# ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

Рассмотрим подробно функционирование программы. Для этого проведем анализ всех классов, которые входят в состав кода программы, и рассмотрим назначение всех методов, свойств и переменных. Статические взаимоотношения классов представлены на диаграмме классов (см. чертеж ГУИР.400201.075 РР.1).

* 1. Выбор алгоритмов предсказания и параметров команд

**3.1.1** Выбор алгоритмов предсказания

На данный момент существует достаточно много алгоритмов предсказания результата события. Так как реализовать планировалось предсказание на основе нескольких из них, а не одного конкретного, то и при выборе учитывалась совместимость данных алгоритмов по входным данным.

Выбор делался из таких алгоритмов, как квалиметрический анализ, нечеткие предсказания, предсказания на основе нейронной сети, двумерное дискретное распределение.

Требовалось выбрать один основной алгоритм и несколько, в данном случае два, вспомогательных. К основному алгоритму предъявлялось требование точности, а также он должен был выдавать вероятности исхода всех трех результатов матча. Вспомогательные алгоритмы могли делать однозначный вывод об исходе матча, то есть присваивать ему вероятность 100%.

Квалиметрический анализ, также называемый методом взвешенной суммы показателей, является довольно простым и быстрым для расчета алгоритмом прогнозирования. Квалиметрия – научная дисциплина, в рамках которой изучаются методология и проблематика комплексной, количественной оценки качества объектов любой природы. Данный алгоритм не нуждается в наличии обучающей выборки, а принимает решение только на основании входных данных. Суть алгоритма состоит в вычислении «силы» каждой команды на основе ее параметров и их весов. Для приемлемой точности работы необходимо подобрать правильные веса параметров и границы «сил» команд для того или иного решения. Это делается вручную и требует затрат времени только на этапе разработки.

Метод нечетких предсказаний также демонстрирует хорошую скорость работы, однако имеет недостаточную точность для его использования в качестве основного. Данный алгоритм требует наличия массива прошедших матчей с их параметров, на основании которых принимается решение о принадлежности предсказываемого события к тому или иному кластеру. На выходе алгоритма мы получаем вероятности всех возможных исходов, что является его плюсом.

Предсказание на основе нейронной сети практически идеально подходит для использования в качестве основного алгоритма. Нейронная сеть может выдавать сколь угодно точный результат, зависящий только от величины обучающей выборки и времени, затрачиваемого на обучение сети. Так же плюсом данного алгоритма является возможность переобучить нейронную сеть уже после завершения разработки программного продукта. Из всех существующих нейронных сетей наиболее часто используются для предсказания результатов спортивных событий сети Кохонена и сети на основе РБФ. Обе эти сети демонстрируют приемлемую точность, однако сеть Кохонена является гораздо более сложной в реализации.

Метод на основе двумерного дискретного распределения не совпадает по входным данным со всеми рассмотренными ранее. Он требует двумерную величину, в то время как параметров у события гораздо больше. Приведение события из многомерного в двумерное потребует дополнительных затрат машинного времени.

Исходя из всего вышеперечисленного, в качестве основного алгоритма предсказания был выбран метод на основе нейронной сети РБФ, а в качестве вспомогательных метод квалиметрического анализа и метод нечетких предсказаний. Итоговые веса вероятностей, предсказанный данными методами, равняются 1, 0.3 и 0.33 соответственно.

**3.1.2** Выбор параметров команд

Так как данная программная модель должна иметь возможность предсказывать результаты матчей всех игровых видов спорта, то было принято решение выбрать такие параметры для команд, которые являются общими для всех игровых видов спорта. Также влияние данных параметров на исход матча должно быть существенном, то есть при сильном изменении любого из этих параметров с большой долей вероятности изменится и исход события.

Всем вышеперечисленным критериям отвечают следующие параметры:

– место команды в турнирной таблице;

– количество набранных очков за 5 последних игр;

– число забитых голов за 5 последних игр;

– число пропущенных голов за 5 последних игр;

– количество игроков, пропускающих матч;

– количество дней отдыха до матча;

– количество матчей подряд, в которых команда не пропускала голы;

– количество матчей подряд, в которых команда забивала голы;

– дома или на выезде играет команда.

