БИБЛИОТЕКА АЛГОРИТМОВ СИЛЬНОГО ИИ.
Алгоритмов автономного оценивания и прогнозирования состояния сложных объектов и процессов на основе интеллектуальной обработки событий в условиях неопределенности и недостоверности данных
(промежуточный)
RU.СНАБ.00837-01 13 YY

Термины и сокращения

Сложный (технический) объект (сокр. *СлО*) – искусственно созданный объект, который имеет внутренне присущую ему структуру, может быть разделен на компоненты (элементы), которые связаны друг с другом большим количеством связей различных типов, выполняет несколько функций и функционирование которого связано с внутренней циркуляцией информационных потоков.

Сложный процесс (сокр. *СлП*) – процесс выполнения сложным объектом функций по своему предназначению.

Состояние (сложного объекта или процесса) – совокупность значений характеристик, которыми обладает сложный объект или процесс, в данный момент времени. Характеристики могут быть количественными (параметры) и качественными.

Оценивание состояния – последовательность действий, целью которой является определение значений характеристик сложного объекта или процесса в текущий или в прошедшие моменты времени с требуемыми точностью и достоверностью.

Прогнозирование состояния – последовательность действий, целью которой является определение значений характеристик сложного объекта или процесса в будущие моменты времени с требуемыми точностью и достоверностью.

Автономное оценивание (прогнозирование) состояния – процедура оценивания (прогнозирования) состояния, выполняемая с максимальным отсутствием вмешательства человека.

ИИ – Искусственный интеллект

Компонент АОПССОП – Компонент автономного оценивания и прогнозирования состояния сложных объектов и процессов

Алгоритм ПОСНД – Алгоритм предварительной обработки сырых и нечетких данных

Алгоритм ИЗСНД – Алгоритм извлечения знаний из собираемых наборов данных

Алгоритм АОТССОП – Алгоритм автономного оценивания текущего состояния сложных объектов и процессов

Алгоритм АПССОП – Алгоритм автономного прогнозирования состояний сложных объектов и процессов

Модуль АОТССОП – Модуль автономного оценивания текущего состояния сложных объектов и процессов

Модуль АПССОП – Модуль автономного прогнозирования состояний сложных объектов и процессов

НС – Нейронная сеть

СлОП – Сложный объект или процесс

1 ОБЩИЕ СВЕДЕНИЯ

Компонент Алгоритмы автономного оценивания и прогнозирования текущего состояния сложных объектов и процессов (сокр. ААОиПТССОиП) библиотеки алгоритмов сильного ИИ RU.СНАБ.00837-01 13 YY разработан в соответствии с мероприятием по выполнении п. 3.9 ТЗ программы ИЦИИ «Сильный ИИ в промышленности» в рамках федерального проекта «Искусственный интеллект».

Он предназначен для определения значений характеристик СлО в текущий, прошедшие и будущие моменты времени с требуемыми точностью и достоверностью.

Компонент разработан на языке Python (версия не ниже 3.7) с использованием следующих библиотек: data_classes, gzip, keraskeras, matplotlib,numpy, os, pandas, pickle, sklearn, tensorflow.

Компонент размещен в репозитории по адресу https://github.com/labcomsec/aopssop_lib.

Для его использования необходим интерпретатор Python (версия не ниже 3.7).

2 ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

2.1 Назначение программного компонента

В состав компонента входят 4 алгоритма, реализующие функции сильного ИИ, а именно следующие:

1) Алгоритм ПОСНД – в части улучшения качества предоставляемых данных и может быть задействован на подготовительной стадии оценивания и прогнозирования состояния;

2) Алгоритм ИЗСНД – в части построения модели СлО, основанной на знаниях. Алгоритм позволяет извлечь множество знаний, описывающих связи между понятиями и действиями в предметной области оценивания и прогнозирования состояния СлО, а также множество действий (выводов), вытекающих из этих знаний;

3) Алгоритм АОТССОП – в части наличия возможности автоматического извлечения высокоуровневых представлений исходных данных и взаимосвязей между ними. Наличие большого числа гиперпараметров и настраиваемых весов, свойственных для глубоких НС, позволяет с достаточно высокой точностью выявлять закономерности между признаками обрабатываемого объекта и оцениваемой меткой его класса;

4) Алгоритм АПССОП – в части выполнения автономного прогнозирования состояний СлОП. Наличие большого числа гиперпараметров и настраиваемых весов, свойственных для глубоких нейронных сетей, позволяет с достаточно высоким качеством выявлять закономерности между

признаками состояния системы и прогнозировать последующие состояния за заданный отрезок времени.

2.2 Классы решаемых задач

Каждый из 4 алгоритмов компонента решает следующие задачи:

1) Алгоритм ПОСНД:
   - коррекция типов входных данных о СлОП;
   - устранение неполноты входных данных о СлОП;
   - анализ информативности данных о СлОП;
   - устранение мультиколлинеарности данных о СлОП.
2) Алгоритм ИЗСНД:

   – извлечения фрагментов знаний, имеющихся в данных о СлОП, в виде ассоциативных правил (вида «ЕСЛИ <посылка>, ТО <следствие>»), содержащих в правой части (следствии) метку класса (англ. class association rules)

3) Алгоритм АОТССОП:

   – автономное оценивание текущего состояния СлОП, включающего наличие или отсутствие в текущий момент определенного вида функциональных неисправностей, дефектов и атакующих воздействий, свойственных целевой системе;

4) Алгоритм АПССОП:

   – обучение модели прогнозирования состояний СлОП на основе исторических данных в виде временного ряда, представляющего собой последовательность векторов характеристик состояний СлОП;

   – автономное прогнозирование состояний СлОП, описываемых определенным вектором характеристик, за заданный отрезок времени.

2.3 Функциональные ограничения на применение

Экспериментально обоснованные функциональные ограничения на применение алгоритмов компонента сильного ИИ являются следующими:

1) для алгоритма ПОСНД: для алгоритма ПОСНД:

- по коррекции типа данных:
  a. порог уникальности значений признаков, устанавливаемый пользователем;
  b. максимальный размер семантического словаря (встроенного или пользовательского) для анализа признаков;
  c. максимальная длина последовательностей значений признака;
- по устранению неполноты данных:
  a. максимальное количество кластеров, на которые предполагается разбивать данные;
  b. максимальное количество узлов, реализующих параллельную обработку кластеров;
- по устранению мультиколлинеарности признаков:

a. минимальное/максимальное значение порога информативности признаков;

2) для алгоритма ИЗСНД:

– максимальный размер / объем обучающих данных, характеризующих СлО, заданных множеством описаний;

– максимальное количество агрегатов, которые объединяют отдельные значения каждого признака;

– максимальное количество ассоциативных правил;

– минимальное/максимальное пороговое значение метрики информативности;

3) для алгоритма АОТССОП:

– экспериментальный набор данных должен быть разбит на две части, одна из которых используется для обучения модели, а другая – для ее тестирования;

– набор данных должен представлять собой набор записей, каждая из которых описывает состояние исследуемого объекта в определенный момент времени;

– каждая запись из набора данных должна представлять собой последовательность числовых признаков фиксированной размерности, при этом величина этой размерности должна быть постоянной для всех записей;

– оценка состояния должна выполняться на основе метки класса, которая должна быть присвоена каждой записи из набора данных;

4) для алгоритма АПССОП:

– прогнозируются только числовые параметры состояний СлОП; работа с категориальными характеристиками не поддерживается, если они не были предварительно закодированы в числовые значения;

– для обучения модели используется фиксированный вектор характеристик для каждого состояния СлОП, длина и последовательность характеристик должна быть неизменной для состояния СЛоП в каждый момент времени;

– прогнозирование с использованием обученной модели осуществляется только для фиксированного вектора характеристик, заложенного в обученную модель;

– прогнозирование с использованием обученной модели осуществляется только на основе текущей или смоделированной последовательности состояний СлОП за промежуток времени, равный длине исторической последовательности, заложенной в обученную модель.

