МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени И.С. ТУРГЕНЕВА»

ФИЗИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА АЛГЕБРЫ И МАТЕМАТИЧЕСКИХ МЕТОДОВ В ЭКОНОМИКЕ

КУРСОВАЯ РАБОТА

**АНАЛИЗ ТЕКСТОВОЙ ИНФОРМАЦИИ В ФОРМИРОВАНИИ РЕКОМЕНДАЦИЙ ПО НАУЧНОМУ МЕРОПРИЯТИЮ В РАМКАХ ПРОВЕДЕНИЯ КОНФЕРЕНЦИИ**

|  |  |
| --- | --- |
|  | Выполнил:  студент 3 курса группы 71ПИ-фм  Направления подготовки  09.03.03 Прикладная информатика  Направленность (профиль)  Прикладная информатика  в экономике  Петухов Д.А.  Руководитель:  Зубкова Л. Н., к.п.н., доцент |

Оценка Подписи членов комиссии

Орел, 2020

**Содержание**

[Введение 3](#_Toc44839784)

[Глава 1 5](#_Toc44839785)

[1.1 Задачи интеллектуальной обработки текстовых данных 5](#_Toc44839786)

[1.2 Обзор технологий решения задачи 9](#_Toc44839787)

[1.3 Задача представления и обработки данных по рекомендациям на научное мероприятие 15](#_Toc44839788)

[Глава 2 18](#_Toc44839789)

[2.1 Математические результаты решения задачи 18](#_Toc44839790)

[2.2 Программная реализация 32](#_Toc44839791)

[Заключение 38](#_Toc44839792)

[Список литературы 39](#_Toc44839793)

**Введение**

В современном мире генерируется огромный объем информации, и он растет с каждым годом. Данные становятся важнейшим организационным ресурсом, обеспечивающим конкурентные преимущества, дают начало инициативам по менеджменту знаний. Ручная обработка и классификация данных становится неэффективной и дорогостоящей. Этим обуславливается потребность в автоматизации различных задач, связанных с обработкой и анализом текстовых данных.

Объектом исследования курсовой работы является процесс обработки полнотекстовых документов. Предмет исследования составляют методы анализа текстовых данных.

Целью курсовой работы является реализация системы обработки полнотекстовых документов, осуществляющая методы интеллектуального анализа данных на примере формирования рекомендаций по научному мероприятию. Достижение поставленной цели предполагает решение следующих задач:

1. выделить задачи интеллектуальной обработки текстовых данных.

2. провести обзор технологий, предназначенных для решения отдельных задач по обработке полнотекстовых документов;

3. описать задачу представления и обработки данных по рекомендациям на научное мероприятие;

4. описать программную реализацию предоставления пользователю рекомендаций на научное мероприятие;

В ходе выполнения курсовой работы применялись следующие теоретические и эмпирические методы проведения исследования: аппарат математической статистики, алгоритмы стемминга, алгоритмы классификации текстов, возможности языка Python.

Курсовая работа включает в себя: введение, первая глава, вторая глава, заключение и список использованной литературы.

Во введении обоснована актуальность темы, определены объект, предмет исследования, сформулированы цель и задачи курсовой работы, определена структура работы.

В первой главе работы рассматриваются задачи интеллектуальной обработки текстовых данных, приводится обзор технологий для решения этих задач, а также поставлена задача представления и обработки данных по рекомендациям на научное мероприятие**.**

Во второй главе представлены результаты проектирования системы анализа текста. В частности, приведены математические результаты решения задачи. Описана программная реализация предоставления рекомендаций пользователю.

В заключении сделаны основные выводы по результатам работы. Список литературы содержит 12 источников, используемых при написании работы.

# **Глава 1. Методы решения задачи анализа текстовых данных для формирования рекомендация по научному мероприятию**

* 1. **Задачи интеллектуальной обработки текстовых данных**

Интеллектуальный анализ текста (Text Mining) – это технология получения информации из неструктурированных текстовых данных путём их преобразования в пригодный для дальнейшей работы набор структурированных данных, представленных в удобном для машинной обработки виде.  То есть, посредством методов Text Mining пользователь может извлекать знания из огромных массивов данных, лишенной "понятной" компьютеру структуры [1].