* 1. Классы блока нечетких предсказаний

Классы данного блока осуществляют предсказание результата события по методу нечетких предсказаний на основании данных, приходящих от блока предсказания.

**3.2.1** Класс FuzzyPredictor

Класс FuzzyPredictor является центральным классом блока нечетких предсказаний. Осуществляет предсказание результата события по методу нечетких предсказаний. Класс FuzzyPredictor реализует методы, необходимые совершения такого предсказания. Основные методы приведены в таблице 3.1. Описание основных полей приведено в таблице 3.2

Таблица 3.1 – Методы класса FuzzyPredictor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| FuzzyPredictor  List<MatchInfo> \_matchesInfo) | Устанавливает начальную конфигурацию, а именно запоминает информацию о матчах, пришедшую в конструктор |
| predict  (MatchInfo matchInfo) | Осуществляет предсказание по методу нечетких предсказаний |
| analizeTeamInfo  (List<Parameter> teamParameters) | Анализирует информацию о команде и выдает результат в формате, необходимом для дальнейшей работы |

Таблица 3.2 – Описание полей класса FuzzyPredictor

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| matchesInfo | List<MatchInfo> | Информация о матчах, присутствующих в системе |

* 1. Классы блока квалиметрического анализа

Классы данного блока осуществляют предсказание результата события по методу квалиметрического анализа на основании данных, приходящих от блока предсказания.

**3.3.1** Класс QualimetricPredictor

Класс QualimetricPredictor является центральным классом блока квалиметрического анализа. Осуществляет предсказание результата события по методу квалиметрического анализа. Класс QualimetricPredictor реализует методы, необходимые для совершения такого предсказания. Основные методы приведены в таблице 3.3. Полей данный класс не содержит.

Таблица 3.3 – Основные методы интерфейса QualimetricPredictor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| predict  (MatchInfo matchInfo) | Осуществляет предсказание по методу квалиметрического анализа |

**3.3.2** Класс TeamPowerCalculator

Класс TeamPowerCalculator осуществляет расчет так называемой «силы» команды для последующего использования в классе QualimetricPredictor. Основные методы приведены в таблице 3.4. Полей данный класс не содержит.

Таблица 3.4 – Методы класса TeamPowerCalculator

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| calculateTeamPower  (List<Parameter> firstTeamParameters, List<Parameter> secondTeamParameters) | Осуществляет расчет сил обеих команд, чьи параметры были переданы в качестве параметров |
| calculateRelativeParameters  (Parameter firstTeamParameter, Parameter secondTeamParameter) | На основе анализа однотипных параметров двух команд вычисляет относительные параметры данного типа для этих команд |

* 1. Классы блока анализа на основе нейронной сети

Классы данного блока осуществляют предсказание результата события при помощи нейронной сети, а именно радиально-базисной сети. Параметры обученной нейронной сети хранятся в базе данных и приходят в данный класс при инициализации. Если эти параметры не пришли, то происходит обучение нейронной сети.

**3.4.1** Класс NeuralNetworkPredictor

Класс NeuralNetworkPredictor является центральным классом блока анализа на основе нейронной сети. Осуществляет предсказание результата события при помощи нейронной сети. Класс NeuralNetworkPredictor реализует методы, необходимые для совершения такого предсказания. Основные методы приведены в таблице 3.5. Описание основных полей приведено в таблице 3.6