2.4 Область применения

Областью применения являются компьютерные системы и сети большой размерности (включая Интернет вещей, КФС), автономные РТК (БПЛА, БПА и др.), объекты в ЧС и т.п.

К направлениям применения интеллектуальных средств оценки и прогнозирования сложных технических объектов и процессов можно отнести

различные программы модернизации инфраструктуры предприятия, совершенствования промышленных установок, повышения срока их службы и снижения вероятности возникновения инцидентов на них.

Представленные интеллектуальные автономные алгоритмы позволят на основе имеющихся исторических и статистических данных бизнес-процессов выявлять их некорректные состояния и переходы, связанные в том числе со следующими событиями:

– неправильная настройка оборудования;
– износ отдельных деталей;
– возможные ошибки и неправильное использование различного технического оборудования персоналом;
– злонамеренными воздействиями со стороны внешних или внутренних нарушителей информационной безопасности.

Кроме того, использованием таких алгоритмов будет способствовать определению основных закономерностей и тенденций в дальнейшем ходе индустриальных процессов и прогнозирования дальнейших состояний технических объектов и особенностей, а также характеристик проистекающих в них процессов.

3 ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1 Используемые методы

Используются следующие методы в каждом из 4 алгоритмов, составляющих компонент.

Алгоритм ПОСНД

В основе алгоритма ПОСНД лежат следующие принципы.

Коррекция типов данных происходит на основе статистических критериев, связанных с количеством уникальных значений признака, обнаружением последовательностей значений признака, анализом текстовых названий признаков, а также анализом дробных значений для представления некорректных FLOAT (*.0(0)) в качестве INT. Порог уникальности выбирается алгоритмом автоматически или может быть задан пользователем. Также алгоритм по умолчанию использует встроенный семантический словарь для анализа названий признаков, однако предусмотрена возможность передачи пользовательского семантического словаря.

Устранение неполноты данных происходит на основе статистического усреднения данных в рамках отдельных кластеров. Разбиение данных на кластеры предполагается на основе таких подходов, как K-means, Mini Batch K-means, Mean-shift, OPTICS, Bisecting K-means. Важной особенностью алгоритма является параллельная обработка каждого кластера, что значительно ускоряет обработку больших объемов данных. Подход к кластеризации данных выбирается алгоритмом автоматически или может быть задан оператором.

Устранение мультиколлинеарности происходит на основе коэффициентов корреляции Пирсона и Спирмена, а также дисперсионного анализа (ANOVA). Ключевой особенностью алгоритма является распараллеливание процесса анализа информативности признаков с последующим удалением неинформативных. Критерий корреляции и порог информативности выбираются алгоритмом автоматически или могут быть заданы оператором.

Алгоритм ИЗСНД

В основе алгоритма ИЗСНД лежат следующие математические соотношения. Обучающие данные, характеризующие СлО, заданы множеством описаний $\mathbf{X} = \{X_i\}$, $i = 1, \ldots, I$, СлО (экземпляров), каждому из которых поставлена в соответствие метка класса из множества $\mathbf{Y} = \{Y_j\}$, $j = 1, \ldots, J$, задающая состояние СлО в некоторый момент времени. Метки классов $Y_j \in \mathbf{Y}$ (состояния СлО) могут быть представлены в категориальный шкале (в простейшем случае, булевыми) или быть линейно упорядоченными.

Признаки $F_k \in \mathbf{F}$, каждый из которых имеет область допустимых значений $\mathbf{D}_k = \{d_k^l\}$, $k = 1, \ldots, K$, $l = 1, \ldots, L$, характеризующие описания СлО, $X_i \in \mathbf{X}$, также могут быть представлены в категориальной, булевой и/или числовой шкалах, а также могут быть текстами на естественном языке. Числовые признаки должны предварительно быть квантованы и далее могут рассматриваться как дискретные линейно упорядоченные признаки. Для этого можно использовать стандартные процедуры дискретизации. Для текстов необходимо предварительно выполнить процедуру векторизации с помощью стандартной процедуры, например, используя метрику TF-IDF.

Для построения ассоциативных правил на множестве обучающих данных на первом шаге необходимо объединить отдельные значения $d_k^l \in \mathbf{D}_k$ каждого признака в агрегаты $\mathbf{A}_k^{(j)}$, обладающие общим свойством по отношению к некоторому состоянию СлО, заданному меткой класса $Y_j \in \mathbf{Y}$. Таким образом, в ходе построения агрегатов область значений $\mathbf{D}_k$ каждого атрибута $F_k \in \mathbf{F}$ разделяется на непересекающиеся подмножества $\mathbf{A}_k^{(j)} \subseteq \mathbf{D}_k$, поставленные в соответствие некоторым классам множества $Y_j \in \mathbf{Y}$. Подмножества $\mathbf{A}_k^{(j)}$ строятся таким образом, чтобы каждое из них содержало те значения признака $F_k \in \mathbf{F}$ из множества его значений $\mathbf{D}_k$, которые более характерны для описаний СлО, $X_i \in \mathbf{X}$, принадлежащих классу Y_j, чем для описаний, принадлежащих другим классам.

Для построений подмножеств-агрегатов могут быть использованы различные метрики оценки частотности. В алгоритме предлагается использовать метод, основанный на наивном байесовском подходе. А именно, предполагается, что некоторое значение $d_k^l \in \mathbf{D}_k$ признака $F_k \in \mathbf{F}$ принадлежит подмножеству $\mathbf{A}_k^{(j)}$, $d_k^l \in \mathbf{A}_k^{(j)}$, в том случае, когда

$$p(Y_j \mid d_k^l) > p(\bar{Y}_j \mid d_k^l) + \delta. \qquad (3.1.1)$$

где $p(Y_j \mid d_k^l)$ – условная вероятность класса Y_j при $F_k = d_k^l$, $p(\bar{Y}_j \mid d_k^l)$ – это условная вероятность любого класса, кроме класса Y_j, при $F_k = d_k^l$, а δ – некоторая положительная константа, значение которой позволяет регулировать мощность формируемых подмножеств-агрегатов. Отметим, что с помощью теоремы Байеса выражение (3.1.1) можно привести к виду, требующему вычисления только $p(d_k^l \mid Y_j)$ u $p(d_k^l \mid \bar{Y}_j)$ и априорных вероятностей классов. И такие вероятности могут быть подсчитаны всего за один проход по обучающим данным, что делает описываемый алгоритм вычислительно эффективным.

Далее предлагается построить унарные предикаты $U_k^{(j)}(d_k^l \in \mathbf{A}_k^{(j)})$, каждый из которых принимает истинное значение в том случае, когда значение d_k^l признака F_k для некоторого описания СлО, $X_i \in \mathbf{X}$ принадлежит подмножеству-агрегату $\mathbf{A}_k^{(j)}$, $d_k^l \in \mathbf{A}_k^{(j)}$. Таким образом, подмножество $\mathbf{A}_k^{(j)}$ является областью истинности унарного предиката $U_k^{(j)}$, а для значений $d_k^l \notin \mathbf{A}_k^{(j)}$ предикат $U_k^{(j)}$ принимает значение «ложь».

Используя построенные предикаты далее в качестве посылки, а метки класса $F_k \in \mathbf{F}$, соответствующие некоторому состоянию СлО, в качестве следствия, могут быть сформированы ассоциативные правила класса следующего вида:

$$U_k^{(j)}\left(d_k^l \in \mathbf{A}_k^{(j)}\right) \rightarrow Y_j, j = 1, \dots, J;\ k = 1, \dots, K;\ l = 1, \dots, L. \quad (3.1.2)$$

Такие правила можно интерпретировать как знания, описывающие связи между понятиями и действиями в предметной области оценивания и прогнозирования состояния СлО, а также множество действий (выводов), вытекающих из знаний.

С целью сокращения множества подученных правил и выявления наиболее информативных из них для каждого правила далее необходимо рассчитать значение метрики информативности $\mu(U_k^{(j)}, Y_j)$. Предполагается, что метрика информативности поступает на вход алгоритма в качестве параметра. В простейшем случае может использоваться классическая метрика уверенности или более экзотические метрики, например, мера Клозгена или коэффициент регрессии событий, которые претендуют на то, чтобы оценивать также силу причинной связи между посылкой и следствием правила. Важным условием для метрики является ее несимметричность.

