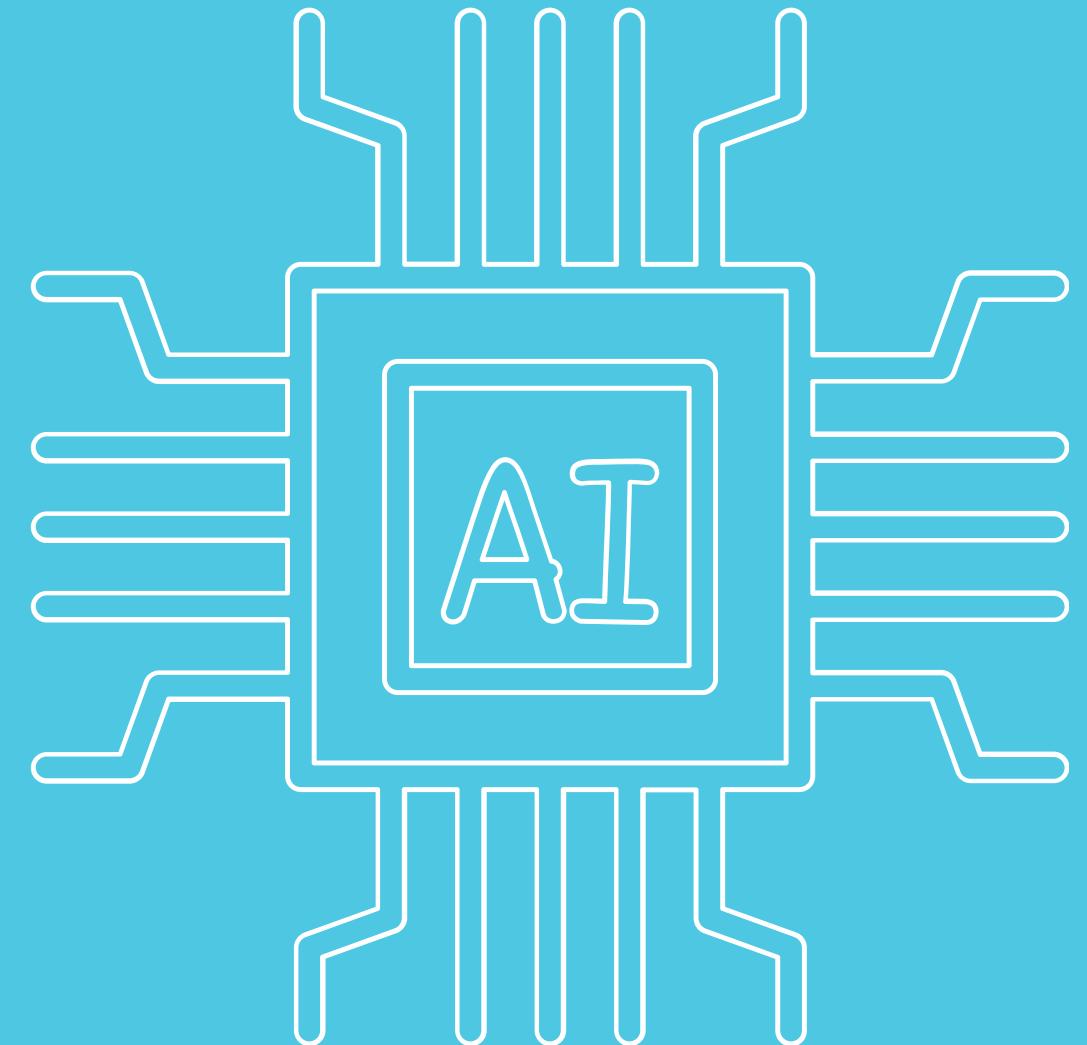


Проект по курсу МЛ

Задача: предсказать, где окажется каждый игрок на футбольном поле, пока мяч летит после паса квотербека

Как оцениваем качество: RMSE (среднеквадратичная ошибка)
— чем меньше, тем точнее наши предсказания в ярдах





Постановка задачи

Что нам дано:

- Данные о каждом игроке **ДО** момента броска мяча: где он стоит, куда бежит, с какой скоростью
- Точка, куда прилетит мяч (квотербек уже бросил – мы знаем траекторию)
- Кто из игроков должен поймать мяч (**целевой принимающий**)

Что нужно предсказать:

- Координаты каждого из 22 игроков в **каждый** момент времени, пока мяч летит
- Камера снимает 10 кадров в секунду, мяч летит в **среднем** 1 секунду

Зачем это нужно:

1. Тренерам: понять, как защита реагирует на разные пасы
2. Аналитикам: оценить качество маршрутов принимающих
3. Командам: предсказать вероятность перехвата мяча защитой
4. Индустрии ставок: live-беттинг на каждый розыгрыш – \$150+ млрд рынок в США
5. Студентам: получить 10 баллов за проект



Описание датасета

Почему датасет подходит для проекта:

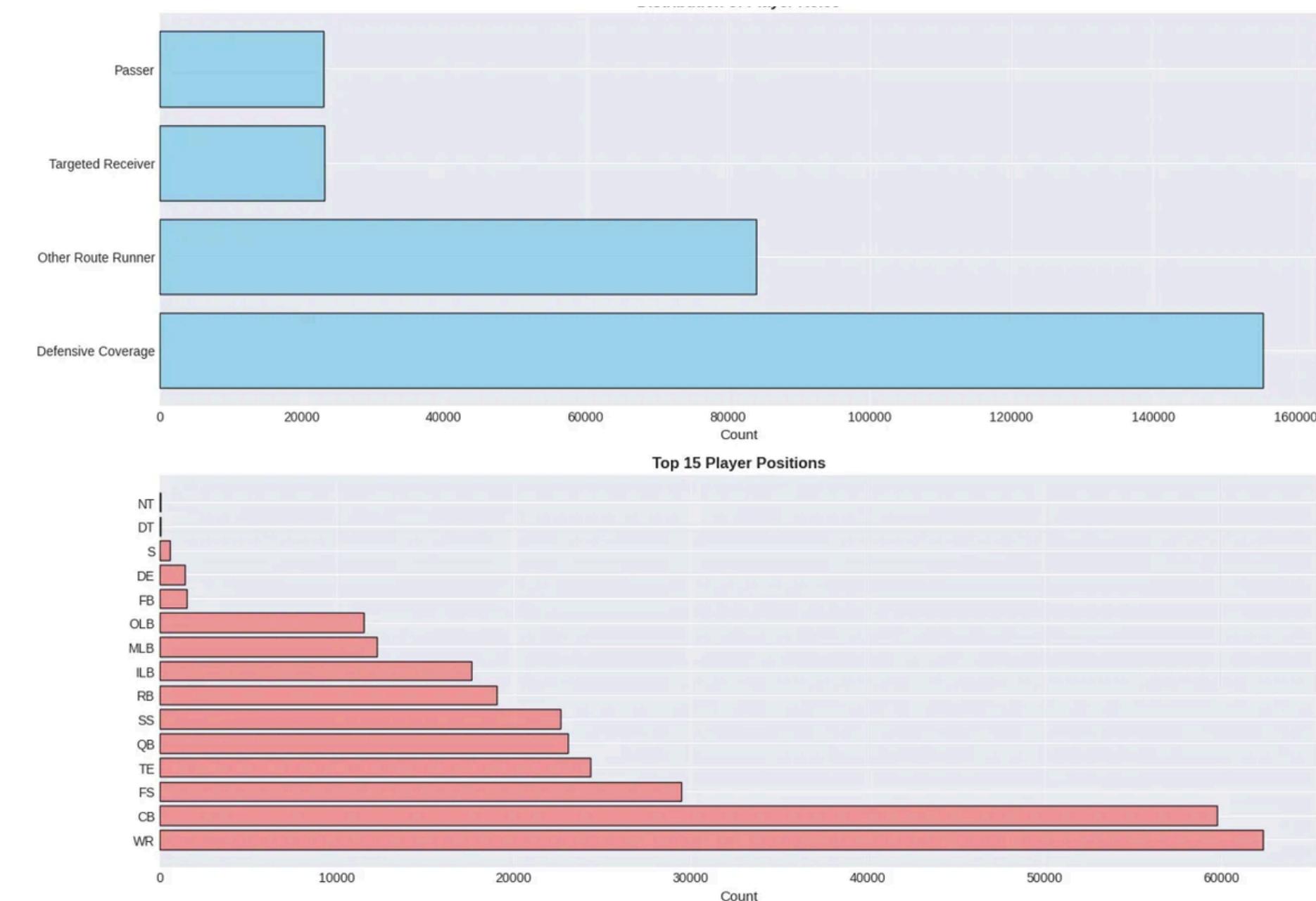
- Есть временная составляющая (кадры идут последовательно)
- Есть пространственные данные (координаты на поле)
- Реальные данные из профессионального спорта
- Более 1000 строк (требование курса)

Что измеряем	Сколько
Записей о позициях игроков (вход)	4.9 миллиона
Записей для предсказания (выход)	560 тысяч
Матчей	272
Отдельных розыгрышей	14000
Уникальных игроков	2157
Недель сезона	18



Кто есть кто на поле

Роль	Количество о записей	Что делает этот игрок
Защитники в покрытии	155397	Следят за принимающими, пытаются перехватить мяч
Другие принимающие	84063	Бегут маршруты, чтобы отвлечь защиту
Целевой принимающий	23151	Тот, кому летит мяч — главный герой розыгрыша
Квотербек	23103	Бросает мяч, после броска почти не двигается



Главный вывод: разные роли = разное поведение.
Модель должна это учитывать!

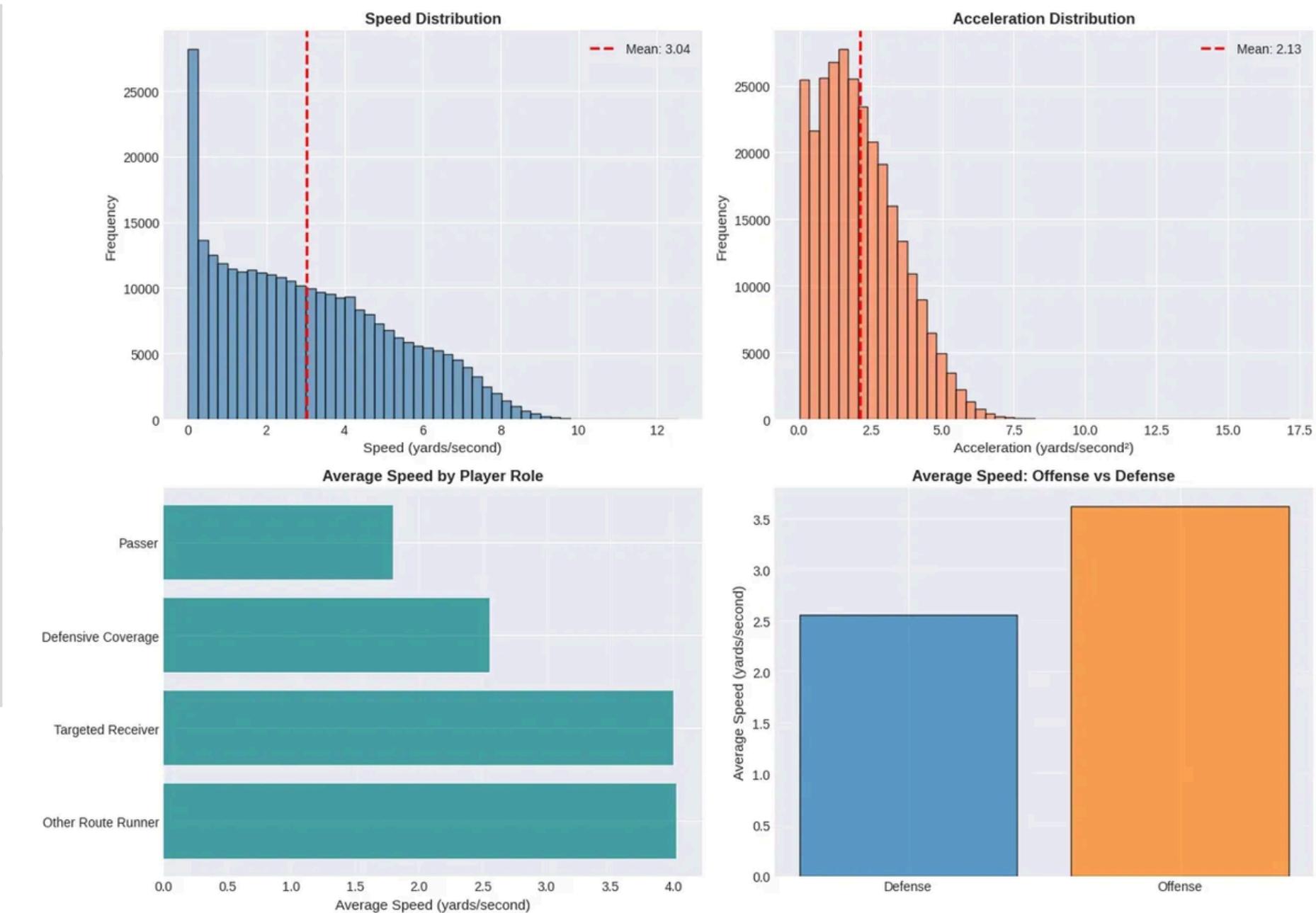
Как быстро бегают игроки



Кто	Средняя скорость	Почему
Принимающие	3.6 м/с	Активно бегут к мячу
Защитники	3.2 м/с	Реагируют на движение, чуть медленнее
Квотербек	1.37 м/с	Стоит на месте после броска

Общая статистика:

- Средняя скорость: 2.78 м/с
- Максимум: 10.97 м/с (это спринт!)
- Большинство игроков двигаются медленно, быстрых мало



Вывод для модели: скорость – один из главных признаков для предсказания!



