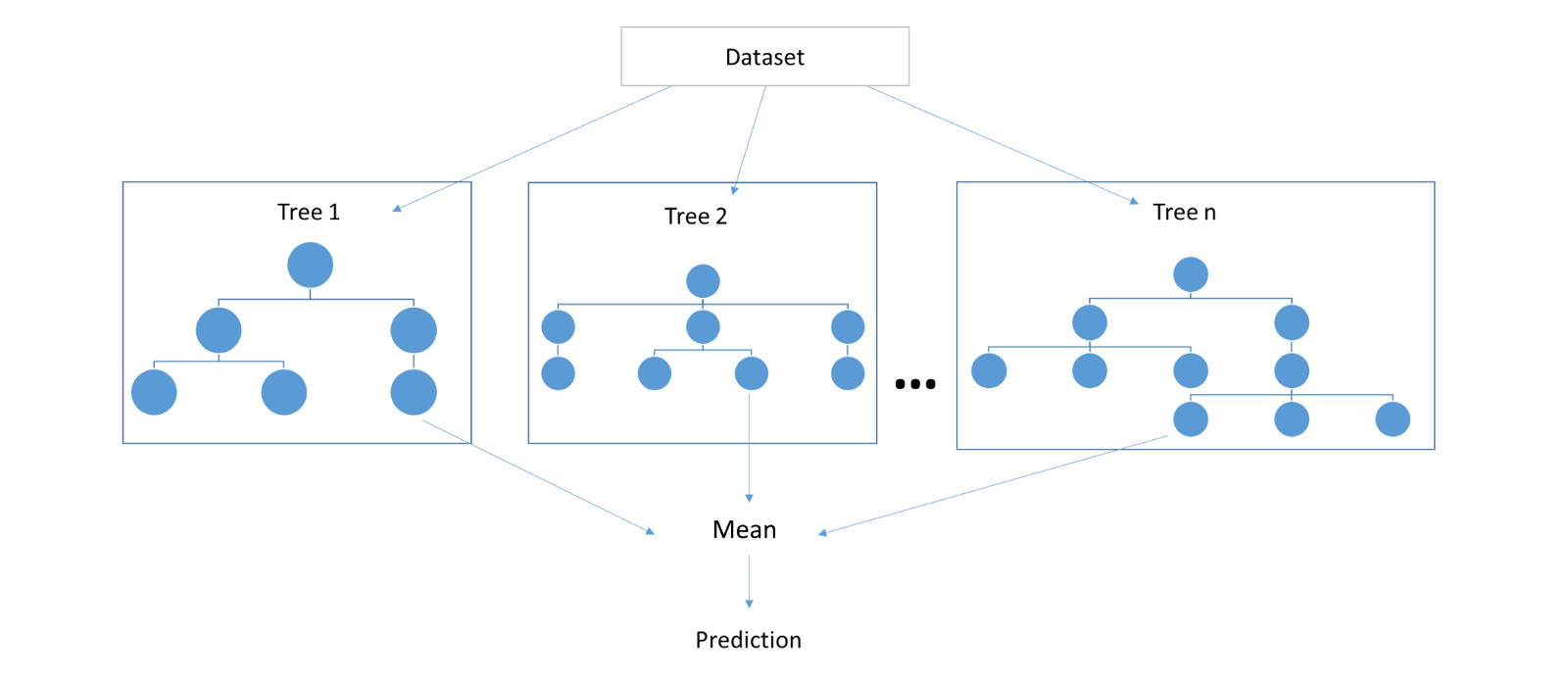


Рисунок 15 – Пример отображения Extra Trees

2.2.4 XGBoost

Последним методом для запуска алгоритма был выбран XGBoost. Данный алгоритм в большей степени отличается от всех других, описанных выше. Его суть заключается в том, что алгоритм использует метод, известный как повышение градиента, для создания мощной и точной модели. Повышение градиента работает за счет рекурсивного построения различных моделей друг над другом, чтобы минимизировать процент возникновения ошибки. Поскольку вся цель XGBoost заключается в том, чтобы минимизировать процент возникновения ошибки, возникшей в обучаемом наборе, то этот алгоритм иногда имеют тенденцию перегружать данные и выполнять подпараллерирование на тестовом наборе. Пример отображения структуры XGBoost и даных приведена на рисунке 16.

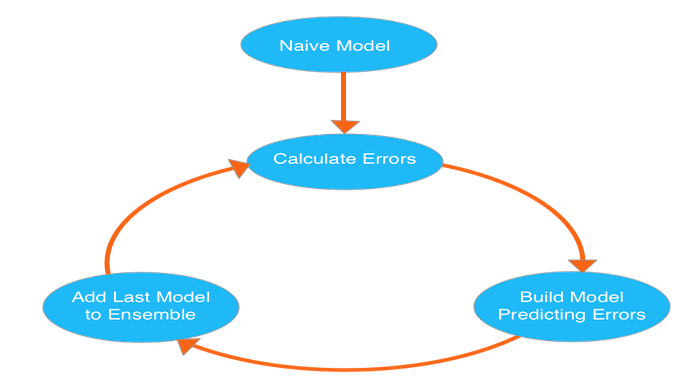


Рисунок 16 – Пример отображения XGBoost

2.3 Подготовка данных

Одним из самых главных этапов создания математической модели является подготовка данных для анализа и представление их в необходимом виде, без лишних столбцов и значений.

Первоначально планировался анализ данных лишь последних двух десятилетий, одноко после разбора предметной области было принято новое решение – проанализировать данные за все время существования НБА, чтобы проследить все тенденции и зависимости возникавшие по ходу всех эпох существования НБА. Для реализации данной задумки сначала необходимо было найти базы данных игроков с их данными и результатами всех сезонов НБА. Базу данных за сезоны удалось найти в открытом источнике баскетбольной аналитики: <https://www.basketball-reference.com/leagues/>. Таблицы с данными были скачаны в формате Excel для дальнейшего анализа и перехода ко второму этапу обработки данных, всего 76 таблиц для каждого сезона НБА.

На втором этапе подготовки данных стояла задача приведения их к необходимому виду, для дальнейшего анализа: убрать ненужные показатели, статистику и идентификаторы. Для приведения таблиц к каноничному виду было принято решение использовать язык програмирования Python и библиотеку для работы с таблицами Excel под названием openpyxl. Исходный код программы приведен в приложении А. После написания программы и ее применения удалось получить одну общую таблицу Excel, которая содержала данные для каждого игрока за все сезоны, проведнные им в лиге. Каждый игрок представляет собой страницу в документе Excel. Это было сделано с целью упрощения дальнейшего обращения к данным в таблице и уменьшению времени для их обработки. Помимо этого, основные физические характеристики игроков хранятся в другом файле, который будет необходим в дальнейшем для построения модели поведения игрока. Пример получившейся структуры данных приведен на рисунке 17. Кроме того, были добавлены столбцы для позиций игроков. Это было необходимо для замены строкового представления численным.

