МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ»
(НИЯУ МИФИ)

ИНСТИТУТ ЯДЕРНОЙ ФИЗИКИ И ТЕХНОЛОГИЙ
КАФЕДРА №40 «ФИЗИКА ЭЛЕМЕНТАРНЫХ ЧАСТИЦ»

УДК 004.8, 539.12

На правах рукописи

ВАН АЛИНА МАОШЭНОВНА

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ ИДЕНТИФИКАЦИИ СТРУЙ, ОБРАЗОВАННЫХ W-БОЗОНОМ

Направления подготовки:
09.04.04 «Программная инженерия»,
14.04.02 «Ядерные физика и технологии»
Диссертация на соискание степени магистра

Научный руководитель,
к.ф.-м.н. _____ Е. Ю. Солдатов

Научный консультант,
к.ф.-м.н. _____ А. Г. Мягков

Москва 2025

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ ИДЕНТИФИКАЦИИ СТРУЙ, ОБРАЗОВАННЫХ W-БОЗОНОМ

Студент	_____ А. М. Ван
Научный руководитель, к.ф.-м.н.	_____ Е. Ю. Солдатов
Научный консультант, к.ф.-м.н.	_____ А. Г. Мягков
Рецензент, к.ф.-м.н.	_____ С. М. Поликарпов
Рецензент	_____ Т. И. Комаров
Секретарь ГЭК, к.ф.-м.н.	_____ Е. Ю. Солдатов
Зав. каф. №40, д.ф.-м.н., проф.	_____ М. Д. Скорохватов
Рук. учеб. прог., к.ф.-м.н.	_____ Е. Ю. Солдатов

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Эксперименты и исследования в физике высоких энергий привели к формированию основной теории строения и взаимодействия частиц. Стандартная модель [1] - это современная теория в физике элементарных частиц, которая описывает сильное, электромагнитное и слабое взаимодействие. Однако, несмотря на все свои преимущества, Стандартная модель не является полной теорией всего. Данная теория не дает описаний таким экспериментальным фактам, как скрытая масса, нейтринные осцилляции, темная энергия, не дает решений проблеме барионной асимметрии и проблеме иерархии масс и структур поколений. Предполагается, что Стандартная модель является частью более общей теории. Возможность наблюдения новой физики за рамками Стандартной модели является одной из главных задач на LHC.

При высоких энергиях соударений встречных пучков, достигаемых на LHC, рождается огромное количество векторных бозонов. Наиболее вероятным каналом распада векторных бозонов является адронная мода. При этом идентификация данных распадов с адронными струями затруднена фоновыми процессами КХД. В CERN на LHC [2] эффективная классификация адронных распадов векторных бозонов, в частности W-бозона, которые восстанавливаются в пределах одной “толстой” струи с $R = 1$, позволила бы значительно повысить чувствительность поиска физических явлений за рамками Стандартной модели. К примеру, в эксперименте по поиску возбужденного лептона [3; 4] с конечным состоянием $e\nu J$ процессом, привносящем основной вклад в фон, является образование пары топ-антитоп. Так как топ-кварк распадается преимущественно по слабому взаимодействию, идентификация струй, образованных W-бозоном, является необходимой для подавления фона от данного процесса.

Для идентификации струй от W-бозона на данный момент используются различные дискриминирующие переменные, обусловленные физикой и геометрией струй, а также кинематические переменные. Однако при использовании данных методов обрезки есть вероятность неправильной идентификации собы-

тий, кинематика которых схожа.

Создание многомерного классификатора по данным переменным в рамках машинного обучения может позволить более эффективно идентифицировать W-бозон. Также следует отметить, что данные подходы машинного обучения в дальнейшем могут адаптироваться в соответствии с потребностями отдельных анализов.

ЦЕЛИ И ЗАДАЧИ

Целью данной работы *является применение методов машинного обучения для решения задачи идентификации событий, образованных W-бозоном.*

В соответствии с поставленной целью ***задачами данной работы*** были:

1) Отбор событий для последующего формирования обучающих и тестовых выборок из Монте-Карло сэмплов с учетом веса каждого события;
2) Предварительный анализ полученных датасетов, формирование признаков и оценка их значимости;
3) Обучение моделей машинного обучения для задачи бинарной классификации: выбор оптимальных гиперпараметров и архитектуры модели;
4) Тестирование и оценка моделей;
5) Исследование отклика обученной модели на реальных данных;
6) Сравнение эффективности идентификации струй от W-бозона при помощи обученных моделей и при помощи дискриминирующей переменной.

1 ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 СТАНДАРТНАЯ МОДЕЛЬ

Стандартная модель (СМ) [5] – теория, описывающая частицы, из которых состоит материя, и их взаимодействие друг с другом. Она появилась в середине 20-го века и приняла свою окончательную форму после всех экспериментальных подтверждений. Модель проясняет обнаруженные к настоящему времени элементарные частицы и их поведение с тремя фундаментальными взаимодействиями: слабым, сильным и электромагнитным. Квантовая хромодинамика дает описание сильному взаимодействию, электрослабая теория описывает слабое и электромагнинтное взаимодействия. Элементарные частицы, определяемые СМ, показаны на рисунке 1.1.

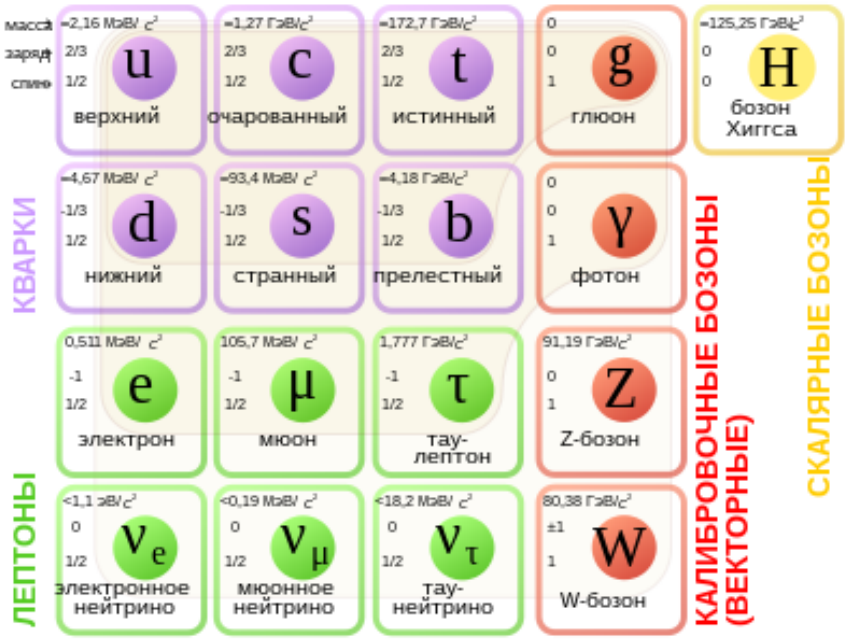

Рисунок 1.1 — Стандартная модель

Стандартная модель [6] содержит фермионы и бозоны. Фермионы в модели подчиняются статистике Ферми-Дирака и имеют спин 1/2. Для описания фермионов используют биспинорное представление группы Лоренца $SO(3,1)$.

Фермионы делятся на две группы: кварки и лептоны. В настоящее время известно 6 лептонов и 6 кварков (по три цвета на каждого), которые, в свою очередь, делятся на три поколения по двое, тем самым образуя дублеты.

Дублеты лептонов удовлетворяют локальной калибровочной симметрии $SU_L(2)\times U_Y(1)$. Электрон, мюон и τ-лептон участвуют в слабом и в электромагнитном взаимодействиях. Их нейтрино участвуют только в слабом взаимодействии. Дублеты кварков удовлетворяют локальной калибровочной симметрии групп $SU(3)\times SU_L(2)\times U_Y(1)$. Кварки участвуют во всех видах взаимодействия.

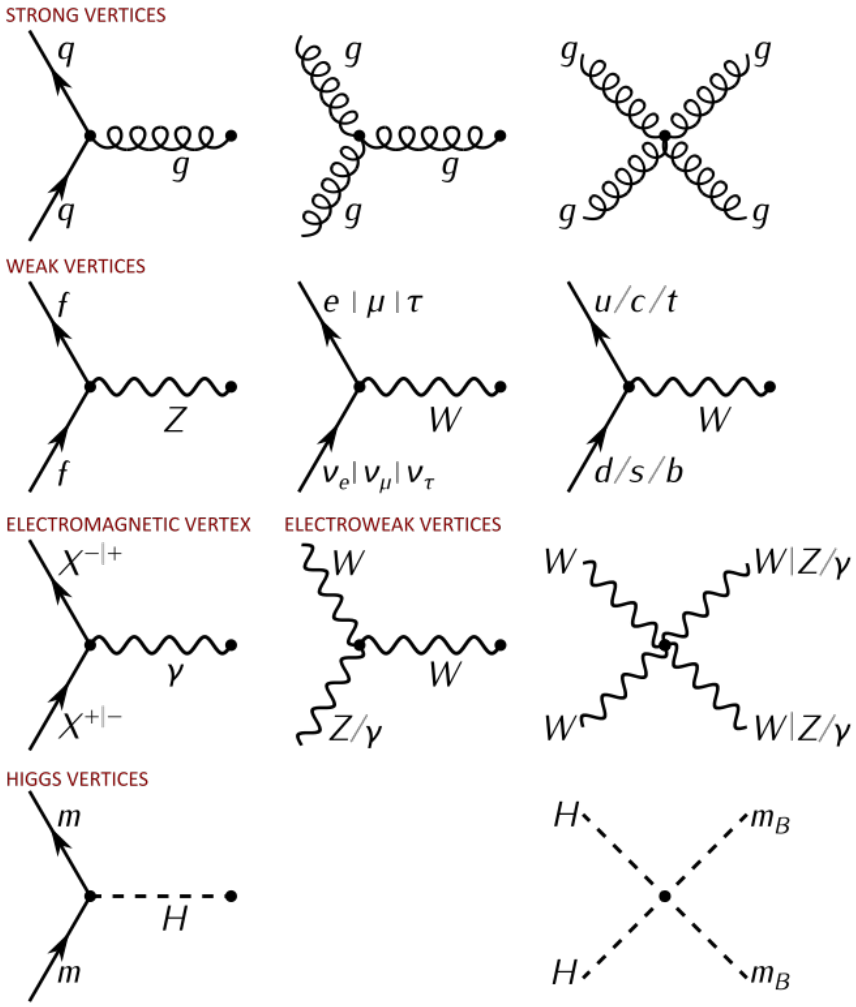

Рисунок 1.2 — Вершины Стандартной модели (m — частица, обладающая массой; m_B — бозон, обладающий массой.)

Бозоны, имеющие целый спин и подчиняющиеся статистке Бозе - Эйнштейна, в свою очередь, являются переносчиками взаимодействия, возникающими вследствие требования локальной калибровочной инвариантности. Частицы W, Z, γ, g имеют спин 1 и являются векторными бозонами. Переносчиками сильного взаимодействия являются 8 глюонов, слабого – $W^{\pm}, Z$-бозоны,

электромагнитного – фотоны.

Для придания масс частицам вводится скалярное поле Хиггса. Механизм Хиггса заключается в нарушении симметрии $SU_L(2)\times U_Y(1)$ до $U_{em}(1)$. Квантом поля Хиггса является бозон Хиггса, имеющий спин 0.

На рисунке 1.2 представлены все вершины взаимодействия Стандартной модели.

1.2 ВЕКТОРНЫЕ БОЗОНЫ

Энергия центра масс в 13 ТэВ на LHC позволяет исследовать физические процессы с образованием векторных бозонов с большим поперечным импульсом. Процедура идентификации векторных бозонов активно изучается на LHC [7; 8]. Особый интерес представляет распад векторных бозонов по адронной моде, так как для W-бозона вероятность этого канала распада составляет около 70%. Распад векторных бозонов в адронной моде имитируется многоструйными фоновыми процессами, которые состоят в основном из легких кварков и глюонов. Поэтому при анализе данных важно идентифицировать распадающиеся по адронной моде W-бозоны и исключить фоновые процессы КХД.

