МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное   
учреждение высшего образования

**«Южно-Уральский государственный университет**

**(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

**Разработка приложения для поиска шаблонов успеваемости студентов на основе данных портала Универис.**

КУРСОВАЯ РАБОТА

по дисциплине «Программная инженерия»

ЮУрГУ – 09.03.04.20231.308-3392.КР

|  |  |
| --- | --- |
| Нормоконтролер3,  Научный руководитель,  д.ф.-м.н., доцент,  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Л. Цымблер  “\_\_\_”\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | Научный руководитель:  д.ф.-м.н., доцент,  \_\_\_\_\_\_\_\_\_\_ М.Л. Цымблер  Автор работы:  студент группы КЭ-303  \_\_\_\_\_\_\_\_\_\_ Е.В. Елисеев  Работа защищена  с оценкой: \_\_\_\_\_\_\_\_\_\_\_  “\_\_\_”\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

Челябинск 2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное   
учреждение высшего образования

**«Южно-Уральский государственный университет**

**(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

УТВЕРЖДАЮ

Зав. кафедрой СП

\_\_\_\_\_\_\_\_\_\_ Л.Б. Соколинский

«\_\_\_».\_\_\_\_\_\_\_\_\_\_\_\_\_.2023

**ЗАДАНИЕ**

**на выполнение курсовой работы**

студенту группы КЭ-303

Елисееву Егору Вадимовичу

обучающемуся по направлению   
09.03.04 «Программная инженерия»

1. **Тема работы**Разработка приложения для поиска шаблонов успеваемости студентов на основе данных портала Универис.
2. **Срок сдачи студентом законченной работы:** 29.05.2023 г.
3. **Исходные данные к работе**

3.1. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques - Morgan Kaufmann Publishers, 2012. – 703 p.

3.2. McKinney W. Python for Data Analysis - O’Reilly Media, 2018. – 540 p.

3.3. Pandas documentation. [Электронный ресурс] URL: <https://pandas.pydata.org/docs/reference/index.html> (дата обращения 15.02.2022 г.).

3.4. NumPy documentation. [Электронный ресурс] URL: <https://numpy.org/doc/stable/reference/index.html> (дата обращения 15.02.2022 г.).

1. **Перечень подлежащих разработке вопросов**

4.1. Провести анализ и спецификацию предметной области.

4.2. Выполнить проектирование пользовательского интерфейса и модульной структуры системы.

4.3. Реализовать систему поиска шаблонов.

4.4. Провести тестирование системы поиска шаблонов.

5.5. Провести эксперименты с применением разработанной системы.

1. **Дата выдачи задания:** 06.02.2023.

**Научный руководитель,**  д.ф.-м.н., доцент М.Л. Цымблер

**Задание принял к исполнению**  Е.В. Елисеев

**ОГЛАВЛЕНИЕ**

[Введение 2](#_Toc136646693)

[Актуальность работы 2](#_Toc136646694)

[Постановка задачи 2](#_Toc136646695)

[Структура и содержание работы 2](#_Toc136646696)

[1. Методы поиска шаблонов 2](#_Toc136646697)

[1.1. Формальные определения 2](#_Toc136646698)

[1.2. Обзор работы по тематике 2](#_Toc136646699)

[2. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 2](#_Toc136646700)

[2.1. Описание предметной области 2](#_Toc136646701)

[2.2. Описание исходных данных 2](#_Toc136646702)

[3. ПРОЕКТИРОВАНИЕ СИСТЕМЫ 2](#_Toc136646703)

[3.1. Требования к программе 2](#_Toc136646704)

[3.2. Диаграмма вариантов использования 2](#_Toc136646705)

[3.3. Архитектура системы 2](#_Toc136646706)

[3.4. Модуль запроса данных 2](#_Toc136646707)

[3.5. Модуль преобразования данных 2](#_Toc136646708)

[3.6. Модуль поиска ассоциативных правил 2](#_Toc136646709)

[3.7. Разработка макета 2](#_Toc136646710)

[4. РЕАЛИЗАЦИЯ СИСТЕМЫ 2](#_Toc136646711)

[4.1. Программные средства реализации 2](#_Toc136646712)

[4.2. Модуль запроса данных 2](#_Toc136646713)

[4.3. Модуль преобразования данных 2](#_Toc136646714)

[4.4. Модуль поиска ассоциативных правил 2](#_Toc136646715)

[4.5. Реализация интерфейса 2](#_Toc136646716)

[5. Тестирование 2](#_Toc136646717)

[5.1. Функциональное тестирование 2](#_Toc136646718)

[ЗАКЛЮЧЕНИЕ 2](#_Toc136646719)

[Список литературы 2](#_Toc136646720)

[Приложения 2](#_Toc136646721)

[Приложение А. Основные функции приложения 2](#_Toc136646722)

Введение

Актуальность работы

На сегодняшний день активно продолжает совершенствоваться процесс цифровизации образовательной системы [1] и вследствие этого каждый день в учебных заведениях собираются и хранятся огромные массивы данных. Одной из главных задач для учебного заведения является повышение качества образования, и для ее достижения руководству важно принимать верные решения, владея достоверной информацией, которая опирается на реальные данные.

Среди данных в образовательной системе естественным образом выделяются данные об успеваемости. Из них можно выявить интересные закономерности, которые могут характеризовать как и самих студентов, так и преподавателей, предметы, методики и многое другое. Также эти закономерности могут являться наиболее яркими маркерами наличия возможных проблем в образовательной системе. Руководствуясь добытой ценной информацией из данных об успеваемости, университет может наиболее эффективно находить проблемы и их источники, а также принимать правильные решения по их устранению.

ЮУрГУ, как крупной образовательной структуре, так же важно уметь извлекать полезные практически знания из хранящихся данных об успеваемости. Именно для этой цели необходимо наличие системы, способной получать данные из системы университета, реализовывать один из методов анализа данных – в данном случае метод ассоциативных правил, и предоставлять результаты. Система позволит рядовому пользователю проводить анализ на нужной ему выборке и находить в ней закономерности, о которых можно будет сделать какой-либо вывод.

Постановка задачи

Целью курсовой работы является разработка приложения для поиска шаблонов успеваемости студентов на основе данных портала Универис. Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ и спецификацию предметной области.
2. Выполнить проектирование пользовательского интерфейса и модульной структуры системы.
3. Реализовать систему поиска шаблонов.
4. Провести тестирование системы поиска шаблонов.
5. Провести эксперименты с применением разработанной системы.

Структура и содержание работы

Работа состоит из введения, пяти глав, заключения и списка литературы. Объем работы составляет 40 страниц, объем списка литературы – 12

источников.

В первой главе описываются формальные определения и производится обзор работ по тематике.

Во второй главе описывается анализ предметной области.

В третьей главе описаны требования к разрабатываемой системе и приведено проектирование архитектуры системы.

В четвертой главе описаны программные средства, участвующие в

разработке сервиса, а также описаны детали реализации всех компонентов системы.

В пятой главе описано функциональное тестирование системы.

В шестой главе описаны эксперименты применения разработанной системы.

В заключении подводится итог проделанной работы.

1. Методы поиска шаблонов
   1. Формальные определения

Поиск ассоциативных правил (Association rules mining) [2] - это статистический метод анализа данных, используемый для выявления сильных связей и закономерностей между элементами в больших наборах данных. Данный метод может применяться в различных областях, таких как маркетинг, биоинформатика, медицина и т.д.

Ассоциативные правила или шаблоны определяются как выражения вида «если А, то В» или «A приводит к B». Слева находится антицендент, справа – консеквент.

Антицендент обычно относится к условию или предпосылке правила или импликации. Это часть высказывания, которая предшествует символу "→" или "если...то" и определяет условие для истинности высказывания в целом.

Консеквент связан с результатом или выводом правила или импликации. Он представляет собой часть высказывания, которая следует после символа "→" или "если...то" и указывает на результат или следствие, которое происходит, если условие (антицендент) истинно.

Предположим, у нас имеется некий набор данных или D, такой, что D = t0…tk, в котором t – транзакция, которая является заданным набором разных элементов i, i ⊆ I, где I – множество всех возможных элементов или товаров. Любую возможную комбинацию из элементов i можно назвать набором элементов (itemset), набор может включать и один элемент.

Обычно для дальнейшего применения алгоритмов набор данных трансформируют в разреженную матрицу, где каждая транзакция представляет собой множество всех товаров с булевым значением, обозначающим присутствие данного товара в транзакции, например: t1 = [{i1: 1}, {i2: 1} … {in : 0}], t2 = [{i1: 1}, {i2: 1} … {in : 0}] … tk = [{i1: 0}, {i2: 1} … {in : 0}], где tk – транзакция, а in – элемент.

Априори (apriori) - алгоритм поиска ассоциативных правил. Одной из главных идей алгоритма является то, что он использует свойство антимонотонности множеств, например, если набор элементов X не является частым, то набор элементов Y, который состоит из набора X и нового элемента, заведомо не будет частым и его можно отбросить. Алгоритм apriori или другие аналоги используют некоторые метрики, которые помогают сделать выводы из набора элементов.

Поддержка (support) – метрика, которая позволяет определить, насколько часто шаблон встречается в транзакциях и вычисляется следующим образом:

где X и Y – наборы элементов, t – транзакция, D – датасет.

Достоверность (confidence) – метрика, которая помогает определить вероятность того, что набор элементов X следует набору Y, ее также можно интерпретировать как «на сколько часто данное правило срабатывает для всего набора данных».