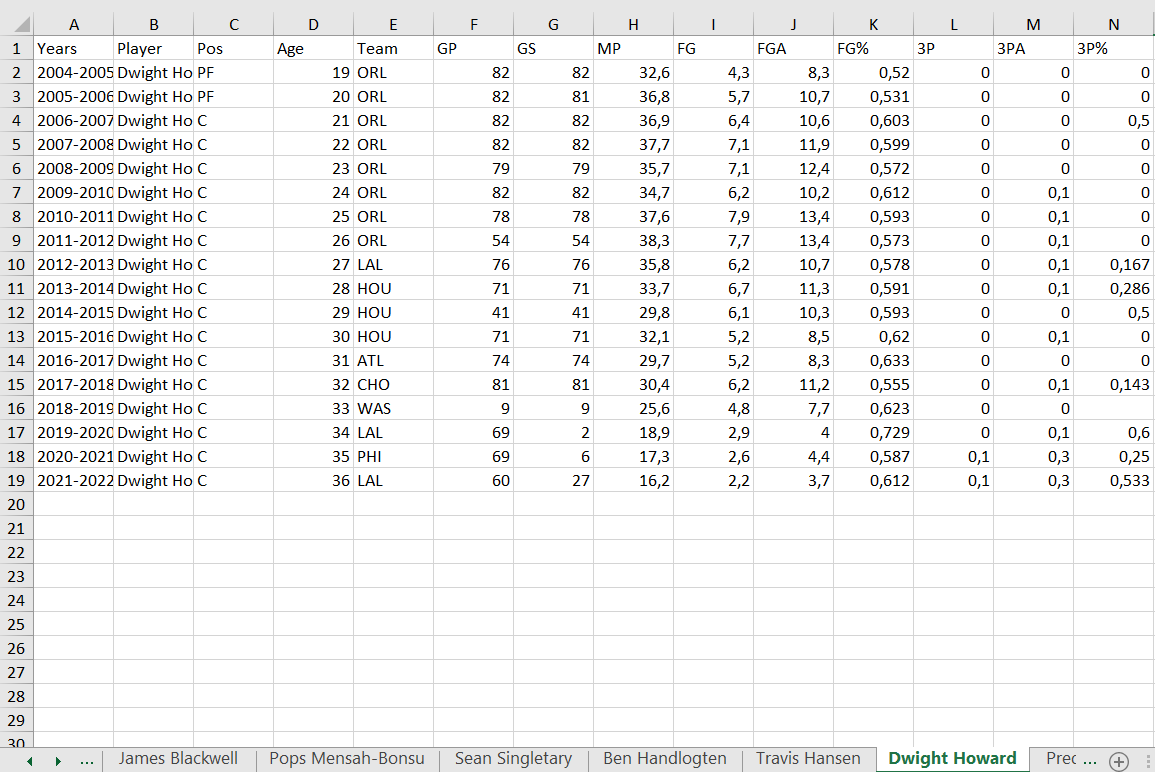


Рисунок 17 – Представление данных после первоначальной обработки

Была проделана работа по удалению нулевых характеристик, которые не относятся к какого-то рода статистике. Были скорректированы данные, имеющие вещественный тип и укорочены до 2-х цифр после запятой. Команды, которые так или иначе больше не существуют в НБА, были заменены на другие, которые пришли им на замену. Так, к примеру, команда Sietl Supersonics была заменена на Oklahoma City Thunder.

После проведения действий, описанных выше, появилась ещё одна потребность: отобразить все в той же общей таблице Excel заслуги всех игроков, которых они добились за годы, проведенные в НБА. Этого удалось добиться при помощи инструментов библиотеки pandas. Под заслугами подразумеваются:

• Индвивидуальные награды, полученные игроком по ходу сезона (MVP, DPOY, 6MOTY, FMVP, MIP, ROTY)

• Лидерство в сезоне по тому или оному показателя (Scoring leader, assists leader, rebound leader, steals leader, blocks leader)

• Смена команды в рассматриваемом году

• Наличие травмы у игрока перед началом сезона

2.4 Параметризация модели

Переходя к последнему этаму пректирования модели, стоит выделить основные параметры, в зависимости от которых и будет строиться модель. Исходя из утверждений и исследований, приведённых выше, удалось прийти к следующим параметрам:

• Возрастной показатель игрока;

• Позиция игрока;

• Травмированность игрока;

• Уровень игры игрока;

• Наличие звездного партнера в команде;

• Ролевая позиция игрока в ротации;

• Количество 3-очковых бросков за матч;

• Статистические показатели игрока за предыдущие года;

• Проценты попадания с поля на различных участках поля;

• Количество сыгранных сезонов;

• Количество игр в стартовой пятерке.

2.5 Выводы по части проектирования программного продукта

Проанализировав предметную область и различные математические модели удалось выделить необходимые метрики и зависимости необходимые для дальнейшеей реализации проекта. Помимио этого, была проделана наиболее трудоемкая работа: приведение исходных данных к необходимому для построения математической модели виду.

3 Реализация модели

После изучения предметной области и определения всевозможных вариантов построения математической модели, а также проведения тестов, было принято решение взять за основу модель Random Forest. Данная модель показала наилучший показатель точности прогнозирования для статистических показателей игроков 2021-2022 сезона на основе 2020-2021 сезона.

3.1 Устранение ненужных показателей

Приведение данных к каноничному для построения модели виду лишь первый шаг. Для ускорения процесса построения модели и увеличению точности было принято решение убрать наиболее несвязанные параметры. При помощи медода RFE, который рекурсивно рассматривает наборы объектов и на каждой иттерации исключает все более и более ненужный показатель, исходя из конкретного атрибута, удалось сократить количество параметров с 31 до 25. Для наглядности была построена корреляционная матрица для целого сезона 2021-2022. Результаты приведены на рисунке 18. Наиболее низкую корелляционную зависимость показали позиционные показатели.

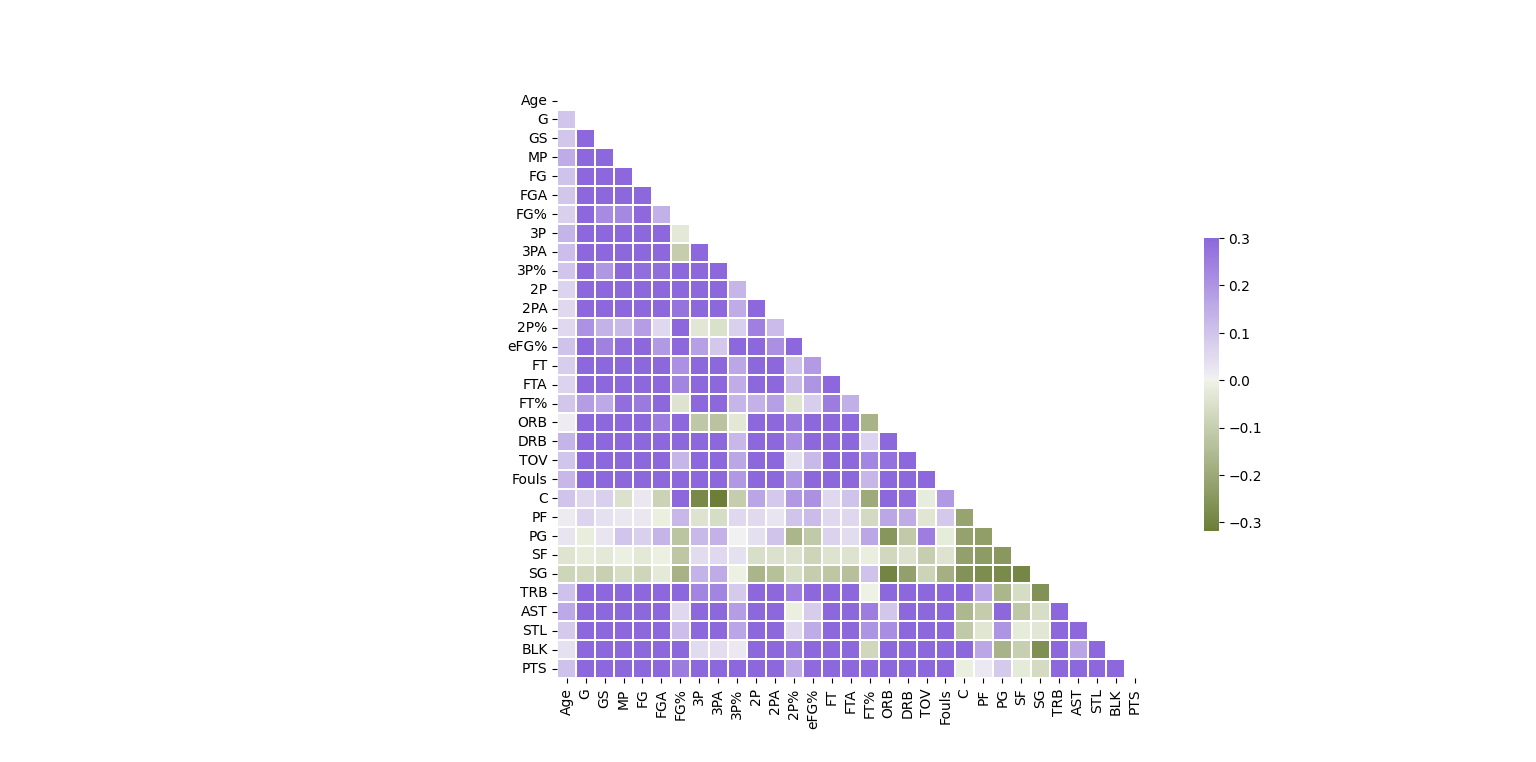


Рисунок 18 – Кореляционная матрица для статистики игроков 2021-2022 сезона

3.2 Представление показаталей в каноничном для математической модели виде

Перед тем как анализировать данные и делать прогнозы любая математическая модель должна преобразовать имеющиеся данные в понятные для неё показатели, а именно – коэффициенты. Пример отображения коэффициентов приведен на рисунке 19. Такого представления удалось добиться при помощи метода StandartScaler().