Реконструкция толстых струй

Стандартный подход к реконструкции струй в эксперименте ATLAS заключается в использовании алгоритма AntiKt. Алгоритм AntiKt [9] - это один из видов алгоритма кластеризации струи. Данный алгоритм строит иерархическую структуру, начиная с отдельных кластеров и постепенно объединяет их в более крупные кластеры, пока не будет достигнута заданная степень кластеризации. При этом расстояние в данном алгоритме задается формулой:

$$d_{ij} = \min(p_{T,i}^{-2}, p_{T,j}^{-2}) \cdot \Delta R_{ij}^2, \tag{1.1}$$

где $p_{T,i}$ и $p_{T,j}$ — поперечные импульсы струй, а ΔR_{ij} — расстояние в пространстве псевдобыстроты и азимутального угла.

При большой величине поперечного импульса струи, образованные ускоренным W-бозоном, после его распада по адронному каналу имеют малый угол разлета. Типичные тонкие струи от распада W-бозона с $p_t > 300$ ГэВ попадают в конус $R = 1$.

1.3 МАШИННОЕ ОБУЧЕНИЕ С УЧИТЕЛЕМ

Машинное обучение (МО) [10] - это динамично развивающаяся область прикладной математики, которая фокусируется на разработке и применении алгоритмов для решения различных задач с использованием обучающих данных. Машинное обучение позволяет решить задачи, которые недопустимо сложны для традиционных подходов или не имеют точных математических моделей, алгоритмов. При этом в данном подходе алгоритм обработки данных и нахождения взаимосвязей между ними заранее не известен и формируется в результате обучения. В рамках данной работы рассматриваются алгоритмы машинного обучения с учителем.

Постановка задачи для применения алгоритмов машинного обучения следующая:на основе размеченных данных, которые представлены в виде пар [матрица признаков; целевое значение], необходимо найти наилучшую функцию аппроксимации, веса которой минимизируют эмпирический риск.

Данные делятся на обучающую, тестовую и валидационную выборки. Обучающая выборка используется для настройки внутренних параметров (весов) модели. В зависимости от задачи обучающая выборка также может быть разделена и на валидационную выборку, которая необходима для оценки обобщающих способностей модели на новых для нее данных. При оценке результатов модели на валидационной выборке в процессе обучения можно вовремя избежать переобучения, т.е. ситуации, в которой модель точно подстраивается под обучающие данные и теряет способность к адекватному прогнозированию. Кроме того, для окончательной оценки результатов модели используется тестовая выборка. Она представляет собой отдельную выборку данных, которые не участвовали в обучении и валидации модели. Это необходимо для объективного оценивания модели на данных, которые модель не видела.

Основными задачами МО с учителем являются задачи регрессии и задачи классификации. При регрессии алгоритм МО учится предсказывать непрерывное число, в задаче классификации - категориальное значение. В данной работе поставлена задача бинарной классификации, информация о которой будет представлена далее.

1.3.1 БИНАРНАЯ КЛАССИФИКАЦИЯ

В данной работе под постановкой задачи идентификации частицы предлагается постановка задачи бинарной классификации. Отклики обучаемой модели должны быть дискретны и принимать только два значения: 0 (фон) и 1 (сигнал). Сама задача бинарной классификации заключается в нахождении гиперплоскости в признаковом пространстве, которая оптимально разделяет объекты двух классов, т.е. в данной задаче струи, образованные W-бозоном, и фоновые процессы КХД.

Функция потерь в контексте бинарной классификации - это мера неточности предсказания обучаемого классификатора для объектов выборки. Она количественно оценивает разницу между фактическими метками класса и предсказанными откликами модели. Алгоритмы машинного обучения направлены на минимизацию функции потерь для нахождения оптимальных внутренних параметров классификатора и, как следствие, улучшения производительности модели на новых данных.

Типичной функцией потери в бинарной классификации является бинарная кросс-энтропия. Формально, функцию потерь для одного объекта выборки можно записать как:

$$L(y, \hat{y}) = -\alpha \left[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}) \right], \tag{1.2}$$

где y — истинная метка класса, $\hat{y}$ — предсказанная вероятность принадлежности классу, α — вес события.

Для регрессии функция потерь является одной из метрик, по которой можно охарактеризовать качество и производительность модели, так как в регрессии отклики принимают непрерывные значения. В случае бинарной классификации оценка модели по среднему значению функций потерь не достаточна, так как отклики модели дискретны. Поэтому необходимы другие механизмы оценки работы модели.

Точность (Accuracy)

Данная метрика определяет долю правильно классифицированных событий среди всех событий. Более формально данную метрику можно представить в виде:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}, \tag{1.3}$$

где

- TP — сумма весов событий, принадлежащих положительному классу и предсказанных как положительный класс;
- TN — сумма весов событий, принадлежащих отрицательному классу и предсказанных как отрицательный класс;
- FN - сумма весов событий, которых классификатор ложно отнес к отрицательному классу;
- FP - сумма весов событий, которых классификатор ложно отнес к положительному классу.

ROC-кривая и AUC

Для оценки работы бинарного классификатора также используют ROC-кривую (Receiver Operating Characteristic). При различных порогах классификации (от 0 до 1) она отображает соотношения между TPR и FPR:

$$TPR = \frac{TP}{TP + FN}, \tag{1.4}$$

$$FPR = \frac{FP}{FP + TN}. \tag{1.5}$$

AUC (Area Under the Curve) - это интеграл под ROC-кривой. Значение данной метрики варьируется между 0 и 1, где 1 указывает на идеальную классификацию. Худшим значением AUC является значение, равное 0.5. Оно означает, что классификатор работает, как случайное подбрасывание монетки. AUC и ROC-кривая полезны тем, что позволяют оценить общую способность модели и учитывают все возможные пороги классификации.

Для решения задачи бинарной классификации существует множество алгоритмов: от простых моделей, как логистическая регрессия, до глубоких нейронных сетей. В работе использованы алгоритмы MLP и BDT, речь о которых пойдет далее.

1.3.2 БУСТИРОВАННЫЕ ДЕРЕВЬЯ РЕШЕНИЙ (BDT)

Одним из базовых подходов к решению задачи бинарной классификации является алгоритм *дерева решений*. Идея данного метода заключается в разбиении признакового пространства на различные множества непересекающихся областей и дальнейшей постановке этим областям меток класса. Для данного процесса разбиения необходимо правило, следуя которому объект выборки попадает в определенную область. Дерево решений - совокупность этих правил.

Правило ветвления в дереве решений определяется так, чтобы среди различных признаков выбрать тот, условие по которому будет сокращать неоднородность в секторах признакового пространства. Количественная мера неоднородности в дереве решений может определяться индексом Джини:

$$G(R_l) = \sum_{k=1}^{K} p_{lk}(1 - p_{lk}), \tag{1.6}$$

где p_{lk} - это вероятность k-го класса в области R_l, K - это количество меток класса. То есть, когда индекс Джини равен 0, можно сделать вывод, что данная область содержит объекты только одного класса.

Критерий останова итераций разбиения на ноды определяется в зависимости от задачи. Популярные из них: ограничение на количество листьев, на глубину дерева и т.д.

По смыслу своего формирования дерево решений при его углублении склонно к переобучению. Чем больше глубина дерева, тем более вероятно то, что модель потеряет обобщающие способности и будет следовать за выбросами. Поэтому часто эту базовую модель используют в *ансамблевых методах*.

Бустинг — это метод ансамблевого обучения, который последовательно соединяет несколько слабых моделей для создания более сильной модели. Он направлен на обучение новых моделей для исправления ошибок, допущенных предыдущими. Тем самым данный процесс повышает общую работоспособность модели.

При градиентном бустинге каждая новая модель обучается для минимизации функции потерь предыдущей с помощью градиентного спуска. На каждой итерации алгоритм вычисляет градиент функции потерь по отношению к прогнозам, а затем обучает новую слабую модель для минимизации этого гра-

диента. Потом прогнозы новой модели добавляются в ансамбль, и обучение повторяется до тех пор, пока не будет достигнут критерий останова.

1.3.3 МНОГОСЛОЙНЫЙ ПЕРСЕПТРОН (MLP)

Нейронная сеть - это сложная дифференцируемая функция, которая преобразует пространство признаков в пространство откликов. Все ее внутренние веса связаны между собой и настраиваются одновременно (когда мы их специально не "замораживаем"). Конкретно данную сложную функцию представляют в виде композиции простых функций, которые называют слоями нейронной сети. Количество данных слоев в сети определяет уровень сложности и глубину модели.

Многослойный персептрон (MLP) - это один из типов нейронной сети, который состоит исключительно из полносвязных слоев. Архитектура MLP довольно проста: в ней содержатся входной слой, в котором обязательно должно содержаться количество нейронов, равное размерности признакового пространства, выходной слой и скрытые слои.

Полносвязные слои - слои, в которых выходные нейроны связаны со всеми входными нейронами. Выходные данные полносвязного слоя формально можно представить с помощью следующей формулы:

$$y = \sigma(Wx + b), \tag{1.7}$$

где y - выход слоя, x — вектор входных данных, поступающих из предыдущего слоя, W — матрица весов, связывающая вход и выход нейрона, b — интерсепт, σ — функция активации.

Функция активации - это нелинейное преображение, которое поэлементно применяется к пришедшим на вход данным. Благодаря данной нелинейности, нейронные сети способны порождать более информативные признаковые пространства, в которых функция ошибки, которую мы пытаемся минимизировать, будет оптимальной. Главные требования для функции активации: быть монотонной и иметь первую производную почти всюду (необходимо для обратного распространения ошибки).

В качестве функции активации могут использоваться разные нелинейные функции, и у каждой из них есть свои плюсы и минусы. Одни из часто исполь-

зуемых функций активации - это ReLu, сигмоида и гиперболический тангенс.

Ключевые механизмы обучения MLP:

1) Прямой проход (forward pass): При прямом распространении данные поступают от входного слоя к выходному, проходя через все скрытые слои. Нейрон вычисляет взвешенную сумму входных, после чего эта сумма проходит через функцию активации для введения нелинейности.
2) Функция потерь: Следующим шагом после выявления отклика модели является вычисление взвешенной функции потерь, которая сравнивает отклики модели с фактическими метками. Для бинарной классификации формульное представление функции потерь представлено выше [1.2].
3) Обратное распространение ошибки (back pass): На данном этапе вычисляется градиент функции потерь по каждому весу модели (в том числе интерсепту) согласно правилу дифференцирования сложной функции. Нейронная сеть обновляет весовые коэффициенты и интерсепт, двигаясь в направлении, противоположном градиенту, чтобы уменьшить потери.

Градиентный спуск

Градиентный спуск представляет собой итеративный алгоритм оптимизации, который применяется для минимизации функции потерь, отражающей степень расхождения между предсказаниями модели и фактическими значениями. Основной целью данного алгоритма является настройка параметров модели, таких как веса и смещения, с целью минимизации ошибки. Движение в направлении, противоположном градиенту функции потерь, позволяет алгоритму постепенно снижать значение функции и, в конечном итоге, достигать её минимума. Градиенты служат направляющими для обновлений параметров, обеспечивая сходимость к оптимальным значениям.

Шаги, используемые при градиентном спуске, определяются гиперпараметром, известным как *скорость обучения*. Скорость обучения является критически важным параметром, который определяет величину шагов, предпринимаемых при движении вниз по градиенту для обновления параметров модели. Она влияет на скорость сходимости алгоритма к минимуму функции потерь: слишком большая скорость обучения может привести к пропуску минимума, в то время как слишком малая может замедлить процесс сходимости.

Существует множество модификаций алгоритма градиентного спуска. Популярные из них: SGD, Adam, RMSprop.