После вычисления значений выбранной метрики для каждого правила, множество правил сортируется в порядке убывания модуля значения метрики. В итоговом множестве правил останутся правила вида (3.1.1), удовлетворяющие условию: $\left|\mu\left(U_k^{(j)}, Y_j\right)\right| \geq \mathrm{M}$, где M – это пороговое значение метрики, которое выбирается экспериментально, исходя из мощности полученного множества правил и среднего значения модуля метрики для правил.

В результате такой фильтрации итоговое множество правил будет содержать только наиболее «сильные» правила, отражающие наиболее «сильные» связи параметров описаний СлО с его состоянием.

Полученное множество правил можно также интерпретировать как новое булево признаковое пространство, соответственно к нему можно применять любые методы сокращения пространства признаков, позволяющие обрабатывать признаки в булевой шкале.

Алгоритм АОТССОП

Алгоритм автономного оценивания текущего состояния сложных объектов и процессов основывается на математических соотношениях из линейной алгебры и дифференциального исчисления. В частности, метод градиентного спуска является основой алгоритма обратного распространения ошибки, который используется для обучения нейросетевых структур данного вида. Концептуальная модель системы, на оценивание и прогнозирование состояний и процессов функционирования которой направлены разрабатываемые алгоритмы, включает элементы $\{V_t\}_{t\in T}$, где $V_t = V_t(p_1, \dots, p_n)$ – кортеж, включающий n переменных p_i, i = 1,...n, описывающих состояние системы в момент времени t. Множество переменных $P = P_d \cup P_e$, где P_d – характеристики (параметры) системы, пригодные для технического диагностирования системы [17], P_e – параметры внешнего окружения системы и внешних воздействий. Множество рассматриваемых кортежей включает следующие выборки: обучающую выборку признаков Xij, j=1..N; обучающую выборку меток классов состояний сложных объектов Yij, j=1..N; тестовую выборку признаков Zij, i=1..M.

Решаемая научная задача по разработке алгоритмов оценивания состояния сложных объектов и процессов описывается функцией F, сопоставляющей следующие входные и выходные данные:

$$F: \left\{\left(p_1^{(t_i)}, \dots p_n^{(t_i)}, t_i\right)\right\}_{i\in\mathbb{N}, t\in T_0, T_0\subseteq T} \rightarrow \left\{\left(m_1^{(t_i)}, \dots m_N^{(t_i)}, pr_i\right)\right\}_{i\in\mathbb{N}, t\in T_0, T_0\subseteq T}, \quad (3.1.11)$$

где $m_j^{(t_i)}$ – значение целевого показателя системы в момент времени t_0. Величина pr_i – скалярная (либо векторная) априорно формируемая вероятностная оценка соответствия результирующего вектора $(m_1^{(t_i)}, \dots m_r^{(t_i)})$ фактическому состоянию системы в момент времени t_0.

Каждый из показателей $m_j^{(t_i)}$ может принимать бинарное или числовое значение. Примерами показателей со значениями $m_j^{(t_i)}$ являются:

– наличие или отсутствие функциональных неисправностей в системе в момент времени t_0;

– наличие или отсутствия в системе дефектов материалов физических конструкций системы в момент времени t_0;

– наличие или отсутствие непреднамеренных и преднамеренных воздействий на систему в момент времени t_0.

Алгоритм АПССОП

Алгоритм автономного прогнозирования состояний сложных объектов и процессов основывается на следующих математических соотношениях. Обучающие данные, характеризующие СлО, заданы множеством описаний (экземпляров), представляющих собой упорядоченную во времени последовательность векторов характеристик СлО:

$$\mathbf{X} = \{X_t\},\ t = 1,\ \ldots,\ T, \tag{3.1.12}$$

где T – длина временного ряда (количество экземпляров).

Каждый экземпляр содержит числовой вектор характеристик состояния СлО:

$$Xi = \{x_{mt}\},\ m = 1,\ \ldots,\ M, \tag{3.1.13}$$

где M – фиксированная длина вектора (количество характеристик).

Для обучаемой модели АПССОП задается параметр длины исторической последовательности данных для прогнозирования L – количество упорядоченных во времени экземпляров $\{X_t,\ \ldots,\ X_{t+L}\}$, необходимых для прогнозирования последующего экземпляра X_{t+L+1}.

На основе обучающих данных при помощи функции G производится формирование генератора временных рядов, представляющего собой формат данных, состоящих из пары (X^h, X^f):

$$\begin{gathered} G\colon \mathbf{X} \rightarrow (\mathbf{X}^h, \mathbf{X}^f) \\ \mathbf{X}^h = \left\{X_1^h, \ldots, X_{T-L-1}^h\right\},\ X_i^h = \{X_t, \ldots, X_{t+L}\},\ X_t^h \in \mathbf{X}^h \\ \mathbf{X}^f = \left\{X_1^f, \ldots, X_{T-L-1}^f\right\},\ X_t^f = X_{t+L+1},\ X_t^f \in \mathbf{X}^f, \end{gathered} \tag{3.1.14}$$

где $\mathbf{X}^h$ – множество пакетов, упорядоченных во времени исторических экземпляров для прогнозирования, $\mathbf{X}^f$ – множество целевых экземпляров для прогнозирования. Любому пакету X_i^h, соответствует последующий во времени экземпляр X_i^f.

Множество рассматриваемых кортежей включает следующие выборки: обучающую выборку $\mathbf{X}r = \{X_1, \ldots, X_t\}$, $t < T$, и тестовую выборку $\mathbf{X}_e = \{X_{t+1}, \ldots, X_T\}$. Выборки строятся таким образом, что обучающая выборка содержит временной ряд данных, предшествующий временному ряду в тестовой выборке.

Прогноз осуществляется таким образом, что для предсказания параметров событий в следующий момент времени необходимо проанализировать предшествующие события во временном окне размерностью L. Задача прогнозирования состояний сложных объектов и

процессов может быть описана при помощи функции Ф, сопоставляющей следующие входные и выходные данные:

$$\phi\colon\ \{X_t, \dots, X_{t+L}\} \to X_{t+L+1},\ t \in T. \tag{3.1.15}$$

Выходными данными являются прогнозируемые состояния СлО за заданный отрезок времени Δ:

$$\begin{gathered} \Phi\colon\ \mathbf{X} \to \mathbf{X}^*, \\ \mathbf{X}^* = \{X^*_{T+1}, \dots, X^*_{T+\Delta}\}, \end{gathered} \tag{3.1.16}$$

где **X*** – множество прогнозируемых векторов состояний СлО.

3.2 Алгоритмы компонента

Алгоритмы компонента программы состоит из 4 следующих, выполняемых последовательно.

3.2.1 Алгоритм ПОСНД

Алгоритм предназначен для работы с подаваемыми на вход наборами данных и включает в себя коррекцию типов данных, а также устранение неполноты и мультиколлинеарности данных о сложных объектах или процессах.

Алгоритм является вспомогательным для других алгоритмов, реализует функции сильного ИИ в части улучшения качества предоставляемых данных и может быть задействован на подготовительной стадии оценивания и прогнозирования состояния.

Блок-схема алгоритма представлена на Рис. 3.2.1.1.

Алгоритм ПОСНД работает с плоскими входными данными, представленными в табличной форме:

$$D^I = (F^M, F^T, F^S, L^R, L^T, L^S, E^C)\ ,$$

где $F^M = \{F^M_{on}\},\ o \in 1..O, n \in 1..N$ – матрица признаков, O – количество объектов, N – количество признаков;

$F^T = \{F^T_n\}, n \in 1..N$ – вектор типов признаков;

$F^S = \{F^S_n\}, n \in 1..N$ – вектор имен признаков;

$L^R = \{L^R_{om}\},\ o \in 1..O, m \in 1..M$ – матрица лейблов, M – количество лейблов;

$L^T = \{L^T_m\}, m \in 1..M$ – вектор типов лейблов;

$L^S = \{L^S_m\}, m \in 1..M$ – вектор имен лейблов;

$E^C = \{E^C_k\}, k \in 1..K$ – вектор координат пустых значений, K – количество пустых значений.