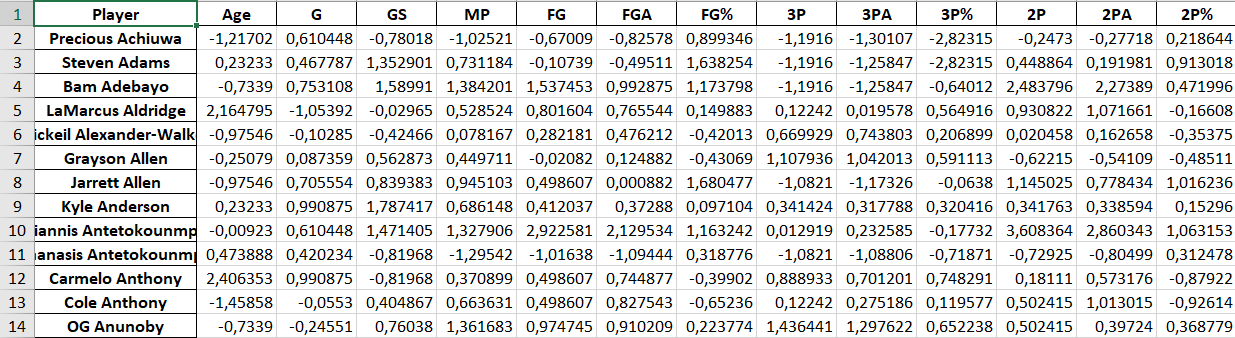


Рисунок 19 – Таблица с коэффициентами

3.3 Прогнозирование результатов

Библиотека sklearn обладает мощными инструментами для работы с данными и их анализа. Так, благодаря методу predict и набору данных для обучения и тестов, можно получить предполагаемый исход того или иного события. В данном случае под событием понимается предполагаемые статистические показатели, которые будет набирать тот или иной игрок в следующем игровом сезоне НБА. Получив заветные коэффициенты, было необходимо правильно переработать их для того, чтобы получить график, отображающий то, насколько точно модель предсказала результаты, и проценты, отвечающие качество прогноза. Данные результаты и графики приведены в разделе тестирования для модели Random Forest. После получения результатов было проведено их преобразование к нормальному виду при помощи метода inverse\_transform. Последний этап – запись результатов со всеми статистическими показателями игроков в общую таблицу Excel. Это было сделано при помощи библиотеки pandas благодаря методу to\_excel. Результат приведен на рисунках 20 и 21.

Без подготовки и ознакомления с предметной областью довольно сложно разобраться в таблице с результатами. Расшифровка обозначений:

• GP – количесвто сыгранных игроком игр в этом сезоне;

• MPG – среднее количество минут, которое игрок проводил на поле за одну игру;

• PPG – среднее количество очков, набираемое игроков за матч;

• RPG – среднее количество подборов, набираемых игроком за матч;

• APG – среднее количество передач, раздаваемое игроком за матч;

• SPG – среднее количество перехватов, которое делает игрок за матч;

• BPG – среднее количество блокшотов, которое делает игрок за матч;

• TOV – среднее количество потерь, делаемое игроком за матч;

• PF – среднее количество фолов, получаемое игроком за матч;

• FGM – среднее количество бросков за игру;

• FGA – среднее количество попыток броска за игру;

• FG% - средний процент попадания с игры;

• 3PM – среднее количество бросков с трёхочковой линии за игру;

• 3PA – среднее количество попыток броска с трёхочковой линии за игру;

• 3P% - средний процент попадания с трёхочковой линии за игру;

• FTM – среднее количество бросков со штрафной линии за игру;

• FTA – среднее количество попыток броска со штрафной линии за игру;

• FT% - средний процент попадания со штрафной линии за игру.

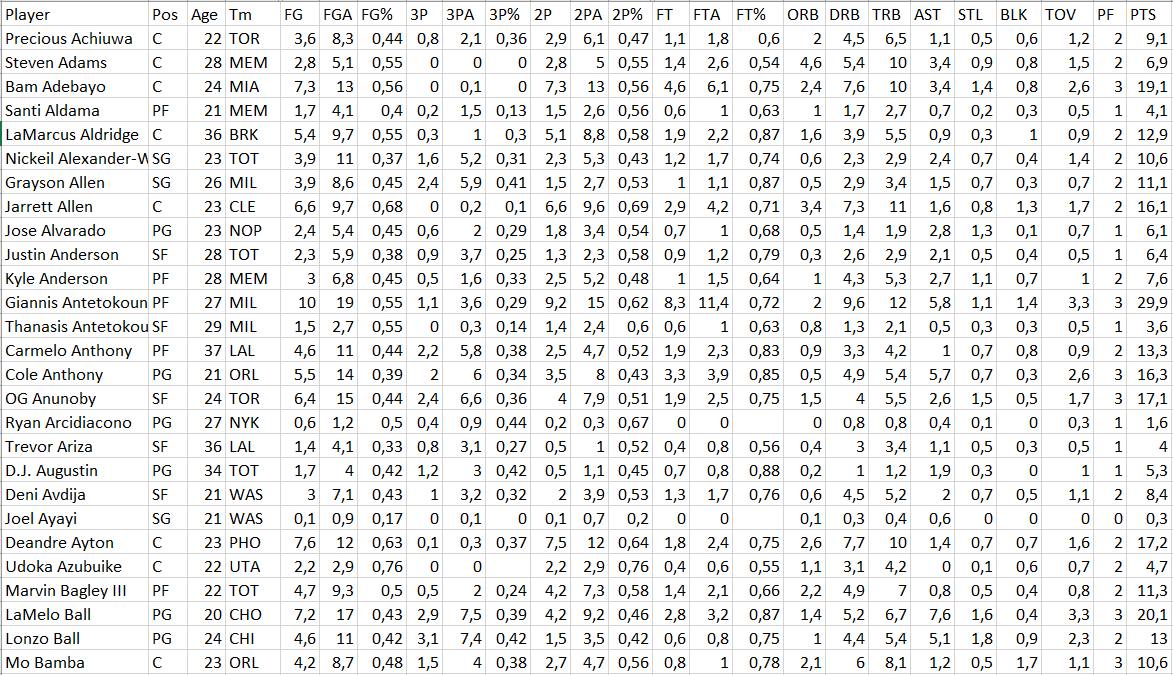


Рисунок 20 – Результаты прогнозирования 1

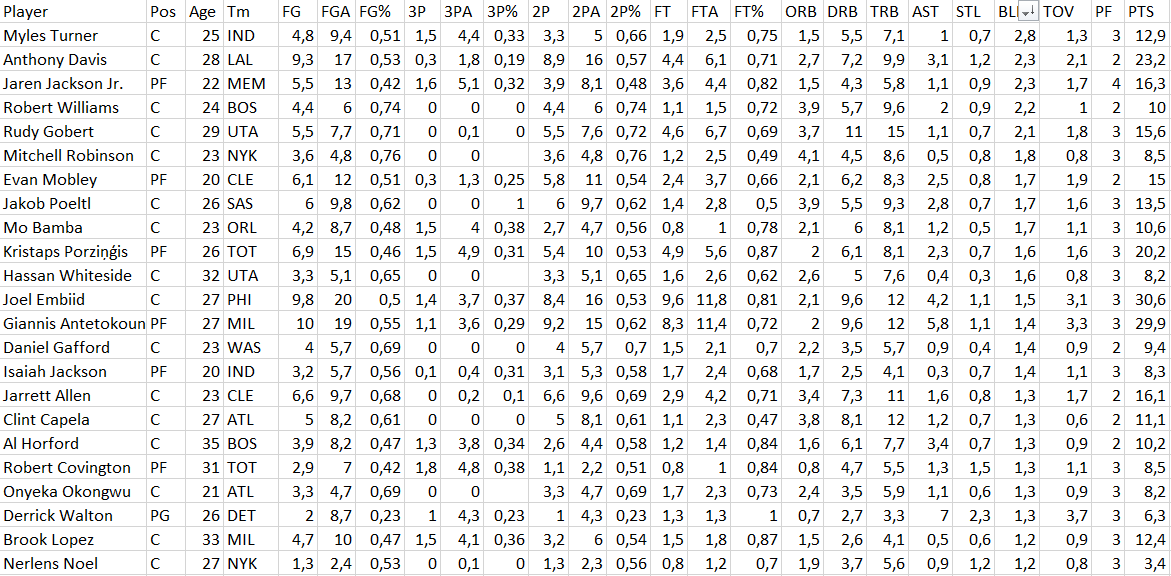


Рисунок 21 – Результаты прогнозирования 2

3.4 Вывод по части реализации программного продукта

В результате проделанной работы удалось получить работоспособную математическую модель, способную довольно точно, исходя из результатов тестирования, предсказывать статистику игроков. Кроме этого, были освоены методы по нормализации исходных данных для модели.

4 Тестирование программной модели

Перед тем как выбирать исходную модель для реализации и прогнозирования статистики игроков, было принято решение построить математические модели несколькими способами. Для проведения тестов были построены 4 базовые модели: линейная модель, модель случайного леса, модель дополнительных деревьев и XGBoost модель. В качестве метрики было принято решение взять показатель среднего количества очков за матч, т.к. это самый динамически меняющийся показатель. Для определения точности спрогнозированных результатов была введена система ранжирования:

• Прогноз с точностью до 2-х очков считается очень точным;

• Прогноз с точностью до 5-и очков считается относительно точным;

• Прогноз с точностью до 10-и очков считается неточным;

• Прогноз с точностью до 15 очков считается крайне неточным.

Для создания такой системы ранжирования были проанализированы последние 10 сезонов НБА и выявлен наибольший показатель среднего количества набираемых очков: он составил 36.43 PPG. Так, опираясь на этот показатель, можно утверждать, что 2 очка – это приблизительно 6% погрешности в показателе PPG и такой прогноз можно считать достаточно точным.