Стохастический градиентный спуск (SGD) является одним из методов оптимизации, используемых для минимизации функции потерь в контексте машинного обучения. В отличие от традиционного градиентного спуска, который использует все данные для вычисления градиента, SGD обновляет параметры модели, основываясь на градиентах, вычисленных на случайных подмножествах данных. Это позволяет значительно ускорить процесс обучения и снижает вероятность застревания в локальных минимумах. Формально механизм обновления параметров можно представить в виде:

$$\theta = \theta - \eta \nabla L(\theta), \tag{1.8}$$

где θ - веса модели, η — скорость обучения, $\nabla J(\theta)$ — градиент функции потерь.

Преимущества SGD включают в себя быструю сходимость и эффективное использование памяти, однако данный метод может быть подвержен шумным обновлениям, что иногда приводит к нестабильности в процессе обучения.

RMSProp (Root Mean Square Propagation) представляет собой адаптивный алгоритм оптимизации, который помогает решать проблемы, возникающие при обучении нейронных сетей, особенно в условиях нестационарных данных. RMSProp адаптирует скорость обучения для каждого параметра, что способствует улучшению сходимости.

$$\theta = \theta - \frac{\eta}{\sqrt{v_t} + \epsilon} g_t, \tag{1.9}$$

где θ - веса модели, η — скорость обучения, g_t — градиент функции потерь, ϵ - поправка, предотвращающая деление на 0, а v_t — скользящее среднее квадратов градиентов, которое задается формулой:

$$v_t = \beta v_{t-1} + (1 - \beta) g_t^2, \tag{1.10}$$

где β - коэффициент затухания.

Этот метод особенно эффективен в условиях шумных или разреженных градиентов, так как он позволяет каждому параметру иметь свою индивидуальную скорость обучения. Кроме того, RMSProp использует экспоненциальное скользящее среднее, что помогает избежать резких изменений в обновлениях параметров, обеспечивая более стабильное и предсказуемое поведение алгоритма.

Алгоритм *Adam (Adaptive Moment Estimation)* объединяет идеи, заложенные в алгоритме RMSProp, и метод моментов. В этом алгоритме обновление весов корректируется с учетом первого и второго моментов градиентов.

$$\theta = \theta - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t, \tag{1.11}$$

где m_t - первый момент градиента,

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \tag{1.12}$$

а $\hat{m}_t, \hat{v}_t$ - корректировки смещения

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \hat{v}_t = \frac{v_t}{1 - \beta_2^t}. \tag{1.13}$$

Одним из значительных преимуществ данного алгоритма является снижение вероятности застревания модели в локальном минимуме. Это достигается за счет того, что веса в Adam обновляются с использованием экспоненциального скользящего среднего прошлых градиентов, что устраняет незначительные колебания и позволяет алгоритму быстрее сходиться к оптимальному решению.

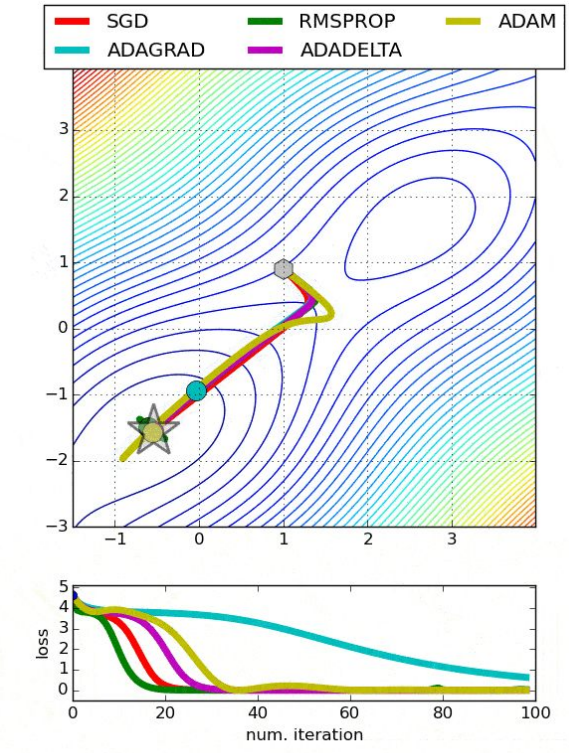

Рисунок 1.3 — Алгоритмы оптимизации

1.3.4 МЕТОДЫ РЕГУЛЯРИЗАЦИИ ДЛЯ УСТРАНЕНИЯ ПЕРЕОБУЧЕНИЯ

Переобучение представляет собой распространенную проблему в области машинного обучения, возникающую в результате того, что модель сильно адаптируется к обучающим данным, что приводит к утрате способности обобщать на новых, ранее не видимых моделью, данных.

Алгоритмы, основанные на *бустированных деревьях решений*, имеют предрасположенность к переобучению из-за своей способности формировать сложные модели, которые могут слишком точно подстраиваться под обучающие данные. В данной работе рассматривается алгоритм градиентного бустинга, который строит ансамбль деревьев решений. Каждое новое дерево в этом ансамбле фокусируется на ошибках, допущенных предыдущими деревьями. Это позволяет модели достигать высокой точности на обучающих данных. Однако такая способность к адаптации может привести к тому, что модель начнет учитывать шум в данных, а не только истинные паттерны, что и вызывает переобучение.

Кроме того, алгоритмы бустинга могут быть чувствительны к небольшим изменениям в данных. Если в обучающем наборе присутствуют выбросы или шум, модель может начать воспринимать их как важные признаки. Это негативно сказывается на её способности обобщать на новых данных. Неправильная настройка гиперпараметров, таких как глубина деревьев, количество деревьев и скорость обучения, также может способствовать переобучению. К примеру, большая глубина деревьев может привести к созданию слишком сложных моделей, которые не обобщают, а подстраиваются под шум в данных.

Полносвязные нейронные сети имеют большое количество настраиваемых весов: с увеличением количества нейронов или слоев в модели увеличивается и количество настраиваемых внутренних параметров. Это позволяет сети подстраиваться под сложные зависимости в обучающих данных, однако также увеличивает риск запоминания шумов. Сложные архитектуры нейронных сетей могут легко адаптироваться к обучающим данным, что делает их уязвимыми к переобучению. В результате модель может начать учитывать несущественные детали, которые не являются общими для всей выборки.

Переобучение можно выявить, анализируя *историю обучения модели*. История обучения модели - это графики, отображающие изменение функции по-

терь на обучающем и валидационном наборах данных в процессе обучения. Обычно, если модель переобучается, график функции потерь на обучающей выборке будет продолжать снижаться, в то время как график функции потерь на валидационной выборке может начать подниматься после определенного количества эпох или итераций обучения, либо выходить на плато.

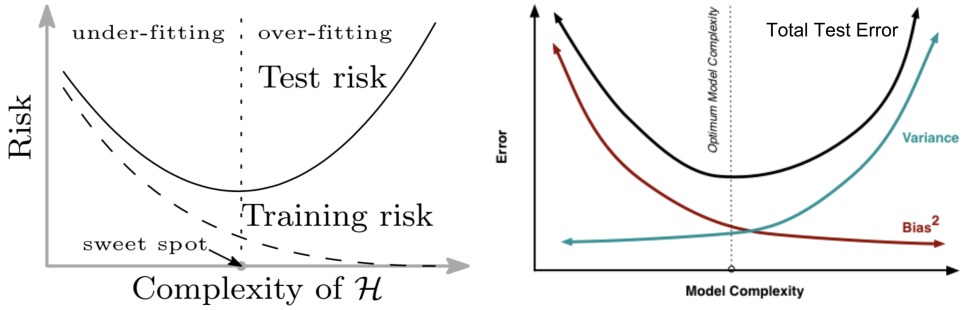

Рисунок 1.4 — График истории обучения

Для предотвращения переобучения моделей применяются различные техники, направленные на ограничение их сложности или на наложение штрафов за значительные изменения в функции потерь.

В контексте моделей, BDT и MLP, используются следующие методы:

- **Регуляризация** представляет собой добавление дополнительного слагаемого в функцию потерь, которое накладывает ограничения на изменения вектора весов, тем самым штрафуя их за избыточные выбросы в данных. Наиболее распространенными видами регуляризации в настоящее время являются регуляризация Риджа и Лассо. Регуляризация L1 (Лассо) накладывает штраф на абсолютные значения коэффициентов модели, в то время как L2 (Ридж) - регуляризация штрафует квадратные значения коэффициентов модели.

  Функция потерь с L1-регуляризацией записывается следующим образом:

$$L(\mathbf{w}) = L_0(\mathbf{w}) + \lambda \sum_{i=1}^{n} |w_i|, \tag{1.14}$$

  где $L_0(\mathbf{w})$ - исходная функция потерь, λ - коэффициент регуляризации, $|w_i|$ - абсолютные значения весов модели, n - количество признаков.

  Также формально функцию потерь с L2-регуляризацией представляется в виде:

$$L(\mathbf{w}) = L_0(\mathbf{w}) + \lambda \sum_{i=1}^{n} w_i^2, \tag{1.15}$$

где w_i^2 - квадраты весов модели.

- **Снижение сложности моделей** также способствует предотвращению переобучения, так как уменьшает количество настраиваемых внутренних параметров.
- **Ранняя остановка обучения** является еще одним методом, используемым для устранения переобучения моделей. В процессе обучения с помощью валидационного набора проверяется производительность модели. Если производительность на валидационном наборе начинает ухудшаться, обучение может быть остановлено, что предотвращает дальнейшее переобучение.
- **Дропаут (Dropout)** представляет собой популярный и эффективный метод регуляризации, применяемый в нейронных сетях для снижения риска переобучения. Основная идея дропаута заключается в случайном исключении определенного процента нейронов из сети на каждом этапе обучения. Это означает, что во время одной итерации обучения нейросеть активирует лишь случайное подмножество нейронов, в то время как оставшиеся временно не участвуют в прямом и обратном распространении ошибки. В результате дропаут способствует формированию более устойчивых и обобщающих признаков.

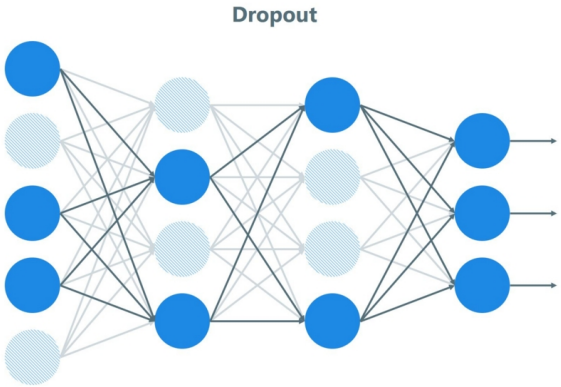

Рисунок 1.5 — Механизм Dropout

2 ДЕТЕКТОР ATLAS

Детектор ATLAS [11] предназначен для исследования широкого спектра физических явлений: от измерения свойств частиц Стандартной модели до поиска новой физики за ее рамками. Он используется для детектирования и идентификации частиц. Детектор ATLAS представляет собой многоцелевой 4π-детектор с симметричной цилиндрической геометрией. Установка состоит из серии больших концентрических цилиндров вокруг линии пучка. Детектор ATLAS состоит из внутреннего трекового детектора, электромагнитного и адронного калориметров, мюонного спектрометра и магнитных систем. Каждый из них в свою очередь сделан из повторяющихся слоев. Трековый детектор предназначен для определения параметров треков заряженных частиц для измерения их импульса. Калориметры необходимы для измерения энерговыделения частиц, мюонная система используется для определения импульса и направления пролёта высокопроникающих мюонов. Магнитная система предназначена для искривления траекторий заряженных частиц для определения их импульса. Устройство детектора ATLAS представлено на рисунке 2.1.