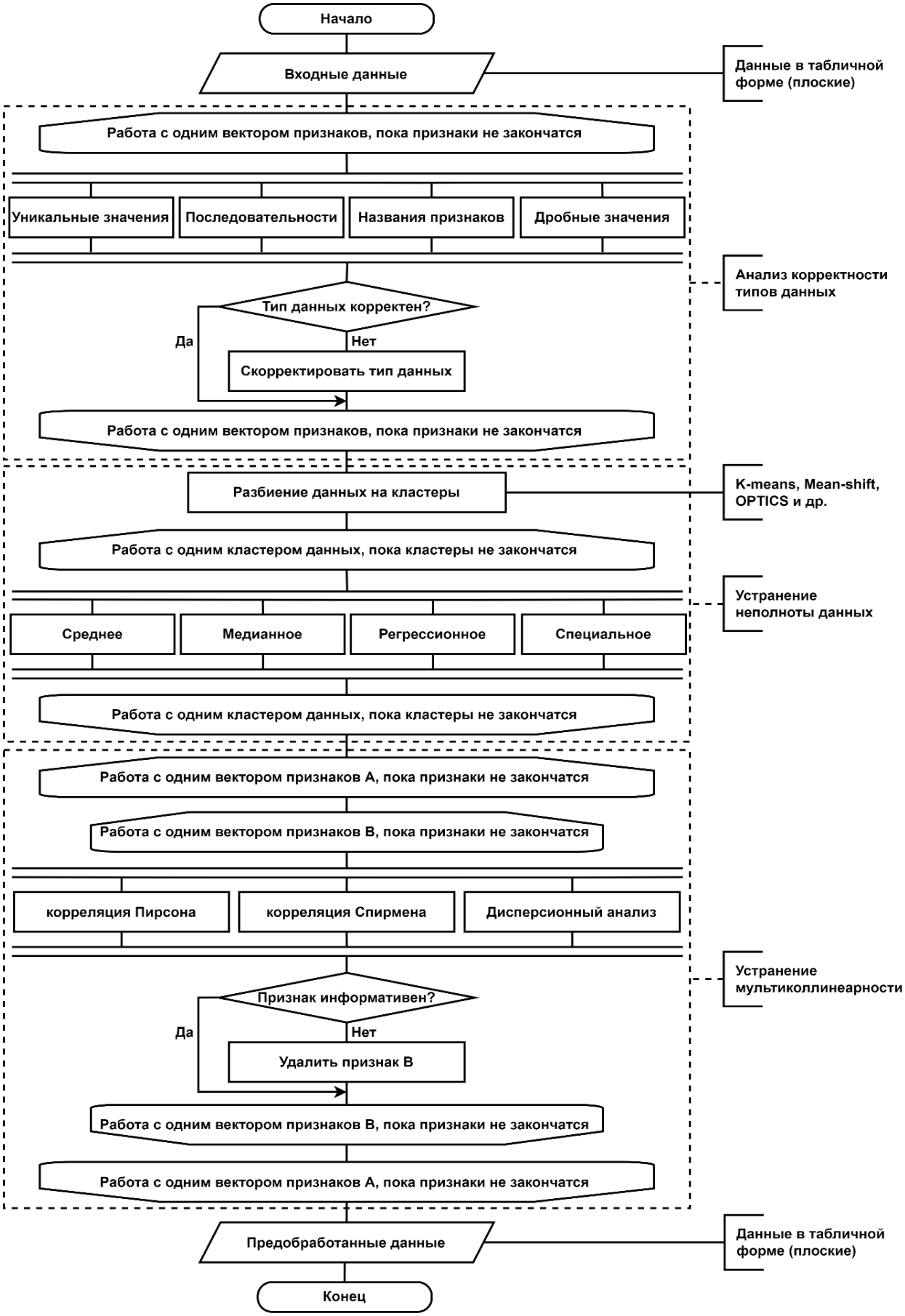

Рисунок 3.2.1.1 – Блок-схема алгоритма ПОСНД

На первом шаге алгоритма осуществляется анализ корректности типов данных для каждого признака из входных данных. При этом обработка каждого признака осуществляется параллельно. Более того, корректность типа данных оценивается на основе количества уникальных значений признака, обнаруженных последовательностей, анализа названий признаков, а также на основе обоснованности использования дробных значений, в случае их наличия. При этом вердикт о корректности типа данных выносится алгоритмами оценки отдельно, в то время как итоговое решение принимается на основе всех четырех оценок с учетом их весовых коэффициентов. Незаполненные поля на данном шаге игнорируются. Информация о корректировке типов данных сохраняется алгоритмом.

Второй шаг алгоритма направлен на устранение неполноты данных на основе статистического усреднения данных в рамках отдельных кластеров. Вначале данные разбиваются на отдельные кластеры на основе таких подходов, как K-means, Mini Batch K-means, Mean-shift, OPTICS, Bisecting K-means. Выбор метода кластеризации осуществляется автоматически, однако возможность выбора метода кластеризации оператором также предусмотрена. В дальнейшем, каждый признак каждого кластера обрабатывается параллельно. Устранение неполноты данных представляет собой автоматическое заполнение значений признаков, которые были пропущены. При этом заполнение может осуществляться на основе среднего, медианного или регрессионного значения по кластеру, а также на основе замены специальным значением. Важно отметить, что данные о произведенных преобразованиях сохраняются алгоритмом.

На третьем шаге алгоритма осуществляется устранение мультиколлинеарности входных данных за счет удаления признаков низкой информативности. Анализ информативности признаков осуществляется параллельно для каждой пары признаков. При этом в зависимости от типов анализируемых признаков (только численные, только категориальные, численные и категориальные), информативность вычисляется с помощью критерия корреляции Пирсона или Спирмена, а также методом ANOVA. Итогом работы алгоритма являются плоские данные в табличной форме, в которых указаны корректные типы данных, заполнены пропущенные значения, а также удалены неинформативные признаки.

3.2.2 Алгоритм ИЗСНД

Алгоритм предназначен для извлечения фрагментов знаний, имеющихся в данных о сложных объектах, в виде ассоциативных правил (вида «ЕСЛИ <посылка>, ТО <следствие>»), содержащих в правой части (следствии) метку класса (англ. class association rules).

Алгоритм ИЗСНД реализует функции сильного ИИ в части построения модели сложных объектов, основанной на знаниях. Алгоритм позволяет извлечь множество знаний, описывающих связи между понятиями и действиями в предметной области оценивания и прогнозирования состояния

сложных объектов, а также множество действий (выводов), вытекающих из этих знаний.

Блок-схема алгоритма ИЗСНД представлена на Рис. 3.2.2.1.