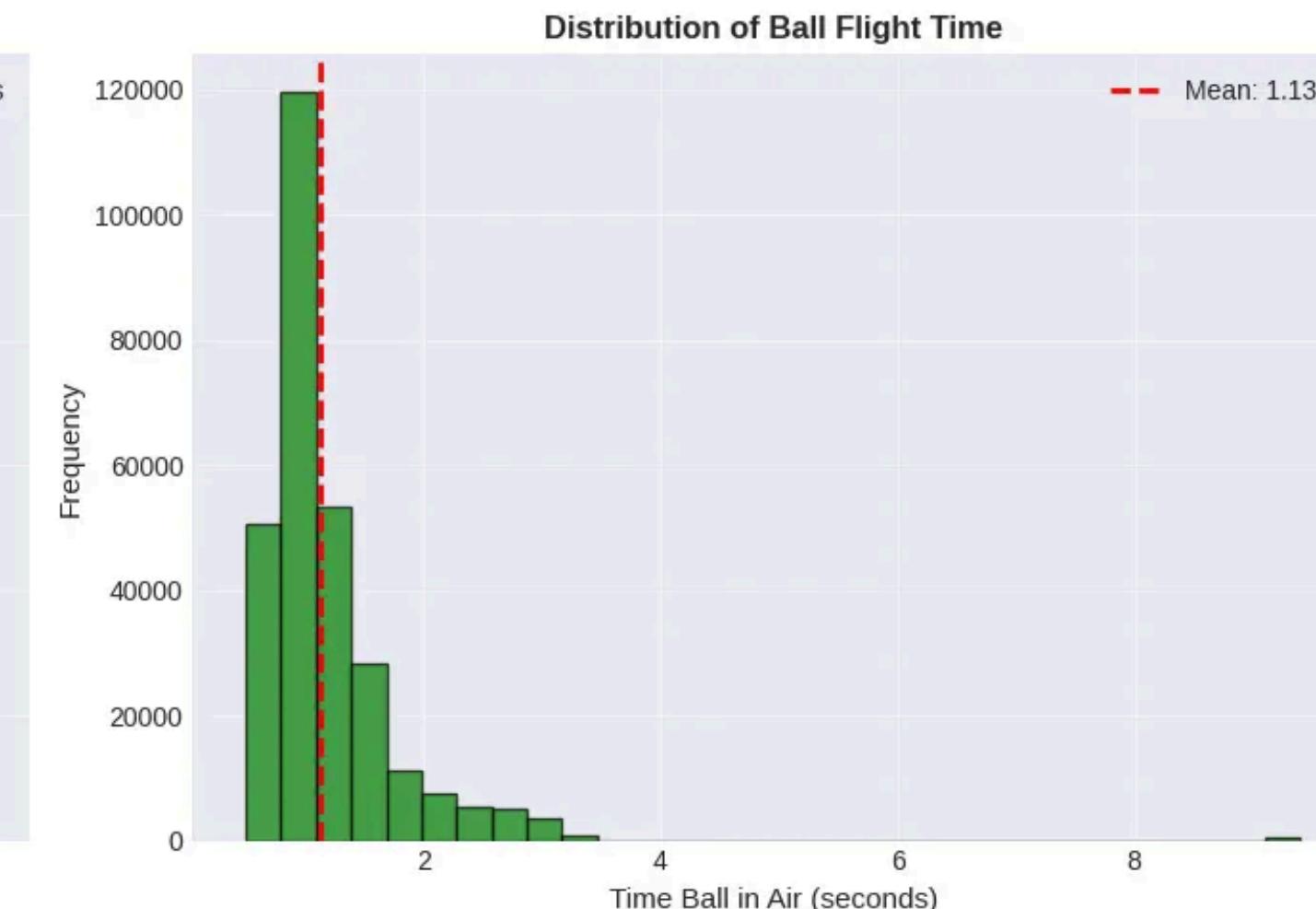
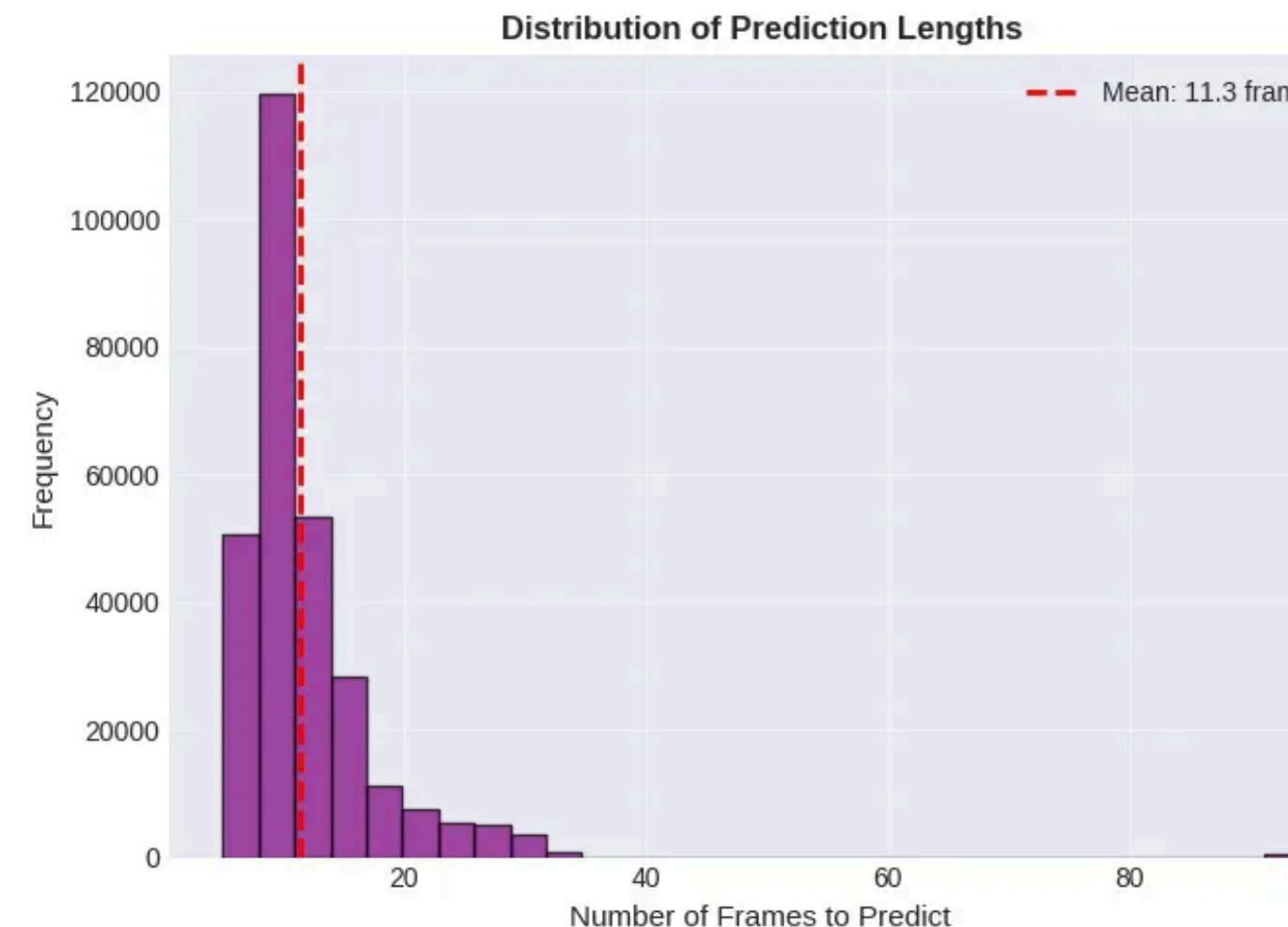
Сколько времени летит мяч

Статистика времени полёта:

- В среднем: 11.3 кадра = 1.13 секунды
- Типичный пас: 10 кадров = 1 секунда
- Самый короткий: 5 кадров = 0.5 секунды
- Самый длинный: 94 кадра = 9.4 секунды
(редкость!)

Что это значит:

- 75% всех пасов – короткие (≤ 1.3 секунды)
- Короткие пасы = меньше неопределённости, легче предсказать
- Длинные пасы (глубокие броски) – редкие, но сложные для модели

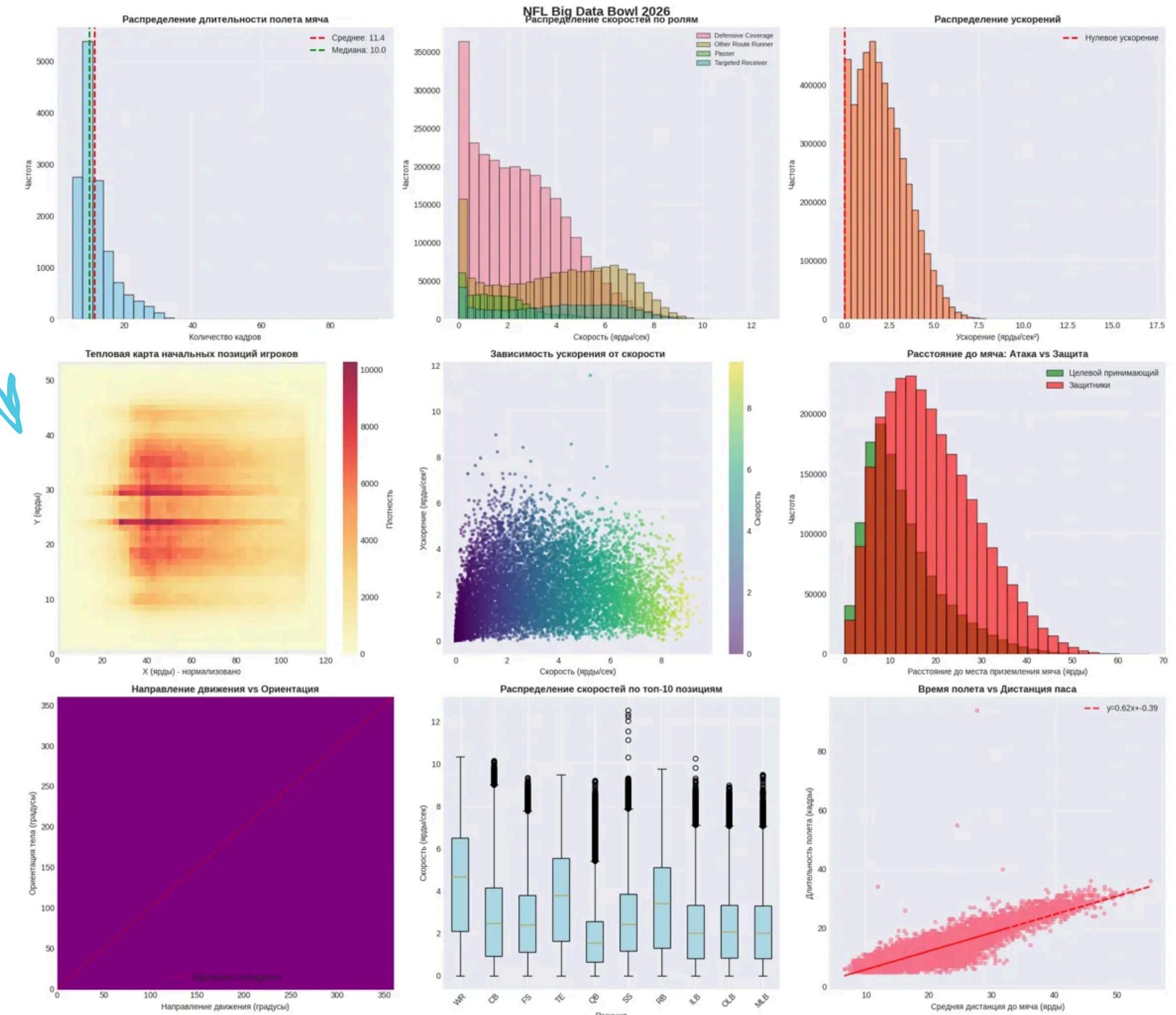


Где находятся игроки на поле



Что видим на карте:

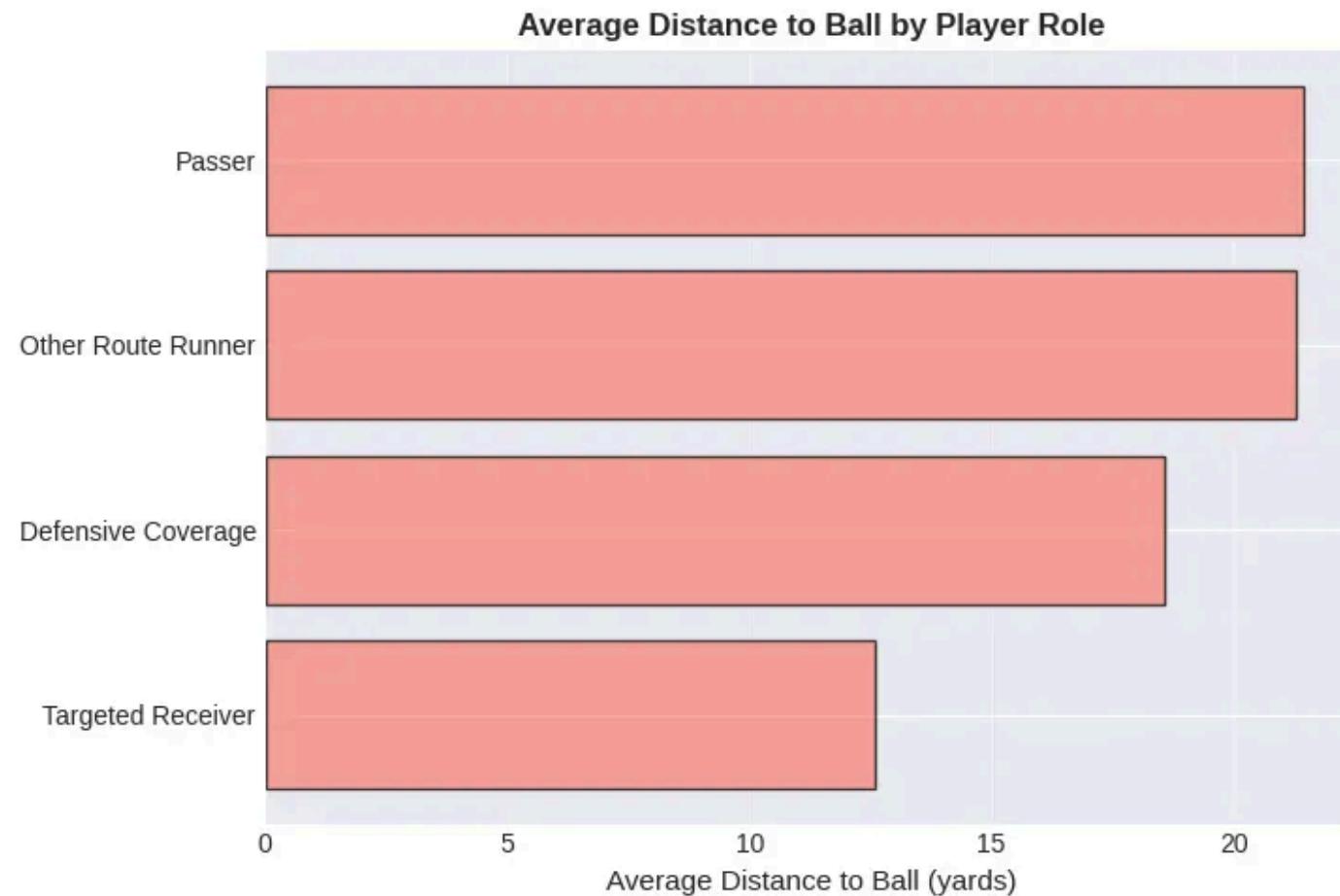
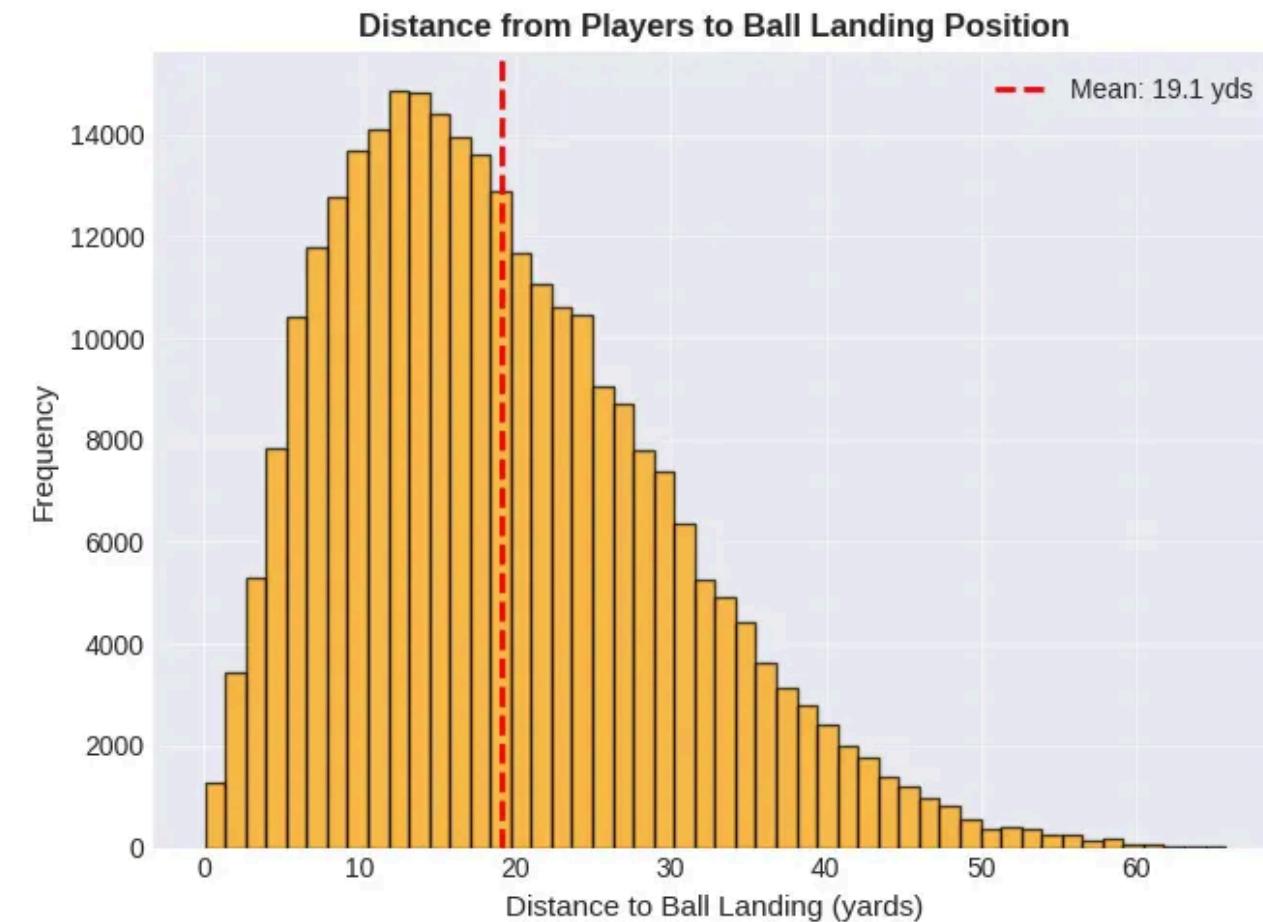
- Поле: 120 ярдов в длину, 53.3 ярда в ширину
- Игровые концентрируются в центре поля (по ширине)
- Горизонтальные "полосы" — это типичные построения команд перед пасом





Кто ближе всех к мячу

Роль	Количество записей	Что делает этот игрок
Задни ки	10.97 м	Реагируют с опозданием
Другие принима ющие	16.46 м	Отвлекают защиту в стороне
Целевой принима ющий	18.29 м	Уже бежит к мячу!
Квотербек	20.12 м	Остается сзади после броска



Ключевой инсайт: целевой принимающий В МОМЕНТ БРОСКА уже движется к мячу — он знает, куда бежать. Это главная подсказка для модели!

Проблемы в данных



Что мы проверяли:

Может ли игрок физически добежать до мяча за время его полёта?