Подъем (lift) – метрика, которая показывает, насколько оба набора элементов зависят друг от друга, то есть показывает корреляцию:

Подъем может являться ключевой в случае оценки зависимости наборов элементов. Если lift > 1, можно говорить о том, что между наборами есть зависимость. Чем больше lift, тем больше зависимость, а если lift < 1, то наборы данных влияют друг на друга негативно.

* 1. Обзор работ по тематике

За время существования метода поиска ассоциативных правил было проделано некоторое число работ в сфере образования, ниже рассмотрены работы, касающиеся поиска шаблонов конкретно в данных успеваемости студентов.

В работе [3] одна часть данных представляет собой успеваемость студентов бакалавров, другая - магистров и аспирантов. Предметы, по которым дана успеваемость, одинаковы в обеих случаях. Этапы получения правил были стандартными – очистка данных, трансформация, применение алгоритма априори. Авторам работы получилось выделить несколько сильных (имеющих прочную взаимосвязь) правил, из которых следовало, что студенты, получившие во время бакалавриата по какому-либо предмету оценку C (примерно тройка), исправляются и получают оценку лучше во время магистратуры или аспирантуры, но другое правило говорит об обратном, что студенты с оценкой E (двойка) не исправляются и получают ту же оценку в магистратуре и аспирантуре. Авторы выделяют несколько возможных причин, таких как незаинтересованность студентов, неудобное расписание, недостаточно хорошая работа преподавателей или недостаточно хорошая учебная программа, которая составлена так, что совсем слабые в предмете студенты так и не могут справиться с ним в дальнейшем.

В работе [4] исходные данные содержали успеваемость за тест, пол, сложность предмета, отношение к обучению, чувство напряжение во время теста, место в классе. Использовался тот же алгоритм Априори, этапы получения правил не отличались от предыдущей работы. Из найденных ассоциативных правил удалось обнаружить, например, что в тестах, где ученик не нервничает, ему удается получить хорошую оценку чаще. Также, ученики, сидящие с первой по третью парту, чаще получают хорошую оценку.

В работе [5] разрабатывали систему рекомендаций на основе поиска ассоциативных правил. Данные представляли собой успеваемость бакалавриата по специальностям коммерции и деловому администрированию. Поиск правил делился на все те же три этапа: очистка данных, трансформирование данных и применение алгоритма априори.

Благодаря поиску ассоциативных правил удалось установить, что если студент наберет 90% баллов на заданиях за курс, то вероятнее всего он наберет 70% и более баллов на экзамене или если он наберет меньше 50% баллов на заданиях, то наберет меньше 50% баллов на экзамене

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ
   1. Описание предметной области

Вследствие получения и хранения большого объема данных, а также интересах университета в улучшении качества образования, в ЮУрГУ возникла задача создания системы извлечения, обработки и анализа данных успеваемости студентов.

В системе Универис хранится информация о журналах дисциплин, студентах, преподавателях, оценках и т.п. Для получения данных необходим специальный URL адрес, через который можно получать данные из API Универис за нужные промежутки времени, либо данные, относящиеся к каким-либо сущностям.

Данная разрабатываемая система будет представлять собой приложение с пользовательским интерфейсом, с которым может взаимодействовать рядовой пользователь. В качестве метода анализа данных будет использоваться достаточно популярный метод - метод поиска ассоциативных правил, выявляющий сильные связи и закономерности между элементами в больших наборах данных. Приложение позволит проанализировать имеющиеся в системе Универис данные посредству нахождения закономерностей, сделать важные выводы из них и принять решения, которые поспособствуют качеству обучения.

* 1. Описание исходных данных

Данные для работы необходимо получить из портала Универис, где хранится успеваемость студентов, информация о дисциплинах, а также информация о студентах и преподавателях. Для получения данных из системы Универис руководством ЮУрГУ был предоставлен доступ к API, с помощью которого можно выполнять get-запросы и получать нужные данные для работы.

На рисунке 1 приведена диаграмма предоставляемых API сущностей, а также показана стрелками возможная последовательность запросов от одной сущности к другой.

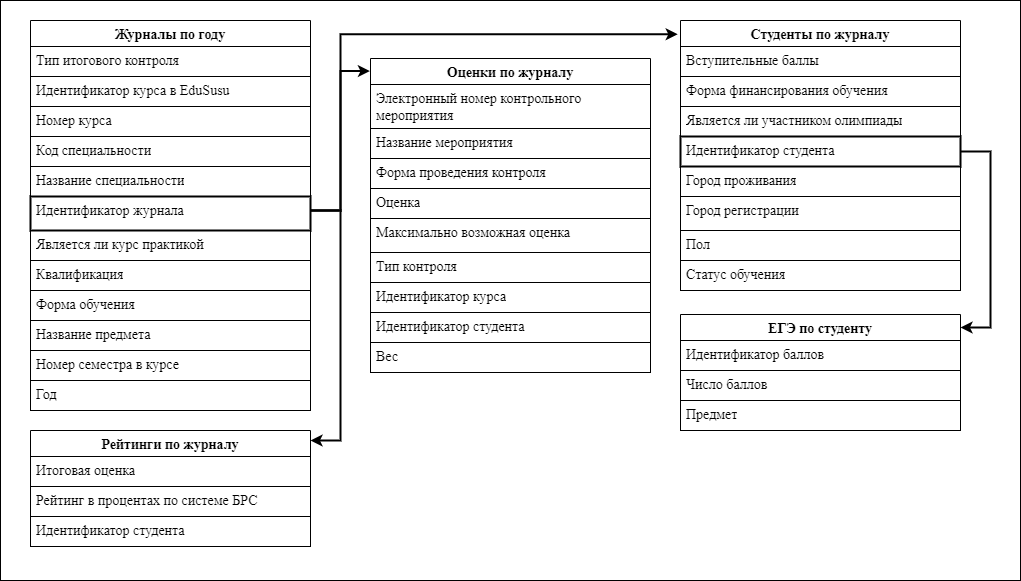


Рисунок 1 - диаграмма предоставляемых API сущностей

Далее подробным образом рассмотрены атрибуты сущностей, которые нужно получить через запросы. Параметры запросов указаны в треугольных скобках.

Запрос «JournalsByYear<Год>» возвращает все журналы за указанный год. Данные запроса являются ключевым связующим, так как дают идентификаторы журналов, по которым будут выполняться другие запросы, и связывают идентификаторы журналов с параметрами, по которым можно будет фильтровать журналы и, вследствие этого, другие запросы.

Таблица 1 - Атрибуты запроса JournalsByYear

|  |  |  |
| --- | --- | --- |
| Атрибут | Семантика | Пример |
| CheckType | Тип итогового контроля | зачет |
| CourseEduId | Идентификатор курса в EduSusu | 115479 |
| CourseNumber | Номер курса | 2 |
| DirectionCode | Код специальности | 13.03.02 |
| DirectionName | Название специальности | Электроэнергетика и электротехника |
| Id | Идентификатор журнала | f1caff4a-a61e-45e4-aff7-0005e88edf5e |
| IsPractice | Является ли курс практикой | False |
| Speciality | Квалификация | бакалавр |
| StudyForm | Форма обучения | очная |
| SubjectName | Название предмета | Общая энергетика |
| Term | Номер семестра в курсе | 2 |
| Year | Год | 2020 |

Запрос «GetStudentsByJournalId<Идентификатор журнала>» возвращает данные всех студентов из конкретного журнала. Некоторые данные студента могут стать возможной причиной его успеваемости в будущем анализе.

Таблица 2 - Атрибуты запроса GetStudentsByJournalId

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Семантика** | **Пример** |
| EnrollScore | Вступительные баллы | 200 |
| FinancialForm | Форма финансирования обучения | бюджет |
| HasOlympParticipation | Является ли участником олимпиады | False |
| Id | Идентификатор студента | 3b84df97-9887-4d99-8ea3-e846b6a4e510 |
| LiveCity | Город проживания | г.Катав-Ивановск |
| RegisterCity | Город регистрации | г.Катав-Ивановск |
| Sex | Пол | Мужской |
| Status | Статус обучения | учится |

Запрос “GetGradesByJournalId<Идентификатор журнала>” возвращает данные всех оценок в конкретном журнале. Данные этого запроса являются ключевыми и будут соотносится с другими данными для установления связи успеваемости студента с другими факторами.

Таблица 3 - Атрибуты запроса GetGradesByJournalId

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Семантика** | **Пример** |
| ElectronicEventNumber | Электронный номер контрольного мероприятия | 4581431.0 |
| EventName | Название мероприятия | Контрольная работа 2 |
| EventType | Форма проведения контроля | Assignment |
| Grade | Оценка | 0 |
| GradeMax | Максимально возможна оценка | 20 |
| GradeType | Тип контроля | Текущий контроль |
| Id | Идентификатор курса | 7b0d416d-1e4e-4c2e-ad36-0506d09b2ffa |
| StudentId | Идентификатор студента | 28e0ce37-60f1-4dd1-92e1-645be6af6649 |
| Weight | Вес | 0.20 |

Запрос “GetRatingsByJournalId<Идентификатор журнала>” возвращает список итогового рейтинга студентов по конкретному журналу. Может служить индикатором того, что студент не сдал какой-то предмет.

Таблица 4 - Атрибуты запроса GetRatingsByJournalId

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Семантика** | **Пример** |
| Mark | Итоговая оценка | 5 |
| Rating | Рейтинг в процентах по системе БРС | 94.00 |
| StudentId | Идентификатор студента | 913e28b7-0ccf-4f05-b7e5-ec75caa1f3a1 |

Запрос GetEgeMarksByStudentId<Идентификатор студента> возвращает оценки за вступительные испытания конкретного студента. В этой работе может служить для возможной связи с успеваемостью студента или другими признаками.