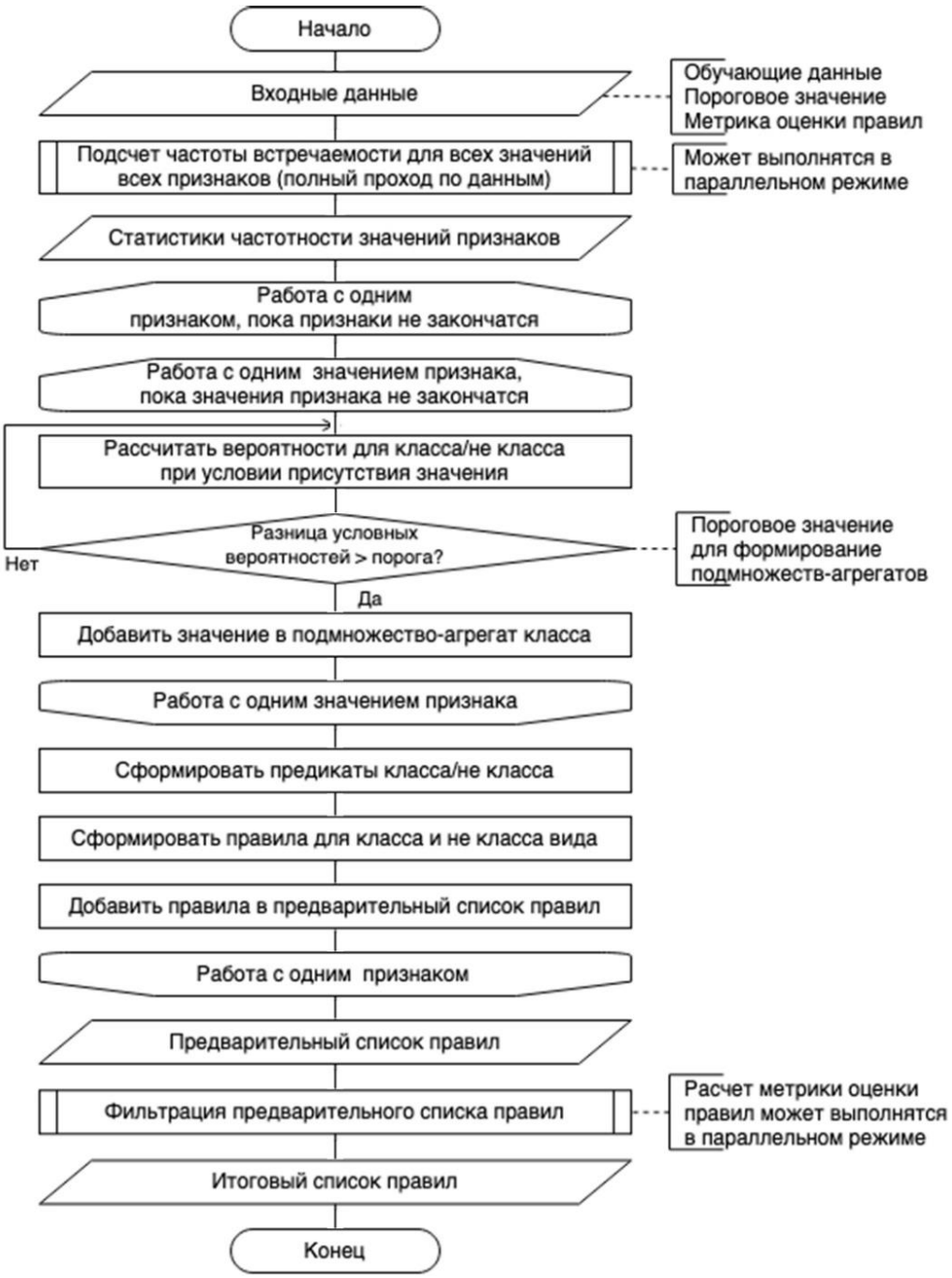

Рисунок 3.2.2.1 – Блок-схема алгоритма ИЗСНД

На первом шаге алгоритма необходимо рассчитать статистики встречаемости всех уникальных дискретных значений всех признаков и условные вероятности для классов на их основе. Это может быть выполнено за один проход по данным. Далее в циклах выполняется проход по всем атрибутам и всем уникальным значениям этих признаков, подсчет условных вероятностей для этих значений и проверка условия (3.1.1). Эти шаги могут выполняться в параллельном режиме. Значения, удовлетворяющие условию

(3.1.1), добавляются в подмножества-агрегаты, для которых в конце каждой итерации внешнего цикла формируются предикаты и правила класса вида (3.1.2) на их основе. В результате выполнения циклов формируется предварительный список правил. Далее для каждого правила из предварительного списка рассчитывается метрика оценки силы правила и выполняется их фильтрация на основании значения метрики. Для этого не требуется дополнительный проход по данным, так как можно использовать статистики, рассчитанные на первом шаге алгоритма. Результатом работы алгоритма будет являться множество ассоциативных правил класса, отражающих наиболее сильные связи параметров описаний СлО с его состоянием.

3.2.3 Алгоритм АОТССОП

Алгоритм предназначен для автономного оценивания текущего состояния сложных объектов и процессов, включающего наличие или отсутствие в текущий момент определенного вида функциональных неисправностей, дефектов и атакующих воздействий, свойственных целевой системе. Алгоритм строится на основе глубокой нейронной сети прямого распространения сигнала, обучение которой производится на объединенном наборе данных, полученном из открытых источников и включающем данные о функционировании нескольких видов индустриальных систем.

Алгоритм автономного оценивания текущего состояния сложных объектов и процессов реализует функции сильного ИИ в части наличия возможности автоматического извлечения высокоуровневых представлений исходных данных и взаимосвязей между ними. Наличие большого числа гиперпараметров и настраиваемых весов, свойственных для глубоких НС, позволяет с достаточно высокой точностью выявлять закономерности между признаками обрабатываемого объекта и оцениваемой меткой его класса.

Блок-схема алгоритма представлена на Рис. 3.2.3.1.

Начало реализации алгоритма заключается в выборе архитектуры нейронной сети (НС) и инициализации ее гиперпараметров. Следующий этап включает задание величины среднеквадратичной ошибки (СКО, *mse*) и числа эпох обучения НС. Далее выполняется разбиение исходного набора данных в заданном пользователем соотношении, одна из образовавшихся подвыборок используется в качестве обучающей (*Xij* – признаки, *Yij* – метки классов) для настройки весовых коэффициентов НС, другая часть – для тестирования. В процессе обучения выполняется корректировка весовых коэффициентов нейронной сети до тех пор, пока не будет достигнута требуемая величина СКО или число эпох обучения не превысит заданного значения. После этого вычисляется показатель точности (*acc*) НС на элементах тестовой выборки (*Zij*), на основании этого значения принимается решение о повторном обучении нейросетевой модели или ее применении для оценки состояния объекта.