4.1 Модель линейной регрессии

Для проведения данного теста необходимо было импортировать метод LinearRegression из библиотеки sklearn.

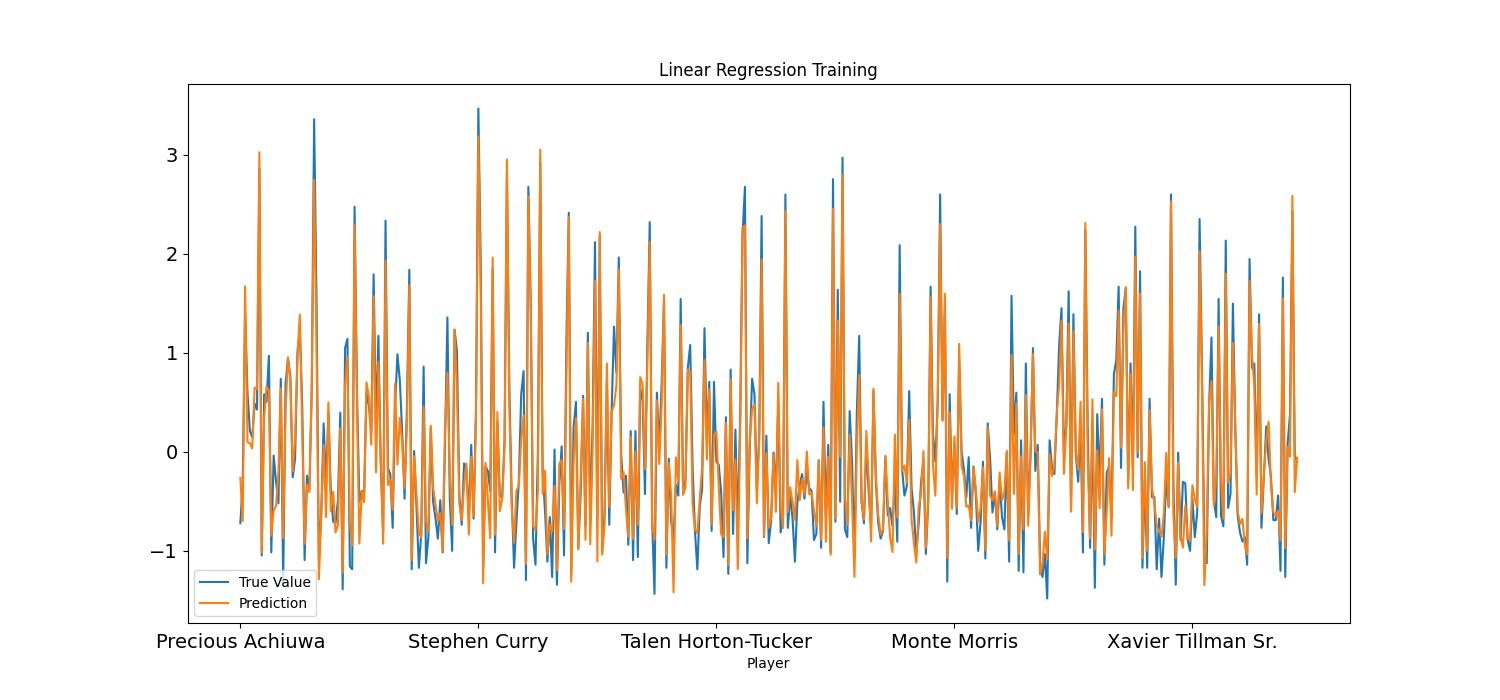


Рисунок 22 – Сравнение истинных значений и спрогнозированных при линейной регрессии

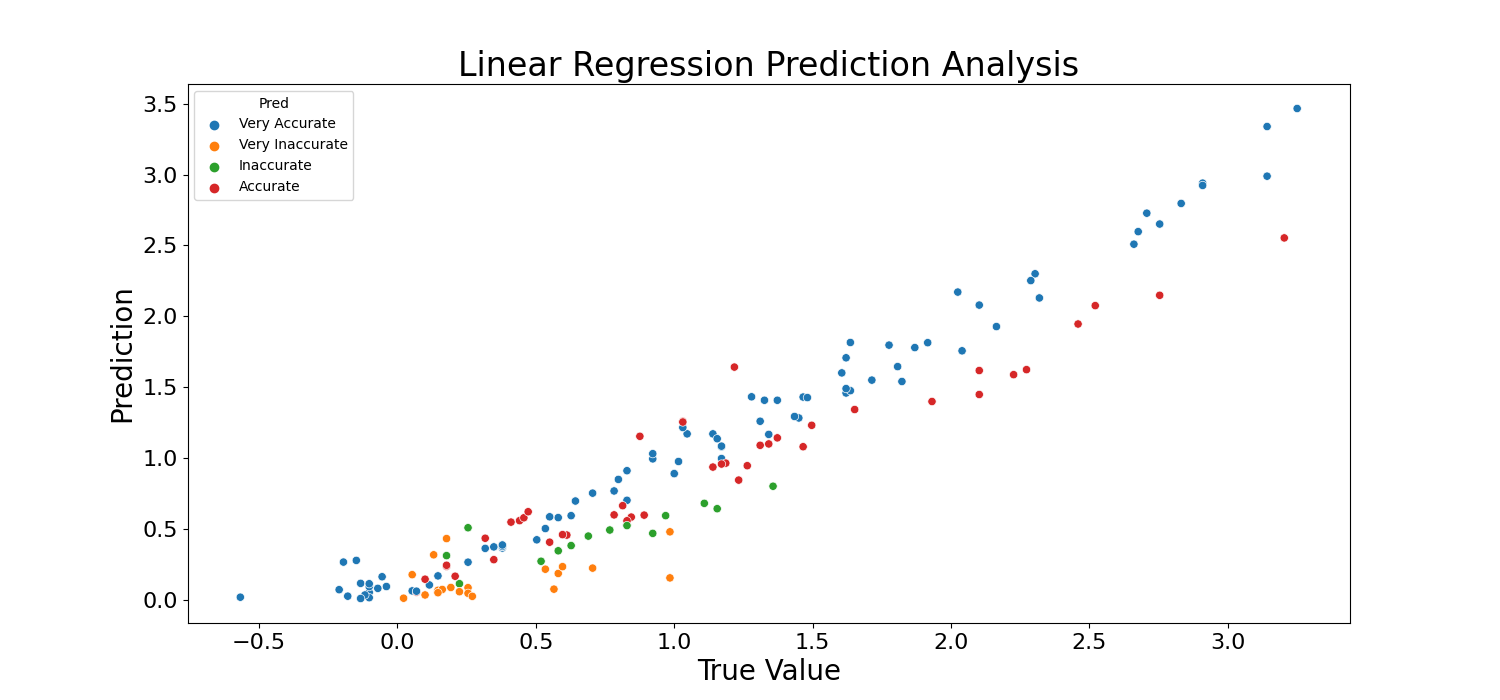


Рисунок 23 – Точечный график для отображения качества спрогнозируемого результата

В результате тестирования было установлено, что модель линейной регрессии способна предсказывать:

• Относительно точно с вероятностью 26.35%;

• Очень точно с вероятностью 52.10%.

4.2 Модель Extra Trees

Для проведения данного теста необходимо было импортировать метод ExtraTrees из библиотеки sklearn.