Таблица 5 - Атрибуты запроса GetEgeMarksByStudentId

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Семантика** | **Пример** |
| Id | Идентификатор баллов | b9e32fff-a5f1-46a1-ac4a-24573b554c08 |
| Mark | Число баллов | 70 |
| Subject | Предмет | Русский язык |

1. ПРОЕКТИРОВАНИЕ СИСТЕМЫ
   1. Требования к программе

В ходе проектирования были определены функциональные и нефункциональные требования к разрабатываемой системе.

Функциональные требования

Функциональные требования устанавливают требования о том, как должна себя вести система и прямым образом относятся к функционалу системы.

Разрабатываемая система должна удовлетворять следующим функциональным требованиям.

1. Система должна предоставлять возможность запроса и загрузки в программу данных из системы Универис.
2. Система должна предоставлять возможность выбора параметров отбора запрашиваемых данных (год, кафедра, специальность и т.д.).
3. Система должна предоставлять возможность экспорта загруженных данных.
4. Система должна предоставлять возможность загрузки ранее экспортированных данных в программу.
5. Система должна предоставлять возможность выбора минимальных значений характеристик, по которым будет произведен поиск ассоциативных правил.
6. Система должна предоставлять возможность поиска ассоциативных правил в загруженных данных.
7. Система должна предоставлять возможность просмотра итогового списка правил.
8. Система должна предоставлять возможность фильтрации найденных правил по критериям поддержки, достоверности и т.д.
9. Система должна предоставлять возможность экспорта итогового списка в формате Excel.

Нефункциональные требования

Нефункциональные требования устанавливают требования ограничений для системы, которые не влияют прямым образом на функционал системы.

Система должна удовлетворять следующим нефункциональным требованиям:

1. Система должна быть написана на языке Python с использованием библиотек;
2. Система должна представлять собой веб-приложение и использовать фреймворк Dash;
   1. Диаграмма вариантов использования

Для описания способов взаимодействия с системой была разработана диаграмма вариантов использования на языке UML, представленная на рисунке 1.

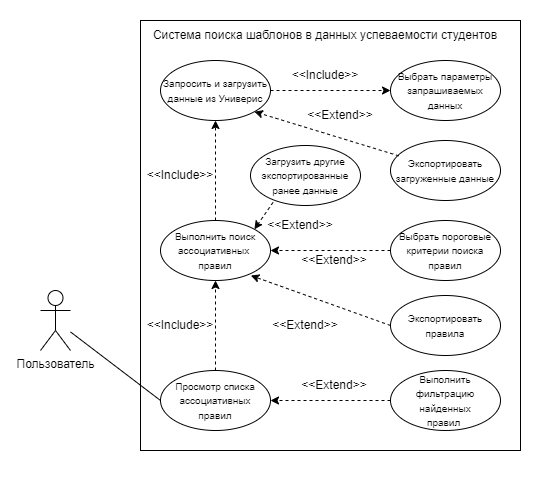


Рисунок 2 - Диаграмма вариантов использования

Единственным актером является пользователь. Для него существуют следующие вариации использования:

1. Экспортировать загруженные данные. Пользователь может выполнить экспорт ранее запрошенных и загруженных данных;
2. Выбрать параметры запрашиваемых данных. Пользователь может выбрать параметры запрашиваемых данных, такие как года специальности и квалификации студентов;
3. Запросить и загрузить данные из Универис. Пользователь выполняет запрос и загрузку данных из системы Универис;
4. Выполнить поиск ассоциативных правил. Пользователь может выполнить поиск ассоциативных правил;
5. Выбрать пороговые критерии поиска правил. Пользователь устанавливает пороговые значения для выбора правил (поддержка, достоверность, подъем);
6. Загрузить другие экспортируемые ранее данные. Пользователь может загрузить другие экспортируемые ранее данные в программу, которые он когда-либо запрашивал и экспортировал
7. Экспортировать правила. Пользователь может экспортировать правила в формате Excel;
8. Просмотр списка ассоциативных правил. Пользователь может просмотреть и проанализировать найденные правила;
9. Выполнить фильтрацию найденных правил. Пользователь может выполнить фильтрацию правил по наличию определенного товара в шаблоне или по значениям характеристик правила (поддержка, достоверность, подъем).
   1. Архитектура системы

Система будет представлять собой веб приложение, состоящее из нескольких модулей: модуль запроса и сохранения данных, модуль преобразования данных, модуль поиска ассоциативных правил и модуль интерфейса.

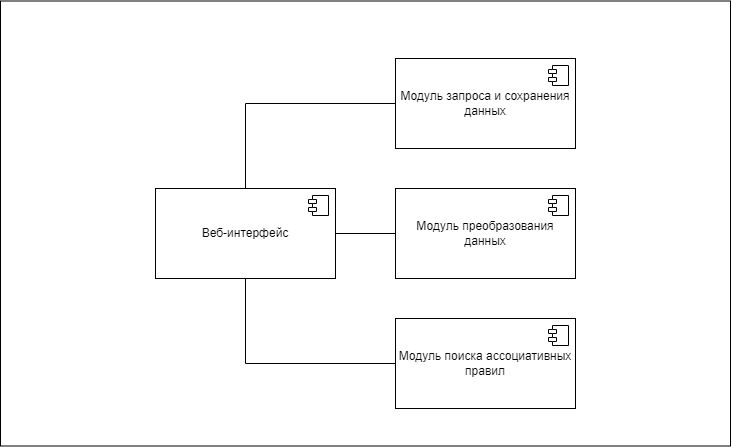


Рисунок 3 - Диаграмма компонентов

* 1. Модуль запроса данных

Для системы, которая анализирует данные, необходима возможность запрашивать и загружать данные, которые можно сохранить и использовать потом или использовать сразу же после запроса. Именно для этого нужен модуль запроса данных.

Модуль будет запрашивать данные используя API системы Универис «Digital trace». В модуль должны передаваться параметры, по которым нужно сделать запрос. Сначала будут производится запросы таблиц журналов по нужным годам, потом их фильтрация по квалификации студентов и названию специальности. По каждому журналу из отфильтрованной таблицы будут производится запросы таблиц контрольных рейтингов, оценок и студентов. По каждому студенту из таблицы студентов будут производиться запросы его баллов вступительных испытаний. После каждого запроса по Id журнала или Id студента к таблице будет добавляться колонка с этим Id, чтобы связать таблицы для будущей работы.

Также необходимо не забывать про сами параметры, по которым будут фильтроваться нужные журналы. В модуле необходимы методы, которые позволят выполнять те же запросы журналов по выбранным годам, но будут сохранять только данные параметров с колонками «Год», «Квалификация», «Специальность». Далее эти данные будут использоваться в интерфейсе для выбора параметров запроса журналов.

* 1. Модуль преобразования данных

Для поиска шаблонов необходимо преобразовать ранее загруженные данные из таблиц в товары, которые будут составлять транзакции. Товары будут формировать собой транзакцию по каждому студенту по каждому семестру.

Преобразование данных в товары описано в таблице 6. В столбцах «Название таблицы» и «Название атрибута» указывается из какой таблицы и поля берутся данные, а в «Описании трансформации» указывается, как преобразуются данные.

Таблица 6 - Преобразование данных из сохраненных таблиц в товары

|  |  |  |
| --- | --- | --- |
| Название таблицы | Название атрибута | Описание трансформации |
| Students | FinancialForm | Добавление «Контрактик», если форма обучения «контракт», добавление «Бюджетник», если форма обучения «бюджет» |
| Students | RegisterCity | Если город Челябинск – добавляется товар «Местный»,  любой другой город– «Иногородний» |
| Ege | Mark | Если среднее число выступительных баллов > 80, добавляется товар «Высокобалльник», в противном случае – «Невысокобалльник» |
| Journals | Term, CourseNumber | Если статус студента «отчислен», берется журнал с самым поздним семестром, где находится студент, добавляется товар «Отчислен на <номер семестра> семестре». В остальных случаях прямое добавление: «Закончил», «В академе» или «Учится» из поля «Status» |
| Students | Status |
| Journals | Year, CourseNumber | Вычисляется год поступления студента по году и номеру курса журнала, в котором он находится. Добавляется товар «Поступил в <Год поступления>» |
| Grades | Grade, GradeMax | Сначала оценки преобразуются в вид от 0 до 5, далее для каждого семестра по каждому студенту считается его процент оценок выше 4 в виде децилей: 50 и меньше, 50, 75, 85, 100. Добавляется товар: «% Оценок выше 4 <Дециль>» |
| Journals | Term, CourseNumber |
| Journals | Term, CourseNumber | Вычисляется номер семестра по номеру семестра и номеру курса журнала. Добавляется товар «<Номер семестра> Семестр» |
| Grades | Mark | Если у студента за семестр есть итоговая оценка ниже тройки – добавляется товар «Пересдача», в обратном случае – «Без пересдачи» |
| Journals | Term, CourseNumber |
| Students | FinancialForm | Вычисляется процент контрактников в группе, по проценту определяются децили 25 и меньше, 25, 50, 75, 100 и добавляется товар «% контрактников в группе <Дециль>» |

Структура транзакции расположена в таблице 7.

Таблица 7 - Структура транзакции

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TID | Items | | | | |
| 1 | <Номер семестра> Семестр | Поступил в <Год начала обучения> | Закончил/отчислен в<номер>семестре | Процент оценок выше 4 за семестр (Децили: 50, 50 и меньше, 75, 85, 100) | Бюджетник/контрактник |
| Местный/ иногородний | Высокобалльник/не высокобалльник | Есть пересдача/нет пересдачи | Процент контрактников в группе (Децили: 25 и меньше, 25, 50, 75, 100) |  |

После формирования списка транзакций, транзакции должны быть преобразованы в вид разряженной матрицы, который необходим для поиска шаблонов.