Таблица 3.5 – Описание основных методов класса NeuralNetworkPredictor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| NeuralNetworkPredictor  (List<MatchInfo> matchesInfo, List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs, bool isTeachNeaded) | Инициализирует объект на основе данных, пришедших от блока предсказания, а именно информации о нейронной сети |
| initializeNeuralNetwork  (List<MatchInfo> matchesInfo, List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs, bool isTeachNeaded) | В зависимости от пришедших данных либо инициализирует ими нейронную сеть, либо проводит ее обучение |
| prepareDataForSaving  (List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs) | Вызывает метод нейронной сети, который подготавливает информацию о ней для сохранения в базу данных |
| predict  (MatchInfo matchInfo) | Выполняет предсказание результата события при помощи нейронной сети |
| isNetworkChanged() | Возвращает информацию о том, была ли нейронная сеть изменена |

Таблица 3.6 – Описание основных полей класса NeuralNetworkPredictor

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| rbfNetwork | RadialBasisFunctionNetwork | Нейронная сеть |
| \_isNetworkChanged | bool | Изменена ли сеть |

**3.4.2** Класс RadialBasisFunctionNetwork

Класс RadialBasisFunctionNetwork реализует нейронную сеть на основе радиально-базисных функций. Содержит в себе алгоритмы обучения нейронной сети и классификации объекта на основе обученной нейронной сети. Основные методы приведены в таблице 3.7. Описание основных полей приведено в таблице 3.8

Таблица 3.7 – Описание основных методов класса RadialBasisFunctionNetwor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| RadialBasisFunctionNetwork  (int \_inputCount, int \_outputCount, int \_rbfCount, List<TeachedRBF> \_teachedRBFs, List<OutputLayerWeights> \_outputLayerWeights, char[] \_classesNames) | Инициализирует нейронную сеть данными, которые раньше были получены при ее обучении |
| RadialBasisFunctionNetwork  (Dictionary<char, List<double[]>> classes, double alfa, double maxError) | Вызывает метод обучения нейронной сети и инициализирует ее полученными значениями |
| TeachNeuralNetwork  (Dictionary<char, List<double[]>> classes) | Обучает нейронную сеть на основе пришедшей обучающей выборки |
| TeachRBFNeurons(Dictionary<char, List<double[]>> classes) | Обучает нейроны нейронной сети |
| CalculateRBFNeuronsForTrainingSet  (Dictionary<char, List<double[]>> classes) | Вычисляет значения РБФ-нейронов для обучающей выборки |
| Randomize() | Случайным образом генерирует веса нейронов выходного слоя |
| ClassifyMatch  (double[] match) | Классифицирует матч на основе обученной нейронной сети |
| prepareDataForSaving  (List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs) | Подготавливает данные о нейронах и весах связей для сохранения в базу данных |

Таблица 3.8 – Описание основных полей класса RadialBasisFunctionNetwork

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| rbf\_teached | RBF\_teachInfo[] | Обученные РБФ-ячейки |
| outputLayer\_weights | double[,] | Веса нейронов выходного слоя |
| inputCount | int | Число нейронов входного слоя |
| outputCount | int | Число нейронов выходного слоя |
| rbfCount | int | Число РБФ-ячеек |
| alfa | double | Скорость обучения |
| maxErrorTreshold | double | Максимальное значение ошибки классификации |
| iterationCount | int | Число итераций обучения |
| classesNames | char[] | Названия кластеров |

**3.4.3** Структура RBF\_teachInfo

Структура RBF\_teachInfo содержит свойств, описывающих обученную РБФ-ячейку Описание основных полей приведено в таблице 3.9.

Таблица 3.9 – Описание основных полей структуры RBF\_teachInfo

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| expectations | double[] | Ожидание |
| standardDeviation | double | Стандартное отклонение |

**3.4.4** Класс RandomProvider

Класс RandomProvider предназначен генерации случайного числа. Описание основных методов и полей класса RandomProvider приведены в таблицах 3.10 и 3.11 соответственно.

Таблица 3.10 – Описание основных методов класса RandomProvider

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| Next() | Генерация случайного числа потокобезопасным методом |

Таблица 3.11 – Описание основных полей класса RandomProvider

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Rnd | Random | Генератор случайных чисел |
| \_sync | object | Объект многопоточной синхронизации |

* 1. Основные классы блока предсказания

Данный блок является управляющим для всех блоков предсказания. Классы данного блока осуществляют вызов методов предсказания всех этих блоков и осуществляют их анализ и расчет итоговых вероятностей на их основе.