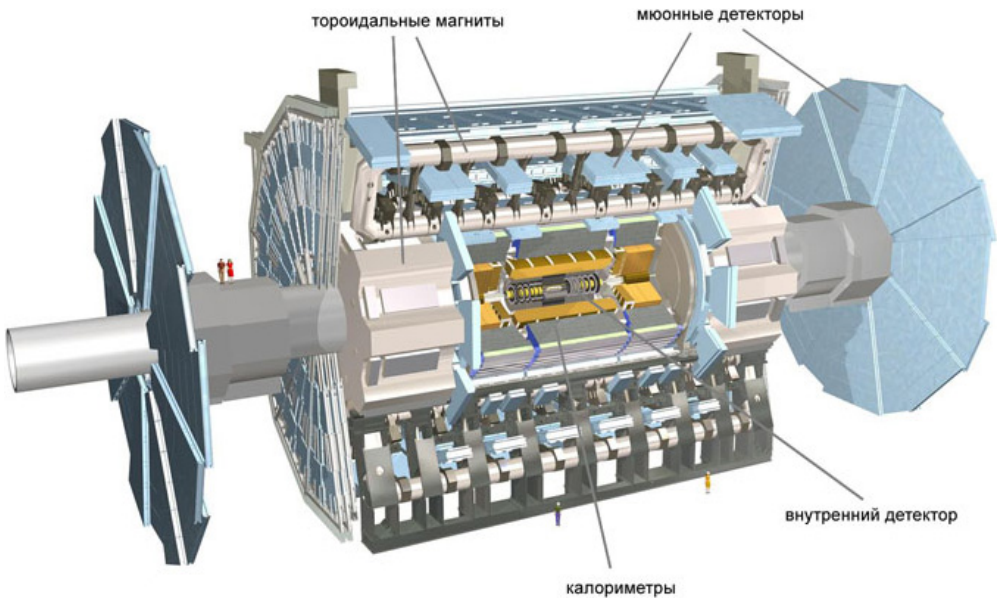

Рисунок 2.1 — Устройство детектора ATLAS

3 ИСХОДНЫЕ ДАННЫЕ

Обучение моделей проводится с данными, полученными методом Монте-Карло моделирования протон-протонного столкновения в детекторе ATLAS на LHC с энергией в системе центра масс 13 ТэВ для конечного состояния **evJ**.

Большое сечение процесса образования пары $t\bar{t}$ на LHC обеспечивает высокую статистику и относительную чистую среду для исследования струй, образованных от W-бозона. После регистрации лептонного распада одного из W-бозона и двух b-струй в данном событии остается еще W-бозон, который при данном конечном состоянии evJ потенциально распадается по адронному каналу. Поэтому в качестве сигнальных объектов используются смоделированные сэмплы процесса образования пары $t\bar{t}$. Данный процесс моделировался с помощью МК генераторов Powheg + Pythia 8.

В качестве фоновых объектов для обучения использовались следующие процессы:

Процесс	**МК-генератор**
$Single - t$	Powheg + Pythia 8
$Z(\to qq)Z(\to ll)$	Sherpa
$W(\to l\nu)Z(\to qq)$	Sherpa
$lll\nu$ + КХД струи	Sherpa
$ll\nu\nu$ + КХД струи	Sherpa
$W(\to e\nu)$ + КХД струи	Sherpa
$Z(\to ee)$ + КХД струи	Sherpa
$Z(\to \tau\tau)$ + КХД струи	Sherpa
$W(\to \nu\tau)$ + КХД струи	Sherpa

Таблица 3.1 — Фоновые процессы

Отклик моделей исследовался на экспериментальных данных с RUN2 2017 года с энергией в системе центра масс 13 ТэВ общей светимостью $L = 6.4$ фб$^{-1}$.

3.1 ВЕС СОБЫТИЯ В МОНТЕ-КАРЛО СИМУЛЯЦИЯХ

По мере увеличения статистики на экспериментальных установках растет и необходимость в более точном моделировании событий. Основными инструментами симуляции процессов в физике высоких энергий, которые используются в данный момент, являются Монте-Карло генераторы и NLO-вычисления (next-to-leading order) [12].

Для более точного моделирования необходимо учитывать поправки, возникшие в результате виртуальных процессов, и поправки, обусловленные реальным излучением. Поэтому при создании симуляций также используется NLO-вычисление, сечение которого формально можно представить в виде суммы поправок от виртуальных взаимодействий и от реальных излучений.

NLO-вычисления формируют поправки к весам событий в МК симуляциях. Также для сравнения МК сэмплов с реальными данными необходимо учитывать набранную в экспериментальных данных светимость и коэффициенты для корректировки весов событий, которые обусловлены устройством детектора.

Итоговая формула расчета веса события в МК симуляциях для сравнения с реальными данными следующая:

$$weight\_of\_event = \frac{L \times \sigma \times k_{fac} \times filter\_eff}{sum\_of\_weight \times prw\_weight}, \tag{3.1}$$

где L - светимость, σ - сечение процесса, k_{fac} - коэффициент корректировки для устранения различий между теоретическими предсказаниями и экспериментальными данными (NLO), $filter\_eff$ - вероятность того, что событие будет зарегистрировано детектором, sum_of_weight - сумма весов всех событий, которые были сгенерированы в процессе симуляции, prw_weight - вес для коррекции на случайные события.

3.2 ОТБОРЫ

В данной работе отбор событий направлен на выделение наборов струй, которые могут быть репрезентативны для струй, происходящих от W-бозонов. Как ранее было упомянуто, струи в данном анализе были сформированы с использованием алгоритма AntiKt с радиусом конуса $R = 1$.

Для составления сигнального датасета из МК симуляций процесса образования пары $t\bar{t}$ первоначально выбираются события, в которых имеется хотя бы одна b-меченная струя. Наличие b-меченных струй в событии является одним из главных следствий распада топ-кварка. Для событий, характеризующих фоновые и сигнальны объекты в данном анализе, добавлен отбор по кинематике b-меченной струи:

$$p_t > 30 \text{ ГэВ}, |\eta| < 2.5$$

Также и на фоновую, и на сигнальную выборку наложены ограничения на поперечный импульс толстой струи и на ее инвариантную массу. Ограничения представлены ниже:

$$p_t > 200 \text{ ГэВ}, 60 \text{ ГэВ} < m < 110 \text{ ГэВ}$$

4 ПРОЦЕСС РАБОТЫ

В рамках данной работы применяются два различных метода МО: деревья решений с алгоритмом градиентного бустинга (BDT) и полносвязные нейронные сети (MLP). Обучение и настройка модели BDT производилась с помощью библиотеки XGBoost [13], модели MLP - с помощью библиотеки Keras [14].

Оба метода используются для мечения струй от W-бозона и исследуют один и тот же набор входных переменных. Данный набор переменных в совокупности для каждого события представляет собой вектор признаков, который подается в алгоритмы МО вместе с метками классов. Для каждого многомерного метода важно оптимально настроить гиперпараметры, алгоритмы оптимизации и архитектуру модели.

4.1 ВХОДНЫЕ ПЕРЕМЕННЫЕ

Поперечный импульс и масса струи

КХД-струя формируется из кварка или глюона, не обладая при этом определенным масштабом массы, в отличие от толстой струи, образованной распадающимся по адронной моде W-бозоном. Ожидается, что масса КХД-струи будет приблизительно пропорциональна её поперечному импульсу p_t. В то же время инвариантная масса толстой струи от ускоренного W-бозона в значительной степени определяется значением массы W-бозона с меньшей зависимостью от его поперечного импульса.

Коэффициенты энергетической корреляции

Функции энергетической корреляции используются во многих работах [15; 16] как один из эффективных методов исследования структуры струи. Данные коэффициенты позволяют лучше выявить неявные коллинеарные особенности в структуре струи, которые невозможно найти с помощью других ме-

тодов. Формулы используемых переменных, характеризующих энергетические корреляции, представлены ниже:

$$E_{CF1} = \sum_{i}^{n} p_{T,i}, \tag{4.1}$$

$$E_{CF2} = \sum_{i,j}^{n} p_{T,i} p_{T,j} \Delta R_{ij}, \tag{4.2}$$

$$E_{CF3} = \sum_{i,j,k}^{n} p_{T,i} p_{T,j} p_{T,k} \Delta R_{ij} \Delta R_{jk} \Delta R_{ki}, \tag{4.3}$$

$$D_2 = E_{CF3} \times \left(\frac{E_{CF1}}{E_{CF2}}\right)^3, C_2 = \frac{E_{CF1} \times E_{CF2}}{E_{CF3}^2}, \tag{4.4}$$

где p_T - поперечный импульс кластера, i, j - i, j-ый кластер струи, а ΔR_{ij} определяется как:

$$\Delta R_{ij}^2 = (\eta_i - \eta_j)^2 + (\phi_i - \phi_j)^2. \tag{4.5}$$

Плоский поток

Плоский поток [17] характеризует геометрическое распределение энергии толстой струи от W, которое отличает ее от КХД-струй, поскольку последние более изотропны. Плоский поток определяется следующим образом. Сначала строится матрица:

$$I_{\omega}^{kl} = \frac{1}{m_{jet}} \Sigma_i \omega_i \frac{p_{i,k}}{\omega_i} \frac{p_{i,k}}{\omega_i}, \tag{4.6}$$

где m_{jet} - масса струи, ω_i - энергия i-го кластера в струе, $p_{i,k}$ - k-я компонента поперечного импульса i-го кластера относительно импульса струи.

Затем плоский поток определяется на основе I_{ω}^{kl} как:

$$P = \frac{4\lambda_1 \lambda_2}{(\lambda_1 + \lambda_2)^2}, \tag{4.7}$$

где $\lambda_{1,2}$ соответсвуют собственным значениям матрицы I_{ω}^{kl}.

Для линейного распределения энергии кластеров в струе $P \to 0$, для изотропного $P \to 1$.

N-субъектность

Данная переменная [18] рассчитывается в предположении, что в струе N объектов, и представляет собой взвешенную по расстоянию между кластерами сумму по поперечным импульсам:

$$\tau_N^{(\beta)} = \sum_i p_{T,i} \min\left(R_{1,i}, R_{2,i}, \ldots, R_{,N,i}\right), \tag{4.8}$$

где сумма проходит по всем кластерам в струе.

Апланарность A

Апланарность определяется тензором сферичности, который задается следующим образом:

$$S^{\alpha,\beta} = \frac{\sum_i p_i^\alpha p_i^\beta}{\sum_i |\vec{p}_i|^2}, \tag{4.9}$$

где где α, β соответствуют x, y и z компонентам импульса каждого энергетического кластера в системе отсчета покоя струи.

При стандартной диагонализации тензора можно найти три собственных значения $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Апланарность определяется как:

$$A = \frac{3\lambda_3}{2}. \tag{4.10}$$

Диапазон значений данной переменной от 0 до 0.5. Для линейного распределения энергии кластеров в струе $A \to 0$, для изотропного $A \to 0.5$..

Отношение моментов Фокса-Вольфрама

Моменты Фокса-Вольфрама определяются как

$$H_l = \sum_i \frac{|\vec{p_i}||\vec{p_j}|}{E^2} P_l(\cos\theta_{ij}), \tag{4.11}$$

где θ_{ij} - угол между энергетическими кластерами i и j, E - полная энергия кластеров в системе отсчета струи, $P_l(x)$ - полиномы Лежандра.

В качестве дискриминирующей переменной в данном анализе будет использоваться отношение второго и нулевого моментов Фокса-Вольфрама.

Более подробно данные переменные описаны в статье [18].

Статистическая мера KtDR

Струя определяется кластерной последовательностью. Ранее упоминалось, что для реконструкции струй используют алгоритм AntiKT, который представляет собой иерархическую кластеризацию. Поэтому кластерная последовательность топологически представляет собой древовидную структуру. Процедура количественной оценки изменчивости последовательности кластеризации [19] заключается в следующем. На каждом шаге кластеризации каждой составляющей паре присваивается вес ω_{ij}, а затем случайным образом выбирается одна из доступных пар и объединяется. Вес по умолчанию определяется как:

$$\omega_{ij} = \exp\{-\alpha \frac{d_{ij} - d^{min}}{d^{min}}\}, \tag{4.12}$$

где $d_{ij} = \Delta R_{ij}^2$, d_{min} - это его минимальное расстояние по всем парам на данном этапе кластеризации, а α - коэффициент уровня случайности.