* 1. Модуль поиска ассоциативных правил

Модуль поиска ассоциативных правил должен получать на вход сформированную ранее разреженную матрицу, минимальные значения метрик, выполнять по ней поиск шаблонов по заданным метрикам и возвращать список правил, который в дальнейшем может быть выведен и экспортирован.

* 1. Разработка макета

Для создания интерфейса системы были разработаны макеты интерфейса. Макет представляет собой нефункциональный прототип интерфейса и используется для отображения содержания страницы и расположения элементов. Для создания макета был использован онлайн редактор Figma [6].

На рисунке 2 можно увидеть главную страницу приложения, которая предоставляет ключевой функционал системы, а именно выбор данных и поиск ассоциативных правил.

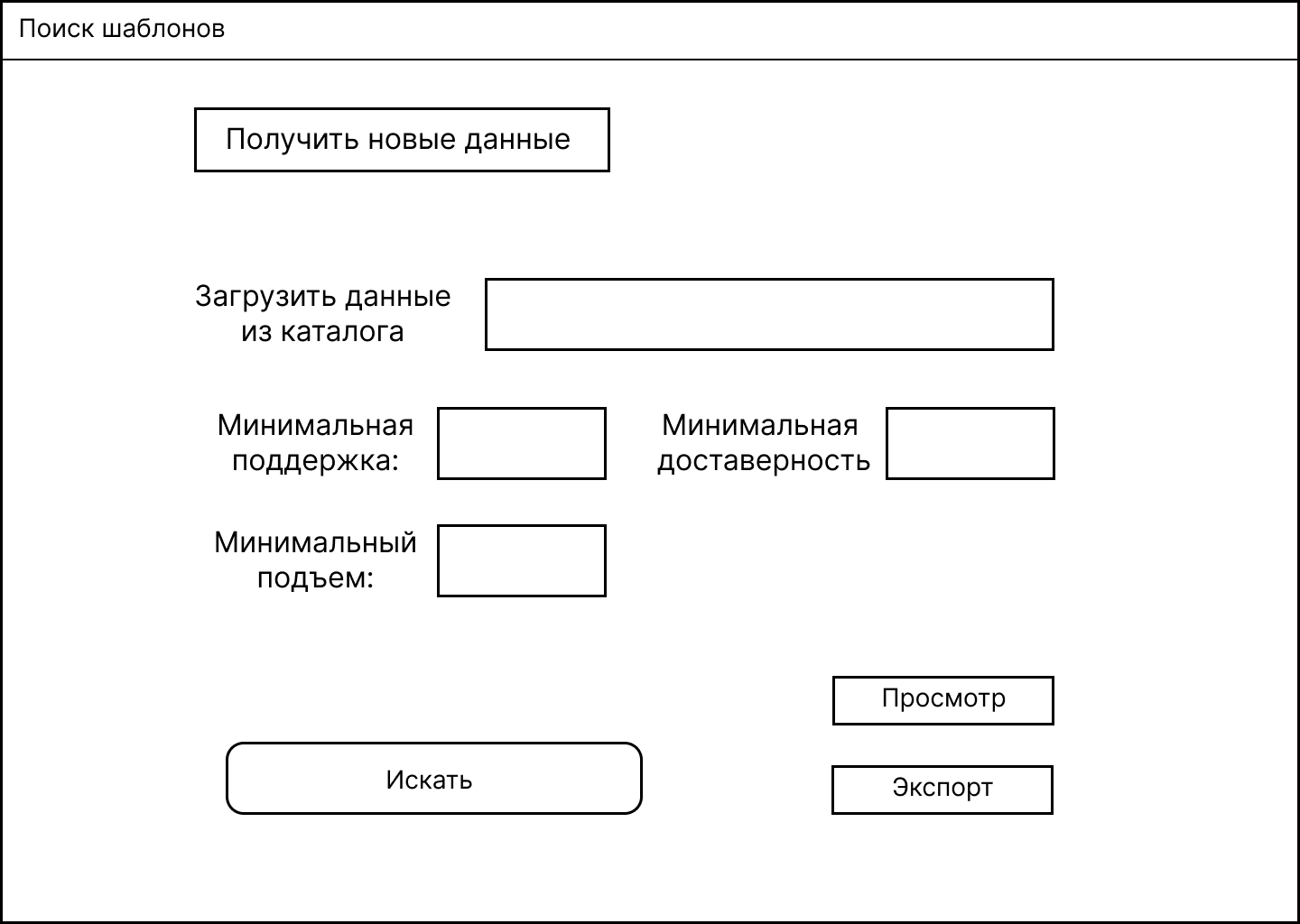


Рисунок 4 - Макет главной страницы

На рисунке 3 изображен макет окна загрузки данных. Так как выбор одних параметров фильтрации сохраняемых данных будет влиять на другие, был выбран древовидный интерфейс выбора параметров.

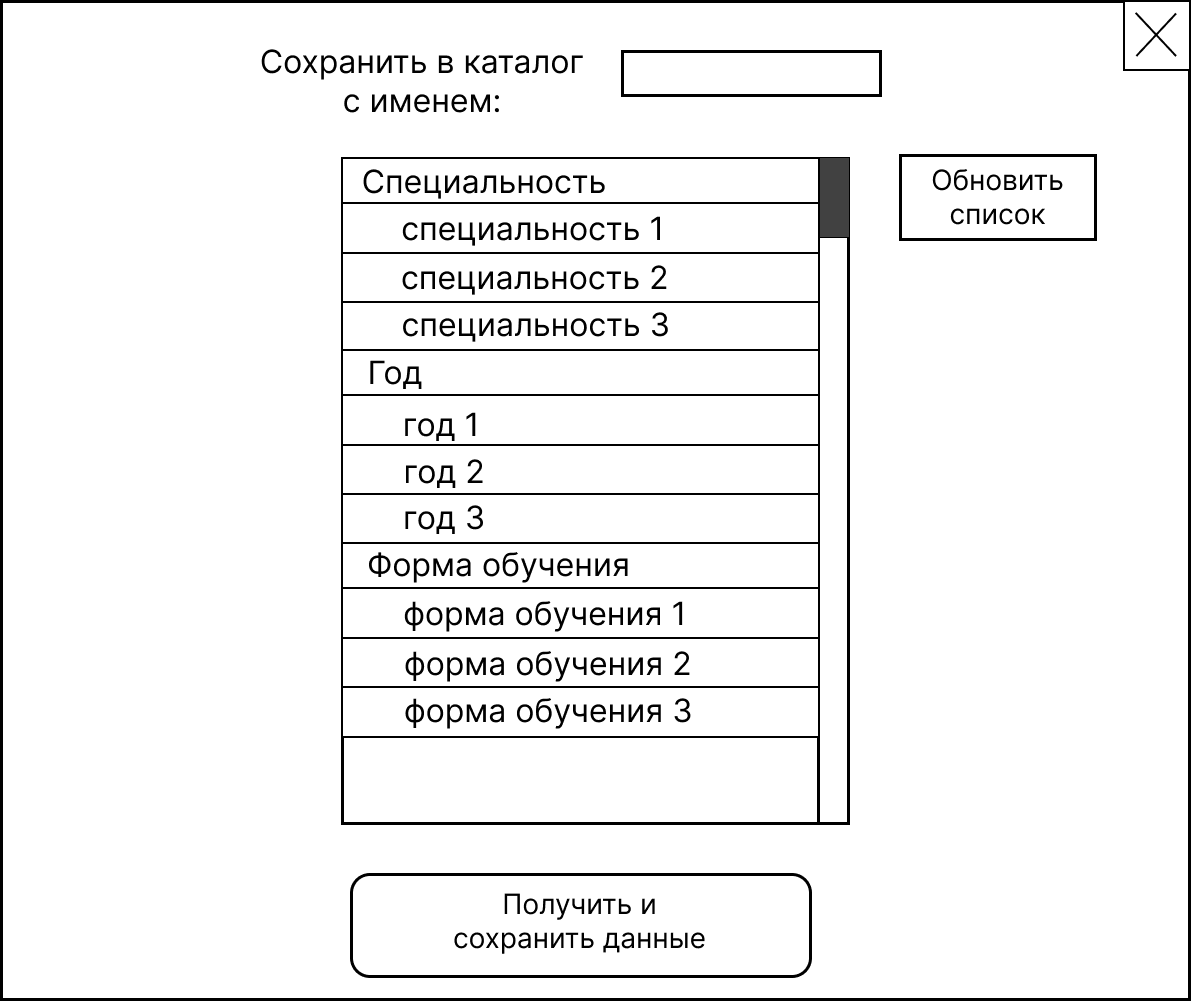


Рисунок 5 - Макет окна запроса данных

На рисунке 4 изображено окно просмотра найденных шаблонов, основной функционал которого – вывод таблицы с правилами, поэтому окно таблицы будет занимать максимально возможное пространство.