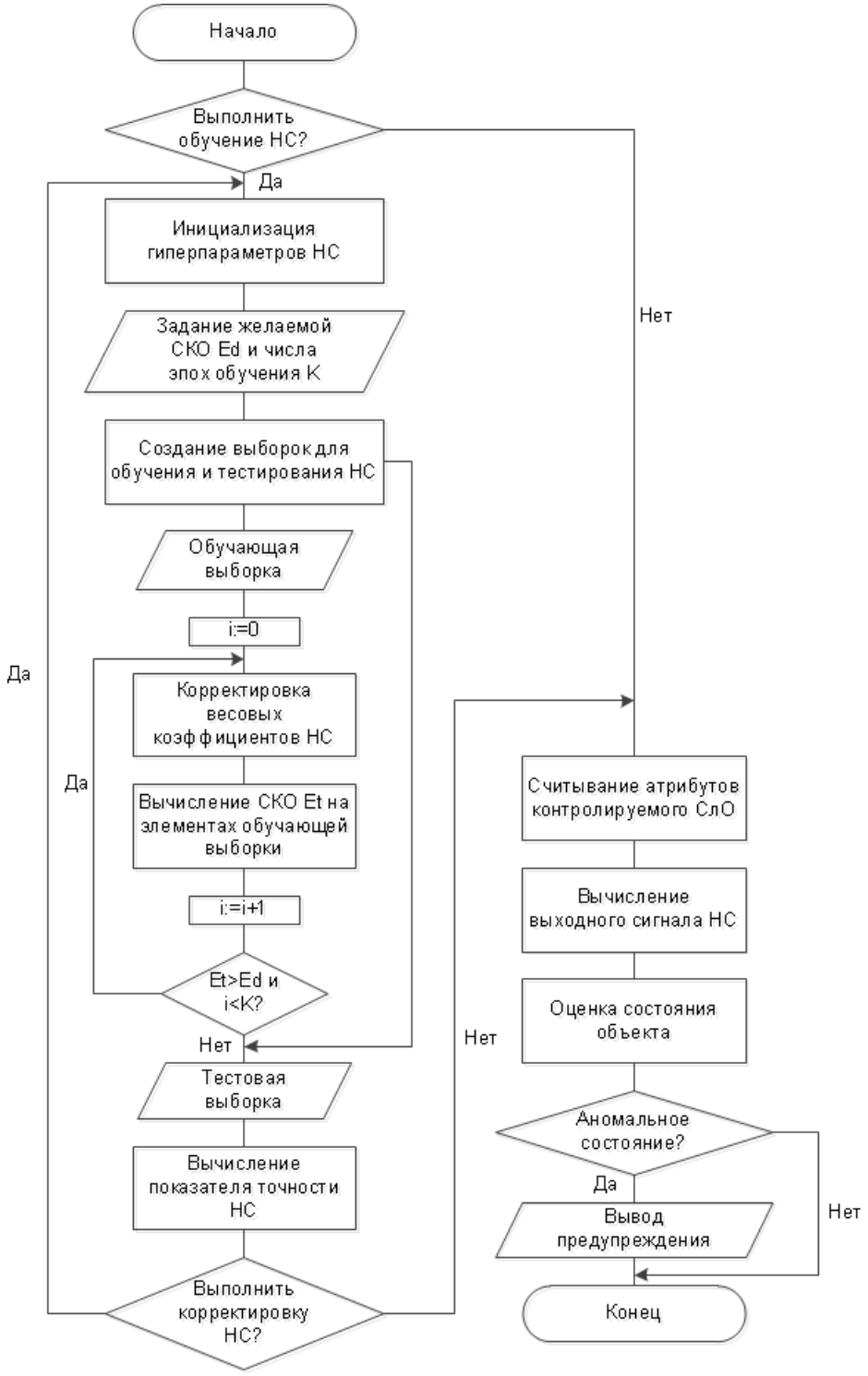

Рисунок 3.2.3.1 – Блок-схема алгоритма АОТССОП

После обучения НС выполняется считывание атрибутов контролируемого сложного объекта, что подразумевает мониторинг параметров его состояния в дискретные моменты времени. На основе этих параметров формируется входной сигнал для НС. Далее выполняется построение выходного сигнала НС, с этой целью последовательно применяются композиции линейных и нелинейных преобразований над входным сигналом. Выходное значение НС рассматривается как признак наличия или отсутствия аномального состояния у контролируемого объекта.

3.2.4 Алгоритм АПССОП

Алгоритм предназначен для автономного прогнозирования состояний сложных объектов и процессов, включающего наличие или отсутствие в некоторый момент времени в будущем определенного вида функциональных неисправностей, дефектов и влияния как непреднамеренных воздействий, так и преднамеренных (атакующих) воздействий, свойственных целевой системе.

Алгоритм строится на основе рекуррентной глубокой нейронной сети, обучение которой производится на семействе наборов данных, полученных из открытых источников и включающих данные о функционировании нескольких видов индустриальных систем.

Алгоритм АПССОП реализует функции сильного ИИ в части выполнения автономного прогнозирования состояний сложных объектов и процессов. Наличие большого числа гиперпараметров и настраиваемых весов, свойственных для глубоких нейронных сетей, позволяет с достаточно высоким качеством выявлять закономерности между признаками прогнозируемого состояния системы и ожидаемой меткой его класса.

Блок-схема алгоритма представлена на Рис. 3.2.4.1.

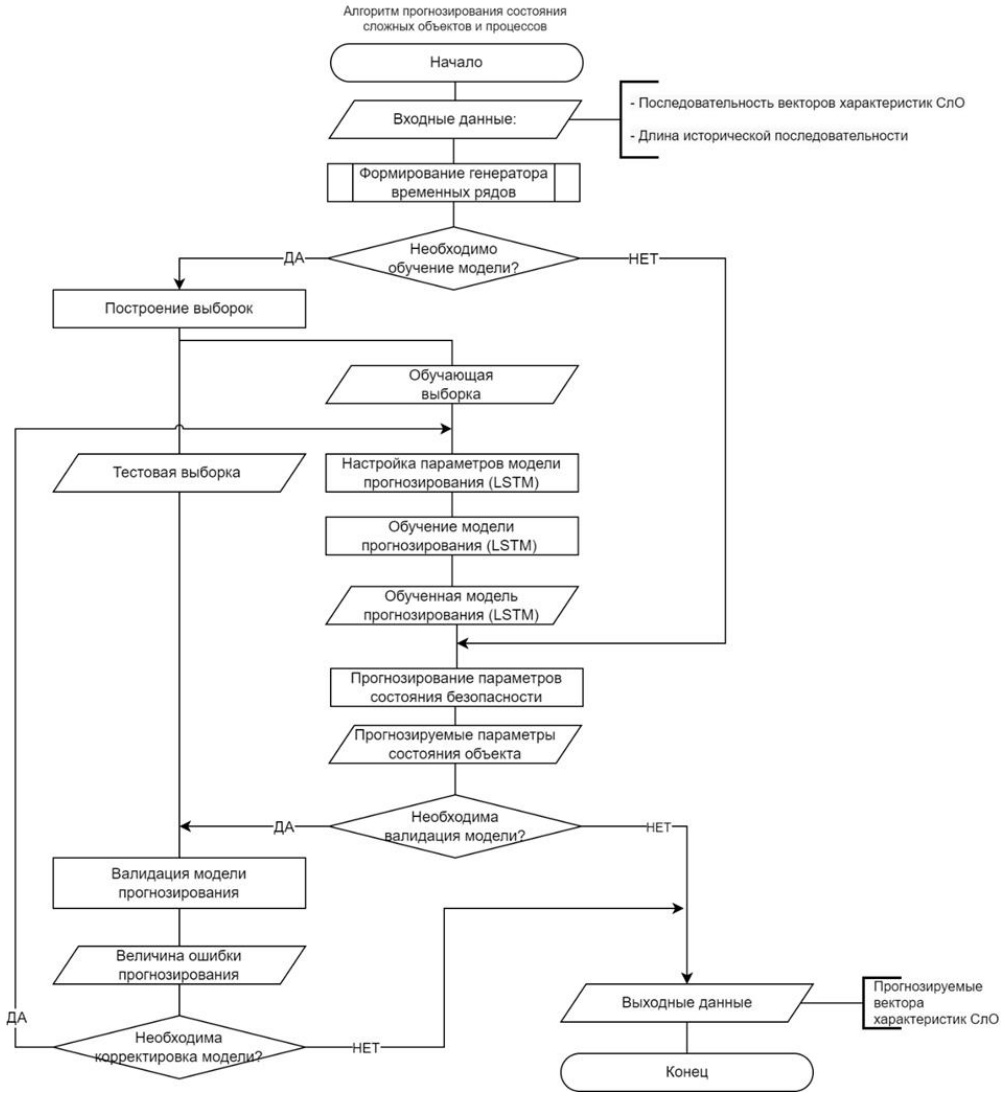

Рисунок 3.2.4.1 – Блок-схема алгоритма АПССОП

Начало реализации алгоритма автономного прогнозирования состояний сложных объектов и процессов состоит в чтении следующих входных данных: упорядоченную во времени последовательность векторов характеристик СлО ($\boldsymbol{X}$) и параметра длины исторической последовательности данных для прогнозирования (L).

Далее производится формирование генератора временных рядов (3.1.14). При необходимости обучить модель прогнозирования на входных данных осуществляется построение выборок – обучающей ($\boldsymbol{X}r$) и тестовой ($\boldsymbol{X}_e$) – таким образом, что обучающая выборка содержит временной ряд данных, предшествующий временному ряду в тестовой выборке. Разбиение исходного набора выполняется в соответствии с заданным пользователем соотношением.

На основе обучающей выборки производятся последовательно настройка параметров модели прогнозирования и ее обучение. В качестве интеллектуальной модели прогнозирования выбрана рекуррентная нейронная сеть (*перев. на англ.* Recurrent Neural Network, *аббр.* RNN), представленная долгой краткосрочной памятью (*перев. на англ.* Long Short-Term Memory, *аббр.* LSTM). Сеть LSTM является разновидностью рекуррентных нейронных сетей и позволяет хорошо анализировать данные в том случае, когда важные события разделены временными лагами неопределенной продолжительности.