KtDR определяется как среднеквадратичное значение распределения массы струи (RMS), деленное на среднее значение массы струи.

$$KtDR = \frac{RMS}{< m >}. \tag{4.13}$$

КХД-струи характеризуются большой изменчивостью распределения масс для каждого под-дерева, в то время как для струй, образованных от W-бозона, кластерные последовательности практически идентичны.

4.2 ПРЕДВАРИТЕЛЬНАЯ ОБРАБОТКА ДАННЫХ

Предварительная обработка реальных данных и Монте-Карло симуляций ранее перечисленных процессов производилась на вычислительном кластере ИФВЭ. Используемые языки и фреймворки для работы с root-файлами: Bash, C++, ROOT.

Первым этапом предварительной обработки root-файлов с набором данных Монте-Карло симуляций было выполнение вспомогательных Bash-скриптов, которые формировали списки с именами файлов и каталогов с данными для дальнейшей работы.

Далее с помощью C++ и ROOT были написаны скрипты для формирования данных с сигнальными и фоновыми объектами по описанным выше от-

борам. Листинг скрипта для формирования сигнального датасета представлен далее.

Структура программы обработки данных из ROOT-файлов

```
set filename path to "filename.txt"
open filename for reading
while there are lines in filename do
    open ROOT file at the specified path
    get TTree from the ROOT file
    set branch addresses for TTree
    create new ROOT file for output, TTree for signal (tS)
    set branches for tS
    get number of entries in the TTree
    for each entry i from 0 to nentries do
        get entry i from TTree
        calculate variables using fatjet properties
        initialize counters count to 0
        for each jet in jet_pt do
            if jet_btag ≥ 1 and jet_pt ≥ 30000 and |jet_eta| ≤ 2.5 then
                increment count
            end if
        end for
        if W_m < 110000 and W_m > 60000 and W_pt > 200000 and count ≥
        1 then
            set values for tS branches
            fill tS tree
        end if
    end for
    write tS to output file
    close output file, input ROOT file
end while
close "filename.txt"
```

Для расчета весов Монте-Карло симуляций был сформирован Bash-скрипт, результатом которого являются csv-файлы с характеристиками модулируемых процессов.

4.3 АНАЛИЗ ПРИЗНАКОВ

Отбор и анализ признаков для обучения модели МО необходим для предотвращения переобучения модели, уменьшения сложности модели и улучшения

производительности модели. К примеру, если в процессе обучения модели дать ей на вход два сильно скоррелированных признака, то это может привести к проблемам, связанным с мультиколлинеарностью. И в итоге модель может потерять все свои обобщающие способности и будет сильно чувствительна к выбросам в обучающей выборке. Также первичная оценка значимости признаков в рамках данного анализа позволит сократить размерность признакового пространства и, как следствие, вычислительные затраты.

Существует множество техник для отбора и оценки значимости признаков для обучения моделей МО. В контексте данной работы будут рассмотрены следующие:

1) Метод отбора признаков с использованием логистической регрессии и регуляризации L1
2) Метод отбора признаков с использованием дерева решений

Анализ признаков производился с помощью языка Python, были использованы библиотеки pandas (для взаимодействия с данными) и sklearn (для обучения базовых моделей МО).

Метод отбора признаков с использованием дерева решений

Основная идея деревьев решений заключается в том, чтобы разбить данные на подмножества, основываясь на значениях признаков. Каждый узел дерева представляет собой условие на определенный признак, а ветви — возможные исходы этого условия. В конечных узлах находятся предсказания модели. Важность признаков может быть оценена на основе критериев, таких как уменьшение энтропии или индекс Джини, что позволяет количественно оценить вклад каждого признака в модель.

Для оценки важности признаков вся полученная выборка проходила через дерево решений с критерием индекса Джини. Полученная важность данных признаков представлена на рисунке 4.1.

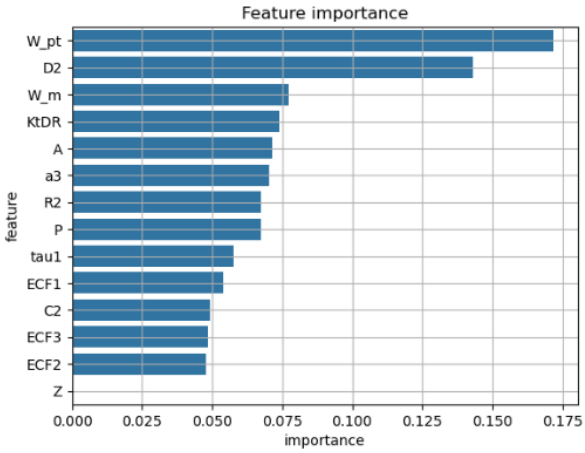

Рисунок 4.1 — Важность признаков (дерево решений)

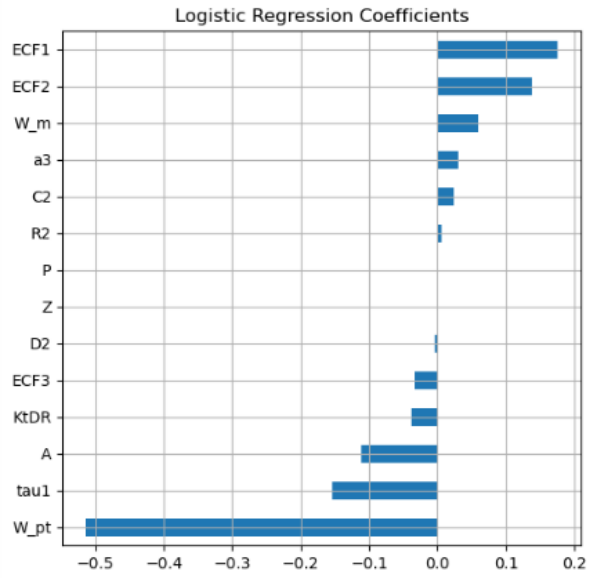

Рисунок 4.2 — Коэффициенты логистической регрессии

Метод отбора признаков с использованием логистической регрессии и регуляризации L1

Логистическая регрессия находит взаимосвязь между целевой переменной и одной или несколькими независимыми переменными, оценивая вероятности с помощью своей логит-функции. Линейные модели работают так, что коэффициенты при признаках указывают насколько сильно изменение признака влияет на целевую переменную. При большом признаковом пространстве для уменьшения риска переобучения используют методы регуляризации. В частности Lasso регуляризация, а именно прибавление к функции потерь суммы абсолютных значений коэффициентов, позволяет обнулять коэффициенты неинформативных признаков.

Предварительно выборки были нормализованы для объективной оценки значимости коэффициентов логистической регрессии. По нормализованной выборке была обучена базовая модель логистической регрессии, полученные коэффициенты данной модели представлены на рисунке 4.2.

По **итогам** исследований из дальнейшего анализа исключена переменная Z_cut, так как при оценке значимости признака деревом решений и логистической регрессией сделан вывод о том, что данная переменная не влияет на дальнейшие результаты моделей.

4.4 ОБУЧЕНИЕ МОДЕЛИ BDT

Процесс обучения бинарного классификатора XGBoost включает несколько ключевых этапов. Сначала данные разделяются на обучающую, валидационную и тестовую выборки, после чего на обучающей выборке создается модель, которая последовательно строит деревья решений, минимизируя ошибку предсказания с помощью градиентного бустинга.

Производительность модели XGBoost зависит не только от входных данных, но и настроек гиперпараметров.

Для модели "древесного" типа нет необходимости в предварительной нормализации входных данных, так как вариативность шкал признаков не влияет на количественную оценку загрязненности подвыборок. Под входными данными в обучении модели XGBoost подразумевается 13 признаков, которые были выбраны в предыдущей главе, и метка класса (0 - фон,1 - сигнал).

Гиперпараметры XGBoost приведены в таблице вместе с оптимальными значениями, определенными для данного анализа. Подбор гиперпараметров был реализован с помощью алгоритма жадного поиска. Данные гиперпараметры приведены в таблице 4.1 вместе с их оптимальным значением.

Параметр	**Значение**	**Описание**
eval_metric	logloss	Функция потерь
tree_method	hist	Метод построения деревьев
max_depth	3	Максимальная глубина дерева
min_child_weight	3	Минимальный вес для дочерних узлов
gamma	0.1	Минимальное уменьшение потерь для разделения узла
subsample	0.6	Доля выборки для обучения дерева
colsample_bytree	0.6	Доля признаков для построения дерева
learning_rate	0.01	Скорость обучения
sampling_method	uniform	Метод создания подвыборки
grow_policy	lossguide	Политика роста деревьев
lambda	0.5	Параметр L2-регуляризации
alpha	0.5	Параметр L1-регуляризации

Таблица 4.1 — Гиперпараметры модели XGBoost для задачи бинарной классификации

Настройка гиперпараметров в XGBoost с помощью алгоритма жадного

поиска включает в себя систематический подход к выбору оптимальных значений параметров модели. Предварительно набор гиперпараметров был раздлен на несколько групп. Данный этап был необходим, поскольку вычислительная сложность алгоритма жадного поиска $O(n^2)$. Для каждой группы отдельно тестировались различные комбинации значений гиперпараметров.

Алгоритм жадного поиска был настроен на 5-кратную перекрестную проверку моделей. Модели обучались на МК-данных с учетом веса событий. Диапазон циклов обучения конкретно одной модели был ограничен 10 итерациями.

Так как ансамбли деревьев с градиентным бустингом склонны к переобучению, одними из настраиваемых гиперпараметров были коэффициенты L1 и L2 регуляризации. Следует отметить, что оптимальные гиперпараметры выбирались исходя из значения метрики AUC.

Для проверки корректности обучения модели необходимо понимать как меняются метрики на обучающей и валидационной выборке. Если ошибка на обучающей выборке стремительно падает, а ошибка на валидационной выборке выходит на плато или начинает расти, то это может указывать на переобучение.

Обучение XGBoost было остановлено на 500 эпохе, так как при увеличении итераций модель начинает переобучаться.

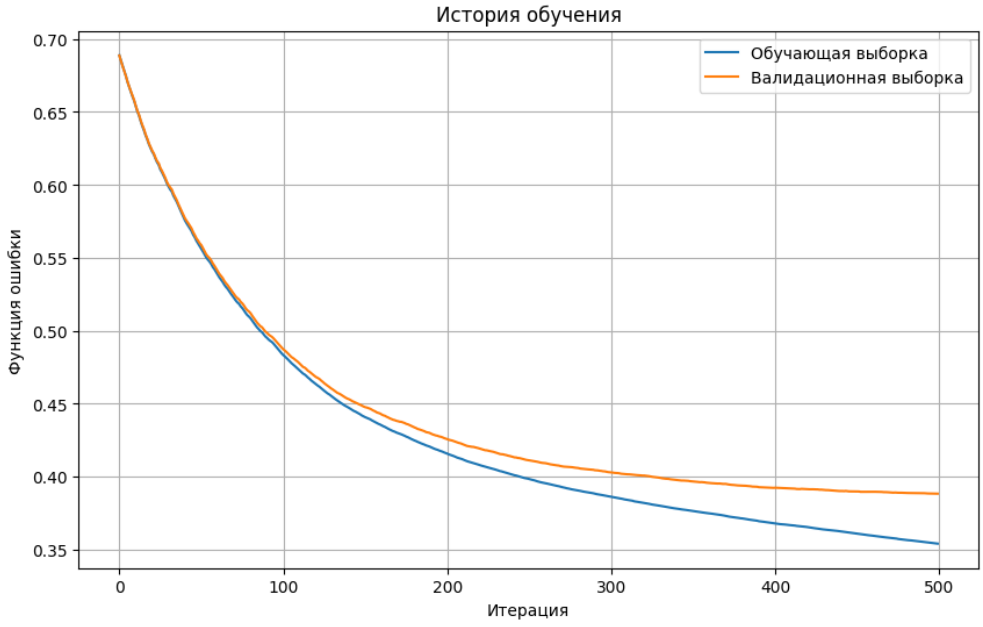

Рисунок 4.3 — История обучения XGBoost

В Приложении А представлено краткое изложение реализации данных действий на python, с которым можно ознакомиться для получения общего представления о процессе работы.