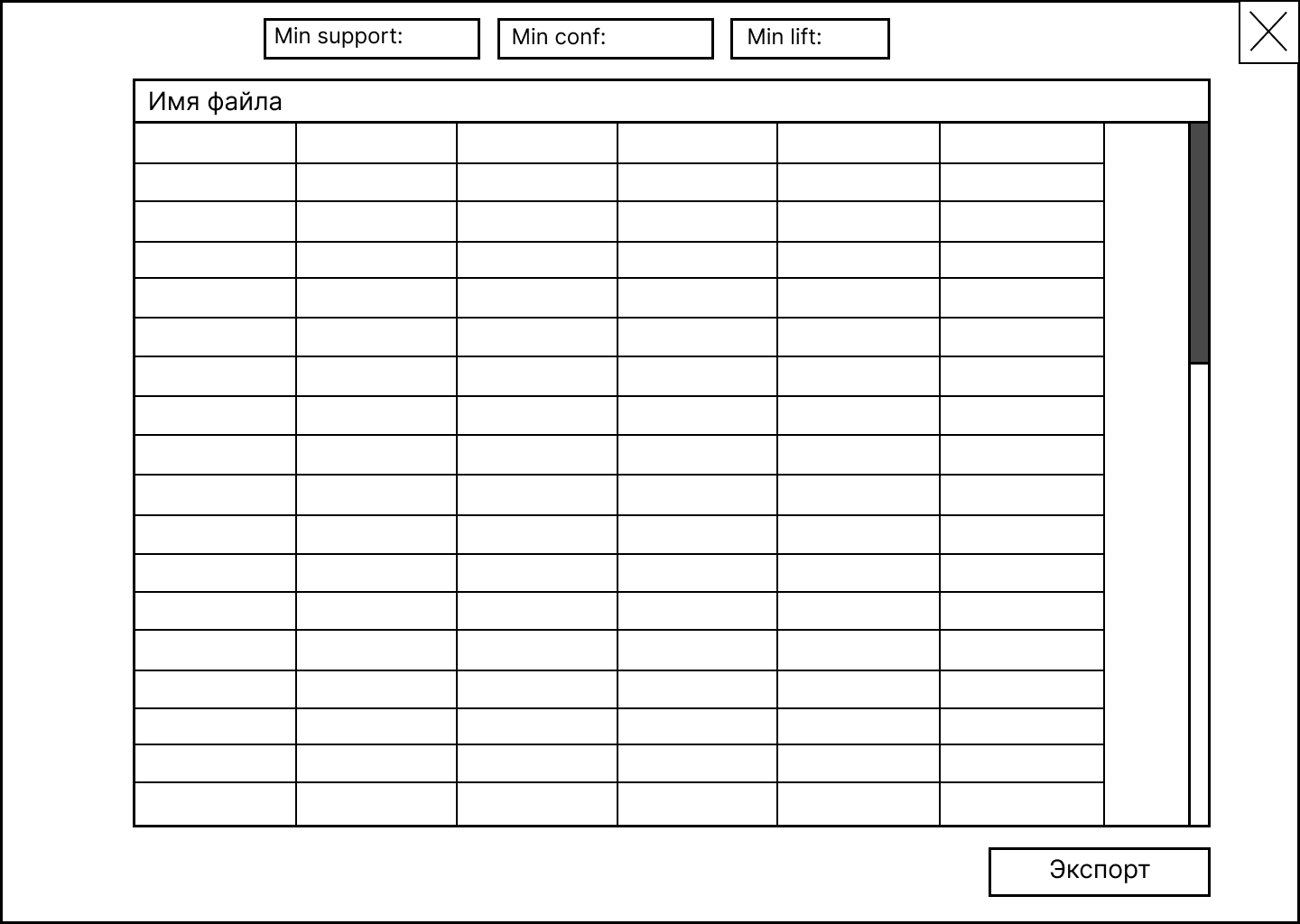


Рисунок 6 - Макет окна просмотра итоговых правил

Ниже представлено описание для основного контента каждой из страниц сайта:

1. На главной странице содержится кнопка запроса данных, кнопка загрузки данных из каталога программы, места ввода минимальных параметров поиска ассоциативных правил, а также кнопки возможности экспорта и просмотра данных и кнопка запуска самого поиска правил.

2. В окне запроса данных содержится место ввода названия папки, в которую будут загружаться данные, древовидный интерфейс выбора параметров загружаемых данных, кнопка обновления списка этих параметров, а также кнопка запуска самой загрузки.

3. В окне просмотра содержатся ячейки для фильтрации правил по метрикам, окно с таблицей правил и еще одна кнопка экспорта, для возможности экспортировать уже отфильтрованные правила.

1. РЕАЛИЗАЦИЯ СИСТЕМЫ
   1. Программные средства реализации

Реализация системы поиска шаблонов в данных успеваемости портала Универис производилась с помощью языка программировани Python версии 3.10.8 и следующего набора программных продуктов и библиотек:

Pandas - это библиотека для работы с данными в Python, предоставляющая высокоуровневые структуры данных, такие как DataFrame и Series [7].

Numpy - это библиотека для научных вычислений в Python, которая предоставляет массивы и матрицы, а также множество функций для работы с ними, таких как арифметические операции, индексирование, фильтрация, транспонирование, сортировка и многое другое [8].

Mlxtend - это библиотека для машинного обучения в Python, которая предоставляет множество инструментов для выполнения различных задач машинного обучения, включая классификацию, регрессию, кластеризацию, ассоциативные правила и так далее [9].

Dash - это фреймворк для создания интерактивных веб-приложений с использованием языка программирования Python. Он позволяет разработчикам создавать элегантные и гибкие веб-интерфейсы для визуализации данных, построения дашбордов, создания форм и других вещей [10].

Asyncio - это стандартная библиотека Python для асинхронного программирования, основанная на концепции сопрограмм. В асинхронном коде, написанном с помощью asyncio, можно выполнять несколько задач одновременно, что увеличивает производительность приложения [11].

Aiohttp - это библиотека для асинхронного HTTP-клиента и сервера в Python. Она позволяет работать с HTTP-соединениями асинхронно, что увеличивает производительность приложения и позволяет выполнять несколько запросов одновременно [12].

* 1. Модуль запроса данных

Модуль запроса данных представляет функционал для запроса данных из API системы Универис. Часть функций модуля реализованы с помощью библиотек Aiohttp и Asyncio, что позволяет производить сразу несколько запросов асинхронно, тем самым экономя время пользователя. Модуль экспортирует следующие функции:

­- request\_and\_save - выполняет запросы и возвращает данные о журналах, студентах, оценках, рейтингах и ЕГЭ по годам, квалификациям и специальностям.

- load\_params – выполняет запрос о журналах и возвращает отфильтрованные параметры журналов (года, квалицикации, специальности) в формате JSON.

Подробное описание интерфейсов экспортируемых и внутренних функций приведено в листингах 1-5 приложения А.

* 1. Модуль преобразования данных

Модуль преобразования данных позволяет преобразовать данные запрошенных таблиц в вид матрицы транзакций, который необходимый для поиска шаблонов. Преобразования осуществляется по средству использования библиотеки Pandas для манипуляции с табличными данными [7. ]. Принцип добавления товаров и вид транзакций описаны в пункте 3.5.

Модуль состоит только из одной экспортируемой функции convert\_to\_transactions, которая получает на вход таблицы Dataframe журналов, студентов, оценок, ЕГЭ, рейтингов, конвертирует их в транзакции и возвращает разреженную матрицу этих транзакций. Подробное описание интерфейса приведено в листинге 6 приложения А.

* 1. Модуль поиска ассоциативных правил

Модуль поиска ассоциативных правил выполняет главную функцию поиска шаблонов в данных успеваемости студентов. Поиск ассоциативных правил осуществляется благодаря библиотеке Mlxtend [9. ].

Модуль состоит только из одной экспортируемой функции mine\_patterns, которая принимает на вход матрицу транзакций и минимальные значения метрик, выполняет поиск и возвращает Dataframe с ассоциативными правилами, а также метрики для каждого правила. Подробное описание интерфейса приведено в листинге 7 приложения А.

* 1. Реализация интерфейса

Интерфейс представляет собой веб-приложение, которое было реализовано с помощью фреймворка Dash [10. ]. Интерфейс состоит из страницы получения данных и страницы поиска шаблонов. На каждой странице сверху присутствует меню, которое осуществляет переадресацию на другую страницу. Страница получения данных изображена на рисунке 7.

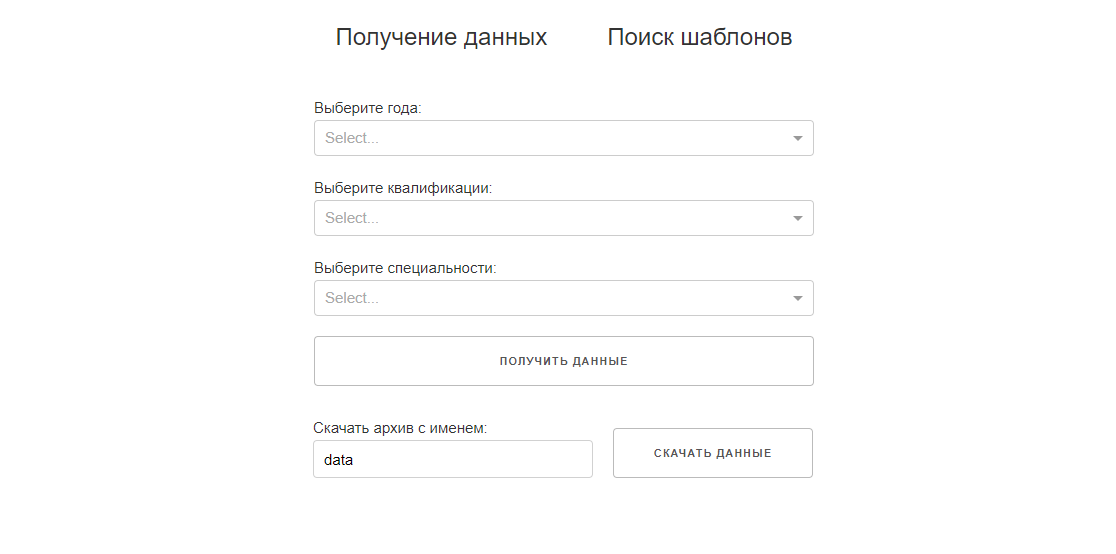


Рисунок 7 - Интерфейс получения данных

Страница получения данных представляет интерфейс для осуществления запросов в API системы Универис по нужным параметрам и содержит поля выбора годов, квалификаций и специальностей, кнопку получения данных, а также поле ввода имени сохраняемого архива и кнопку скачивания запрошенных данных. Поля выбора параметров реализованы динамически. После выбора нужных годов обновляется содержимое поля выбора квалификаций, после выбора квалификации обновляется содержимое поля выбора специальностей. После отстройки всех нужных параметров пользователь может сделать запрос, который далее можно будет либо сохранить, введя имя для архива и нажав кнопку «Скачать данные», либо использовать на другой странице.