Модель прогнозирования обучается путем поиска закономерностей в векторах характеристик состояний СлО для заданного временного окна и прогнозированием вектора значений в последующий момент времени (3.1.15).

Обученная модель прогнозирования LSTM применяется затем для прогнозирования параметров состояния (3.1.16). Далее, при необходимости, выполняется валидация модели прогнозирования на тестовых данных, результатом которой является величина ошибки прогнозирования, представленная кортежем <*mse*, *mae*>, где *mse* – среднеквадратичная ошибка, *mae* – средняя абсолютная ошибка. После этого принимается решение о необходимости корректировки модели прогнозирования, и при положительном решении заново происходит настройка параметров модели прогнозирования. Результатом выполнения данного алгоритма является набор прогнозируемых характеристик состояний объекта.

3.3 Структура компонента

Компонент состоит из двух следующих модулей, каждый из которых реализует описанные ранее алгоритмы (см. раздел 3.2).

Модуль ПОСНД

Модуль ПОСНД состоит из следующих программных элементов (реализованных с помощью программных классов), обладающих собственным целевым предназначением:

1) *CheckDataTypes* – класс для коррекции типов данных;
2) *ClusterFilling* – класс устранения неполноты данных;
3) *Informativity* – класс для анализа информативности признаков;
4) *MultiCollinear* – класс для устранения мультиколлинеарности;

5) *Data* – класс для хранения данных и результатов их обработки;
6) *PrintLog* – класс для журналирования работы модуля и представления результатов пользователю.

Модуль ИЗСНД

Модуль ИЗСНД состоит из следующих программных элементов (реализованных с помощью программных классов), обладающих собственным целевым предназначением:

1) *Predicate* – класс для представления логических предикатов;
2) *IzdapAlgo* – класс, реализующий логику алгоритма ИЗДАП.

Модуль АОТССОП

Модуль состоит из следующих программных элементов (реализованных с помощью программных классов), обладающих собственным целевым предназначением:

1) *SAIClassifier* – элемент для селективной классификации, позволяющий выбрать один из следующих их типов: дерево решений, наивный Байес, логистическая регрессия и искусственная нейронная сеть;
2) *FormatDetector* – элемент для определения формата входных данных;
3) *DataLoader* – элемент для загрузки набора входных данных;
4) *ClsEstimator* – элемент для оценивания эффективности классификаторов.

Модуль АПССОП

Модуль состоит из следующих программных элементов (реализованных с помощью программных классов), обладающих собственным целевым предназначением.

1) *AIForecaster* – элемент для прогнозирования СлОП:

Атрибуты класса:

а) *model* – нейросетевая модель прогнозирования (объект keras.models.Sequential);

б) *time_window_length* – длина исторической последовательности (целое число);

в) *n_features* – количество признаков (целое число);

г) *model_path* – путь для внешнего сохранения модели прогнозирования (текстовая строка);

д) *epochs* – количество эпох для обучения модели прогнозирования (целое число);

е) *batch_size* – размер пакетов для обучения (целое число);

ж) *early_stop* – объект EarlyStopping (библиотека keras).

Методы класса:

а) *def __init__(self, time_window_length=0, n_features=0, model_path='', n_epochs=1, open=False)* – конструктор класса для базовой инициализации параметров, где *open* – указатель на необходимость загрузить существующую модель (булевая переменная);

б) *def train(self, train_generator, validation_generator=None, save=True)* – обучение и валидация модели нейронной сети на данных, *train_generator* – генератор временных рядов обучающих данных (объект keras TimeseriesGenerator), *validation_generator* – генератор временных рядов данных валидации (объект keras TimeseriesGenerator); *save* – указатель на необходимость сохранить модель во внешний файл (булевая переменная);

в) *def forecasting(self, current_batch, forecasting_data_length, verbose=True)* – прогнозирование значений, где *current_batch* – исходный пакет данных для прогнозирования (многомерный массив), *forecasting_data_length* – длина прогнозируемой последовательности (целое число), *verbose* – указать на необходимость отображать процесс прогнозирования в командной строке (булевая переменная);

г) *def save_model(self)* – сохранение модели прогнозирования во внешний файл;

д) *def open_model(self)* – загрузка модели прогнозирования из внешнего файла;

е) *def data_to_generator(self, data)* – преобразование массива данных в генератор временных рядов, где *data* – исходные данные (многомерный массив);

ж) *def get_batch(self, generator, current_batch_id)* – извлечение отдельного пакета из генератора временных рядов, где *generator* – генератор временных рядов (объект keras TimeseriesGenerator), *current_batch_id* – порядковый номер извлекаемого пакета (целое число).

2) *DataScaler* – элемент для нормализации данных:

Атрибуты класса:

а) *scaler* – объект модели нормализации (объект MinMaxScaler);

б) *scaler_path* – путь для внешнего сохранения модели нормализации (текстовая строка).

Методы класса:

а) *def fit(self, data, save=True)* – обучение модели нормализации, где *data* – исходные данные (многомерный массив), *save* – указатель на необходимость сохранить модель нормализации во внешний файл (булевая переменная);

б) *def save(self)* – сохранение модели нормализации во внешний файл;

в) *def open(self)* – загрузка модели нормализации из внешнего файла;

г) *def transform(self, data)* – нормализация данных;

д) *def inverse(self, data)* – обратное преобразование нормализованных данных.

3) *ForecastEstimator* – элемент для оценки качества модели прогнозирования:

Атрибут класса:

а) *quality* – матрица результатов оценки качества (объект pandas DataFrame).

Методы класса:

а) *def estimate(self, true, pred, feature_names=[])* – оценка качества модели прогнозирования, где *true* – фактические значения (многомерный массив), *pred* – прогнозируемые значения (многомерный массив), *feature_names* – имена признаков (список);

б) *def save(self, file_name)* – сохранение результатов оценки во внешний файл, где *file_name* – путь к внешнему файлу (текстовая строка).

4 ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Техническими средствами являются электронно-вычислительные машины и устройства, которые используются при работе программы, должны иметь минимально необходимые характеристики, представленные в Табл. 4.1.

Таблица 4.1 – Минимально необходимые характеристики электронно-вычислительных машин и устройств для выполнения программы

Тип компьютера	**Кол-во CPU x кол-во ядер**	**Тактовая частота CPU, ГГц**	**Кол-во GPU x кол-во ядер**	**Тактовая частота GPU, ГГц**	**Оперативная память, Гб**	**Дисковая память, Гб**
Рабочая станция	1 x 8	3.8	1 x 3584	5.505	32	2000

5 ВЫЗОВ И ЗАГРУЗКА

5.1 Базовые функции компонента

Модуль ПОСНД

Начало работы с набором данных

```
data = Data()
titanic_path = '../datasets/titanic.csv'
titanic = pd.read_csv(titanic_path)
data.features_names = ["PassengerId", "Pclass", "Age", "SibSp", "Parch"]
data.labels_names = ["Survived", "Fare"]
data.features_matrix = np.array(titanic[data.features_names])
data.labels_matrix = np.array(titanic[data.labels_names])
data.features_types = ["cat", "cat", "num", None, None]
data.labels_types = ["cat", None]
```

Подключение журналирования

```
__Verbose__.PrintLog.instance().set_print_mode(True)
__Verbose__.PrintLog.instance().set_severity_level("status")
```

Запуск алгоритмов модуля

```
CheckDataTypes.CheckDataTypes.correct_types(data)
ClusterFilling.ClusterFilling.fill(data)
Informativity.Informativity.calculate_informativity(data)
Multicolinear.MultiCollinear.remove_uninformative_features(data)
```

Модуль ИЗСНД

Создание объекта IzdapAlgo

algo = IzdapAlgo(0.1)

Обучение модели

algo.fit(test_data, class_column = class_column, positive_class_label = positive_class_label)

Модуль АОТССОП

Создание объекта SAIClassifier

SAIClassifier('neural_network', 10, 1)