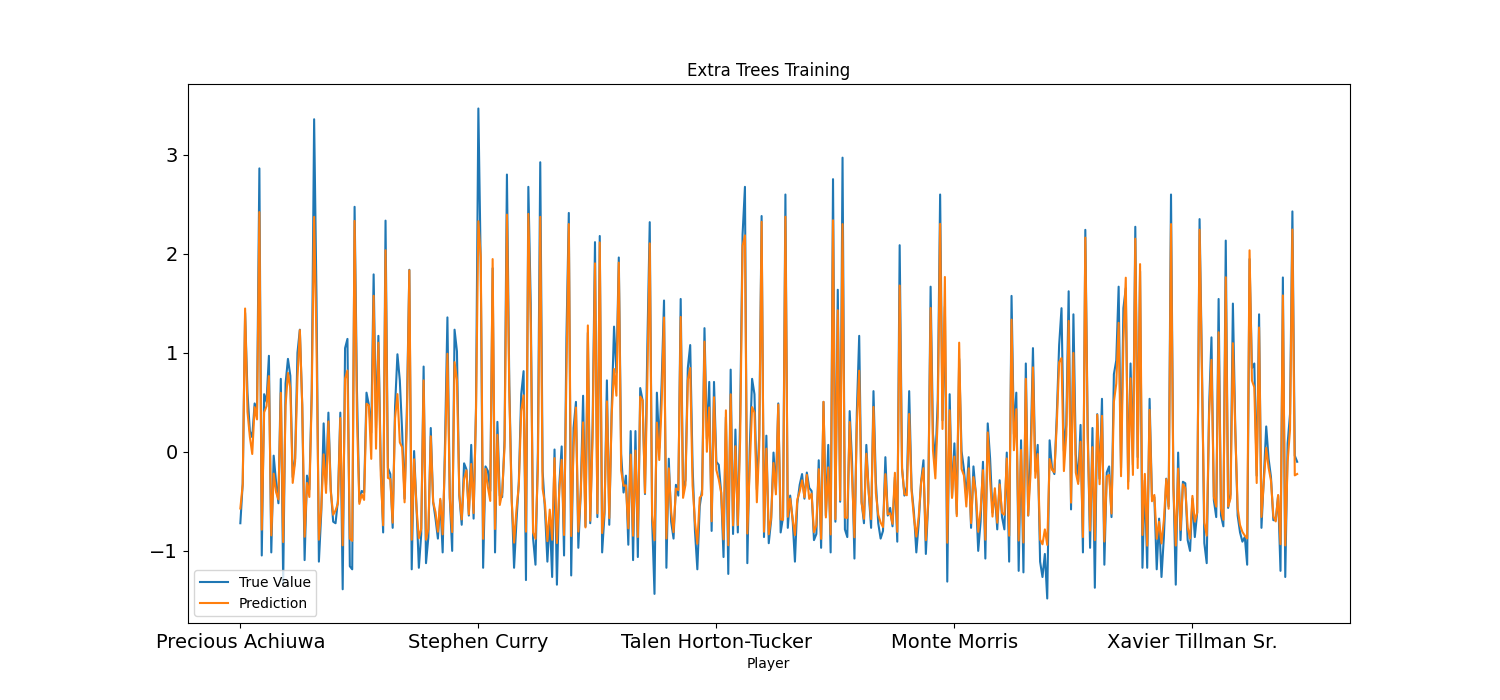


Рисунок 24 – Сравнение истинных значений и спрогнозированных при Extra Trees

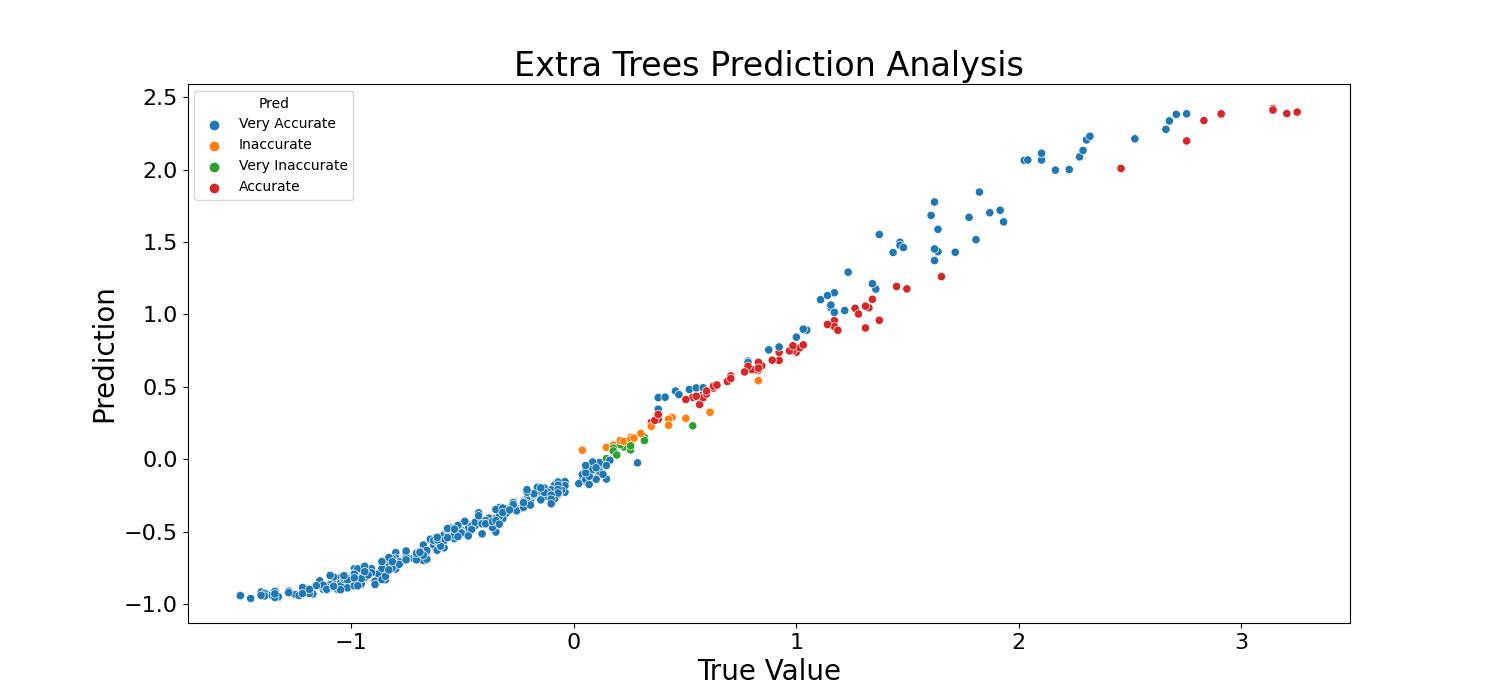


Рисунок 25 – Точечный график для отображения качества спрогнозируемого результата

В результате тестирования было установлено, что модель Extra Trees способна предсказывать:

• Относительно точно с вероятностью 13.71%;

• Очень точно с вероятностью 79.1%.

4.3 Модель Random Forest

Для проведения данного теста необходимо было импортировать метод RandomForest из библиотеки sklearn.