Следующая страница для поиска шаблонов изображена на рисунке 8.

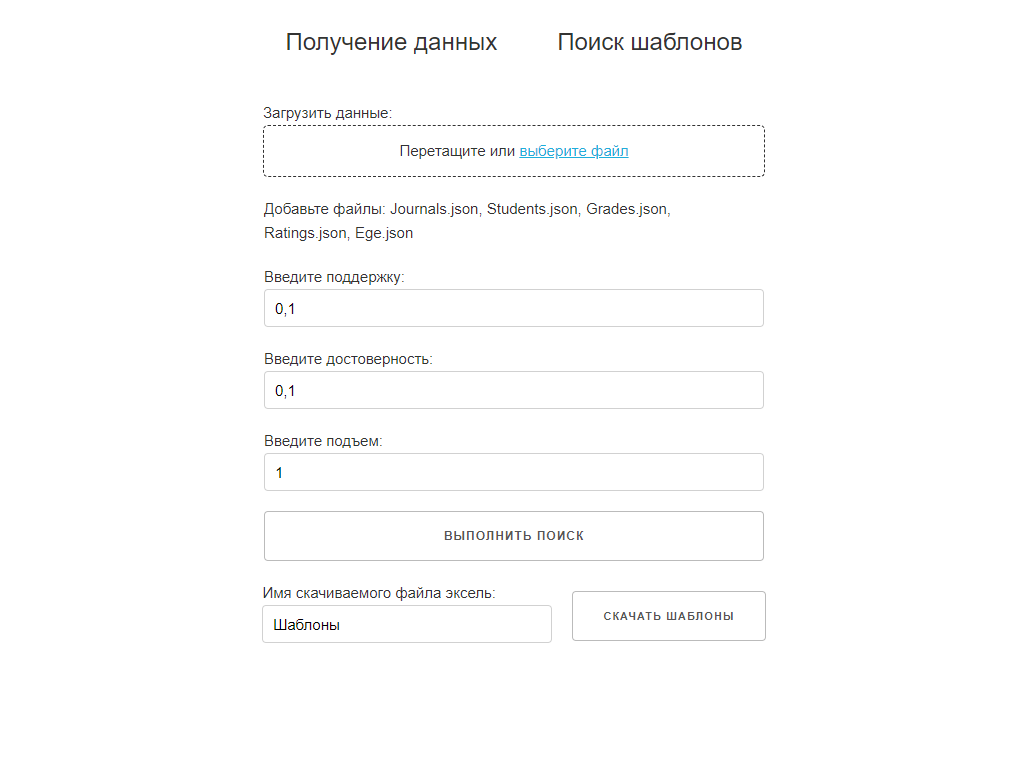


Рисунок 8 - Интерфейс поиска шаблонов

Страница для поиска шаблонов содержит интерфейс, необходимый для загрузки ранее сохраненных данных, настройки минимальных метрик (поддержка, достоверность и подъем), поиска, просмотра и скачивания шаблонов. Чтобы загрузить данные в поле загрузки данных, пользователь может либо нажать на выделенную надпись и откроется проводник, либо самостоятельно перетащить в поле файлы. Снизу от поля отображается информация о добавленных файлах, а также о файлах, которые нужно добавить для поиска. Поля для ввода минимальных метрик содержат параметры по умолчанию, при желании можно ввести собственные. Чтобы скачать полученные шаблоны можно ввести имя для скачиваемого файла и нажать на кнопку «Скачать шаблоны». При нажатии на кнопку «Выполнить поиск» будет выполнен поиск шаблонов по загруженным данным и введенным метрикам, результат отобразится ниже, пример вывода таблицы изображен на рисунке 9.

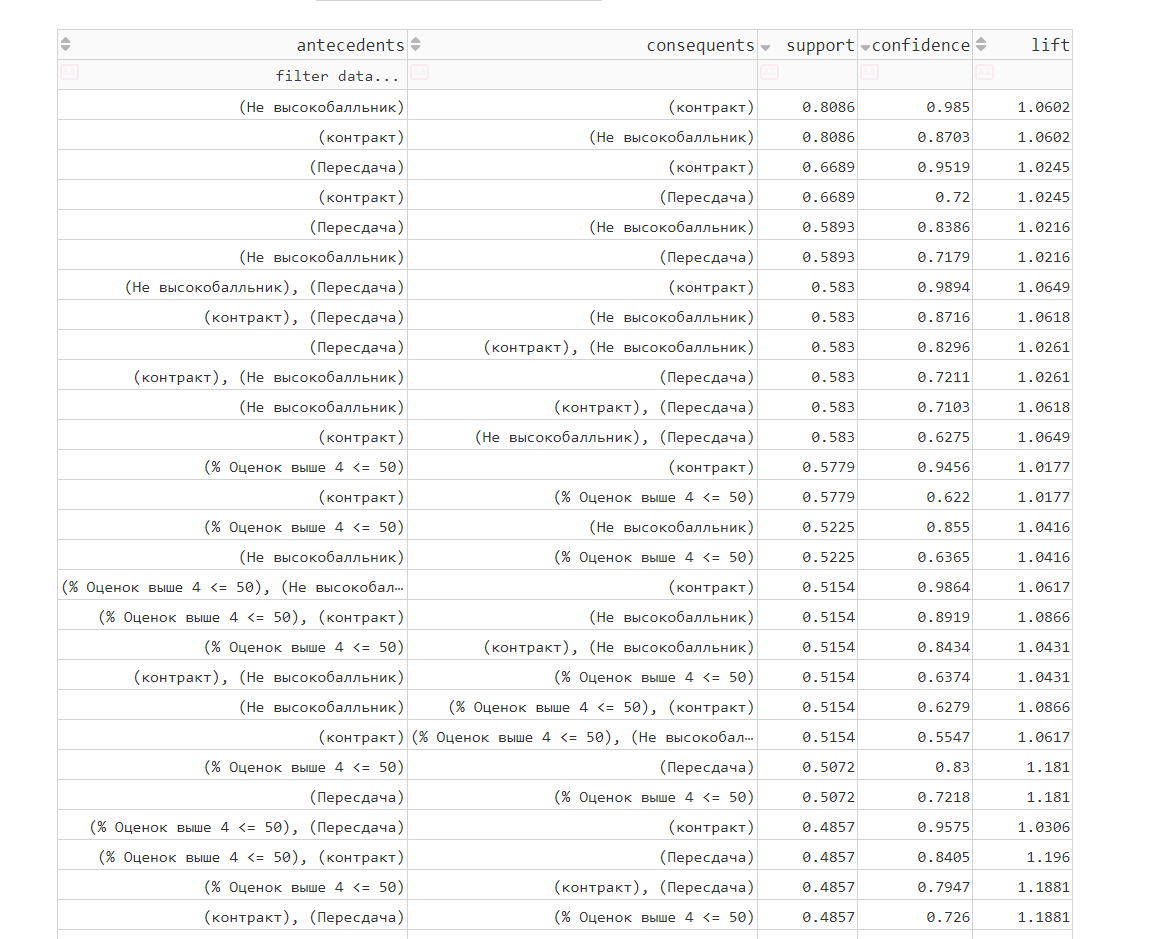


Рисунок 9 - Вывод таблицы шаблонов

В таблице пользователь может сортировать и фильтровать данные, используя элементы управления, находящиеся на верху таблицы. С помощью стрелок слева от названия колонки можно осуществлять сортировку по возрастанию и убыванию, при этом сортировка работает на несколько столбцов. Чтобы отфильтровать строчные данные по ключевому слову, нужно ввести слово снизу от названия столбца. Числовые данные можно фильтровать вводом выражений, например «> 0.5» в строке с поддержкой отобразит все строки, в которых поддержка больше 0.5.

1. Тестирование
   1. Функциональное тестирование

Этап тестирования в разработке системы играет ключевую роль в обеспечении надежности. Цель тестирования - обнаружение ошибок и дефектов в функциональности и работоспособности системы. Это позволит исправить проблемы до выпуска готового продукта, чтобы система соответствовала требованиям и ожиданиям пользователя. Тестирование также направлено на выявление уязвимостей и проблем безопасности, чтобы предотвратить несанкционированный доступ и неправильное использование системы.

Одним из подходов к тестированию является функциональное тестирование, которое проверяет работоспособность системы, включая ввод-вывод данных и взаимодействие с другими компонентами. Это позволяет убедиться, что система выполняет требуемые функции и работает правильно. Функциональное тестирование также выявляет ошибки и дефекты, связанные с функциональностью, и помогает разработчикам обеспечить стабильность и надежность системы.