**3.5.1** Интерфейс IPredictor

Интерфейс IPredictor обеспечивает одинаковую сигнатуру для методов предсказания всех классов-предсказателей. Основные методы приведены в таблице 3.12

Таблица 3.12 – Описание основных методов интерфейса IPredictor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| Predict  (MatchInfo matchInfo) | Сигнатура метода предсказания результата |

**3.5.2** Класс Serializer

Класс Serializer производит сериализацию данных в строку и обратно для сохранения и чтения из базы данных. Основные методы описаны в таблице 3.13

Таблица 3.13 – Описание основных методов класса Serializer

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| SerializeToString  (double[] array) | Сериализует массив в строку |
| DeserializeFromString  (string serializedArray) | Десериализует строку в массив |

**3.5.3** Класс GeneralPredictor

Класс GeneralPredictor является основным в блоке предсказания. Он вызывает все существующие в программе предсказатели и на их основе получает итоговый результат. Основные методы приведены в таблице 3.14. Описание основных полей приведены в таблице 3.15

Таблица 3.14 – Описание основных методов класса GeneralPredictor

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| GeneralPredictor  (List<MatchInfo> matchesInfo, List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs) | Инициализирует все предсказатели на основе полученных данных |
| Predict  (MatchInfo matchInfo) | Дает итоговое предсказание |

Таблица 3.15 – Описание основных полей класса GeneralPredictor

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| qualimetricPredictor | Qualimetric  Predictor | Квалиметрический предсказатель |
| fuzzyPredictor | FuzzyPredictor | Нечеткий предсказатель |
| neuralNetworkPredictor | NeuralNetwork  Predictor | Предсказатель на основе нейронной сети |

* 1. Классы блока работы с базами данных

Классы данного блока представляют собой модели сущностей, хранящихся в базах данный, позволяют считывать эти сущности и преобразовывать их к бизнес-моделям.

**3.6.1** Класс Rate

Класс Rate представляют собой модель сущности, хранящейся в базе ставок. Описание основных полей приведено в таблице 3.16

Таблица 3.16 – Описание основных полей класса Rate

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Id | int | Id ставки |
| UserId | string | Id пользователя, сделавшего ставку |
| MatchId | int | Id матча, на который сделана ставка |

*Продолжение таблицы 3.16*

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Result | int | Результат, на который сделана ставка |
| Amount | decimal | Сумма ставки |
| Coefficient | string | Коэффициент на данный результат |

**3.6.2** Класс RatesContext

Класс RatesContext является входной точкой для работы с базой данных ставок. Описание основных полей приведено в таблице 3.17

Таблица 3.17 – Описание основных полей класса RatesContext

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Rates | DbSet<Rate> | Все ставки, содержащиеся в базе данных ставок |

**3.6.3** Класс Team

Класс Team представляют собой модель сущности, хранящейся в таблице команд в базе игр. Описание основных полей приведено в таблице 3.18

Таблица 3.18 – Описание основных полей класса Team

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Id | int | Id команды |
| Name | string | Название команды |
| Position | int | Место в турнирной таблице |
| Points | int | Набрано очков за последние 5 игр |
| Goals | int | Забито голов |
| MissingBalls | int | Пропущено голов |
| InjuredPlayers | int | Игроков пропускает матч |
| DaysOfRest | int | Дней отдыха |
| ZeroGames | int | Не пропускали голов подряд |
| GoalGames | int | Забивали голы подряд |
| HomeMatch | int | Домашний или выездной матч |
| FuzzyInfo | string | Информация о команде в формате для нечеткого предсказания |

**3.6.4** Класс Match

Класс Match представляют собой модель сущности, хранящейся в таблице матчей в базе игр. Описание основных полей приведено в таблице 3.19