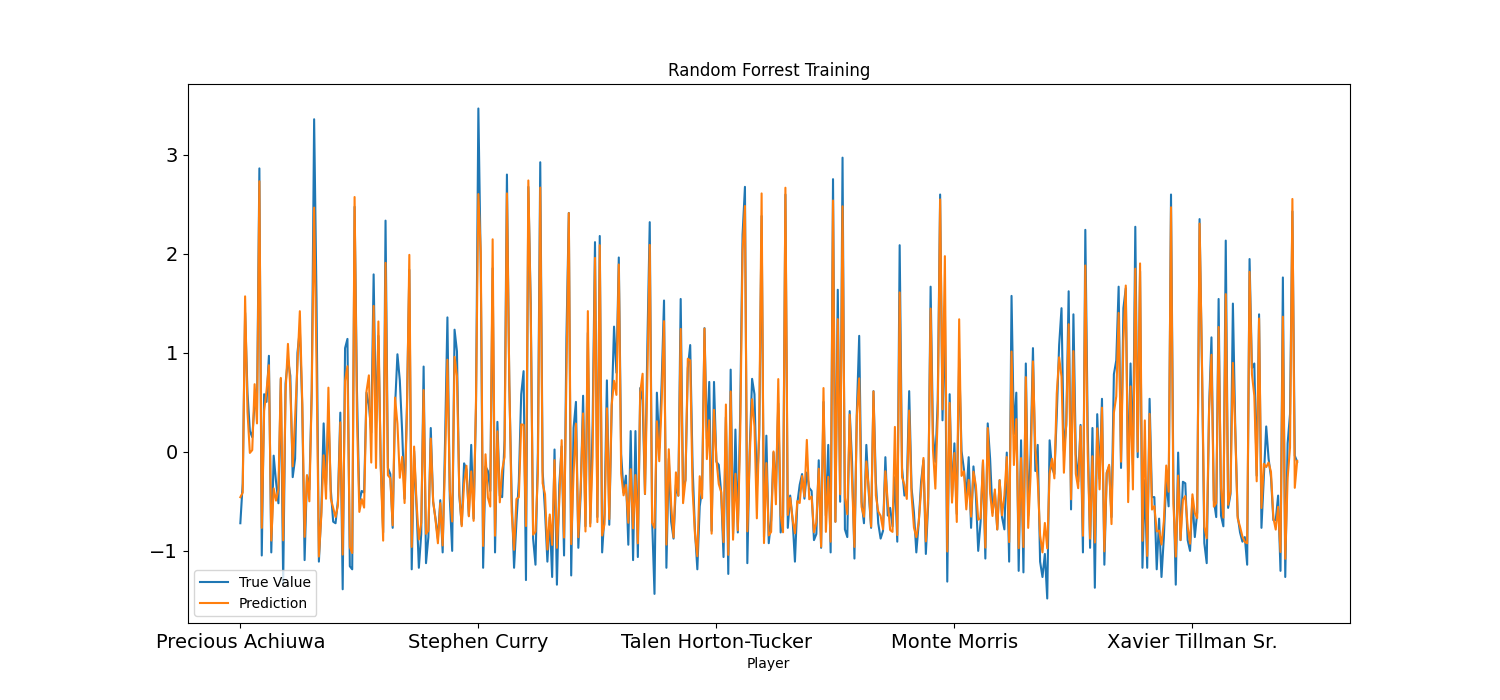


Рисунок 26 – Сравнение истинных значений и спрогнозированных при Random Forest

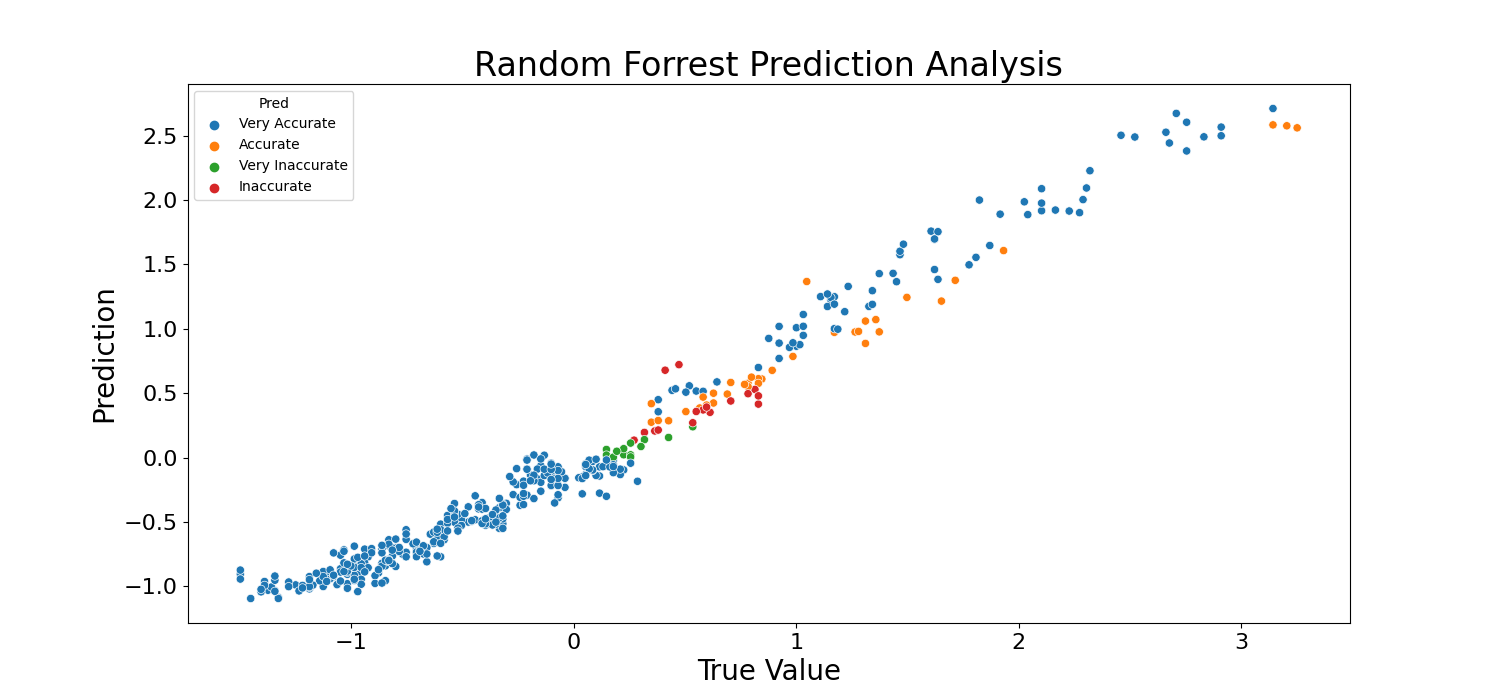


Рисунок 27 – Точечный график для отображения качества спрогнозируемого результата

В результате тестирования было установлено, что модель Random Forest способна предсказывать:

• Относительно точно с вероятностью 7.64%;

• Очень точно с вероятностью 85.17%.

4.4 Модель XGBoost

Для проведения данного теста необходимо было импортировать метод xgboost из библиотеки sklearn и XGBRegressor.