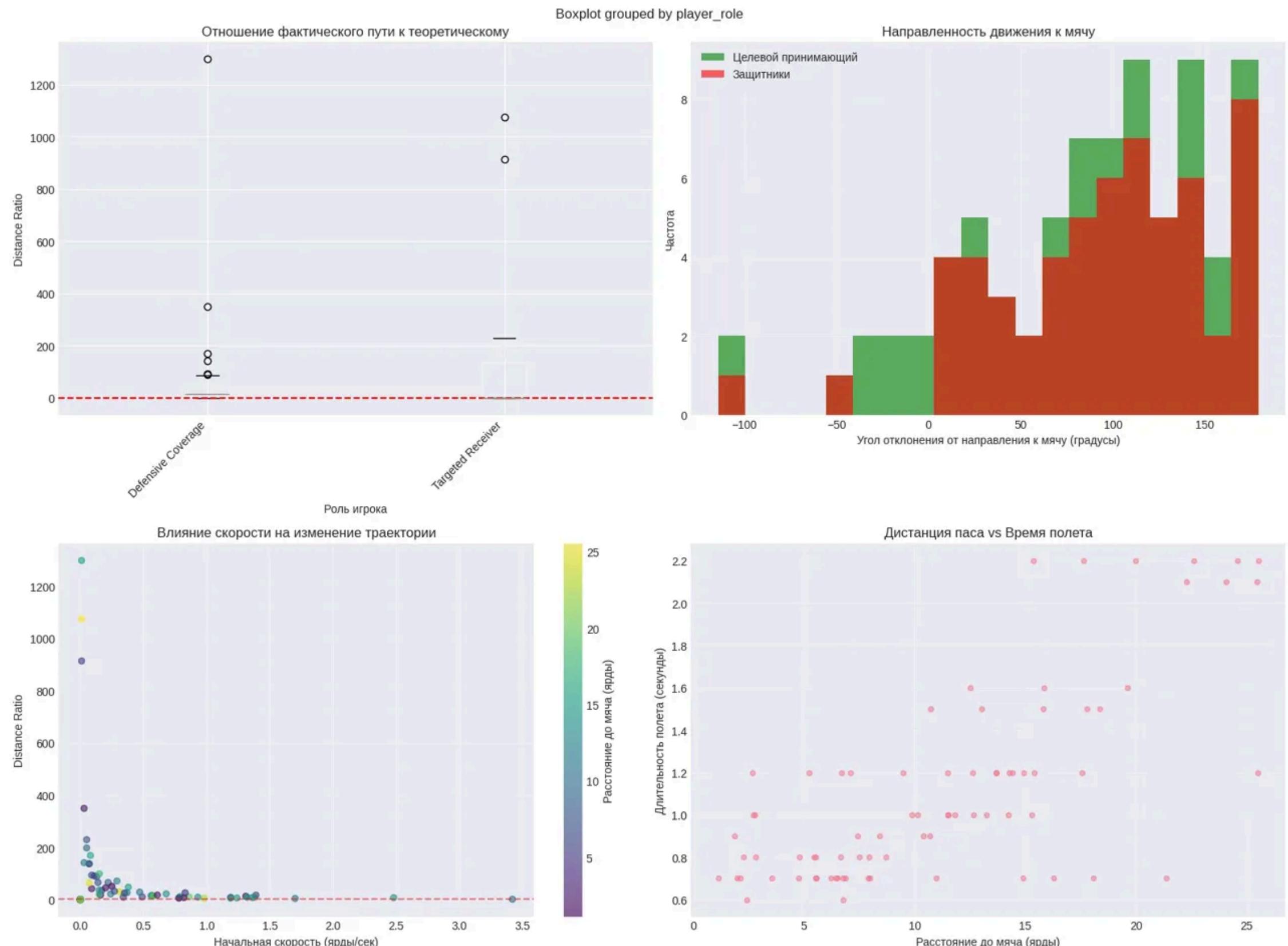
Обнаруженная проблема:

- Расчётное отношение "реальное расстояние / теоретически возможное" = 77 (должно быть 1)
- Причина: когда игрок почти стоит на месте (скорость = 0), формула даёт бессмысленные числа

Ещё одна находка:

- Игроки НЕ бегут прямо к мячу – отклонение 87-94 градуса
- Это нормально: они бегут по заранее заученным маршрутам

Решение: Отметить выбросы флагами, но не удалять – это реальные игровые ситуации





Какие признаки мы создали

Группа признаков (всего более 40)	Что это значит	Примеры
Расстояния	Как далеко игрок от мяча	Расстояние до точки приземления, отдельно по X и Y
Скорости	Как быстро и куда движется	Скорость вправо/влево, вверх/вниз по полю
Углы	Куда смотрит и куда бежит	Угол между направлением игрока и мячом
Время	Сколько времени до приземления	Оценка времени, чтобы добраться до мяча
Позиция на поле	Где именно находится	Близость к боковой линии, к зачётной зоне
Роль игрока	Кто он в розыгрыше	Это принимающий? Защитник? Квотербек?
Физические данные	Характеристики игрока	Рост, вес, индекс массы тела



Первая модель (Baseline)

Результаты:

Что сравниваем	RMSE (ошибка в ярдах)
Наивный подход (просто продолжить движение)	5.2 ярда
Наша модель XGBoost	3.886 ярда
Улучшение	-25%

Что мы построили:

- Две модели машинного обучения XGBoost (градиентный бустинг на деревьях)
- Одна предсказывает координату X (вдоль поля)
- Вторая предсказывает координату Y (поперёк поля)

Настройки модели:

- 300 деревьев в ансамбле
- Глубина каждого дерева: 8 уровней
- Скорость обучения: 0.05 (учимся осторожно, чтобы не переобучиться)

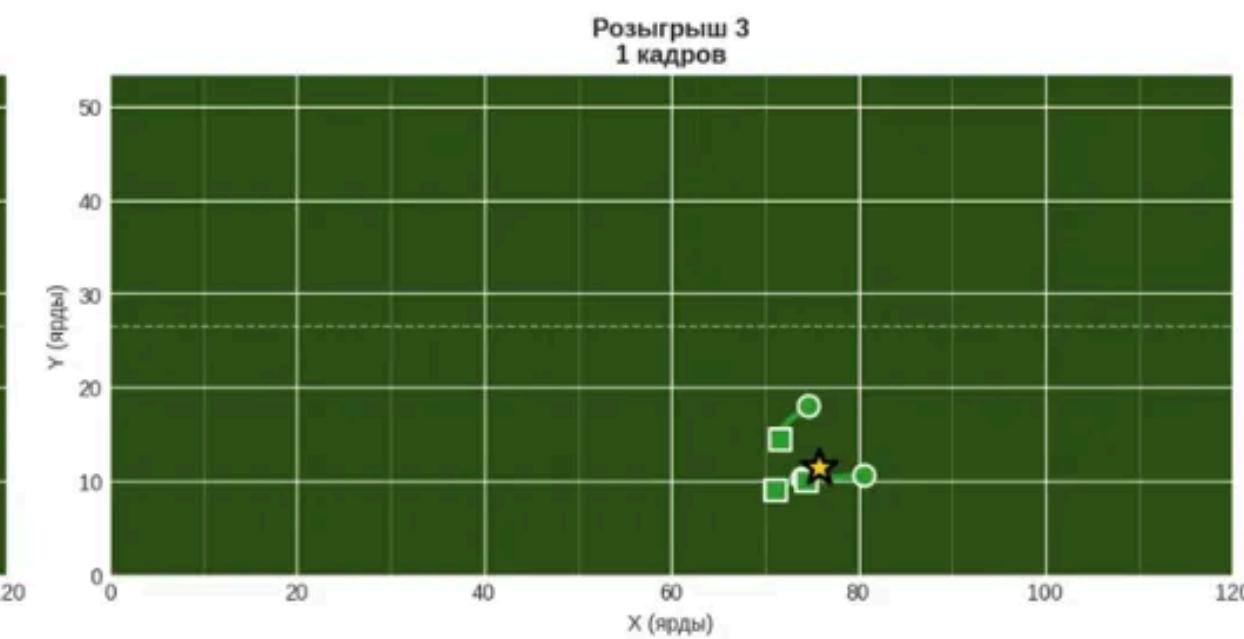
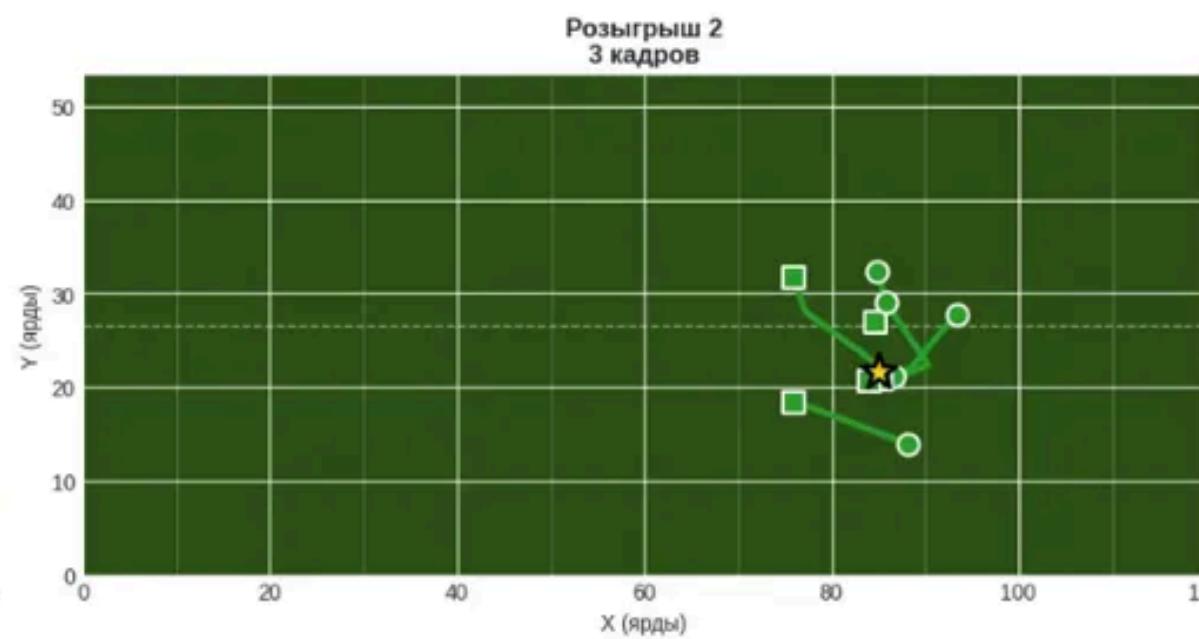
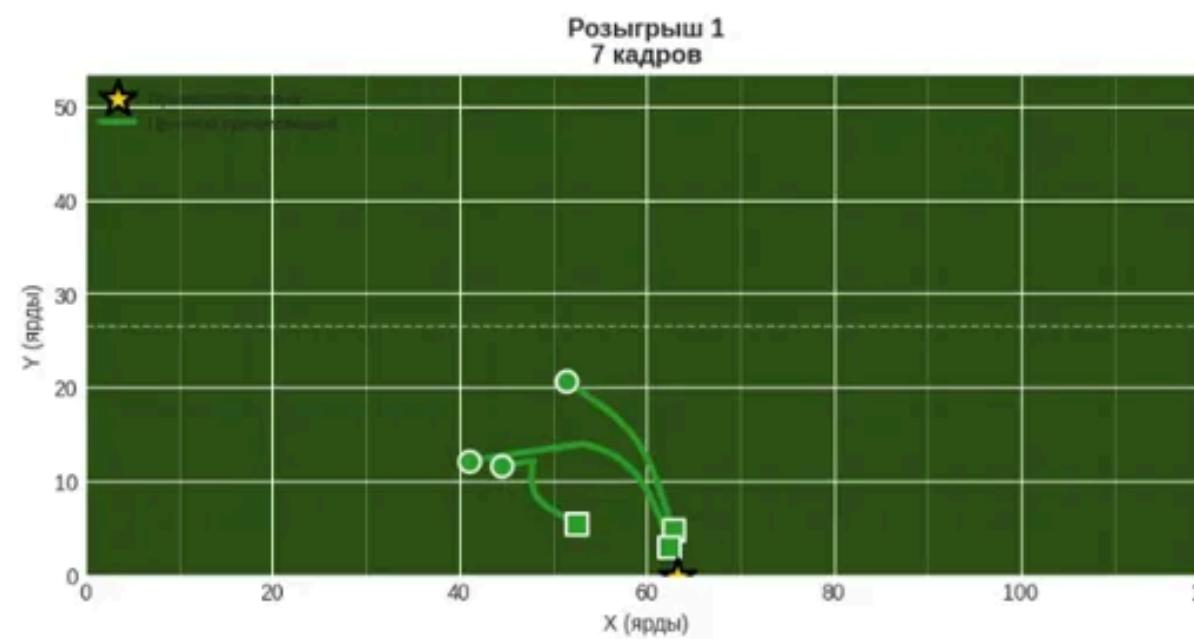


Как выглядят траектории

Что изображено на графиках: зелёные линии — пути, по которым двигались игроки, жёлтая звезда — точка, куда приземлился мяч, круги — где игроки начали движение, квадраты — где игроки оказались в конце

Три примера:

1. Розыгрыш на 7 кадров — стандартный короткий пас
2. Розыгрыш на 3 кадра — очень быстрый пас
3. Розыгрыш на 1 кадр — мгновенная передача



Итоги первого этапа



Что мы сделали:

- Изучили данные: 4.9 миллиона записей о движении игроков
- Нашли закономерности: кто как двигается в зависимости от роли
- Создали 40+ полезных признаков для модели
- Построили первую модель с ошибкой 3.886 ярда



Главные открытия:

- Целевой принимающий предсказуем – он уже бежит к мячу
- Защитники реагируют медленнее и не бегут напрямую
- Расстояние до мяча – самый важный признак
- В данных есть выбросы, но их нельзя просто удалить



Этап 2 – Цели и задачи

Что мы делаем на этом этапе:

1. Ищем странные данные – выбросы и аномалии
2. Решаем, что с ними делать: удалить или оставить
3. Создаём новые признаки, которые помогут модели лучше предсказывать
4. Используем умные алгоритмы для поиска скрытых аномалий

Почему это важно именно для футбольных данных:

- Игрок может резко ускориться или затормозить – это не ошибка, а реальная игра
- Необычные позиции (у самой боковой линии) – редкие, но важные ситуации
- Нужен баланс: не потерять ценную информацию, но и не испортить модель шумом