4.5 ОБУЧЕНИЕ МОДЕЛИ MLP

Производительность MLP зависит от гиперпараметров обучения, алгоритма оптимизации и архитектуры нейронной сети.

Так как каждое событие в МК-симуляциях имеет свой собственный вес, в процессе обучения нейронная сеть должна учитывать это при обновлении внутренних параметров. Вес события помогает нейронной сети учитывать важность каждого события при расчете функции потерь. В данной работе вес события интегрирован в лосс-функцию бинарной кросс-энтропии (формула 1.2).

На первом этапе необходимо определить наилучший алгоритм оптимизации для обучения модели на полученных выборках. Работа алгоритмов оценивалась на нейронной сети с простой архитектурой: входной слой - скрытый слой с 32 нейронами - выходной слой. Функция активации для скрытого слоя - relu. Для каждой модели были выбраны learning_rate = 0.001 и размер батча = 64. Обучение всех моделей останавливалось на 20 эпохе.

По метрике AUC и ROC-кривой можно оценить работу моделей с разными алгоритмами оптимизации. По полученным зависимостям и значениям можно сделать вывод, что наилучший алгоритм оптимизации RMSProp.

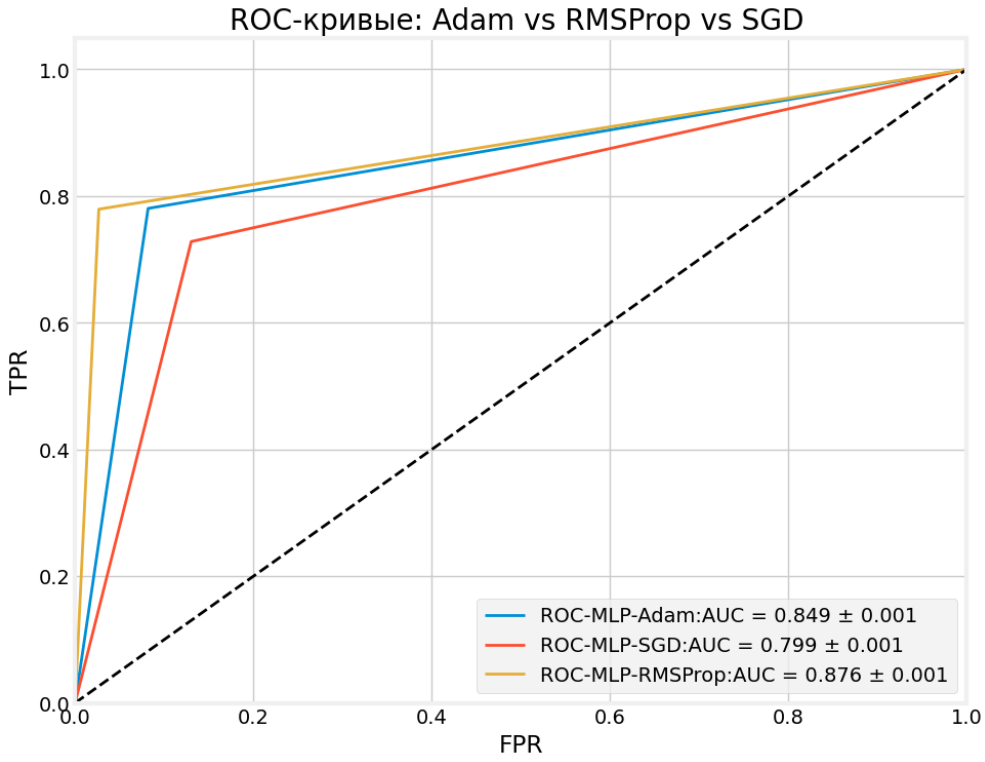

Рисунок 4.4 — ROC-кривые для MLP-Adam, MLP-SGD и MLP-RMSProp

Для обучения нейронной сети далее используется алгоритм оптимизации RMSProp. Следующим шагом является настройка гиперпараметров и архитектуры нейронной сети. Для нахождения оптимальных настроек на разных наборах значений слоев и параметров сети были обучены модели MLP. Выбор оптимальных настроек нейронной сети обусловлен наилучшим показателем метрики AUC.

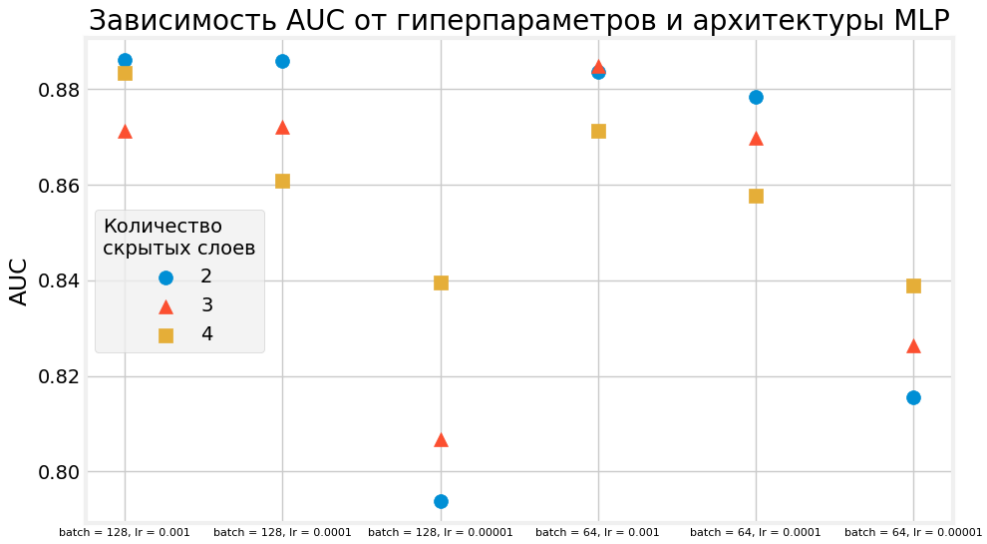

Рисунок 4.5 — AUC для MLP, обученных с различной скоростью обучения (lr), размером батча (batch) и количеством скрытых слоев

Итоговые гиперпараметры и архитектура MLP представлены в таблицах 4.2,4.3.

Обучение MLP было завершено на 20-й эпохе, поскольку после этого момента наблюдалась стабильность в динамике ошибки на валидационной выборке. Изменения значения функции потерь перестали демонстрировать значительный прогресс, что свидетельствует о достижении оптимума модели без явного переобучения. Такой подход позволяет избежать излишних итераций обучения, экономя вычислительные ресурсы и сохраняя при этом качество обобщения модели на новых данных. Визуальное представление изменения функций ошибки на обучающей и валидационной выборке представлено на рисунке 4.6.

В Приложении А представлено краткое изложение реализации указанных действий на языке python с использованием библиотеки Keras. С этим материалом можно ознакомиться для получения общего понимания ключевых этапов выполнения задач.

Параметр	**Значение**
Скорость обучения	0.001
Размер батча	128
Функция активации	relu
Алгоритм оптимизации	RMSProp
Количество эпох	20

Таблица 4.2 — Гиперпараметры модели MLP

Слой	**Характеристика**
Входной	13
Скрытый	32
Dropout	0.1
Скрытый	64
Dropout	0.1
Выходной	1

Таблица 4.3 — Архитектура MLP

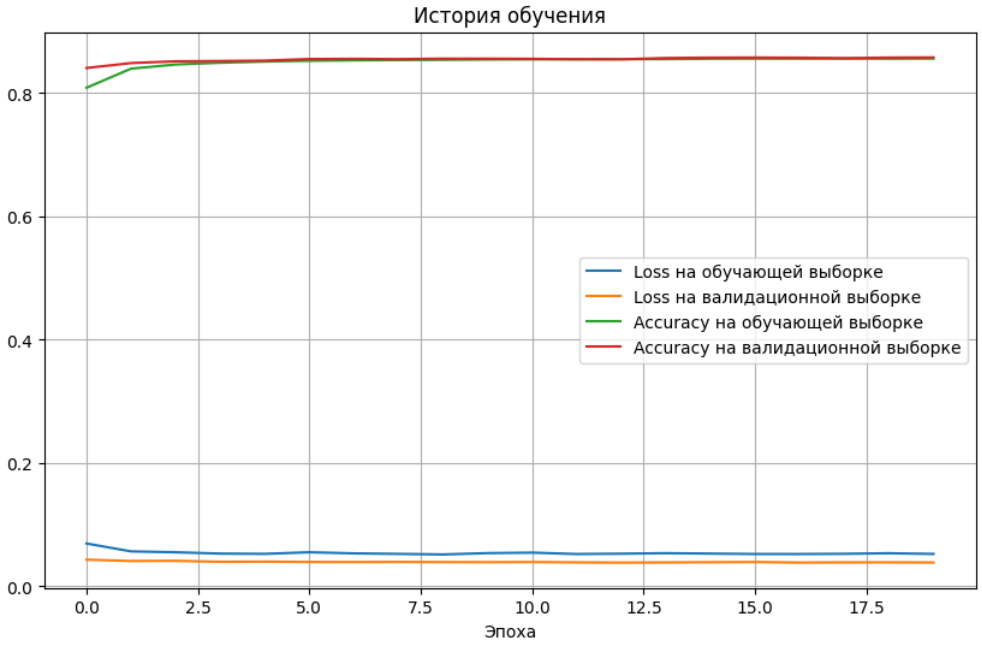

Рисунок 4.6 — История обучения MLP

5 РЕЗУЛЬТАТЫ

5.1 ТЕСТИРОВАНИЕ МОДЕЛЕЙ НА МОНТЕ-КАРЛО СИМУЛЯЦИЯХ

Тестирование моделей машинного обучения является важным этапом в процессе разработки, позволяющим оценить их производительность и обобщающую способность. В задаче бинарной классификации, где цель состоит в том, чтобы классифицировать объекты на два класса, тестирование моделей заключается в оценивании метрик, которые количественно отражают качество работы модели на тестовом наборе данных. В работе под тестовым набором данных понимается выборка Монте-Карло симуляций по всем процессам, которая не участвовала в процессе обучения модели.

Так как в данной задаче одинаково важны точности принадлежности объекта к любому классу, основной метрикой для оценки служит accuracy и ROC-AUC. Accuracy, как доля правильно классифицированных объектов, дает общее представление о производительности модели. ROC-AUC измеряет способность модели различать классы независимо от их частоты встречаемости в выборке для различных порогов классификации. Значение точности для каждой модели представлено в таблице 5.1.

Модель	**Accuracy**
MLP	0.860 ± 0.001
XGBoost	0.836 ± 0.002

Таблица 5.1 — Значение accuracy для обученных моделей на тестовой выборке

Также немаловажной оценкой производительности модели служит матрица ошибок. В данной задаче под значениями в матрице предполагается сумма весов при 4 различных вариантах. Анализ матрицы ошибок позволяет выявить, какие типы ошибок модель совершает чаще, и, следовательно, какие аспекты

модели требуют улучшения.