Таблица 3.19 – Описание основных полей класса Match

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Id | int | Id матча |
| Result | string | Результат матча |
| FirstTeamId | int? | Id первой команды |
| SecondTeamId | int? | Id второй команды |
| FirstTeam | Team | Первая команда |
| SecondTeam | Team | Вторая команда |

**3.6.5** Класс GamesContext

Класс GamesContext является входной точкой для работы с базой данных игр. Описание основных полей приведено в таблице 3.20

Таблица 3.20 – Описание основных полей класса RatesContext

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Matches | DbSet<Match> | Все матчи, содержащиеся в базе данных игр |
| Teams | DbSet<Team> | Все команды, содержащиеся в базе данных игр |

**3.6.6** Класс FuzzyLimit

Класс FuzzyLimit представляет собой информацию о верхнем и нижнем пределах параметров для нечеткого предсказания. Основные методы приведены в таблице 3.21. Описание основных полей приведено в таблице 3.22

Таблица 3.21 – Описание основных методов класса FuzzyLimit

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| FuzzyLimit  (int \_leftLimit,  int \_rightLimit) | Инициализирует объект входным данным |

\_

\_

Таблица 3.22 – Описание основных полей класса FuzzyLimit

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| leftLimit | int | Нижняя граница |
| rightLimit | int | Верхняя граница |

**3.6.7** Класс Parameter

Класс Parameter представляет собой расширенную информацию об определенном показателе команды. Основные методы приведены в таблице 3.23. Описание основных полей приведено в таблице 3.24

Таблица 3.23 – Описание основных методов класса Parameter

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| Parameter  (bool \_isStimalutor, int \_weigth, double \_value) | Инициализирует объект без пределов параметра |
| Parameter  (bool \_isStimalutor, int \_weigth, double \_value, FuzzyLimit \_fuzzyLimit) | Инициализирует объект с пределами параметра |

Таблица 3.24 – Описание основных полей класса Parameter

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| isStimulator | bool | Является параметр полезным для команды |
| weight | int | Вес параметра |
| value | double | Значение параметра |
| fuzzyLimit | FuzzyLimit | Пределы параметра |

**3.6.8** Класс TeamInfo

Класс TeamInfo является бизнес-моделью класса Team. Основные методы приведены в таблице 3.25. Описание основных полей приведено в таблице 3.26

Таблица 3.25 – Описание основных методов класса TeamInfo

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| TeamInfo  (Team teamInfo) | Инициализирует объект на основе модели Team |
| List<Parameter> getListOfParameters  () | Получает массив параметров команды |

Таблица 3.26 – Описание основных полей класса TeamInfo

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| name | string | Название команды |
| position | Parameter | Место в турнирной таблице в формате Parameter |
| points | Parameter | Набрано очков за последние 5 игр в формате Parameter |
| goals | Parameter | Забито голов в формате Parameter |
| missingBalls | Parameter | Пропущено голов в формате Parameter |
| injuredPlayers | Parameter | Игроков пропускает матч в формате Parameter |
| daysOfRest | Parameter | Дней отдыха в формате Parameter |
| zeroGames | Parameter | Не пропускали голов подряд в формате Parameter |
| goalGames | Parameter | Забивали голы подряд в формате Parameter |
| homeMatch | Parameter | Домашний или выездной матч в формате Parameter |
| fuzzyInfo | string | Информация о команде в формате для нечеткого предсказания |

**3.6.9** Класс MatchInfo

Класс MatchInfo является бизнес-моделью класса Match. Основные методы приведены в таблице 3.27. Описание основных полей приведено в таблице 3.28

Таблица 3.27 – Описание основных методов класса MatchInfo

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| MatchInfo  (Match matchInfo, TeamInfo \_firstTeam, TeamInfo \_secondTeam) | Инициализирует объект на основе модели Match |
| getMatchParams  () | Получает массив параметров обеих команд матча |

Таблица 3.28 – Описание основных полей класса MatchInfo

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| id | int | Id матча |
| firstTeam | TeamInfo | Первая команда |
| secondTeam | TeamInfo | Вторая команда |
| realResult | int | Результат матча |