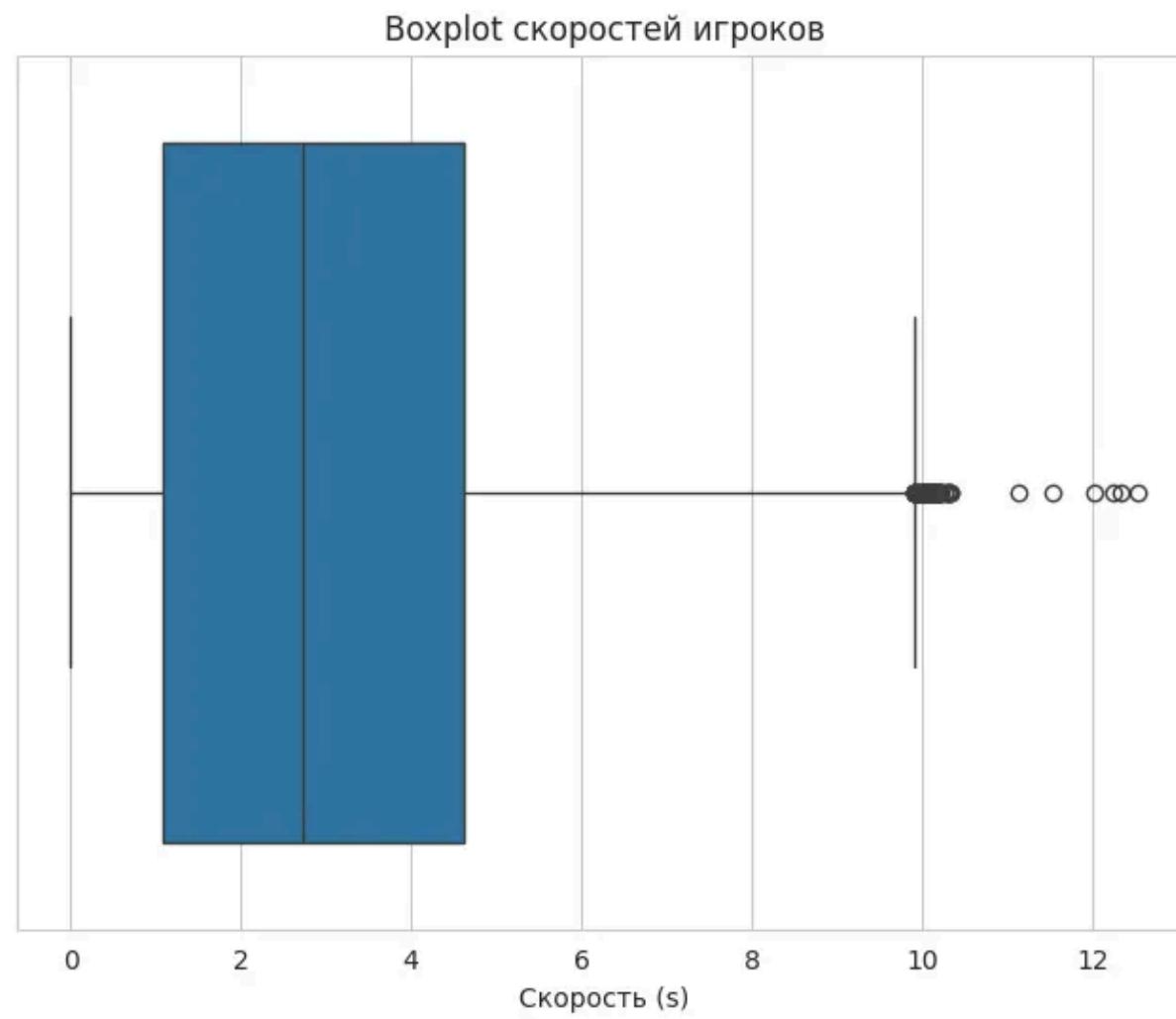


Как мы искали выбросы – статистические методы



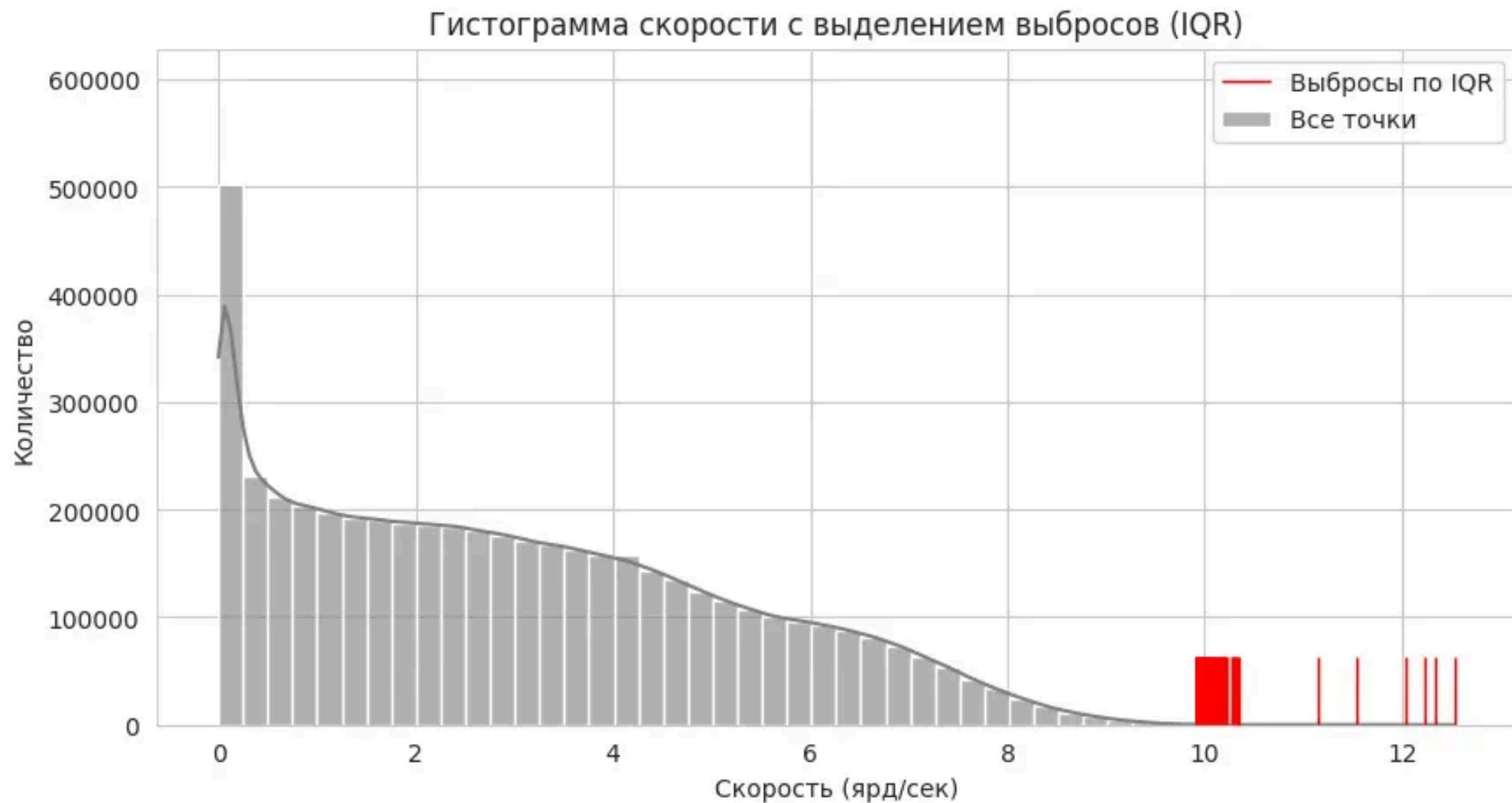
Метод 1: Z-score (Z-оценка)

- Что это: Смотрим, насколько значение отличается от среднего
- Как работает: Если скорость игрока отличается от средней более чем на 3 стандартных отклонения – это выброс
- Результат: Нашли 546 подозрительных записей по скорости



Метод 2: IQR (Межквартильный размах)

- Что это: Смотрим на "типичный диапазон" значений (средние 50% данных)
- Как работает: Всё, что сильно выше или ниже этого диапазона – выброс
- Результат: Нашли 166 подозрительных записей (более строгий метод)

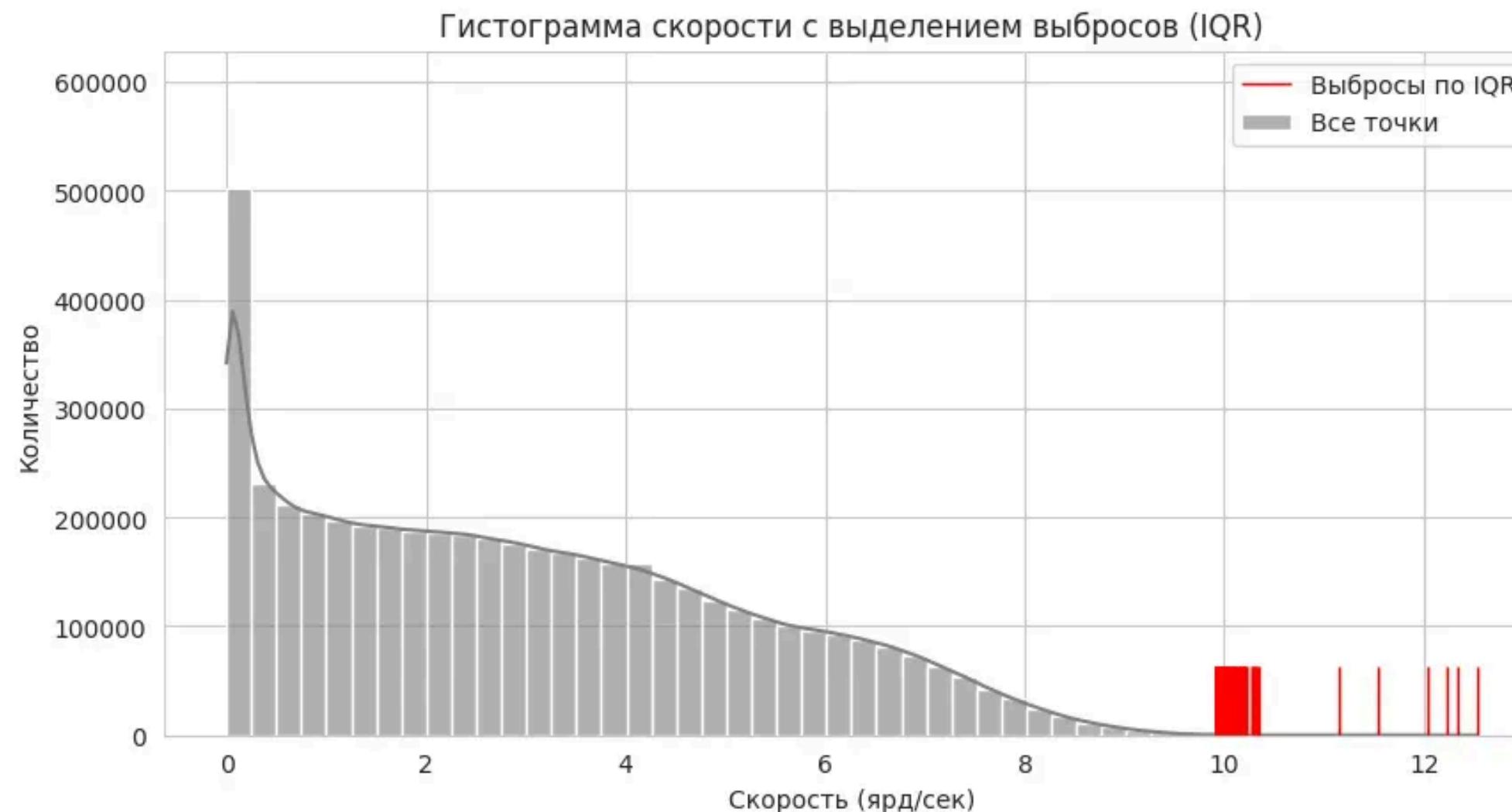


Что означают эти выбросы в реальности



Переводим цифры в реальный мир:

- Скорость > 10 ярдов/сек = 33 км/ч – это спринт профессионального атлета
- Для сравнения: Усэйн Болт бегает ~44 км/ч (мировой рекорд)
- NFL игроки реально могут развивать ~35-40 км/ч на коротких дистанциях



Что проверяли	Сколько выбросов	Доля от всех данных
Скорость бега	546	0.011%
Ускорение	1,200	0.025%
Позиция по X	200	0.004%
Позиция по Y	150	0.003%

Вывод: Выбросов очень мало (меньше 0.1%), но они могут нести важную информацию

Наше решение – НЕ удалять выбросы! ●●●

Почему мы решили сохранить странные данные:

Это реальный футбол:

- Игрок может резко рвануть за мячом – это не ошибка датчика
- Столкновения игроков создают непредсказуемые траектории
- Самые важные моменты игры часто и есть "экстремальные"

Модель должна уметь предсказывать редкие случаи:

- Если мы уберём все необычные ситуации из обучения – модель их не поймёт
- А на реальных данных такие ситуации будут встречаться

Наш подход – пометить, а не удалить:

- Создаём специальный флаг "это выброс" для каждой записи
- Модель сама решит, как использовать эту информацию

Результат: Создали флаги-метки для 6 разных характеристик игроков

Умный поиск аномалий – Isolation Forest



Что такое Isolation Forest:

- Это алгоритм машинного обучения, который сам находит странные точки
- Идея: аномалии легче изолировать от остальных данных
- Не нужно задавать правила вручную – алгоритм сам их находит

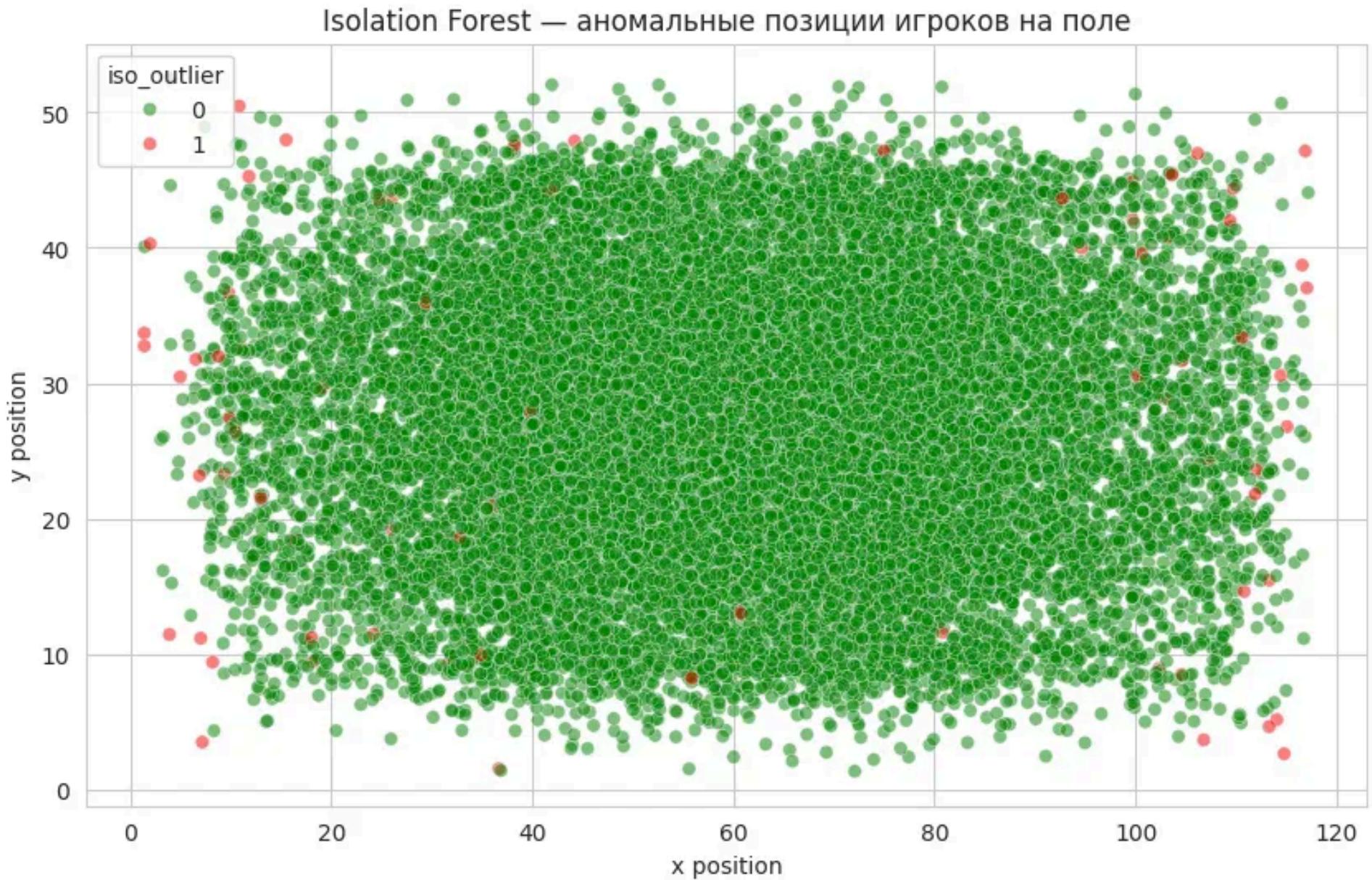
Что мы ему дали на вход:

- Скорость, ускорение, позицию на поле, направление движения

Что он нашёл (смотрим на картинку):

- Зелёные точки – обычные позиции игроков
- Красные точки – аномалии

Главное открытие: Аномалии – это игроки у самых краёв поля!