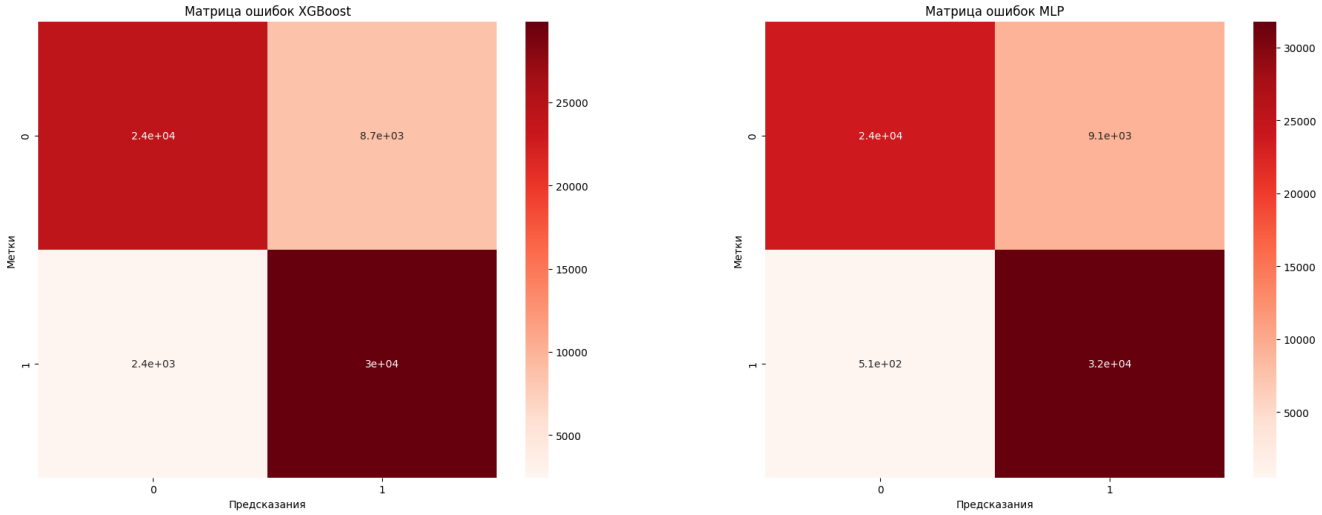

Рисунок 5.1 — Матрицы ошибок для XGBoost и MLP

По данным матрицам ошибок можно сделать вывод о том, что модели чаще всего совершают ошибки второго рода. Данные ошибки связаны с отбором событий при конструировании сигнального датасета из процесса образования пары топ-кварков. Несмотря на то, что в данном датасете существенная часть событий связана с образованием W-бозона, распавшимся по адронной моде, не исключаются и события, связанные с другими процессами.

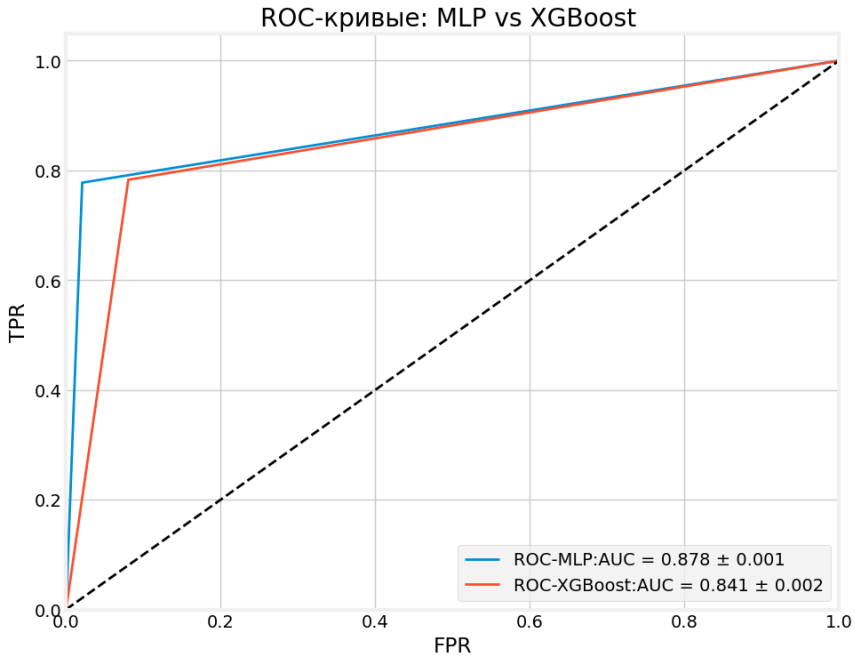

Рисунок 5.2 — ROC-кривые MLP и XGBoost

ROC-кривая позволяет сравнить и оценить результаты двух обученных моделей, визуализируя их способность различать классы при различных поро-

гах классификации. Модель, чья ROC-кривая находится выше и левее, обычно демонстрирует лучшую производительность, поскольку она обеспечивает более высокий TPR при более низком FPR. Сравнение ROC-кривых позволяет выбрать модель, которая лучше всего соответствует требованиям поставленной задачи.

Анализ распределения откликов моделей XGBoost и MLP показывает, что модели в целом успешно разделяют классы в задаче бинарной классификации. Однако, наблюдаются ошибки второго рода, которые связаны с особенностями формирования сигнального датасета.

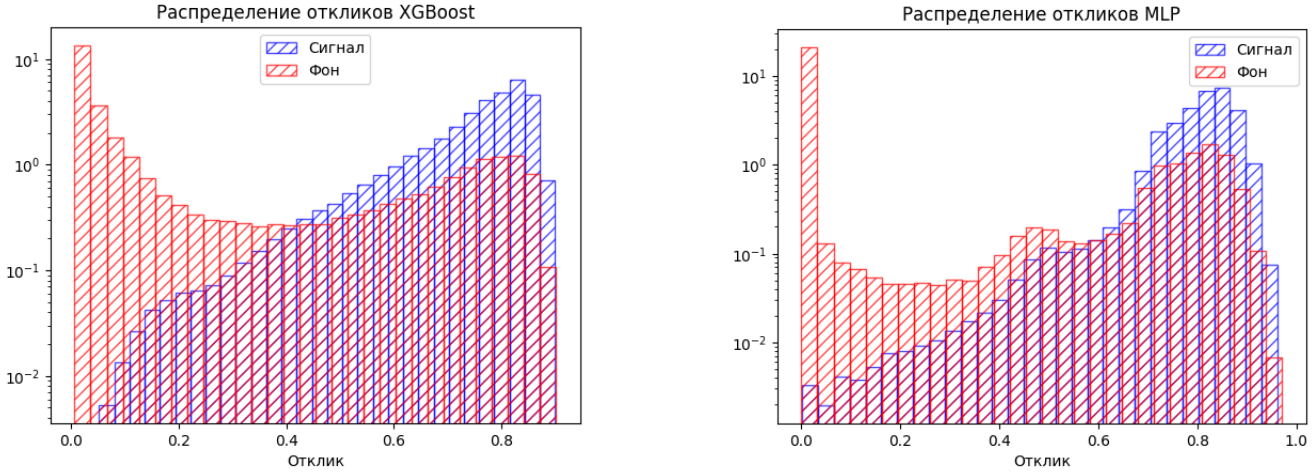

Рисунок 5.3 — Распределение откликов MLP и XGBoost

ВЫВОД

Тестирование моделей машинного обучения XGBoost и MLP с использованием выборки Монте-Карло симуляций показало, что обе модели демонстрируют хорошую производительность, о чем свидетельствуют значения accuracy и анализ ROC-кривых. Однако, анализ матриц ошибок и распределения откликов моделей выявил преобладание ошибок второго рода, что связано с особенностями формирования сигнального датасета.

5.2 ИССЛЕДОВАНИЕ ОТКЛИКА МОДЕЛЕЙ НА РЕАЛЬНЫХ ДАННЫХ

В рамках работы были подготовлены реальные данные с эксперимента ATLAS RUN2 с общей светимостью 6.1 фм$^{-1}$. Для исследования работы моделей машинного обучения необходимо сравнить плотность распределения откликов моделей на реальных данных и на Монте-Карло датасетах. Это сравнение является важным для оценки способности моделей адекватно классифицировать объекты в реальных данных.

Первым этапом исследования работы модели на реальных данных является сравнение этих данных с полным набором данных Монте-Карло процессов и фона от ложных электронов, который определялся из экспериментальных данных. На данном этапе была проведена по упрощенной схеме оценка фона, обусловленного ложными электронами, при конечном состоянии evJ.

В результате данного анализа было получено распределение по инвариантной массе струи, которое иллюстрирует соотношение между реальными данными и Монте-Карло симуляциями по всем процессам. Данное распределение представлено ниже.

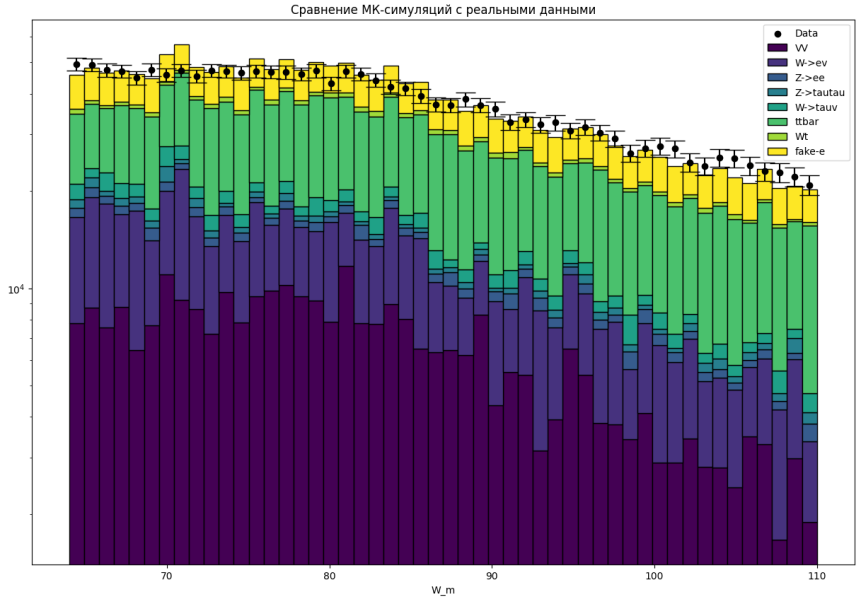

Рисунок 5.4 — Сравнение реальных данных с МК-симуляциями

Исходя из данного распределения можно сделать вывод, что расхождения между МК-симуляциями и реальными данными незначительны и обусловлены приблизительной оценкой фона от ложного электрона.

Для сравнения плотности распределения откликов моделей на реальных данных и Монте-Карло датасетах для всех процессов построены гистограммы с распределением для каждой модели.

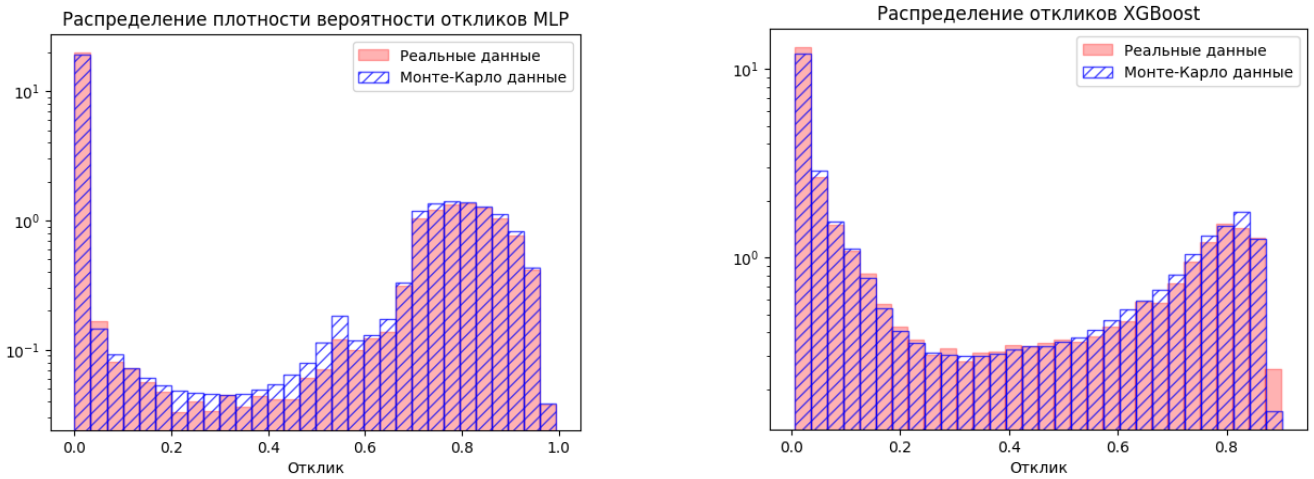

Рисунок 5.5 — Сравнение распределения откликов моделей MLP И XGBoost на реальных данных и на МК-симуляциях

По полученным гистограммам можно сделать вывод, что результаты работы моделей на реальных данных и на Монте-Карло симуляциях сходятся, что говорит о хороших обобщающих способностях модели и адекватности в классификации объектов.