В таблице 1 представлено функциональное тестирование системы

Таблица 8 - Функциональное тестирование

| **№** | **Название теста** | **Шаги** | **Ожидаемый результат** | **Тест пройден?** |
| --- | --- | --- | --- | --- |
|  | Загрузка данных бакалавриата кафедры системного программирования за 2019 и 2020 годы | На странице получения данных нужно:   1. В поле «выберите года» выбрать «2019» и «2020»; 2. в поле «выберите квалификации» выбрать «бакалавр» 3. в поле «выберите специальности» выбрать «Программная инженерия» и «Фундаментальная информатика и информационные технологии» 4. нажать на кнопку «Получить данные». | На этой же странице под кнопкой «Получить данные» отобразится сообщение «Данные успешно получены» и будет доступно скачивание данных. На странице поиска шаблонов под полем загрузки данных будет отображено сообщение «Найдены все нужные файлы» и список найденных файлов. Будет доступен поиск шаблонов. | Да |
|  | Проверка скачивания загруженных данных | На странице получения данных нужно:   1. осуществить загрузку данных 2. написать имя файла в поле «Скачать архив с именем» или оставить имя по умолчанию; 3. нажать на кнопку «Скачать данные» | Выполнится скачивание архива .zip с введенным именем, в котором будет находиться 5 файлов .json с таблицами. | Да |
|  | Проверка загрузки ранее скаченных данных в виде архива zip с 5 файлами с именами Journals.json, Students.json, Grades.json, Ratings.json, Ege.json | На странице поиска шаблонов нужно перетащить или выбрать нужный архив. | Выполнится загрузка данных. После этого снизу обновится информация о том, что найдены все нужные файлы. | Да |
|  | Проверка загрузки ранее скаченных данных в виде 5 файлов с именами Journals.json, Students.json, Grades.json, Ratings.json, Ege.json | На странице поиска шаблонов нужно перетащить или выбрать нужные файлы. | Выполнится загрузка данных. После этого снизу обновится информация о том, что найдены все нужные файлы. | Да |
|  | Проверка загрузки только файла Grades.json. | На странице поиска шаблонов нужно перетащить или выбрать нужный файл. | Выполнится загрузка данных. После этого снизу обновится информация о том, что найден файл «Grades.json» и что нужно добавить остальные файлы. | Да |
|  | Проверка загрузки файла с другим именем | На странице поиска шаблонов нужно перетащить или выбрать нужный файл. | Выполнится попытка загрузки данных. После этого снизу отобразится сообщение о неверном имени файла. | Да |
|  | Проверка поиска шаблонов | На странице поиска шаблонов нужно:   1. выбрать поддержку, достоверность и подъем, либо оставить значения по умолчанию; 2. Нажать на кнопку «Выполнить поиск» | Выполнится поиск шаблонов.  В низу экрана отобразится таблица с найденными шаблонами. | Да |
|  | Проверка фильтрации шаблонов по убыванию поддержки. | На странице поиска шаблонов нужно:   1. выполнить поиск шаблонов; 2. в поле support нажимать на иконку сортировки до тех пор, пока не останется только стрелочка вниз. | Строчки шаблонов отобразятся в порядке убывания поля «support» | Да |
|  | Проверка скачивания шаблонов | На странице поиска шаблонов нужно:   1. выполнить поиск шаблонов; 2. написать имя файла в поле «Имя скачиваемого файла эксель» или оставить имя по умолчанию; 3. нажать на кнопку «Скачать шаблоны» | Выполнится скачивание файла excel с введенным именем и находящимися внутри шаблонами. | Да |

1. ЭКсперименты

Подготовка данных

Для проведения экспериментов было загружено 2 набора данных. Первый набор данных представляет собой данные бакалавров гуманитарных специальностей «Психология», «Социология», «Лингвистика», «Политология» и «Филология» за 2018-2022 года. Второй набор – данные бакалавров программистских специальностей «Программная инженерия» и «Фундаментальная информатика и информационные технологии» за 2018-2022 года.

Для поиска шаблонов в обоих случаях были установлены следующие минимальные границы метрик: поддержка - 0.01, достоверность – 0.1, подъем – 1.

Серии экспериментов первого набора данных

Стоит отметить, что шаблоны вынимались с помощью сортировки по убыванию поддержки и достоверности и брались первые из представленных категорий, так, чтобы они отражали самый частый и вероятностный результат в категории.

Шаблоны вида «Местный/Иногородний -> Успеваемость» представлены в таблице 9 (меры поддержки, достоверности и подъема для краткости обозначены как sup, conf и lift соответственно):

Таблица 9 - Шаблоны вида «Местный/Иногородний -> Успеваемость» первого набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | Местный | Пересдача | 0.4689 | 0.7722 | 1.004 |
| 2 | Местный | % оценок выше 4 < 50 | 0.2816 | 0.4638 | 1.034 |
| 3 | Местный | Не высокобалльник | 0.0925 | 0.2356 | 1.0205 |
| 4 | Иногородний | Без пересдачи | 0.4957 | 0.8163 | 1.0095 |
| 5 | Иногородний | % оценок выше 4 => 50 | 0.1664 | 0.4238 | 1.0488 |
| 6 | Иногородний | Высокобалльник | 0.0798 | 0.2032 | 1.0619 |

Данные шаблоны показывают, что у местных студентов вышеупомянутых гуманитарных специальностей часто и с большой вероятностью низкие оценки за семестр, низкие вступительные баллы и больше пересдач. Точно противоположная ситуация у студентов из другого города. Стоит отметить, что по метрике подъема, корреляция у этих шаблонов есть, но она невысока.

Далее в таблице 10 представлены шаблоны вида «Бюджетник/Контрактник" -> Успеваемость»:

Таблица 10 - Шаблоны вида «Бюджетник/Контрактник" -> Успеваемость» первого набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | Контрактник | Пересдача | 0.4825 | 0.794 | 1.0323 |
| 2 | Контрактник | % оценок выше 4 < 50 | 0.3451 | 0.5679 | 1.2662 |
| 3 | Бюджетник | (% оценок выше 4 >= 50) | 0.1927 | 0.4913 | 1.2159 |
| 4 | Бюджетник | Без пересдачи | 0.1057 | 0.2694 | 1.1669 |

Шаблоны имеют в среднем большую меру подъема, относительно предыдущих, но показывают предсказуемый результат о том, что контрактник чаще и с большой долей вероятности будет иметь низкую успеваемость за семестр и пересдачи, бюджетник – в точности наоборот.

Далее, приведены самые частые шаблоны, в которых антицендент является семестром, а консевент – успеваемостью:

Таблица 11 - Шаблоны вида «Семестр -> Успеваемость» первого набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | 2 Семестр | % оценок выше 4 >= 50 | 0.0848 | 0.4758 | 1.1776 |
| 2 | 1 Семестр | % оценок выше 4 >= 50 | 0.083 | 0.4828 | 1.1949 |
| 3 | 3 Семестр | % оценок выше 4 < 50 | 0.0726 | 0.5556 | 1.2386 |
| 4 | 4 Семестр | % оценок выше 4 < 50 | 0.0608 | 0.4637 | 1.0338 |
| 5 | 4 Семестр | % оценок выше 4 >= 50 | 0.0531 | 0.4048 | 1.0019 |
| 6 | 5 Семестр | % оценок выше 4 < 50 | 0.0494 | 0.458 | 1.0211 |
| 7 | 7 Семестр | % оценок выше 4 >= 50 | 0.0381 | 0.4398 | 1.0884 |
| 8 | 5 Семестр | % оценок выше 4 >= 75 | 0.0168 | 0.1555 | 1.7579 |
| 9 | 6 Семестр | % оценок выше 4 >= 75 | 0.0118 | 0.1088 | 1.2301 |
| 10 | 8 Семестр | % оценок выше 4 >= 75 | 0.0109 | 0.1277 | 1.4435 |

Исходя из этих шаблонов видно, что в 3 семестре чаще и вероятнее всего низкая успеваемость, относительно первых двух семестров, причем корреляция между семестром и успеваемостью значительно больше у 3 семестра, в котором зафиксировано падение успеваемости, чем у других семестров. Далее успеваемость довольно спорная у 4 и 5 семестра, с 6 по 8 чаще и вероятнее всего хорошая успеваемость.

Серии экспериментов второго набора данных

Шаблоны вида «Местный/Иногородний -> Успеваемость» второго набора данных приведены в таблице 11:

Таблица 12 - Шаблоны вида «Местный/Иногородний -> Успеваемость» второго набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | Иногородний | Не высокобалльник | 0.3662 | 0.766 | 1.2045 |
| 2 | Иногородний | Пересдача | 0.3574 | 0.7476 | 1.0674 |
| 3 | Местный | Высокобалльник | 0.2522 | 0.4832 | 1.3272 |
| 4 | Местный | Без пересдачи | 0.179 | 0.3429 | 1.1444 |

Исходя из шаблонов видно, что местные студенты имеют чаще и с большей вероятностью высокие вступительные баллы и меньше пересдач, а иногородние имеют низкие вступительные баллы и больше пересдач, что является противоположным результатом относительно части шаблонов из экспериментов над набором данных гуманитариев.

Далее в таблице 10 представлены шаблоны вида «Бюджетник/Контрактник" -> Успеваемость»:

Таблица 10 - Шаблоны вида «Бюджетник/Контрактник" -> Успеваемость» второго набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | Контрактник | Пересдача | 0.2354 | 0.7872 | 1.1239 |
| 2 | Контрактник | % оценок выше 4 < 50 | 0.2085 | 0.6972 | 1.4935 |
| 3 | Бюджетник | % оценок выше 4 >= 50 | 0.3625 | 0.5171 | 1.1818 |
| 4 | Бюджетник | Без пересдачи | 0.236 | 0.3366 | 1.1236 |

Исходя их этих шаблонов видна похожая и предсказуемая ситуация, как и в первой серии экспериментов: контрактник чаще и с большой долей вероятности будет иметь низкую успеваемость за семестр и пересдачи, бюджетник – в точности наоборот.