Почему игроки у края поля — это важно



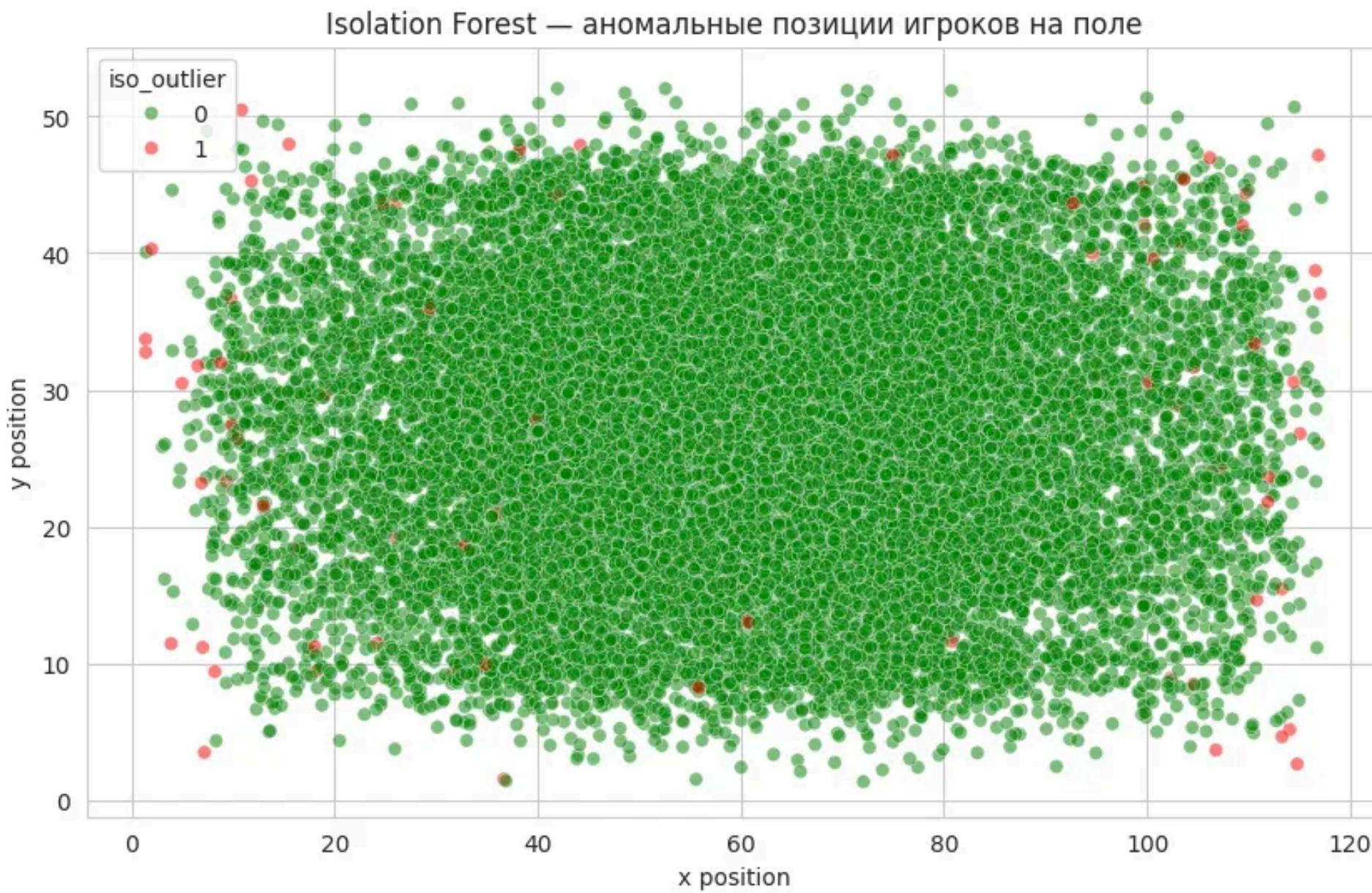
Где алгоритм нашёл аномалии:

- У боковых линий (игрок прижат к краю поля)
- В зачётных зонах (эндзонах)

Почему это важно для предсказаний:

- Игровой у боковой линии - может двигаться только в одну сторону — в поле
- Игровой в эндзоне - другая тактика, другие цели
- Игровой в углу поля - самая ограниченная свобода манёвра

Решение: добавляем признак "игровой в аномальной позиции" — модель будет это учитывать



Новые признаки на основе аномалий



Для каждой характеристики игрока (скорость, ускорение, позиция и т.д.) создали три флага:

Дополнительные признаки:

Флаг	Что означает
Выброс по Z-score	Значение сильно отличается от среднего
Выброс по IQR	Значение выходит за типичный диапазон
Любой выброс	Сработал хотя бы один из методов

Название	Что показывает
Общий выброс	Игрок аномален хоть по какому-то параметру
Аномалия по Isolation Forest	Алгоритм ML считает эту точку странной
Расстояние перемещения	Как далеко игрок переместился за время полёта мяча

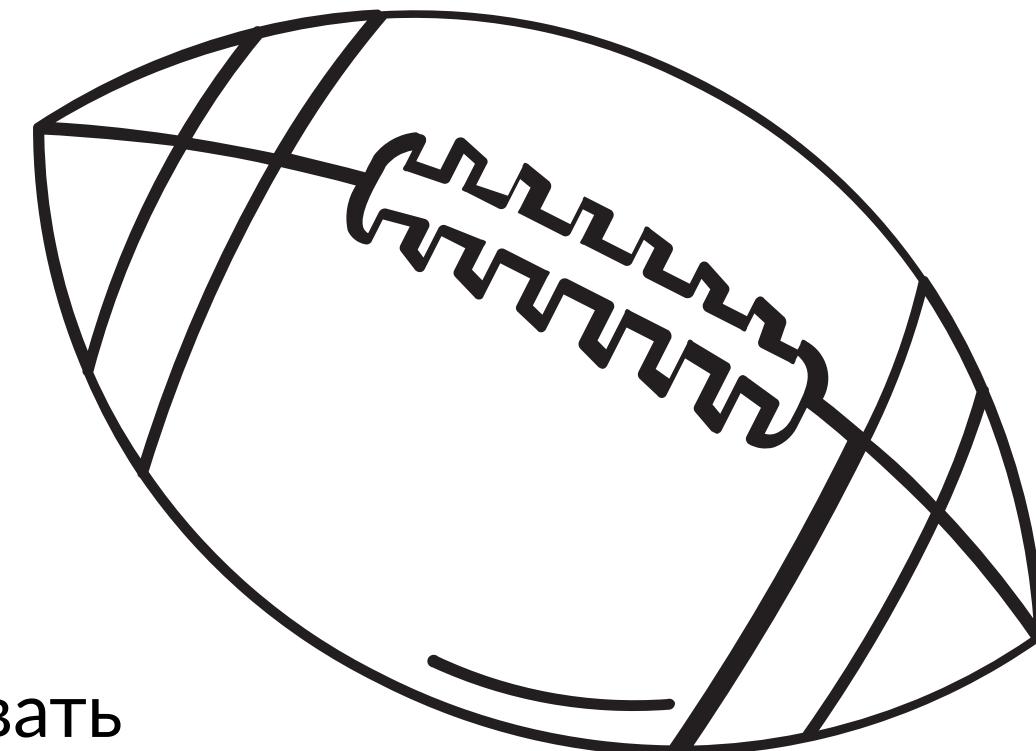
Итого: +18 новых признаков, описывающих необычность каждой записи

Признаки на основе соседей (KNN)



Идея:

Важно не только где находится игрок, но и кто рядом с ним



Признак 1: Плотность окружения

- Среднее расстояние до 5 ближайших игроков
 - Маленькое значение → игрок в толпе, сложно маневрировать
 - Большое значение → игрок на свободном пространстве

Признак 2: Расстояние до ближайшего защитника

- Как далеко от принимающего находится ближайший защитник
 - Маленькое значение → принимающего плотно опекают, сложно поймать мяч
 - Большое значение → принимающий "открыт", хорошие шансы на приём

Признаки, связанные со временем



Признак 1: Относительное время в розыгрыше

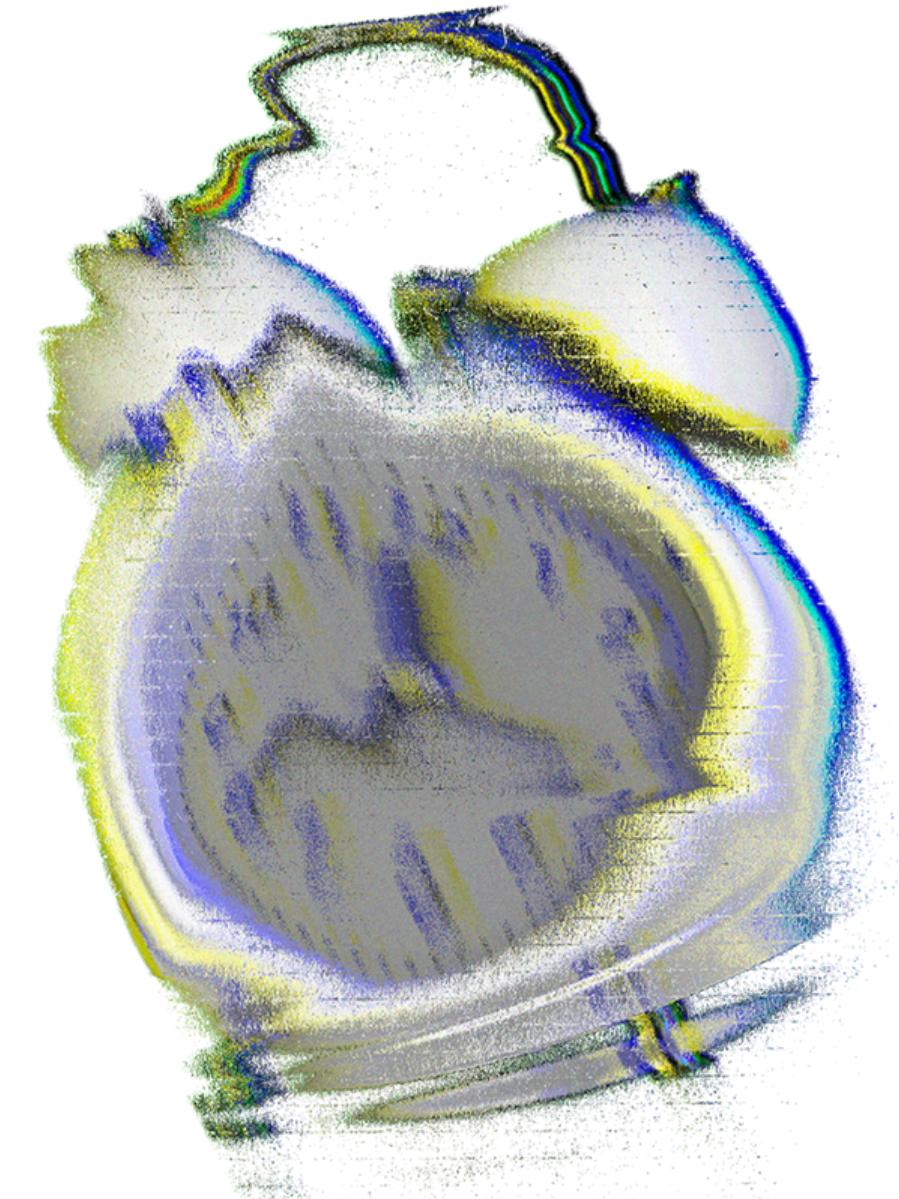
- Что это: На каком этапе полёта мяча мы находимся (от 0 до 1)
- Зачем: В начале полёта игроки ещё реагируют, в конце – уже у мяча

Признаки 2-3: Циклическое кодирование времени (\sin/\cos)

- Проблема: Для модели значения 0.99 и 0.01 кажутся далёкими
- Решение: Кодируем время как точку на круге
- Результат: Модель понимает, что конец одного цикла близок к началу следующего

Зачем это нужно:

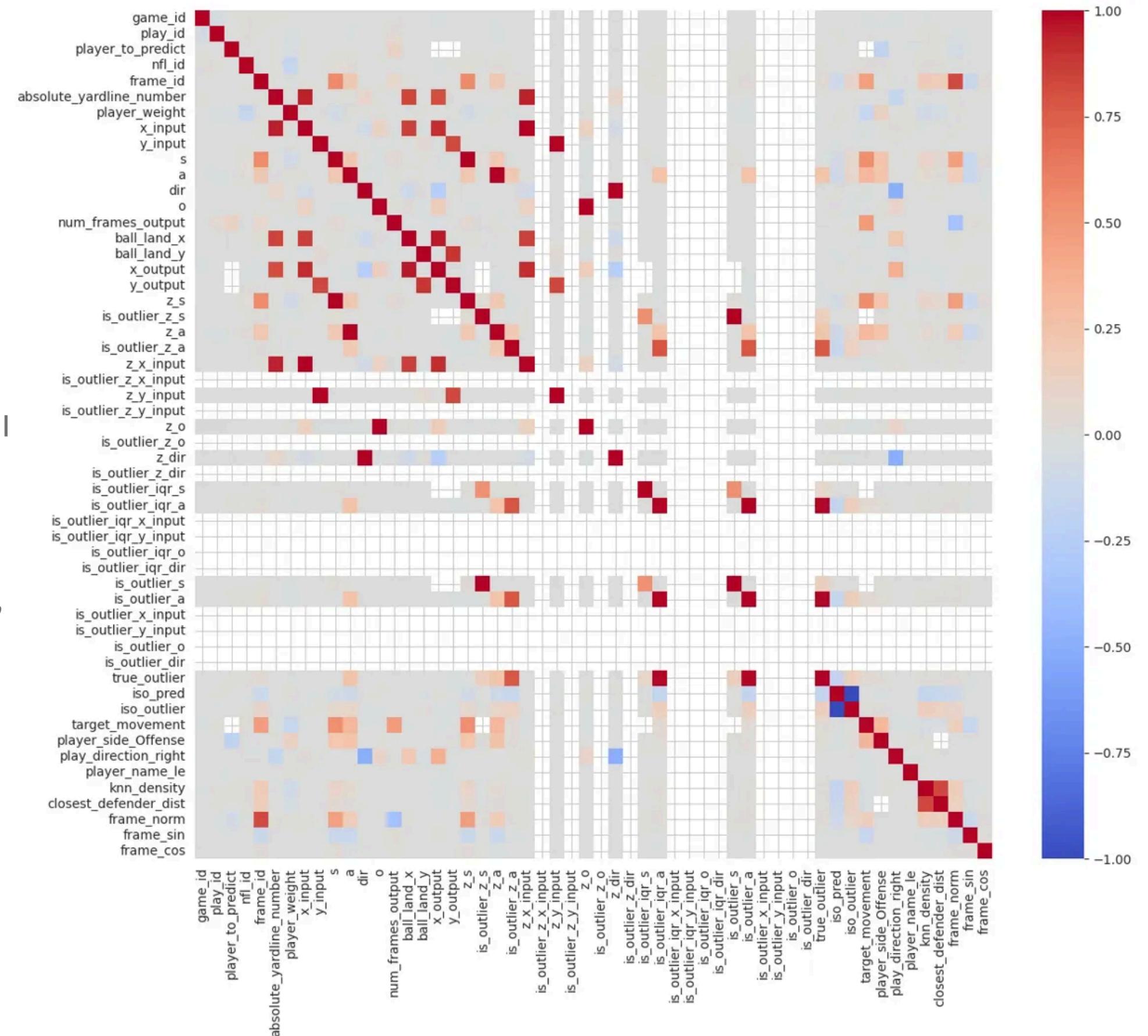
- Движение игроков имеет определённые фазы
- Модель лучше понимает динамику, когда время закодировано правильно



Проверяем связи между признаками

Что мы увидели:

- Сильные связи (хорошо – логичные зависимости):
 - Позиция игрока связана с местом приземления мяча (0.88)
 - Начальная и конечная позиции связаны (игроки не телепортируются)
- Новые признаки-аномалии:
 - Слабо связаны с остальными – значит, несут новую информацию!
 - Это хорошо: мы не дублируем то, что уже знаем
- Признаки соседей и времени:
 - Добавляют уникальную информацию
 - Не копируют существующие признаки

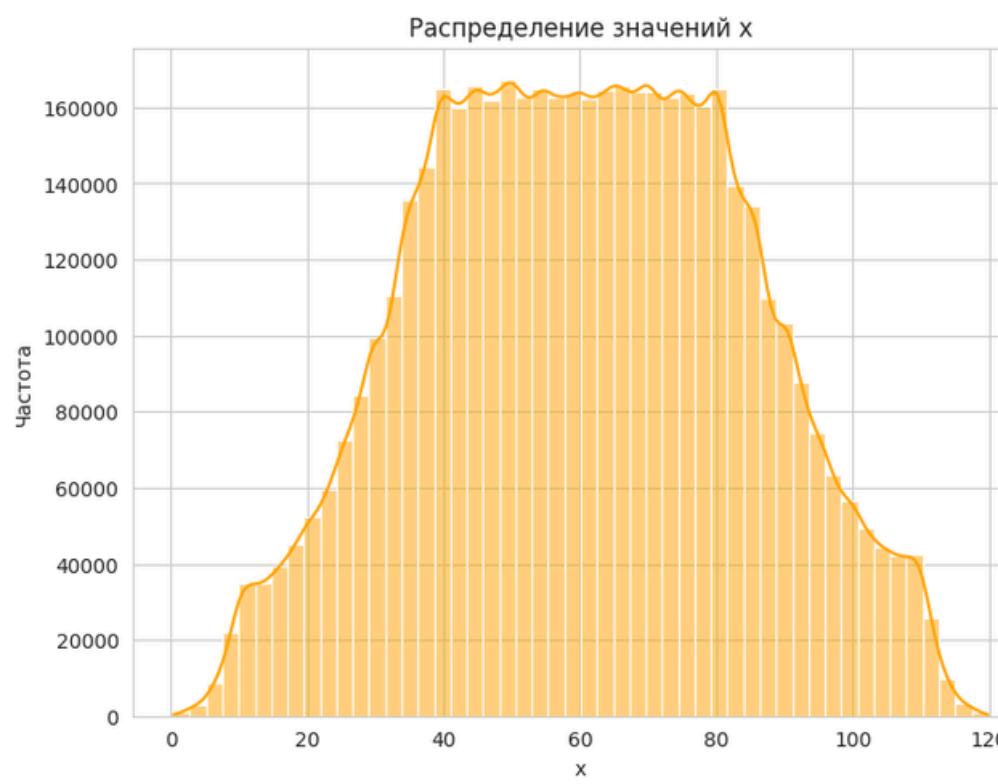


Как распределены позиции игроков



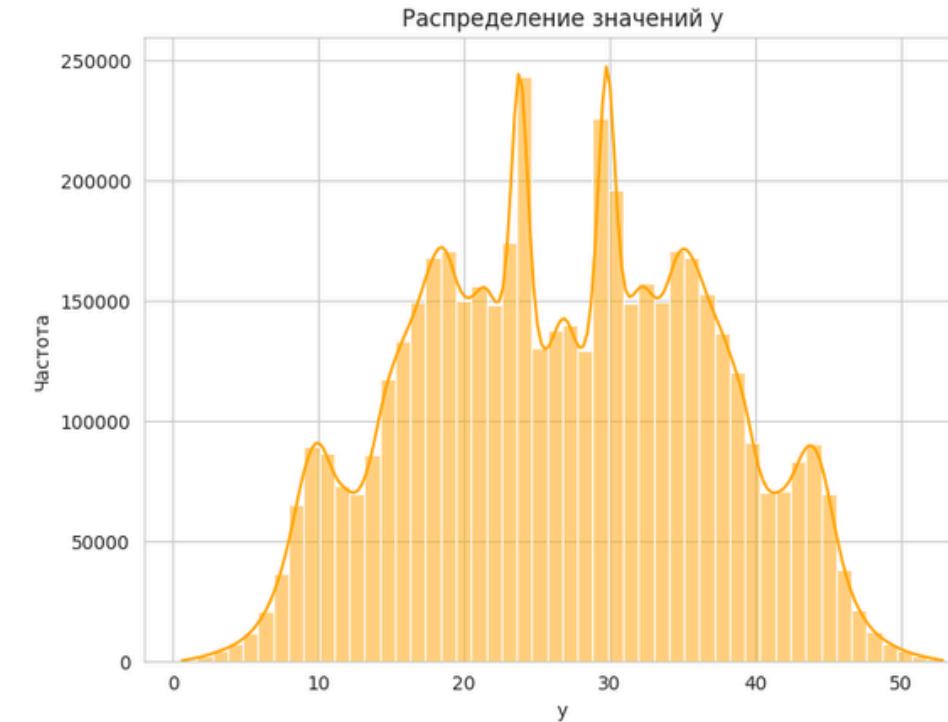
По длине поля (координата X):