5.3 СРАВНЕНИЕ ЭФФЕКТИВНОСТИ ИДЕНТИФИКАЦИИ W-БОЗОНА МЕЖДУ МОДЕЛЯМИ И ДИСКРИМИНИРУЮЩЕЙ ПЕРЕМЕННОЙ

В настоящий момент в анализе данных с эксперимента ATLAS для идентификации струй, образованных от распада W-бозона по адронной моде, используется переменная wtag_mass_d2. В данном методе мечения струй от W-бозона используются соотношения между комбинированной массой струи и функции энергетической корреляции D2. Если струя попадает в массовый интервал и коэффициент энергетической корреляции достаточно мал, то данная струя может быть помечена как кандидат в W-бозоны.

Переменная wtag_mass_d2 оптимизирована для достижения эффективности сигнала 50%. При фиксированной эффективности сигнала, равной 50%, была посчитана эффективность фона классификаторов MLP, XGBoost и дискриминирующей переменной wtag_mass_d2.

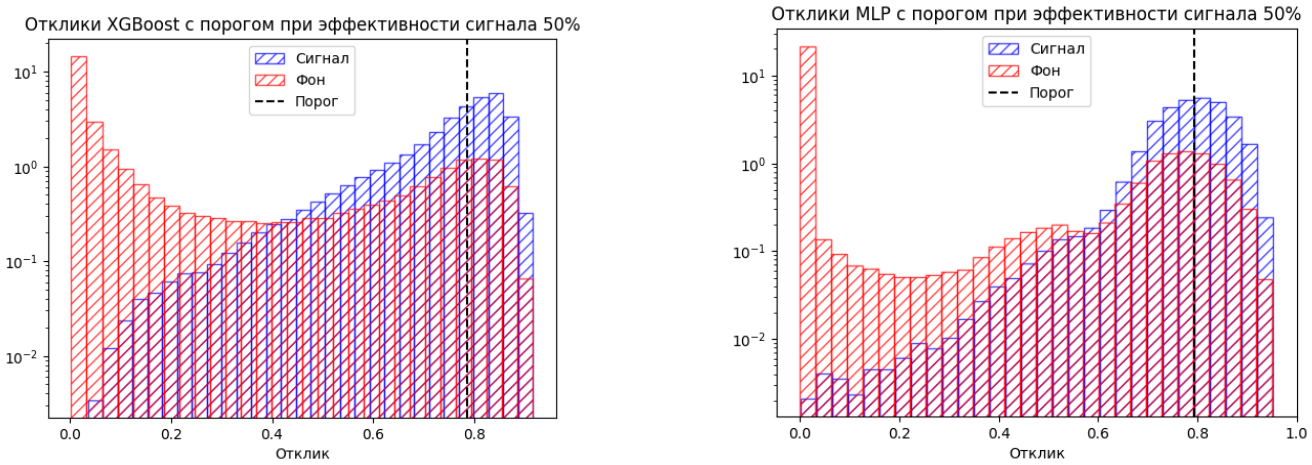

Рисунок 5.6 — Расчет порога классификации моделей при заданной эффективности сигнала 50%

Полученные значения представлены в таблице 5.2. По полученным эффективностям фона можно сказать, что результаты классификаторов MLP и XGBoost имеют меньше фонового содержания при заданной эффективности сигнала, чем результаты работы дискриминирующей переменной.

ЗАКЛЮЧЕНИЕ

В рамках ВКР были выполнены задачи, направленные на разработку эффективных методов идентификации W-бозонов с использованием машинного обучения. В частности:

1) На основе Монте-Карло симуляций сформирован датасет, учитывающий веса событий. Также был сформирован датасет реальных данных с $L = 6.1$ фм$^{-1}$;
2) Выбраны оптимальные признаки для последующего обучения моделей на основе анализа их значимости;
3) Проведена настройка гиперпараметров, архитектуры и алгоритмов оптимизации моделей машинного обучения;
4) С учетом веса каждого события, были обучены и протестированы модели MLP и XGBoost. Полученные метрики точности моделей на тестовой выборке представлены ниже:

Модель	**Accuracy**	**AUC**
MLP	0.860 ± 0.001	0.878 ± 0.001
XGBoost	0.836 ± 0.002	0.841 ± 0.002

5) Проведено исследование отклика моделей на реальных данных. Анализ полученных гистограмм демонстрирует хорошее соответствие результатов работы моделей на реальных данных;
6) При заданной эффективности сигнала 50% посчитаны эффективности фона для двух классификаторов и дискриминирующей переменной:

Классификатор	**Эффективность фона**
MLP	$10.5\% \pm 0.1\%$
XGBoost	$10.6\% \pm 0.1\%$
wtag_mass_d2	$30.9\% \pm 0.1\%$

Таблица 5.2

СПИСОК ЛИТЕРАТУРЫ

1. *Okun L. B.* Leptons and Quarks: Special Edition Commemorating the Discovery of the Higgs Boson. — Amsterdam, Netherlands : North-Holland, 1982. — ISBN 978-981-4603-14-0, 978-981-4603-00-3, 978-0-444-86924-1.
2. *Evans L.*, *Bryant P.* LHC Machine // Journal of Instrumentation. — 2008. — Т. 3, № 08. — S08001.
3. Search for excited electrons singly produced in proton–proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS experiment at the LHC / M. Aaboud [и др.] // Eur. Phys. J. C. — 2019. — Т. 79, № 9. — С. 803. — arXiv: 1906.03204 [hep-ex].
4. *Cheremushkina E.*, *Kamenshchikov A.*, *Zenin O.* Search for singly produced excited electrons in two leptons and jets final state using proton-proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector at the LHC : тех. отч. / CERN. — Geneva, 2016.
5. *Perkins D. H.* Introduction to high energy physics. — 1982. — ISBN 978-0-521-62196-0.
6. *Novaes S. F.* Standard Model: An Introduction. — 2000. — arXiv: hep-ph/0001283 [hep-ph].
7. Boosted hadronic vector boson and top quark tagging with ATLAS using Run 2 data : тех. отч. / CERN. — Geneva, 2020.
8. Performance of top-quark and W-boson tagging with ATLAS in Run 2 of the LHC / M. Aaboud [и др.] // Eur. Phys. J. C. — 2019. — Т. 79, № 5. — С. 375. — arXiv: 1808.07858 [hep-ex].
9. *Cacciari M.*, *Salam G. P.*, *Soyez G.* The anti-ktjet clustering algorithm // Journal of High Energy Physics. — 2008. — Т. 2008, № 04. — С. 063—063.
10. Школа Анализа Данных: Учебник по машинному обучению. — https://education.yandex.ru/handbook/ml/article/about.

11. The ATLAS Experiment at the CERN Large Hadron Collider // Journal of Instrumentation. — 2008. — Т. 3, № 08. — S08003.

12. *Nason P.* A New Method for Combining NLO QCD with Shower Monte Carlo Algorithms // Journal of High Energy Physics. — 2004. — Т. 2004, № 11. — С. 040—040.

13. *Chen T.*, *Guestrin C.* XGBoost: A Scalable Tree Boosting System // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. — ACM, 2016. — С. 785—794. — (KDD '16).

14. *Chollet F.* Deep learning with python. — New York, NY : Manning Publications, 2017.

15. *Larkoski A. J.*, *Salam G. P.*, *Thaler J.* Energy correlation functions for jet substructure // Journal of High Energy Physics. — 2013. — Т. 2013, № 6.

16. *Larkoski A. J.*, *Moult I.*, *Neill D.* Power counting to better jet observables // Journal of High Energy Physics. — 2014. — Т. 2014, № 12.

17. *Cui Y.*, *Han Z.*, *Schwartz M. D.* W-jet Tagging: Optimizing the Identification of Boosted Hadronically-Decaying W Bosons // Phys. Rev. D. — 2011. — Т. 83. — С. 074023. — arXiv: `1012.2077 [hep-ph]`.

18. *Khachatryan V.* Identification techniques for highly boosted W bosons that decay into hadrons // Journal of High Energy Physics. — 2014. — Т. 2014, № 12.

19. Longitudinally invariant K_t clustering algorithms for hadron hadron collisions / S. Catani [и др.] // Nucl. Phys. B. — 1993. — Т. 406. — С. 187—224.

Приложение А
Обучение и настройка моделей МО

BDT

```
from sklearn.model_selection import GridSearchCV
import xgboost as xgb
# Grid search for the first group of hyperparameters
param_grid = {
    'max_depth': [3, 4, 5, 6, 7, 8],
    'min_child_weight': [2, 3, 4, 5]
}
xgb_model = xgb.XGBClassifier(n_estimators=10, objective='binary:logistic',
    nthread=4, seed=42)
grid_search = GridSearchCV(estimator=xgb_model, param_grid=param_grid, scoring=
    'roc_auc', n_jobs=4, cv=5, verbose=1)
grid_search.fit(X_train, y_train, sample_weight=event_weight_train)
print(grid_search.best_params_)
# XGBoost Training
dtrain = xgb.DMatrix(data=X_train, label=y_train, weight=event_weight_train)
dtest = xgb.DMatrix(data=X_test, label=y_test, weight=event_weight_test)
dvalid = xgb.DMatrix(data=X_valid, label=y_valid, weight=event_weight_valid)
params = {
    'objective':'binary:logistic',
    'eval_metric':'logloss',
    'tree_method':'hist',
    'max_depth':3,
    'min_child_weight':3,
    'gamma':0.1,
    'subsample':0.6,
    'colsample_bytree':0.6,
    'learning_rate': 0.1,
    'sampling_method':'uniform',
    'grow_policy':'lossguide',
    'lambda':0.5,
    'alpha':0.5
}
evals_result = {}
xgbmodel = xgb.train(params=params,dtrain=dtrain,num_boost_round=1000,
```

```
                    evals=[(dtrain,'train'),(dvalid,'valid')],
                    evals_result=evals_result,
                    early_stopping_rounds=10,
                    verbose_eval=10)
```

MLP

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers import Adam, RMSprop, SGD
# Choosing an optimization algorithm. Example for RMSprop
model_rms = Sequential()
model_rms.add(Dense(32,activation='relu',input_dim=13))
model_rms.add(Dense(1,activation='sigmoid'))
model_rms.compile(optimizer=RMSprop(learning_rate=0.001,use_ema=True),loss='
    binary_crossentropy',metrics=['accuracy', 'AUC'])

history = model_rms.fit(
X_train_norm,
y_train,
epochs=20,
sample_weight=event_weight_train,
batch_size=64,
validation_split=0.2
)
# MLP Training
model_rms = Sequential()
model_rms.add(Dense(32, activation='relu', input_dim=13))
model_rms.add(Dropout(0.1))
model_rms.add(Dense(64, activation='relu'))
model_rms.add(Dropout(0.1))
model_rms.add(Dense(1, activation='sigmoid'))
model_rms.compile(optimizer=RMSprop(learning_rate=0.001, use_ema=True), loss='
    binary_crossentropy', metrics=['accuracy'])

history = model_rms.fit(
X_train_norm,
y_train,
epochs=20,
sample_weight=event_weight_train,
batch_size=128,
validation_split=0.2
)
```