**3.6.10** Класс OutputLayerWeights

Класс OutputLayerWeights представляет собой модель сущности, хранящейся в таблице выходных нейронов в базе нейронной сети. Описание основных полей приведено в таблице 3.29

Таблица 3.29 – Описание основных полей класса OutputLayerWeights

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Id | int | Id выходного нейрона |
| OutputLayerWeight | string | Веса выходного нейрона |

**3.6.11** Класс TeachedRBF

Класс TeachedRBF представляет собой модель сущности, хранящейся в таблице РБФ-ячеек в базе нейронной сети. Описание основных полей приведены в таблице 3.30

Таблица 3.30 – Описание основных полей класса TeachedRBF

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| Id | int | Id РБФ-ячейки |
| StandartDeviation | double | Стандартное отклонение |
| Expectation | string | Ожидание |

**3.6.12** Класс NeuralNetworkContext

Класс NeuralNetworkContext является входной точкой для работы с базой данных нейронной сети. Описание основных полей приведено в таблице 3.31

Таблица 3.31 – Описание основных полей класса NeuralNetworkContext

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| OutputLayer  Weights | DbSet<OutputLayer  Weights> | Все веса выходных нейронов, содержащиеся в базе нейронной сети |
| TeachedRBFs | DbSet<TeachedRBF> | Все РБФ-ячейки, содержащиеся в базе нейронной сети |

* 1. Классы блока корректировки коэффициентов

Данный блок вычисляет реальные коэффициенты на основе данных блока предсказания, а также производит корректировку реальных коэффициентов с учетом обязательной выгоды для владельца программы.

**3.7.1** Класс CoefficientsCalculator

Класс CoefficientsCalculator вычисляет реальные коэффициенты на основе данных блока предсказания, а также производит корректировку реальных коэффициентов с учетом обязательной выгоды для владельца программы. Основные методы приведены в таблице 3.32

Таблица 3.32 – Описание основных методов класса CoefficientsCalculator

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| calculateRealCoefficients  (double[] prediction) | Вычисляет реальные коэффициенты |
| calculateAdjustedCoefficients  (double[] prediction) | Вычисляет откорректированные коэффициенты |

* 1. Классы блока клиентского интерфейса

Интерфейс пользователя представляет собой веб-приложение, спроектированное на основе многоуровневой архитектуре. *Многоуровневая архитектура* (multilayered architecture) сосредоточена на иерархическом распределении отдельных частей системы при помощи эффективного разделения отношений.

Уровень представления (presentation layer) ответственен за взаимодействие с пользователем, ввод и вывод информации.

Бизнес-уровень или уровень бизнес-логики (business logic layer) обрабатывает информацию, реализуя конкретные бизнес-правила.

Уровень доступа к данным (data access layer) обеспечивает загрузку и сохранение информации, используя источник данных (файл, база данных) или внешний сервис.

При этом уровень представления считается высшим, за ним идёт уровень бизнес-логики, а за уровнем бизнес-логики – уровень доступа к данным.

Классы данного блока предназначены для корректного представления данных пользователю, описанные выше функции реализуются в других блоках.

**3.8.1** Класс ResultsEncoder

Класс ResultsEncoder преобразует информацию о результате матча из внутреннего формата программы в формат, который понятен пользователю. Описание основных полей приведено в таблице 3.33

Таблица 3.33 – Описание основных методов класса ResultsEncoder

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| results | string[] | Массив строковых представлений результатов матчей, где индекс массива соответствует внутреннему представлению результата в программе |

* 1. Классы блока управления

Являясь центральным компонентом системы, отвечает за такие задачи, как запуск изменения баз данных, передачу данных на клиентский интерфейс, запуск предсказания результата события.