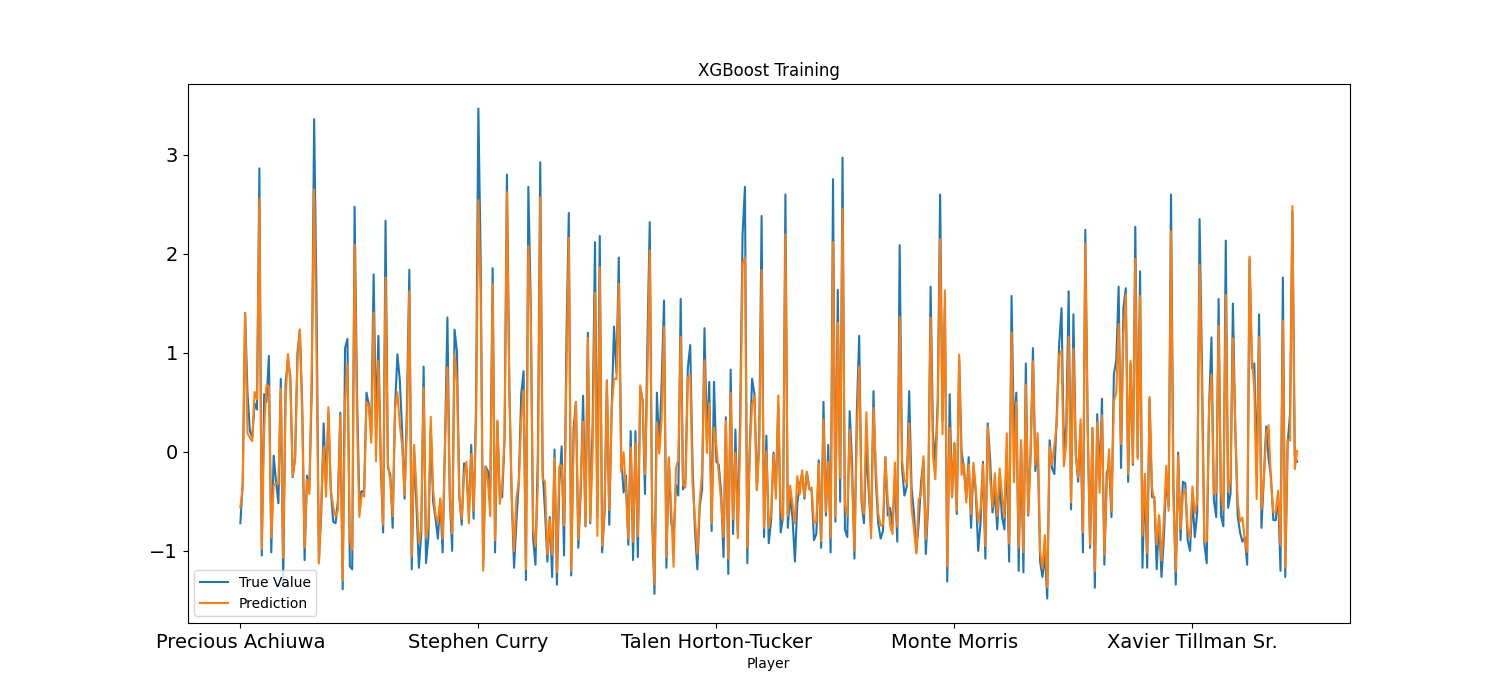


Рисунок 28 – Сравнение истинных значений и спрогнозированных при XGBoost

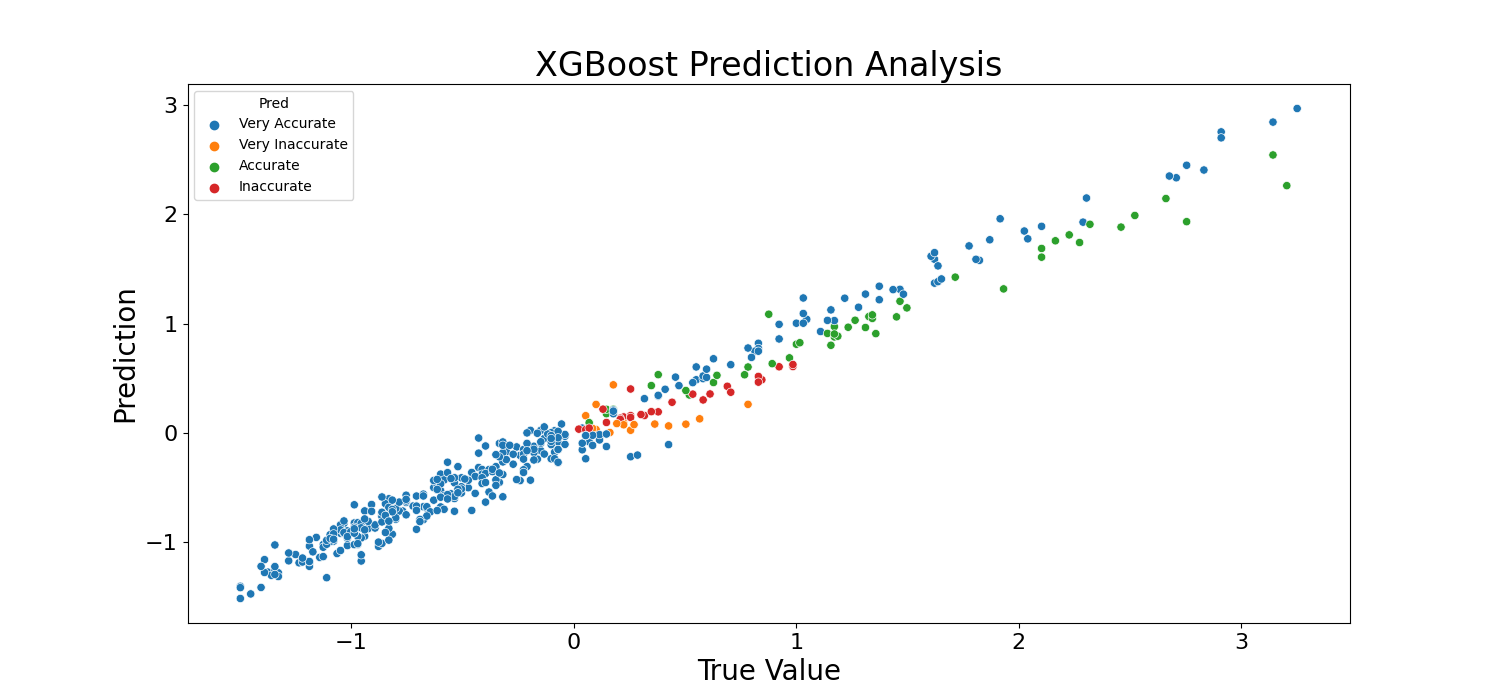


Рисунок 29 – Точечный график для отображения качества спрогнозируемого результата

В результате тестирования было установлено, что модель Random Forest способна предсказывать:

• Относительно точно с вероятностью 10.56%;

• Очень точно с вероятностью 79.78%.

4.5 Выводы по тестовой части

По результатам тестирования удалось выяснить, что наиболее точный прогноз дает модель Random Forest. Вероятнее всего, это связано с особенностью организации входных данных для обучения. Наиболее неточный результат показала модель линейной регрессии, что было достаточно ожидаемо. Модель линейной регрессии, в данном случае, строилась лишь с целью получения начальной точки для других моделей и сравнения с ней. XGBoost и Extra Trees показали неплохие результаты для игроков менее задействованных в игре, но проиграли Random Forest.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта была разработана математическая модель, позволяющая в полной мере предсказать статистику игроков НБА как для предыдущих, так, надеюсь, и для будущих игровых сезонов.

В ходе выполнения проекта были получены теоретические и практические навыки в областях Data Science, Data Analysis и Data Engineering.

В дальнейшем планируется расширить область анализируемых показателей и сделать графический интерфейс для математической модели. Помимо статистики игроков, будут анализироваться профессиональные командные показатели. Графический интерфейс будет представлять из себя логотипы команд и игроков, а необходимы они будут для сопоставления определенному игроку наиболее подходящих для его стиля игры других игроков.

На сегодняшний день программный продукт полностью соответствует требованиям, представленным ранее, и с его помощью можно смело вступать в аргументированные диалогами с лучшими аналитиками баскетбольного мира. Теперь на все вопросы и возражения есть достойный математический аппарат, помогающий выйти победителем из любой дискуссии и в полной мере показать свои профессиональные навыки в области прогнозирования статистических показателей игроков НБА.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. basketball-reference.com [Электронный ресурс] – Режим доступа:

https://www.basketball-reference.com/

1. realgm.com [Электронный ресурс] – Режим доступа:

<https://basketball.realgm.com/nba/teams/Washington-Wizards/30/stats/2020/Averages/All/points/All/desc/1/Regular_Season>

1. skitit-learn.org [Электронный ресурс] – Режим доступа:

https://scikit-learn.org/stable/index.html

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг программного кода

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import Charts as ch

from sklearn.metrics import r2\_score

from sklearn.feature\_selection import RFE

from sklearn.linear\_model import LinearRegression

from sklearn.preprocessing import MinMaxScaler

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean\_squared\_error

from sklearn.ensemble import ExtraTreesRegressor

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import cross\_val\_score

from scipy import stats

from sklearn.ensemble import RandomForestRegressor

from xgboost import XGBRegressor

# ch.players\_effectiveness('Shaquille O' + "'" + 'Neal', "D", "AD", "E")

# ch.average\_decade\_stats(2010, 2000, '2PA')

# ch.average\_pts\_stats(1946, 76, 'PTS')

# ch.matrix\_correlogram('LeBron James')

# ch.player\_position\_efficient(1977, "AST")

# Приводим таблицу в порядок

df = pd.read\_excel('pred.xlsx')

df.pop('Rk')

df['Fouls'] = df['PF']

df.pop('PF')

dum1 = pd.get\_dummies(df['Pos'])

df = pd.concat([df, dum1], axis=1)

df.pop('Pos')

player\_team = df.Tm.unique()

tm\_dict = dict(zip(player\_team, range(len(player\_team))))

df = df.replace({'Tm': tm\_dict})

df.pop('Tm')

df = df.set\_index('Player')

cols\_at\_end = ['TRB', 'AST', 'STL', 'BLK', 'PTS']

df = df[[c for c in df if c not in cols\_at\_end] + [c for c in cols\_at\_end if c in df]]

cols = []

co = df.columns

for col in co:

cols = cols + [col]

dum = ['TRB', 'AST', 'STL', 'BLK']

for du in dum:

cols.remove(du)

# Стандартизация данных для модели

sc2 = StandardScaler()

for col in cols:

df[[col]] = sc2.fit\_transform(df[[col]])

# df['PTS'] = sc2.fit\_transform(df['PTS'])

sc3 = StandardScaler()

df[['BLK']] = sc3.fit\_transform(df[['BLK']])

sc4 = StandardScaler()

df[['STL']] = sc4.fit\_transform(df[['STL']])

sc5 = StandardScaler()

df[['AST']] = sc5.fit\_transform(df[['AST']])

sc6 = StandardScaler()

df[['TRB']] = sc6.fit\_transform(df[['TRB']])

df = df.fillna(0)

# table2021 = norm\_table('2020-2021.xlsx')

# table2022 = norm\_table('2021-2022.xlsx')

# Конечное представление табличек

df3 = df[0:445]

df4 = df[445:]

#print(df3, df4)

# Тестовые наборы данных

train = df3 # [0:(round(len(df3)\*.8))]

test = df4 # [0:(round(len(df3)\*.8))]

trains = np.array\_split(train, [31], axis=1)

train\_x = trains[0]

train\_y = trains[0]

tests = np.array\_split(test, [31], axis=1)

test\_x = tests[0]

test\_y = tests[0]

d = df3

# Попарная корреляция столбцов

corr = d.corr()

mask = np.zeros\_like(corr, dtype=np.bool)

mask[np.triu\_indices\_from(mask)] = True

f, ax = plt.subplots(figsize=(11, 9))

cmap = sns.diverging\_palette(100, 275, as\_cmap=True)

sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0, square=True, linewidths=.01, cbar\_kws={"shrink": .5})

plt.show()

# Сокращение строк

model = LinearRegression()

rfe = RFE(model, n\_features\_to\_select=25)

reg = rfe.fit(train\_x.values, test\_x.PTS.values)

#print(reg.support\_)

#print(reg.ranking\_)

# reg = ExtraTreesRegressor(min\_samples\_split=4, n\_estimators=1000,bootstrap=True).fit(train\_x.values, train\_y.PTS.values)

# reg.feature\_importances\_

def adj\_r2\_score(r2, n, k):

return 1 - ((1 - r2) \* ((n - 1) / (n - k - 1)))

reg = LinearRegression().fit(train\_x.values, test\_x.PTS.values)

# reg.score(train\_x.values, train\_y.PTS.values)

train\_y2 = train\_y.copy(deep=True)

test\_y2 = test\_y.copy(deep=True)

train\_y2['prediction'] = reg.predict(train\_x.values)

test\_y2['prediction'] = reg.predict(test\_x.values)

train\_y2['PTS'] = train\_y2['PTS']

test\_y2['PTS'] = test\_y2['PTS']

train\_y2['PTS'].plot(figsize=(15, 7), title='Linear Regression Training', fontsize=14, label='True Value')

train\_y2['prediction'].plot(figsize=(15, 7), title='Linear Regression Training', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj train r^2 = ' + str(

# adj\_r2\_score((r2\_score(train\_y2.PTS, train\_y2.prediction)), train\_x.shape[0], train\_x.shape[1])))

#print('train mse = ' + str(mean\_squared\_error(train\_y2.PTS, train\_y2.prediction)))

test\_y2['PTS'].plot(figsize=(15, 7), title='Linear Regression Testing', fontsize=14, label='True Value')

test\_y2['prediction'].plot(figsize=(15, 7), title='Linear Regression Testing', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj test r^2 = ' + str(