- Распределение похоже на колокол (нормальное)
- Больше всего данных в центре поля (40-80 ярдов)
- Мало данных у зачётных зон – там редко происходят пасы

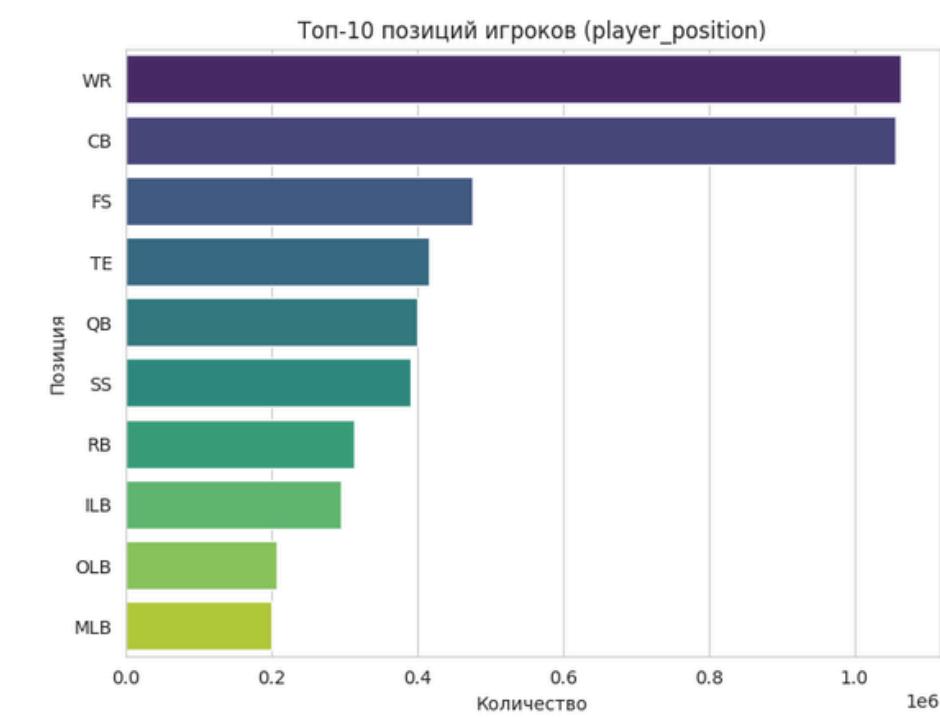
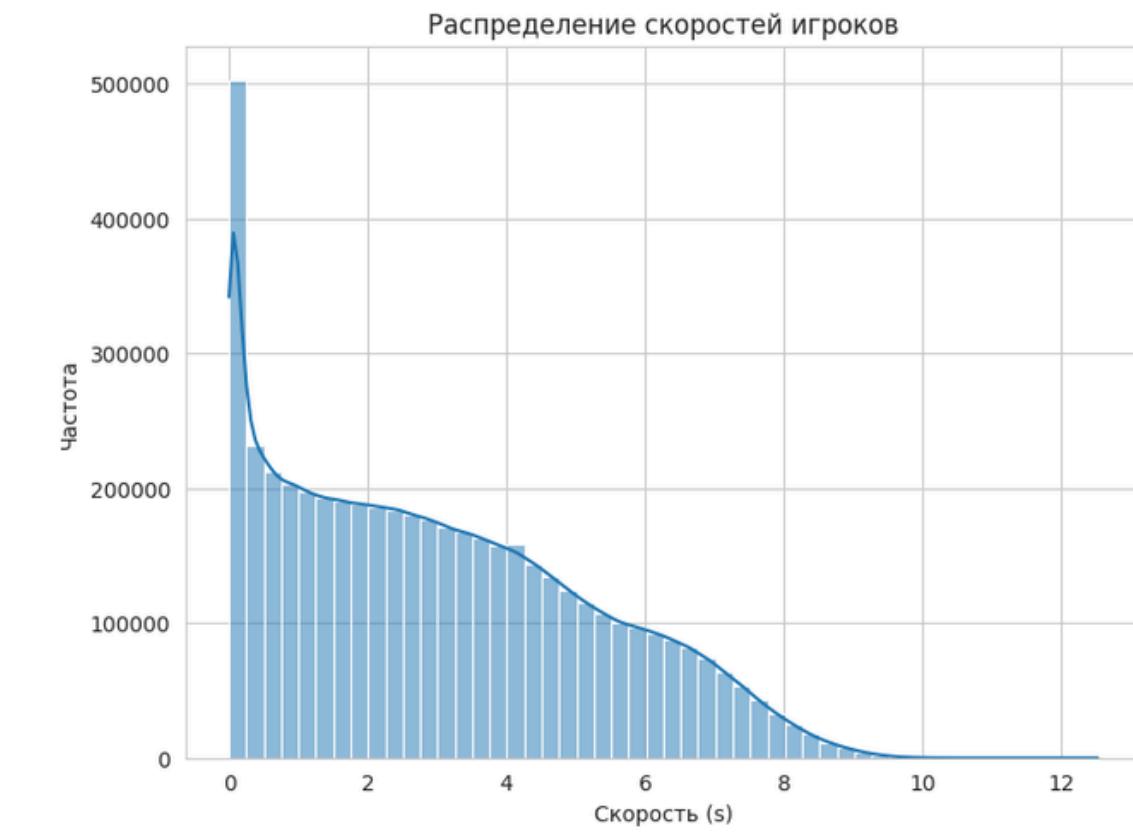


По ширине поля (координата Y):

- Интересная находка: два пика!
- Пики на ~20 и ~35 ярдах
- Это соответствует типичным построениям команд (хэш-марки на поле)



Вывод: Модель должна учитывать, что Y-координата имеет сложную структуру



Итоговый набор признаков после этапа 2



Группа признаков (всего около 50)	Количество	Что описывают
Исходные данные	10	Позиция, скорость, направление
Расстояния	5	Как далеко от мяча
Скорости	4	Куда и как быстро движется
Временные	4	На каком этапе розыгрыш
Аномалии (новые!)	18	Насколько ситуация необычна
Соседи (новые!)	2	Кто рядом с игроком
Роли игроков	6	Кто этот игрок в розыгрыше



Выводы по Этапу 2

Что мы сделали:

- Нашли выбросы тремя способами:
 - Статистика (Z-score): 546 необычных скоростей
 - Статистика (IQR): 166 выбросов
 - Машинное обучение (Isolation Forest): аномалии у краёв поля
- Приняли решение: не удалять, а помечать
 - Выбросы – это реальные игровые ситуации
 - Модель получит флаги и сама решит, как их использовать
- Создали 23 новых признака:
 - 18 флагов аномалий
 - 2 признака про соседей (плотность, расстояние до защитника)
 - 3 временных признака





Этап 3 – Цели и задачи

01

Понимаем, КАК модель принимает решения (глобальная интерпретация)

02

Разбираем конкретные предсказания (локальная интерпретация)

03

Превращаем объяснения модели в новые признаки

04

Группируем похожие ситуации (кластеризация)

05

Ищем лучшие настройки модели (оптимизация)



Какие модели мы обучили

Как комбинируем модели:

- Ridge вносит всего 3% (слишком простая)
- LightGBM – 48%
- CatBoost – 49%

Модель	Что это	RMSE	Зачем нужна
Ridge	Линейная регрессия	4.02 ярда	Простая, легко понять
LightGBM	Ансамбль деревьев решений	3.52 ярда	Быстрая, точная
CatBoost	Ансамбль деревьев решений	3.53 ярда	Хорошо работает с категориями
Ensemble	Комбинация всех трёх	3.51 ярда	Объединяя лучшее

Как понять, что важно для модели — метод SHAP



Что такое SHAP:

- Метод, который показывает вклад каждого признака в предсказание
- Чем больше вклад — тем важнее признак для модели

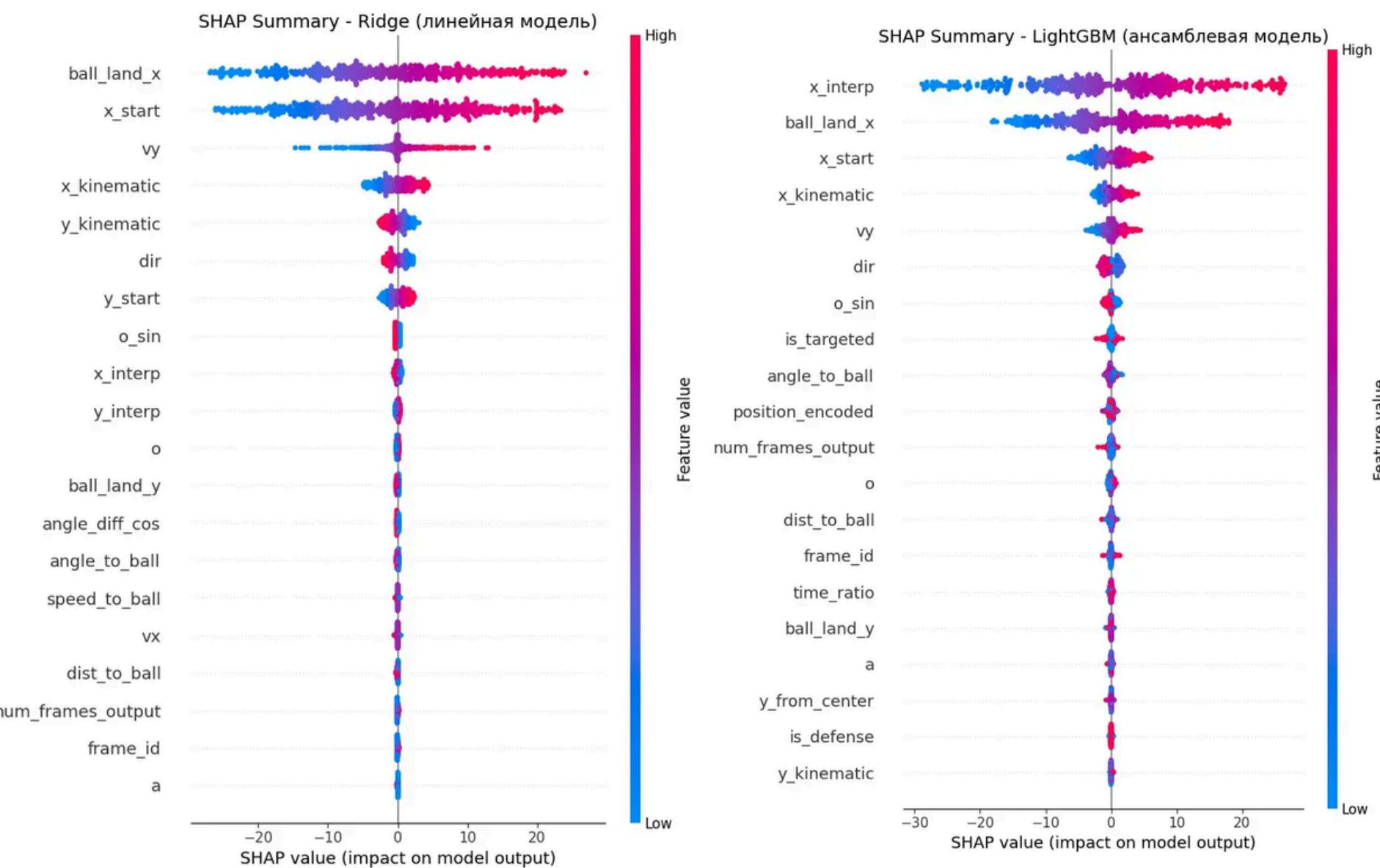
Что важно для простой модели (Ridge):

1. Место приземления мяча (куда летит мяч)
2. Начальная позиция игрока (где он стоял)
3. Скорость движения вбок

Что важно для сложной модели (LightGBM):

1. Расчётная позиция игрока (наш созданный признак!) — где игрок ДОЛЖЕН быть по физике
2. Место приземления мяча
3. Начальная позиция игрока

Важный вывод: Сложная модель научилась использовать наш рассчитанный признак лучше, чем сырье данные!