**3.9.1** Класс GamesController

Класс GamesController является управляющим классом для работы с информацией о матчах и командах, участвующих в них. Позволяет добавлять и редактировать, а также выводить пользователю данную информацию. Основные методы приведены в таблице 3.34. Описание основных полей приведено в таблице 3.35

Таблица 3.34 – Описание основных методов класса GamesController

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| Results  () | Подготавливает список всех матчей, результат которых уже известен, и выводит этот список пользователю |
| EditMatches  () | Выводит список всех доступных матчей с возможностью добавить новый или изменить уже существующий. Доступен только администратору. |
| CreateMatch  (bool? useTeamIds) | Возвращает страницу для создания нового матча. Если в параметрах пришло true, то в Id команд матча поместит \_firstTeamId и \_secondTeamId |
| CreateMatch  (Match match) | Создает новый матч и возвращает администратора на страницу редактирования матчей |
| EditMatch  (int id) | Возвращает страницу редактирования матча, id которого пришло в параметре |
| EditMatch  (Match match) | Сохраняет отредактированный матч в базе данных игр |
| CreateTeam  (bool isFirstTeam) | Доступен только при создании нового матча. Возвращает страницу для создания новой команды. Данная команда будет считаться первой, если в параметре пришло true |
| CreateTeam  (Team team) | Сохраняет созданную команду в базе данных игр и возвращает администратору к созданию матча |
| getMatchesInfo  () | Если к методу обратились первый раз, то подготавливает и запоминает массив всех матчей, а затем его возвращает, иначе просто возвращает ранее запомненный массив |
| updateTeamsTable  (List<Team> teamsToUpdate) | Сохраняет изменения в полученных командах в базе данных игр |

Таблица 3.35 – Описание основных полей класса GamesController

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| db | GamesContext | Подключение к базе данных игр |
| matchesInfo | List<MatchInfo> | Массив всех содержащихся в базе данных матчей |
| addFirstTeam | bool | Создается ли в данный момент первая команда для нового матча |
| \_firstTeamId | int? | Id первой созданной команды для нового матча |
| \_secondTeamId | int? | Id второй созданной команды для нового матча |

**3.9.2** Класс PredictionController

Класс PredictionController является управляющим классом для работы предсказаниями результатов матчей и непосредственно запускает переобучение нейронной сети. Основные методы приведены в таблице 3.36. Описание основных полей приведено в таблице 3.37

Таблица 3.36 – Описание основных методов класса PredictionController

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| FutureEvents  () | Подготавливает список всех матчей, результат которых еще не известен, и выводит этот список пользователю. Также вычисляет и выводит коэффициенты на все возможные исходы данного матча |
| TeachNN  () | Запускает процесс обучения нейронной сети и после его вызывает метод сохранения информации о нейронной сети |
| saveNeuralNetwokToDataBase  (List<OutputLayerWeights> outputLayerWeights, List<TeachedRBF> teachedRBFs) | Сохраняет информацию о нейронной сети |

Таблица 3.37 – Описание основных полей класса PredictionController

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| neuralNetworkDb | NeuralNetworkContext | Подключение к базе данных нейронной сети |

**3.9.3** Класс RatesController

Класс RatesController является управляющим классом для работы со ставками, позволяет совершить ставку на исход доступного матча, а также просмотреть информацию о всех своих ставках. Основные методы приведены в таблице 3.38. Описание основных полей приведено в таблице 3.39

Таблица 3.38 – Описание основных методов класса RatesController

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| MakeRate(int matchId, int result, double coefficient) | Возвращает форму для совершения ставки на выбранный исход определенного матча |
| MakeRate(Rate rate) | Сохраняет совершенную ставку в базе данных ставок |
| RatesInfo() | Возвращает страницу с информацией о всех ставках данного пользователя. Перед этим проверяет, не произошли ли события, на которые сделана ставка. И если да, то изменяет состояние ставки в соответствии с результатом |

Таблица 3.39 – Описание основных полей класса RatesController

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| 1 | 2 | 3 |
| ratesDb | RatesContext | Подключение к базе данных ставок |

* 1. Структура данных

База данных программы хранит в себе информацию о зарегистрированных пользователях, их ставках, матчах, командах и информацию об обученной нейронной сети. Логически выделены блоки базы данных пользователей и базы данных игр, которые физически представлены четырьмя базами данных:

– база данных игр;

– база данных ставок;

– база данных пользователей;

– база данных нейронной сети.