Далее, приведены самые частые шаблоны, в которых антицендент является семестром, а консеквент – успеваемостью:

Таблица 11 - Шаблоны вида «Семестр -> Успеваемость» второго набора данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Антицендент | Консеквент | sup | conf | lift |
| 1 | 2 Семестр | % оценок выше 4 < 50 | 0.1108 | 0.6082 | 1.3028 |
| 2 | 1 Семестр | % оценок выше 4 >= 50 | 0.1007 | 0.5377 | 1.2289 |
| 3 | 3 Семестр | % оценок выше 4 < 50 | 0.0663 | 0.4752 | 1.0179 |
| 4 | 3 Семестр | % оценок выше 4 >= 50 | 0.0615 | 0.4408 | 1.0075 |
| 5 | 4 Семестр | % оценок выше 4 >= 50 | 0.0586 | 0.4641 | 1.0608 |
| 6 | 6 Семестр | % оценок выше 4 < 50 | 0.0554 | 0.5279 | 1.1308 |
| 7 | 5 Семестр | % оценок выше 4 < 50 | 0.0511 | 0.491 | 1.0518 |
| 8 | 8 Семестр | % оценок выше 4 >= 50 | 0.0429 | 0.5689 | 1.3002 |
| 9 | 7 Семестр | % оценок выше 4 < 50 | 0.0386 | 0.4932 | 1.0564 |

Исходя из приведенных выше шаблонов видно, что успеваемость семестра наиболее часто и вероятнее всего падает на втором семестре, также плохая успеваемость чаще и вероятнее всего следует из 5-го, 6-го и 7-го семестра, третий семестр является спорным. Из 1-го, 4-го и 8-го семестра следует хорошая успеваемость.

Вывод

Проведенные эксперименты смогли выявить несколько наглядных категорий шаблонов, которые показали, что иногородние студенты выбранных гуманитарных дисциплин чаще и с большей вероятностью имеют хорошую успеваемость, чем местные студенты, чего не наблюдается на программистских специальностях. На обеих выборках из студента-бюджетника чаще всего следует хорошая успеваемость, а из контрактника – плохая. Также была рассмотрена зависимость успеваемости от семестра, которая, показала на обеих выборках, что в первых и последних семестров периода обучения чаще и вероятнее хорошая успеваемость, в середине – плохая и спорная.

ЗАКЛЮЧЕНИЕ

В рамках работы была разработана системы поиска шаблонов успеваемости студентов на основе данных портала Универис.

В ходе работы были выполнены следующие задачи:

1. Проведен анализ и спецификация предметной области;
2. Выполнено проектирование пользовательского интерфейса и модульной структуры системы;
3. Реализована система поиска шаблонов;
4. Проведено тестирование системы поиска шаблонов;
5. Проведены эксперименты с применением разработанной системы.

Разработанная системы может использоваться для дальнейшего исследования имеющихся данных успеваемости студентов различных специальностей путем нахождения в них шаблонов.

Исходные коды компонентов разработанной в рамках работы системы свободно доступны в сети Интернет по адресу:

Список литературы

1. Nadrljanski D., Nadrljanski M., Pavlinovic M. Digitalization of Education // Learning and Analytics in Intelligent Systems book series, 2022, Vol. 29, P. 17-39. DOI: 10.1007/978-3-031-04662-9\_2
2. Agrawal R., Imielinski T., Swami A. Mining Association Rules between Sets of Items in Large Databases // ACM SIGMOD Record, 1993, Vol. 22, Issue 2, P. 207-216. DOI: 10.1145/170036.170072
3. Kumar V., Chadha A. Mining Association Rules in Student’s Assessment Data // International Journal of Computer Science Issues, 2012, Vol. 9, Issue 5, No. 3, P. 211-216.
4. Zheng J., Zhang J. The Application of Association Rules Mining in the Analysis of Students Test Scores // International Conference on Education, Management and Computer Science, 2016.
5. Nanjund G. Association Rule Mining in Educational Recommender Systems // International Journal of Advance Research in Science and Engineering, 2017, Vol. 5, Issue 8, P. 1285-1299.
6. Figma [Электронный ресурс] URL: https://www.figma.com (дата обращения 27.04.2022 г.).
7. Pandas [Электронный ресурс] URL: https://pandas.pydata.org (дата обращения 27.04.2022 г.).
8. NumPy [Электронный ресурс] URL: https://numpy.org (дата обращения 27.04.2022 г.).
9. Mlxtend [Электронный ресурс] URL: http://rasbt.github.io/mlxtend (дата обращения 27.04.2022 г.).
10. Dash [Электронный ресурс] URL: <https://dash.plotly.com/dash-core-components/upload> (дата обращения 27.04.2022 г.).
11. Asyncio [Электронный ресурс] URL: https://docs.python.org/3/library/asyncio.html (дата обращения 27.04.2022 г.).
12. Aiohttp [Электронный ресурс] URL: https://docs.aiohttp.org/en/stable/ (дата обращения 27.04.2022 г.).

Приложения

Приложение А. Основные функции приложения

Листинг 1 – Интерфейс функции make\_request

@retry(wait=wait\_fixed(1), stop=stop\_after\_attempt(3))

async def make\_request(url):

"""

Функция отправляет HTTP GET-запрос на указанный `url` и возвращает

ответ в формате JSON.

Входные данные:

- url (str): URL, на который отправляется HTTP GET-запрос.

Выходные данные:

- Объект `dict`, представляющий JSON-ответ на GET-запрос.

"""

Листинг 2 – Интерфейс функции get\_journals\_by\_year

async def get\_journals\_by\_year(years):

"""

Функция получает данные о журналах за указанные годы.

Она создает несколько задач для выполнения запросов с использованием

функции `make\_request` и объединяет результаты в один объект Pandas

DataFrame.

Входные данные:

- years (list): Список целых чисел, представляющих годы, для которых

запрашиваются данные о журналах.

Выходные данные:

- Объект Pandas DataFrame, содержащий данные о журналах.

"""

Листинг 3 – Интерфейс функции get\_and\_bind\_table\_with\_id

async def get\_and\_bind\_table\_with\_id(url, foreign\_ids, foreign\_id\_column\_name):

"""

Функция получает данные с указанного `url` для каждого `foreign\_id`.

Она создает несколько задач для выполнения запросов с использованием

функции `make\_request` и объединяет результаты в один объект Pandas

DataFrame. Кроме того, она добавляет столбец в DataFrame, содержащий

внешний идентификатор (`foreign\_id`).

Входные данные:

- url (str): URL, на который отправляются запросы.

- foreign\_ids (list): Список внешних идентификаторов.

- foreign\_id\_column\_name (str): Название столбца, в котором будет

сохранен внешний идентификатор.

Выходные данные:

- Объект Pandas DataFrame, содержащий данные из запросов.

"""

Листинг 4 – Интерфейс функции load\_params

async def load\_params(years):

"""

Эта функция сохраняет параметры журналов в формате JSON.

Она вызывает функцию get\_journals\_by\_year для получения данных о

журналах, затем фильтрует и сохраняет только необходимые параметры.

Входные данные:

- years (list): Список целых чисел, представляющих годы, для которых

запрашиваются данные о журналах.

Выходные данные:

- Объект Pandas DataFrame, содержащий параметры журналов.

"""

Листинг 5 – Интерфейс функции request\_and\_save

async def request\_and\_save(years, qualifications, direction\_names):

"""

Эта функция выполняет запросы и сохраняет данные о журналах, студентах,

оценках, рейтингах и ЕГЭ.

Входные данные:

- years (int, list): Год или список лет, для которых запрашиваются данные

о журналах.

- qualifications (str, list): Квалификация или список квалификаций, по

которым фильтруются данные о журналах.

- direction\_names (str, list): Направление или список направлений, по

которым фильтруются данные о журналах.

Выходные данные:

- Объекты Pandas DataFrame, содержащие данные о журналах, студентах,

оценках, рейтингах и ЕГЭ.

"""

Листинг 6 – Интерфейс функции convert\_to\_transactions

def convert\_to\_transactions(journals, students, ege, grades, ratings):

"""

Конвертирует 5 таблиц с данными в матрицу транзакций для поиска шаблонов.

Входные данные:

- journals (pd.DataFrame): Таблица с данными о журналах.

- students (pd.DataFrame): Таблица с данными о студентах.

- ege (pd.DataFrame): Таблица с данными об экзаменах ЕГЭ.

- grades (pd.DataFrame): Таблица с данными об оценках.

- ratings (pd.DataFrame): Таблица с данными о рейтингах.

Выходные данные:

- pd.DataFrame: Таблица транзакций для поиска шаблонов.

"""

Листинг 7 – Интерфейс функции mine\_patterns

def mine\_patterns(transactions\_matrix, min\_supp, min\_lift, min\_conf):

"""

Функция для поиска ассоциативных правил в матрице транзакций.

Входные данные:

- transactions\_matrix (DataFrame): Матрица транзакций, представленная в виде DataFrame.

- min\_supp (float): Минимальное значение поддержки для поиска частых наборов.

- min\_lift (float): Минимальное значение подъема для поиска ассоциативных правил.

- min\_conf (float): Минимальное значение достоверности для фильтрации ассоциативных правил.

Выходные данные:

- DataFrame: DataFrame с найденными ассоциативными правилами.

"""