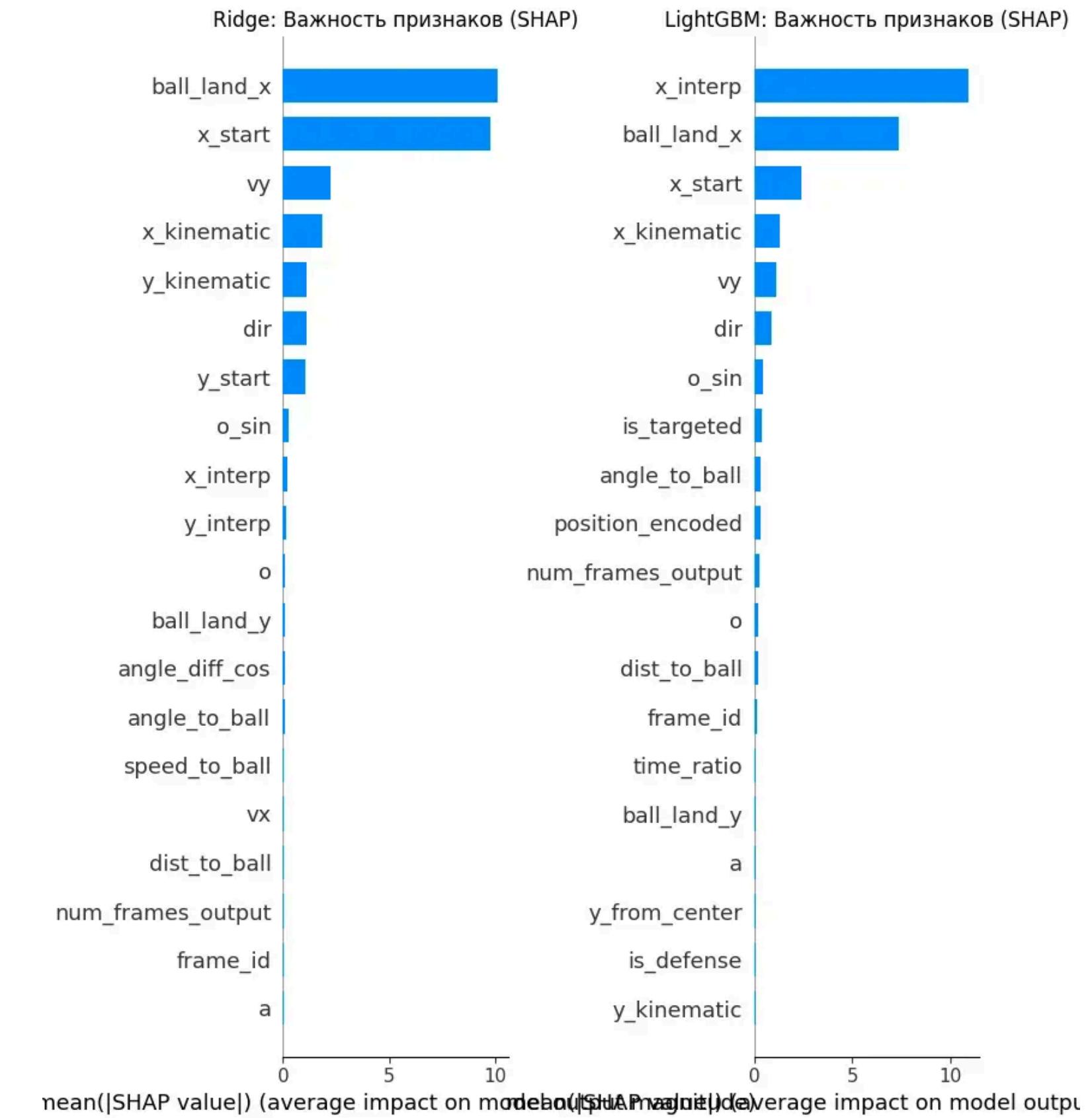


Сравнение важности признаков между моделями

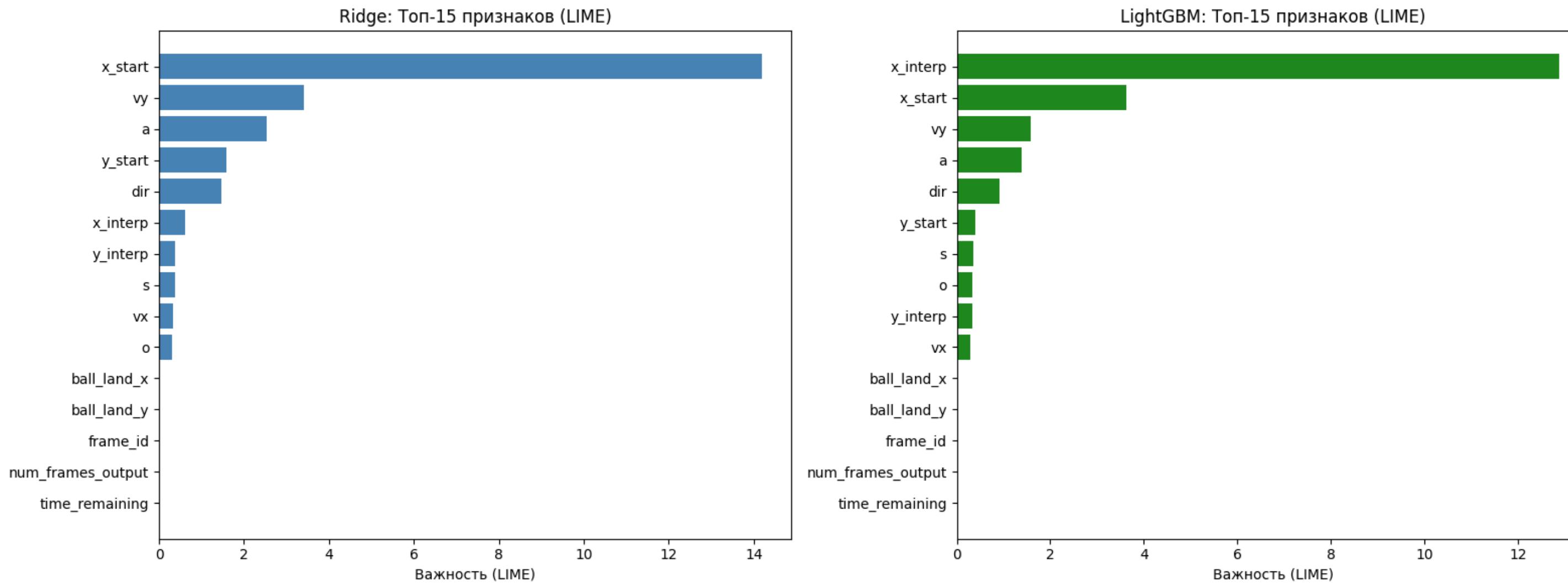


Признак	Простая модель	Сложная модель	Что значит
Место приземления мяча	1	2	Обе модели используют
Расчетная позиция	Низкий ранг	1	Сложная модель лучше использует
Физическая модель движения	Средний	4	Физика помогает
Флаг целевой принимающий	Низкий	8	Нелинейный эффект

Вывод: Сложные модели лучше извлекают информацию из рассчитанных признаков



Второй метод интерпретации — LIME



Результаты:

- LIME и SHAP дают похожие ответы (корреляция 0.65)
- Это хорошо: значит, наши выводы надёжны

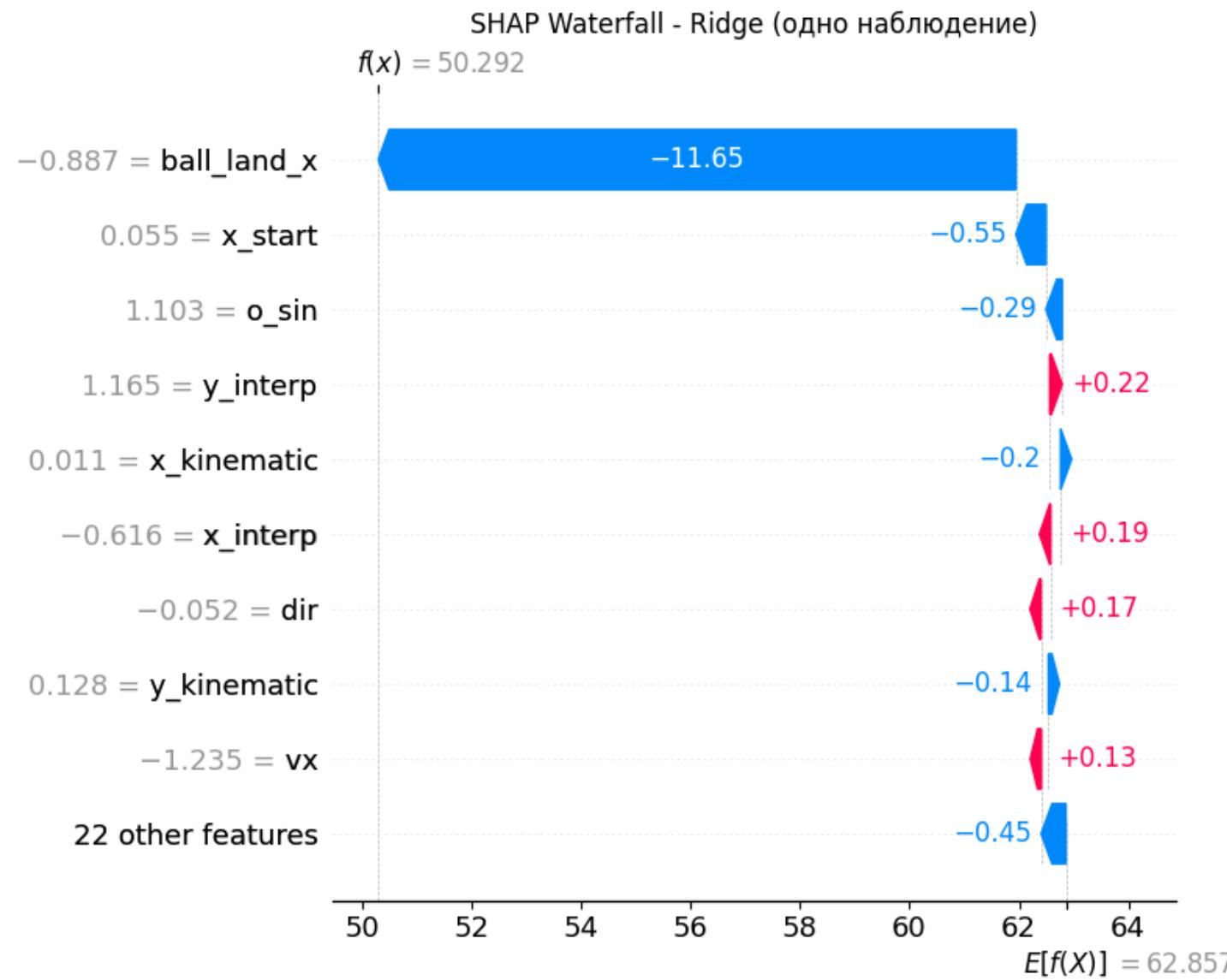
Что подтвердилось:

- Простая модель опирается на начальную позицию
- Сложная модель использует рассчитанные признаки

Разбор конкретного предсказания

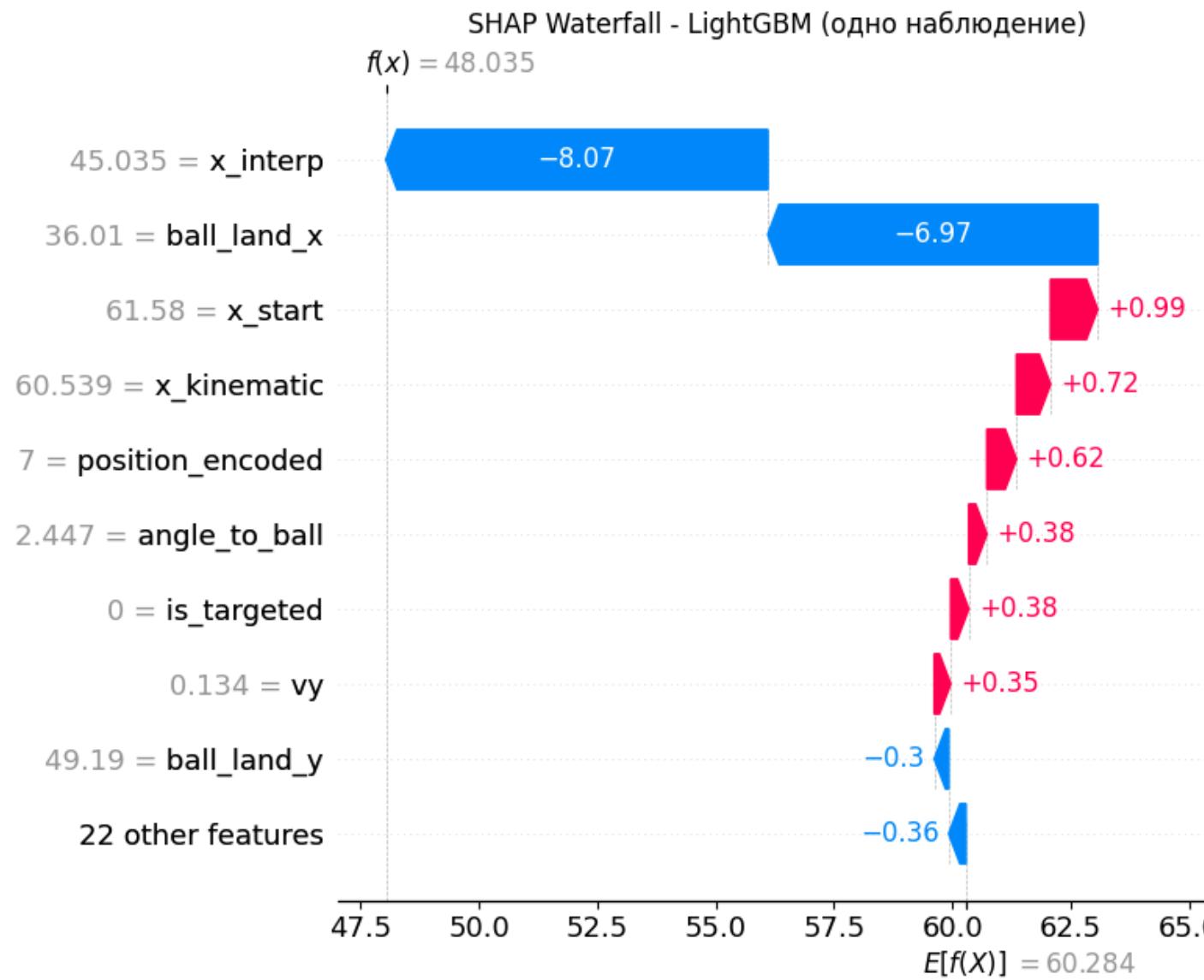


Вывод: Сложная модель распределяет влияние между несколькими факторами, а не полагается на один



Простая модель (Ridge):

- Базовое предсказание: 62.86 ярда
- Место приземления мяча сдвинуло предсказание на -11.65 ярда (главный фактор!)
- Итоговое предсказание: 50.29 ярда



Сложная модель (LightGBM):

- Базовое предсказание: 60.28 ярда
- Расчётная позиция сдвинула на -8.07 ярда
- Место приземления сдвинуло на -6.97 ярда
- Итоговое: 48.04 ярда

Превращаем объяснения в признаки (SHAP-эмбеддинги)



Признак	Средний вклад	Разброс вклада
Место приземления мяча	0	± 8.48
Начальная позиция	0	± 2.71
Направление движения	0	± 0.99

Что получилось:

- Обучающие данные:
44,834 игрока × 31
число
- Тестовые данные:
11,209 игроков × 31
число



Ищем аномалии в "объяснениях модели"

Вопрос: Есть ли игроки, которых модель "видит" совсем иначе?

Метод 1: Статистический (Z-score)

- Нашли 23.7% "необычных" объяснений в обучающих данных
- В тестовых данных – 23.6% (почти столько же!)

Метод 2: Алгоритм Isolation Forest

- Нашли 5% аномальных в обучающих
- В тестовых – 4.3%

Проверка стабильности (KS-тест):

- Сравнили распределения обучающих и тестовых данных
- Значимое различие только по 1 признаку из 31

Вывод: Модель работает одинаково на обучающих и тестовых данных – это хорошо!



Эксперимент – что если удалить аномалии?

Гипотеза: Может, странные наблюдения мешают модели?

Что сделали:

- Удалили 10,726 записей (24% данных), которые были аномальными
- Переобучили модель на оставшихся данных

Вариант	RMSE
До удаления	3.5197 ярда
После удаления	3.5245 ярда
Изменение	+0.14% (стало хуже!)

Вывод:
Удалять аномалии нельзя – они содержат полезную информацию о редких, но важных ситуациях!

Группируем похожие ситуации (кластеризация)



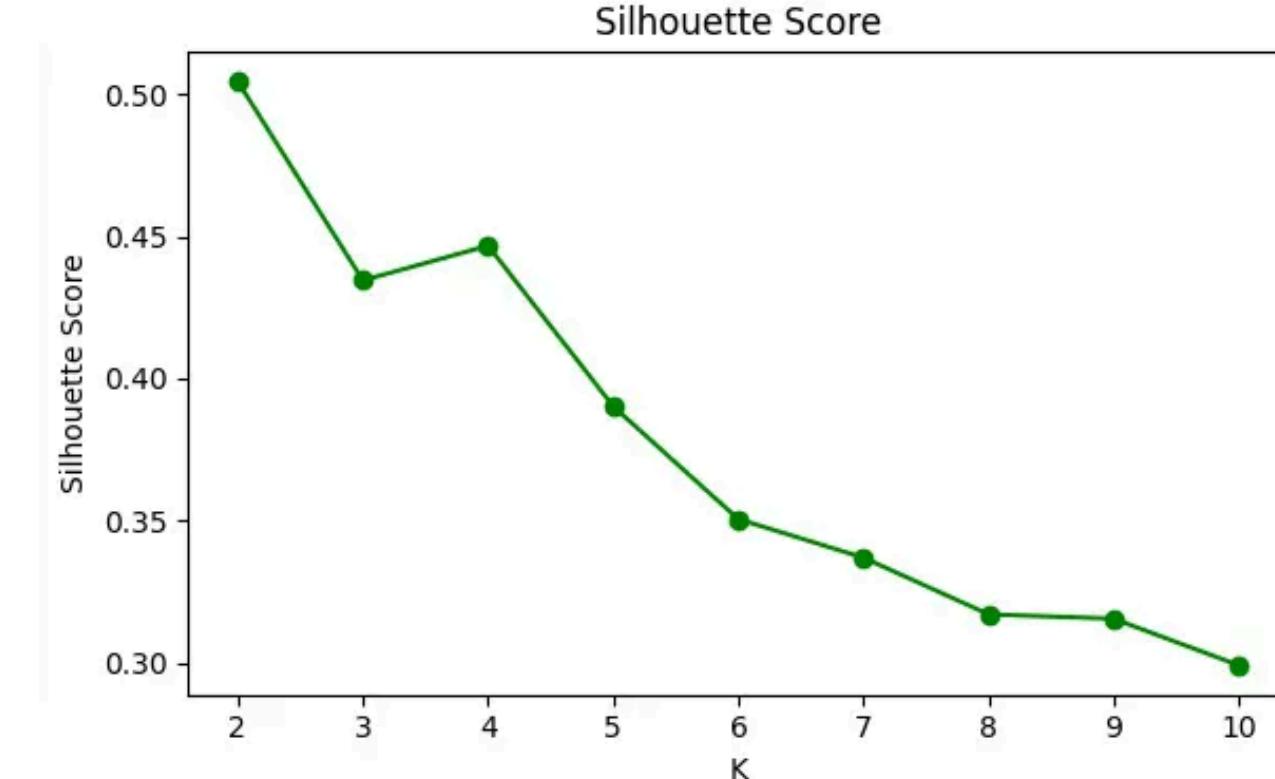
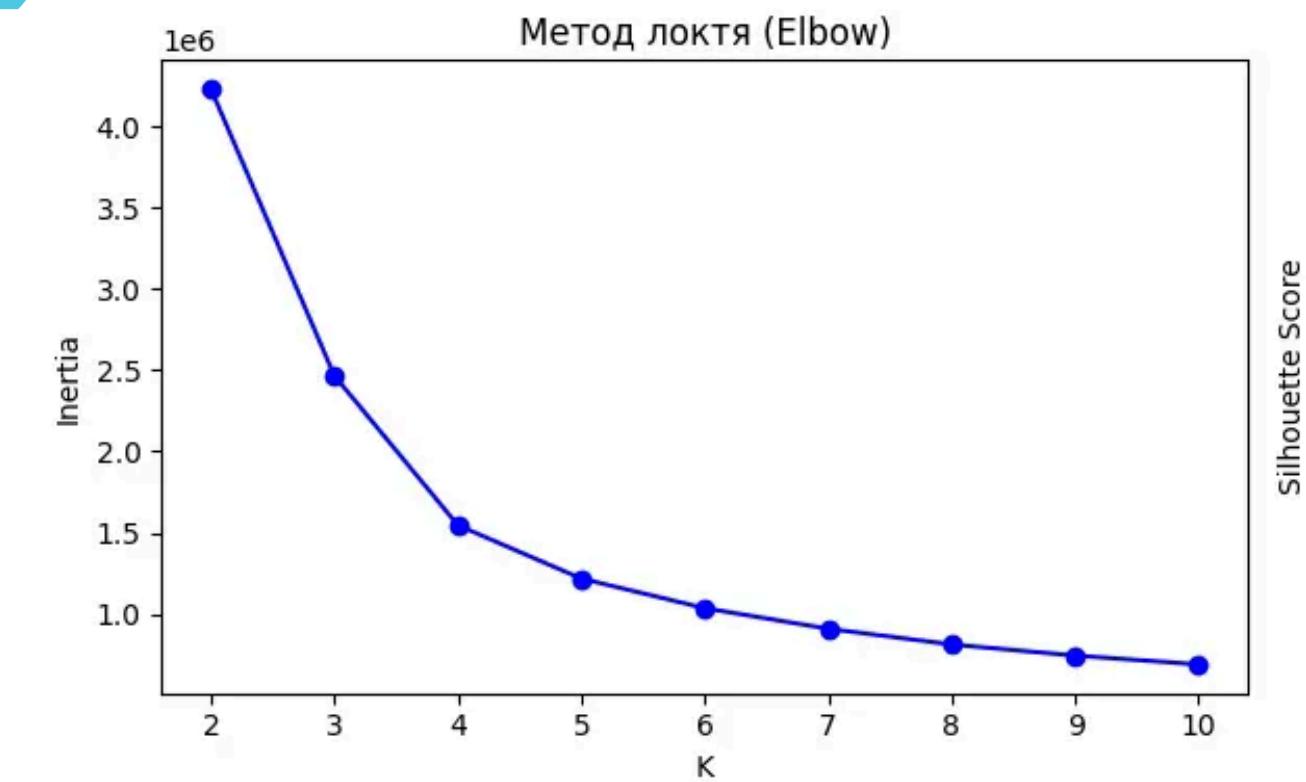
Идея: Может, есть группы игроков, которых модель видит похоже?