Отношения между таблицами представлены в модели данных (см. чертеж ГУИР.400201.075 РР.3).

**3.10.1** База данных игр

База данных игр состоит из двух таблицы: Matches и Teams. Описание полей данных таблиц приведено в таблицах 3.40 и 3.41 соответственно.

Таблица 3.40 – Описание полей таблицы Matches

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 2 |
| Id | Хранит идентификатор данного матча, является уникальным первичным ключом таблицы |
| FirstTeamId | Хранит идентификатор первой команды, принимающей участие в матче. Является первым внешним ключом таблицы |
| SecondTeamId | Хранит идентификатор второй команды, принимающей участие в матче. Является вторым внешним ключом таблицы |
| Result | Хранит результат матча в числовом формате. Цифры ноль, один, два, три обозначают соответственно ничью, выигрыш, проигрыш первой команды или то, что матч еще не состоялся |

Таблица 3.41 – Описание полей таблицы Teams

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 2 |
| Id | Хранит идентификатор данной команды, является уникальным первичным ключом таблицы |
| Name | Название команды |
| Position | Место в турнирной таблице |
| Points | Набрано очков за последние 5 игр |

*Продолжение таблицы 3.41*

|  |  |
| --- | --- |
| 1 | 2 |
| Goals | Забито голов |
| MissingBalls | Пропущено голов |
| InjuredPlayers | Игроков пропускает матч |
| DaysOfRest | Дней отдыха |
| ZeroGames | Не пропускали голов подряд |
| GoalGames | Забивали голы подряд |
| HomeMatch | Домашний или выездной матч |
| FuzzyInfo | Информация о команде в формате для нечеткого предсказания |

**3.10.2** База данных ставок

База данных ставок состоит из одной таблицы Rates. Описание полей данной таблицы приведено в таблице 3.42

Таблица 3.42 – Описание полей таблицы Rates

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 3 |
| Id | Хранит идентификатор данной ставки, является уникальным первичным ключом таблицы |
| UserId | Id пользователя, сделавшего ставку |
| MatchId | Id матча, на который сделана ставка |
| Result | Результат, на который сделана ставка |
| Amount | Сумма ставки |
| Coefficient | Коэффициент на данный результат |

**3.10.3** База данных пользователей

База данных пользователей содержит в себе пять таблиц с информацией о пользователях, их ролях и входах в систему, но интерес представляет только таблица Users. Описание полей данной таблицы приведено в таблице 3.43

Таблица 3.43 – Описание полей таблицы Users

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 3 |
| Id | Хранит идентификатор данной пользователя в строковом формате, является уникальным первичным ключом таблицы |

*Продолжение таблицы 3.43*

|  |  |
| --- | --- |
| 1 | 2 |
| Email | Email пользователя, он же логин в системе |
| PasswordHash | Хешированный пароль пользователя |

**3.10.4** База данных нейронной сети

База данных нейронной сети состоит из таблиц OutputLayerWeights и TeachedRBFs, содержащих информацию о весах выходного слоя нейронов и РБФ-ячейках сети соответственно. Описание полей данных таблиц приведено в таблицах 3.44 и 3.45 соответственно.

Таблица 3.44 – Описание полей таблицы OutputLayerWeights

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 3 |
| Id | Хранит идентификатор нейрона выходного слоя, является уникальным первичным ключом таблицы |
| OutputLayerWeight | Веса выходного нейрона |

Таблица 3.45 – Описание полей таблицы TeachedRBFs

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 3 |
| Id | Хранит идентификатор РБФ-ячейки, является уникальным первичным ключом таблицы |
| StandartDeviation | Стандартное отклонение |
| Expectation | Ожидание |