Создание объекта FormatDetector

FormatDetector('../hai/hai-20.07/test1.csv.gz')

Создание объекта DataLoader

DataLoader('../hai/hai-20.07/test1.csv.gz', 64, ‘,’)

Создание объекта ClsEstimator

classifiers = [SAIClassifier(c, in_size, out_size, plot=False) for c in classifier_types]

xor_ds = {'features': np.array([[0, 0], [0, 1], [1, 0], [1, 1]]), 'labels': np.array([[0], [1], [1], [0]])}

ClsEstimator(xor_ds['features'], xor_ds['labels'], xor_ds['labels'], [classifier]

Модуль АПССОП

Создание объекта AIForecaster

forecasting_model = AIForecaster(n_epochs=3,
time_window_length = 10,
n_features = len(features_names),
model_path = ”…/forecasting_model)

Создание существующей модели для объекта AIForecaster

forecasting_model = AIForecaster (model_path = ”…/forecasting_model, open = True)

Формирование генератора временных рядов

generatorX = forecasting_model.data_to_generator(trainX)

Обучение модели объекта AIForecaster

forecasting_model.train(generatorX)

Прогнозирование с использованием модели объекта AIForecaster

prediction = forecasting_model.forecasting(current_batch = batch, forecasting_data_length = 1000)

Создание объекта DataScaler
scaler = DataScaler(scaler_path = ”scaler.pkl”)

Открытие существующей модели для объекта DataScaler
scaler = DataScaler(scaler_path = “scaler.pkl”, open=True)

Обучение модели объекта DataScaler
scaler.fit(data = trainX, save = True)

Нормализация данных с использованием DataScaler
scaled_trainX = scaler.transform(trainX)

Создание объекта ForecastEstimator
estimator = ForecastEstimator()

Оценка эффективности прогнозирования
quality = estimator.estimate(true = scaled_testX, pred = prediction)

6 ВХОДНЫЕ ДАННЫЕ

6.1 Состав и структура входных данных

Модуль ПОСНД

Входными данными для модуля ПОСНД являются следующие (имеющие определенный формат):

1) *features_matrix (numpy.array)* – двумерный массив (только численные значения), состоящий из объектов и их признаков;
2) *features_types (numpy.array)* – одномерный массив (возможные значения элементов *["num", "cat", None]*), длина которого соответствует количеству признаков;
3) *features_names (numpy.array)*: одномерный массив (только строковые значения), длина которого соответствует количеству признаков;
4) *labels_matrix (numpy.array)* – двумерный массив (только численные значения), состоящий из объектов и их меток;
5) *labels_types (numpy.array)* – одномерный массив (возможные значения элементов ["num", "cat", None]), длина которого соответствует количеству меток;
6) *labels_names (numpy.array)* – одномерный массив (только строковые значения), длина которого соответствует количеству меток.

Модуль ИЗСНД

Входными данными для модуля ИЗСНД являются следующие (имеющие определенный формат):

1) *data* – признаки и метки обучающей выборки (многомерный массив);

2) *class_column* – имя признака, содержащего метки обучающей выборки, в виде строки;
3) *positive_class_label* – метка положительного класса;
4) *rule_metric* – метрика оценки правил (название в строковом формате);
5) *probability_threshold* – вероятность, задающая порог отсечения значений для построения предикатов.

Модуль АОТССОП

Входными данными для модуля АОТССОП являются следующие (имеющие определенный формат):

1) *x_train* – признаки обучающей выборки (многомерный массив);
2) *y_train* – метки обучающей выборки (многомерный массив);
3) *x_test* – признаки тестовой выборки (многомерный массив);
4) *y_test* – метки тестовой выборки (многомерный массив);
5) *x* – объект для определения класса с помощью глубокой нейронной сети (многомерный массив);
6) *saved_file* – путь к файлу для сериализации модели (текстовая строка);
7) *loaded_file* – путь к файлу для десериализации модели (текстовая строка);
8) *file* – путь к файлу для определения его типа и загрузки данных (текстовая строка);
9) *features* – признаки данных сложного объекта (массив с данными);
10) *labels(num_labels)* – метки данных сложного объекта (массив с данными);
11) *classifiers* – выбранные ранее классификаторы (программный объект).

Модуль АПССОП

Входными данными для модуля АПССОП являются следующие:

1) *model_path* – путь к модели прогнозирования (текстовая строка);
2) *scaler_path* – путь к модели нормализации (текстовая строка);
3) *time_window_length* – размер временного окна при обучении, длина источеской последовательности (число);
4) *forecast_len* – размер временного окна для прогнозирования (число),
5) *data* – матрица признаков исходного набора данных (многомерный массив);

6.2 Подготовка входных данных

Дополнительной предобработки входные данные не требуют, однако модули компонента загружают часть данных их внешних файлов следующим образом.

Модуль ПОСНД

Данные для работы модуля ПОСНД передаются в виде пути к файлу, содержащему данные, нуждающиеся в предобработке. При этом необходимо отделить признаки данных от их меток.

Модуль ИЗСНД

Данные для работы модуля ИЗСНД передаются в виде пути к файлу, содержащему обучающие данные.

Модуль АОТССОП

Модель десериализуется из внещнего файла – loader_file; файл имеет бинарный формат, реализованный в библиотеке pickle.

Входной набор данных для определения типа и загрузки загружается из файла по пути к нему – file; файл имеет текстовый формат в виде набора строк, элементы которых разделяются символами ‘,’ или ‘;’ (также, он может быть запакован с помощью gzip).

Модуль АПССОП

Модель прогнозирования загружается из файла по пути к нему – model_path; файл имеет бинарный формат, реализованный в библиотеке keras.models.

Модель нормализации загружается из файла по пути к нему – scaler_path; файл имеет бинарный формат, реализованный в библиотеке pickle.

7 ВЫХОДНЫЕ ДАННЫЕ

7.1 Состав и структура выходных данных

Модуль ПОСНД

Выходными данными для модуля ПОСНД являются следующие (имеющие определенный формат):

1) Вывод в консоль результатов анализа корректности типов данных.
2) Вывод в консоль результатов устранения неполноты данных.
3) Вывод в консоль результатов анализа информативности признаков.
4) Вывод в консоль результатов анализа мультиколлинеарности.

Модуль ИЗСНД

Выходными данными для модуля ИЗСНД являются следующие (имеющие определенный формат):

1) Вывод в консоль построенных ассоциативных правил класса.
2) Возврат преобразованных данных в формате pandas.DataFrame (функция transform).

Модуль АОТССОП

Выходными данными для модуля АОТССОП являются следующие (имеющие определенный формат):

1) оцениваемые значения (числовые значения);
2) показатели качества оценивания (числовые значения);
3) вывод обученной модели по некоторому датасету (программный формат модели).

Модуль АПССОП

Выходными данными для модуля АПССОП являются следующие:

1) прогнозируемые значения (числовые значения);
2) показатели качества прогнозирования (числовые значения);
3) вывод обученной модели прогнозирования по некоторому датасету (программный формат модели);
4) вывод обученной модели нормализации по некоторому датасету (программный формат модели).

7.2 Интерпретация выходных данных

Дополнительной постобработки выходные данные не требуют, однако модули компонента сохраняют часть данных во внешние файлы следующим образом.

Модуль ПОСНД

Модуль ПОСНД не сохраняет выходные данные во внешние файлы.

Модуль ИЗСНД

Модуль ИЗСНД не сохраняет выходные данные во внешние файлы.

Модуль АОТССОП

Модель сериализуется во внешний файл – saved_file; файл имеет бинарный формат, реализованный в библиотеке pickle.

Модуль АПССОП

Модель прогнозирования сериализуется во внешний файл – model_path; файл имеет бинарный формат, реализованный в библиотеке keras. Модель нормализации сериализуется во внешний файл – scaler_path; файл имеет бинарный формат, реализованный в библиотеке pickle. Результаты оценки прогнозирования сохраняются во внешний файл формата CSV – file_name.