Как выбрали количество групп:

- Метод "локтя": резкий изгиб на 4 группах
- Метод силуэта: хороший показатель для 4 групп
- Выбрали: 4 группы

Метод	Результат
K-Means	4 группы
Иерархическая кластеризация	4 группы
DBSCAN	не сработал

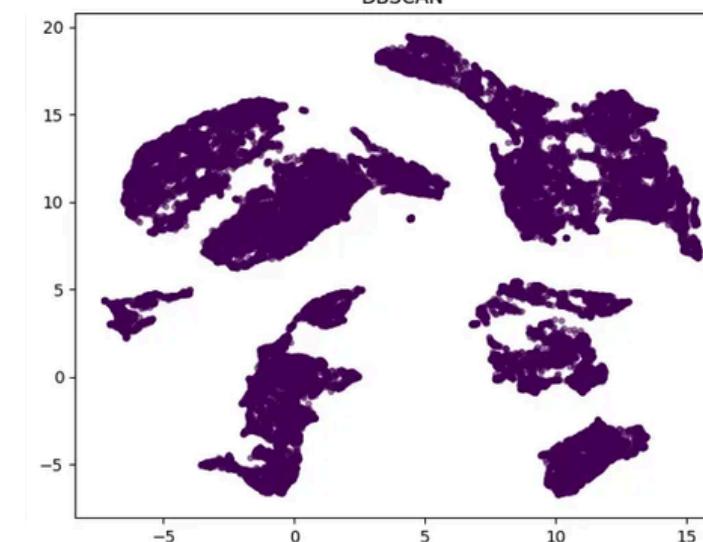
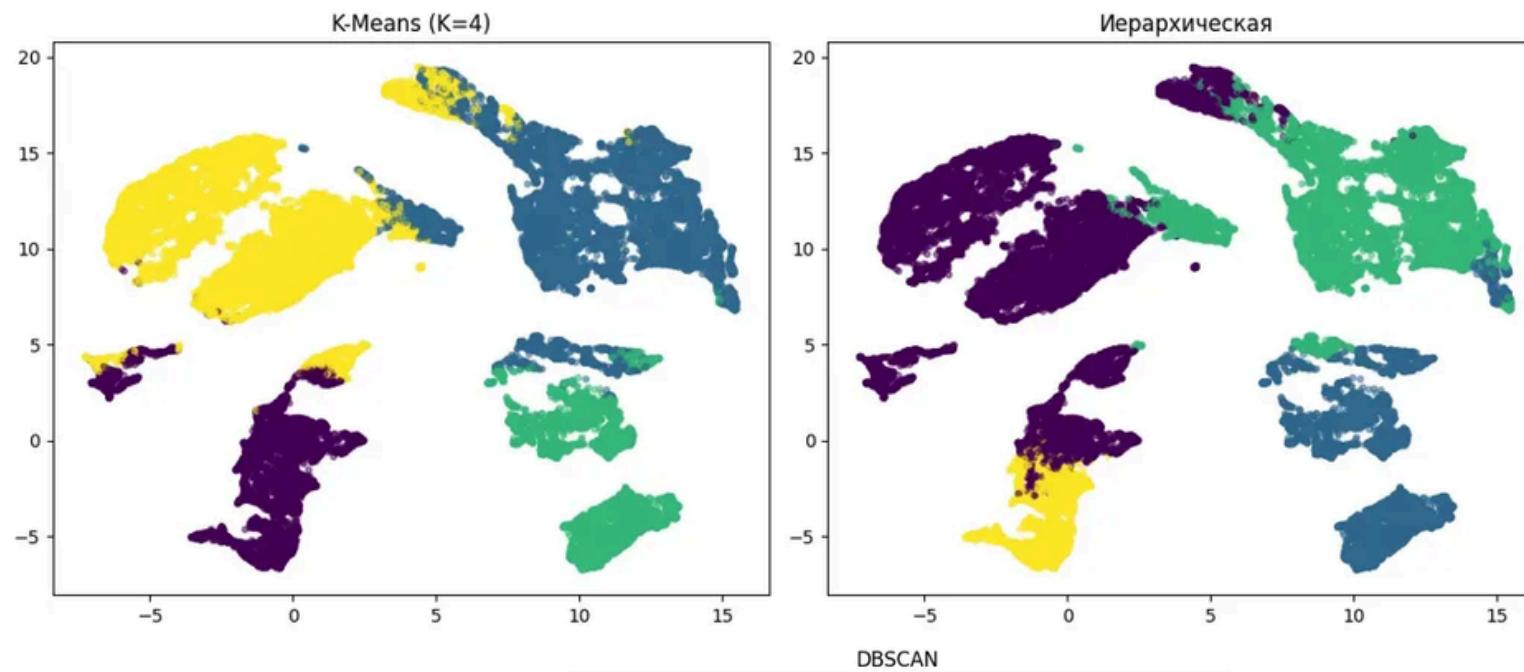
Почему DBSCAN не сработал: Данные слишком плотные, алгоритм не смог найти границы



Что означают эти группы?



Главное открытие: Модель автоматически разделила игроков по их позиции на поле!
Это логично: предсказывать движение у зачётной зоны и в центре поля — разные задачи



Группа	Размер, %	Где на поле (X), ярды	Интерпретация
0	16	92.9	Правая часть поля
1	34	48.3	Центр поля
2	15	27.5	Левая часть поля
3	35	71.1	Правее центра



Эксперимент — помогут ли группы как признак?

Идея: Добавим номер группы как новый признак — может, это поможет?

Что сделали:

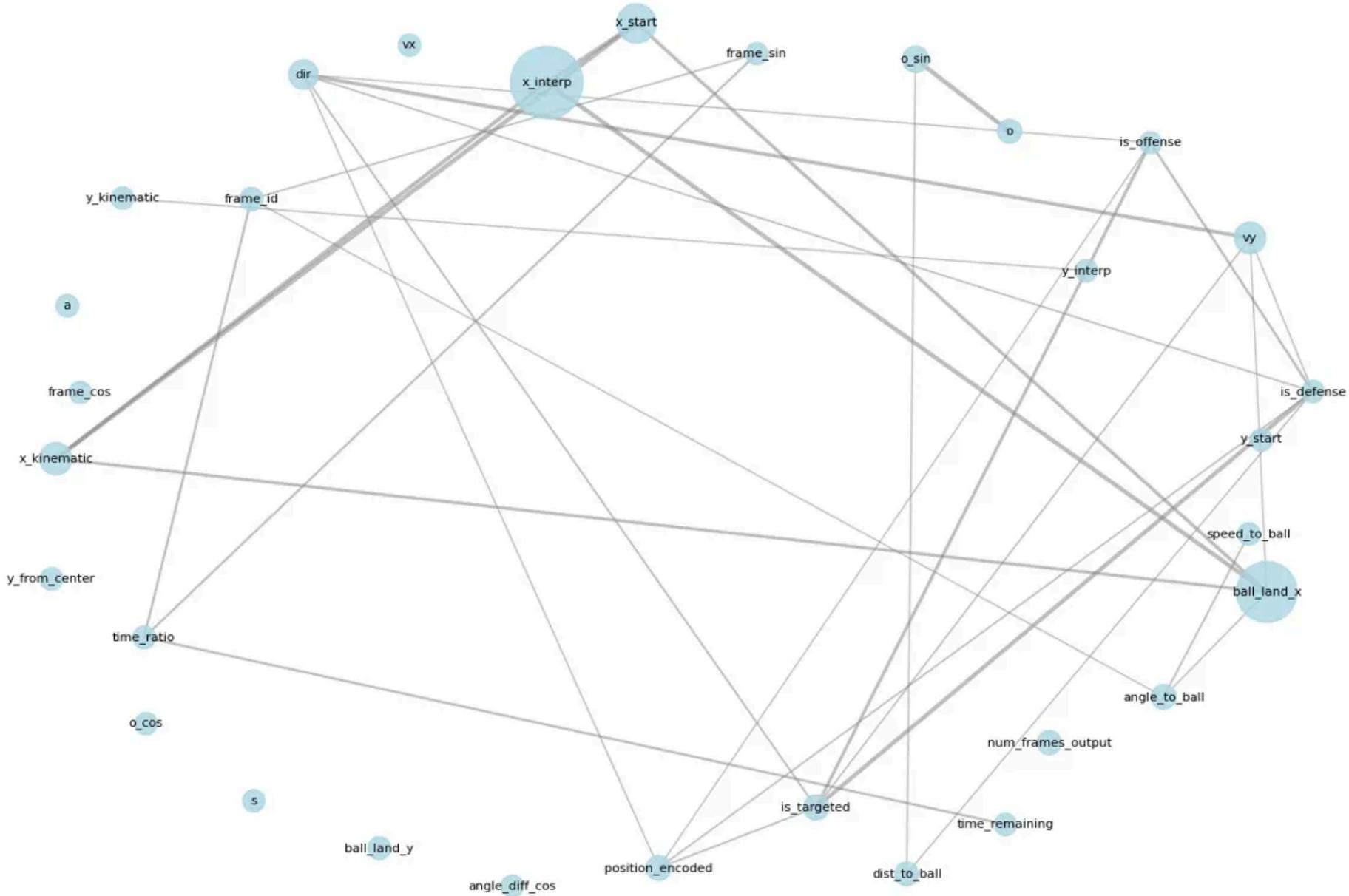
- Каждому игроку присвоили номер его группы (0, 1, 2 или 3)
- Преобразовали в 4 отдельных признака (один равен 1, остальные 0)
- Переобучили модель

Вариант	RMSE
Без групп	3.5197 ярда
С группами	3.5220 ярда
Изменение	35

Граф связей между признаками



Граф взаимосвязей признаков ($|corr| > 0.3$) Размер узла = важность



Что мы построили:

- Каждый признак – это точка (узел)
 - Если два признака влияют на модель похожим образом – рисуем линию между ними
 - Чем важнее признак – тем больше точка

Что увидели:

- Расчётная позиция — центральный узел, связан со многими другими
 - Место приземления мяча — тоже важный хаб
 - Скорость и ускорение — изолированы (влияют независимо)
 - Y-координата мяча — почти не связана с остальными

Инсайт: Всё, что связано с X-координатой (движение вдоль поля) – сильно взаимосвязано. Y-координата (поперёк поля) – независима.

Главный эксперимент – кросс-валидация



Что использовал и	RMSE	Разброс
Только исходные	3.79 ярда	±0.02
Только SHAP-эмбеддинги	3.10 ярда	±0.03
Все вместе	3.09 ярда	±0.02

Вопрос: Какие признаки лучше использовать?

Три варианта:

1. Только исходные данные (позиция, скорость и т.д.)
2. Только SHAP-эмбеддинги (как модель "видит" игрока)
3. Всё вместе (конкатенация)

Улучшение:

- SHAP vs исходные: -18.6%
- Конкатенация vs SHAP: -0.4% (почти не добавляет)

Главный инсайт: SHAP-эмбеддинги содержат почти ВСЮ полезную информацию!

Подбор лучших настроек модели (Optuna)



Параметр	Что означает	Диапазон поиска
Количество деревьев	Сколько деревьев в ансамбле	300-800
Глубина дерева	Насколько сложные правила	7-11
Скорость обучения	Как быстро модель учится	0.03-0.1
Количество листьев	Сколько конечных правил	24-64

Лучшие настройки (после 50 попыток):

- 777 деревьев
- Глубина 11
- Скорость обучения 0.081
- 63 листа

Финальные результаты проекта



Этап	Что сделали	RMSE	Улучшение
Старт	Первая модель	3.886 ярда	-
Этап 1	Улучшенная модель	3.5 ярда	-10%
Этап 2	LightGBM	3.52 ярда	-9.4%
Этап 3	+ SHAP-эмбеддинги	3.09 ярда	-20.5%
Финал	+ Оптимизация Optuna	2.87 ярда	-26.2%

Итого: улучшили точность на 26%!

Было 3.886 → Стало 2.87 (на 1 ярд точнее ≈ на 1 метр)



Главные выводы проекта

1. Про данные:

- Данные NFL качественные и информативные
- Аномалии (выбросы) нельзя удалять – они полезны
- Обучающие и тестовые данные похожи (модель стабильна)

2. Про признаки:

- Расчётная позиция игрока – лучший созданный признак
- Признаки про соседей не дали большого улучшения
- SHAP-эмбеддинги – мощнейший инструмент

3. Про модели:

- Сложные модели (LightGBM) лучше простых на нелинейных зависимостях
- Комбинация моделей даёт минимальный прирост
- Оптимизация настроек критически важна для финального качества



Что НЕ сработало

Что пробовали	Результат	Почему не сработало
Удалить аномалии	Стало хуже на 0.14%	Аномалии = редкие, но важные игровые ситуации
Добавить кластеры как признак	Без изменений	Модель уже сама извлекает эту информацию
Кластеризация DBSCAN	0 групп	Данные слишком плотные, нет чётких границ
Shapley Flow	Совпадает с SHAP на 97%	Не даёт новой информации



Итоговое резюме проекта



Что мы сделали:

- ✓ Анализ данных: Изучили 4.9 миллиона записей, нашли закономерности по ролям игроков
- ✓ Работа с аномалиями: Нашли выбросы тремя методами, решили пометить, а не удалять
- ✓ Создание признаков: 50+ признаков, включая физические расчёты и характеристики соседей
- ✓ Интерпретация: Полностью разобрали, как модель принимает решения (SHAP, LIME)
- ✓ Результат: Ошибка снизилась с 3.886 до 2.87 ярдов (улучшение 26%)



Приложение – наше топ решение

Модель	Что это	Индивидуальная ошибка	Вес в итоговом ответе
Bi-GRU	Рекуррентная сеть с вниманием	0.584	32.3%
ST-Transformer	Трансформер для временных рядов	0.580	32.5%
ST-Transformer v2	Другая конфигурация трансформера	0.562	35.2%

Как комбинировали:

- Чем лучше модель – тем больше её вес
- Итоговый ответ = взвешенная сумма трёх предсказаний

Наш результат на Kaggle: 0.534 RMSE в МИРОВОМ рейтинге 500 позиций

Для сравнения: 1 место = 0.460



Приложение – наше топ решения

Группа признаков	Кол-во	Что это
Базовая физика	30	Скорость, импульс, кинетическая энергия
Геометрическая цель	13	Где игрок ДОЛЖЕН оказаться по геометрии
Эмбеддинги от нейросети	17	Как нейросеть "видит" взаимодействие игроков
Временные лаги	30	Позиции на предыдущих кадрах
Взаимодействие с соперником	15	Расстояние до защитников, скорость сближения
Паттерны маршрутов	8	Прямолинейность траектории, тип маршрута

Главный прорыв – геометрические признаки:

- Для принимающего: "он должен оказаться ТАМ, где приземлится мяч"
- Модель учит только ОТКЛОНЕНИЯ от этой идеальной точки
- Это резко упрощает задачу для нейросети!

2. Продвинутые техники:

- 5 случайных зёрен × 5 фолдов = 25 моделей на каждую архитектуру (всего 75!)
- Аугментация на тесте: 6 вариантов с шумом и отражением
- Специальная функция потерь для плавных траекторий

Благодарим
за внимание!

AI

...