# adj\_r2\_score((r2\_score(test\_y2.PTS, test\_y2.prediction)), train\_x.shape[0], train\_x.shape[1])))

#print('test mse = ' + str(mean\_squared\_error(test\_y2.PTS, test\_y2.prediction)))

test\_y2 = test\_y2[test\_y2['prediction'] > 0]

test\_y2.loc[test\_y2[['prediction', 'PTS']].max(axis=1) / test\_y2[['prediction', 'PTS']].min(

axis=1) >= 2, 'Pred'] = 'Very Inaccurate'

test\_y2.loc[

test\_y2[['prediction', 'PTS']].max(axis=1) / test\_y2[['prediction', 'PTS']].min(axis=1) < 2, 'Pred'] = 'Inaccurate'

test\_y2.loc[

test\_y2[['prediction', 'PTS']].max(axis=1) / test\_y2[['prediction', 'PTS']].min(axis=1) < 1.5, 'Pred'] = 'Accurate'

test\_y2.loc[test\_y2[['prediction', 'PTS']].max(axis=1) / test\_y2[['prediction', 'PTS']].min(

axis=1) < 1.2, 'Pred'] = 'Very Accurate'

fig, ax1 = plt.subplots(figsize=(15, 7))

ax = sns.scatterplot(x="PTS", y="prediction", hue="Pred", data=test\_y2)

ax.set\_title("Linear Regression Prediction Analysis", fontsize=24)

ax.set\_xlabel("True Value", fontsize=20)

ax.set\_ylabel("Prediction", fontsize=20)

ax.tick\_params(labelsize=16)

plt.show()

print("This model is able to relatively accurately predict " + str(

round(100 \* len(test\_y2[test\_y2['Pred'] == 'Accurate']) / len(test\_y2), 2)) + "% PTS of NBA Players.")

print("This model is able to very accurately predict " + str(

round(100 \* len(test\_y2[test\_y2['Pred'] == 'Very Accurate']) / len(test\_y2), 2)) + "% PTS of NBA players.")

reg = ExtraTreesRegressor(min\_samples\_split=22, n\_estimators=1000,max\_depth=4, bootstrap=True)

reg.fit(train\_x.values, test\_x.PTS.values)

train\_y2 = train\_y.copy(deep=True)

test\_y2 = test\_y.copy(deep=True)

train\_y2['prediction'] = reg.predict(train\_x.values)

test\_y2['prediction'] = reg.predict(test\_x.values)

train\_y2['PTS'] = train\_y2['PTS']

test\_y2['PTS'] = test\_y2['PTS']

train\_y2['PTS'].plot(figsize=(15,7), title= 'Extra Trees Training', fontsize=14, label='True Value')

train\_y2['prediction'].plot(figsize=(15,7), title= 'Extra Trees Training', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj train r^2 = ' + str(adj\_r2\_score((r2\_score(train\_y2.PTS, train\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('train mse = ' + str(mean\_squared\_error(train\_y2.PTS, train\_y2.prediction)))

test\_y2['PTS'].plot(figsize=(15,7), title= 'Extra Trees Testing', fontsize=14, label='True Value')

test\_y2['prediction'].plot(figsize=(15,7), title= 'Extra Trees Testing', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj test r^2 = ' + str(adj\_r2\_score((r2\_score(test\_y2.PTS, test\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('test mse = ' + str(mean\_squared\_error(test\_y2.PTS, test\_y2.prediction)))

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) >= 2, 'Pred'] = 'Very Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 2, 'Pred'] = 'Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.5, 'Pred'] = 'Accurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.2, 'Pred'] = 'Very Accurate'

fig, ax1 = plt.subplots(figsize=(15, 7))

ax = sns.scatterplot(x="PTS", y="prediction", hue="Pred", data=test\_y2)

ax.set\_title("Extra Trees Prediction Analysis", fontsize=24)

ax.set\_xlabel("True Value",fontsize=20)

ax.set\_ylabel("Prediction",fontsize=20)

ax.tick\_params(labelsize=16)

plt.show()

print ("This model is able to relatively accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

print ("This model is able to very accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Very Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

reg = RandomForestRegressor(n\_estimators=1000, max\_depth=5, min\_samples\_split=14)

reg.fit(train\_x.values, test\_x.PTS.values)

train\_y2 = train\_y.copy(deep=True)

test\_y2 = test\_y.copy(deep=True)

train\_y2['prediction'] = reg.predict(train\_x.values)

test\_y2['prediction'] = reg.predict(test\_x.values)

train\_y2['PTS'] = train\_y2['PTS']

test\_y2['PTS'] = test\_y2['PTS']

train\_y2['PTS'].plot(figsize=(15,7), title= 'Random Forrest Training', fontsize=14, label='True Value')

train\_y2['prediction'].plot(figsize=(15,7), title= 'Random Forrest Training', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj train r^2 = ' + str(adj\_r2\_score((r2\_score(train\_y2.NBAPTS, train\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('train mse = ' + str(mean\_squared\_error(train\_y2.NBAPTS, train\_y2.prediction)))

test\_y2['PTS'].plot(figsize=(15,7), title= 'Random Forrest Testing', fontsize=14, label='True Value')

test\_y2['prediction'].plot(figsize=(15,7), title= 'Random Forrest Testing', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj test r^2 = ' + str(adj\_r2\_score((r2\_score(test\_y2.NBAPTS, test\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('test mse = ' + str(mean\_squared\_error(test\_y2.NBAPTS, test\_y2.prediction)))

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) >= 2, 'Pred'] = 'Very Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 2, 'Pred'] = 'Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.5, 'Pred'] = 'Accurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.2, 'Pred'] = 'Very Accurate'

fig, ax1 = plt.subplots(figsize=(15, 7))

ax = sns.scatterplot(x="PTS", y="prediction", hue="Pred", data=test\_y2)

ax.set\_title("Random Forrest Prediction Analysis", fontsize=24)

ax.set\_xlabel("True Value",fontsize=20)

ax.set\_ylabel("Prediction",fontsize=20)

ax.tick\_params(labelsize=16)

plt.show()

print ("This model is able to relatively accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

print ("This model is able to very accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Very Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

reg = XGBRegressor(n\_estimators=1000, max\_depth=16, reg\_lambda=.48, gamma=.3, booster='gblinear')

reg.fit(train\_x.values, test\_x.PTS.values)

train\_y2 = train\_y.copy(deep=True)

test\_y2 = test\_y.copy(deep=True)

train\_y2['prediction'] = reg.predict(train\_x.values)

test\_y2['prediction'] = reg.predict(test\_x.values)

train\_y2['PTS'] = train\_y2['PTS']

test\_y2['PTS'] = test\_y2['PTS']

train\_y2['PTS'].plot(figsize=(15,7), title= 'XGBoost Training', fontsize=14, label='True Value')

train\_y2['prediction'].plot(figsize=(15,7), title= 'XGBoost Training', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj train r^2 = ' + str(adj\_r2\_score((r2\_score(train\_y2.NBAPTS, train\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('train mse = ' + str(mean\_squared\_error(train\_y2.NBAPTS, train\_y2.prediction)))

test\_y2['PTS'].plot(figsize=(15,7), title= 'XGBoost Testing', fontsize=14, label='True Value')

test\_y2['prediction'].plot(figsize=(15,7), title= 'XGBoost Testing', fontsize=14, label='Prediction')

plt.legend(loc='lower left')

plt.show()

#print('adj test r^2 = ' + str(adj\_r2\_score((r2\_score(test\_y2.NBAPTS, test\_y2.prediction)),train\_x.shape[0], train\_x.shape[1])))

#print('test mse = ' + str(mean\_squared\_error(test\_y2.NBAPTS, test\_y2.prediction)))

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) >= 2, 'Pred'] = 'Very Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 2, 'Pred'] = 'Inaccurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.5, 'Pred'] = 'Accurate'

test\_y2.loc[test\_y2[['prediction','PTS']].max(axis=1)/test\_y2[['prediction','PTS']].min(axis=1) < 1.2, 'Pred'] = 'Very Accurate'

fig, ax1 = plt.subplots(figsize=(15, 7))

ax = sns.scatterplot(x="PTS", y="prediction", hue="Pred", data=test\_y2)

ax.set\_title("XGBoost Prediction Analysis", fontsize=24)

ax.set\_xlabel("True Value",fontsize=20)

ax.set\_ylabel("Prediction",fontsize=20)

ax.tick\_params(labelsize=16)

plt.show()

print ("This model is able to relatively accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

print ("This model is able to very accurately predict " + str(round(100\*len(test\_y2[test\_y2['Pred'] == 'Very Accurate'])/len(test\_y2),2)) + "% of NBA Players.")

# df3.to\_html('table.html')