Неструктурированные данные — информация, которая либо не имеет заранее определенной структуры данных, либо не организована в установленном порядке [10]. Неструктурированные данные, как правило, представлены в форме текста, который может содержать такие данные, как даты, цифры и факты. Это приводит к трудностям анализа, особенно в случае использования традиционных программ, предназначенных для работы со структурированными данными [3].

Структурированные данные — это данные, которые соответствуют модели данных, имеют четко определенную структуру, следуют последовательному порядку и могут быть легко доступны и использоваться человеком или компьютерной программой [3].

Text Mining обычно включает в себя процесс структурирования исходного текста с применением синтаксического анализа, использования некоторых лингвистических функций с последующей загрузкой в базу данных и интерпретацией результатов. Главной целью является превращение текста в структурированные данные, пригодные для анализа методами интеллектуального анализа данных. К таким методам относится [2]:

* Удаление стоп-слов. Стоп-словами называются слова, которые являются вспомогательными и несут мало информации о содержании документа. Обычно заранее составляются списки таких слов, и в процессе предвари­тельной обработки они удаляются из текста. Типичным примером таких слов являются вспомогательные слова и артикли, например: "так как", "кроме того" и т. п.;
* Стемминг — морфологический поиск. Он заключается в преобразовании каждого слова к его нормальной форме. Нормальная форма исключает склонение слова, множественную форму, особенности устной речи и т. п. Например, слова "сжатие" и "сжатый" должны быть преобразованы в нор­мальную форму слова "сжимать". Алгоритмы морфологического разбора учитывают языковые особенности и вследствие этого являются языково­зависимыми алгоритмами;
* N-граммы — это альтернатива морфологическому разбору и удалению стоп-слов. N-грамма — это часть строки, состоящая из N символов. На­пример, слово "дата" может быть представлено 3-граммой "\_да", "дат", "ата", "та\_" или 4-граммой "\_дат", "дата", "ата\_", где символ подчеркива­ния заменяет предшествующий или замыкающий слово пробел. По срав­нению со стеммингом или удалением стоп-слов, N-граммы менее чувстви­тельны к грамматическим и типографическим ошибкам. Кроме того, N-граммы не требуют лингвистического представления слов, что делает данный прием более независимым от языка. Однако N-граммы, позволяя сделать текст более строгим, не решают проблему уменьшения количества неинформативных слов;
* Приведение регистра. Этот прием заключается в преобразовании всех сим­волов к верхнему или нижнему регистру. Например, все слова "текст", "Текст", "ТЕКСТ" приводятся к нижнему регистру "текст"

Результаты анализа текста оцениваются с точки зрения некоторых критериев качества, включающих актуальность, новизну и интерес. Типичные задачи анализа текста включают [4]:

* Классификацию. Целью классификации является определение для каждого документа одной или несколь­ких заранее заданных категорий, к которым этот документ относится. Осо­бенностью задачи классификации является предположение, что множество классифицируемых документов не содержит "мусора", т. е. каждый из доку­ментов соответствует какой-нибудь заданной категории.
* Кластеризацию. Целью кластеризации документов является автоматическое выяв­ление групп семантически похожих документов среди заданного фиксиро­ванного множества. Отметим, что группы формируются только на основе по­парной схожести описаний документов, и никакие характеристики этих групп не задаются заранее.
* Извлечение концептов (сущностей). Задача извлечения концептов состоит в нахождении нескольких ссылок на один и тот же объект в тексте. Это особенно важно, поскольку соответствующее содержание может быть выражено местоимениями и обозначениями (например, «она устала»).
* Разработку таксономий. Таксономия – это метод систематизации областей знания со сложной организацией согласно иерархическому положению каждого из рассматриваемых элементов.
* Тематическое индексирование. Под термином «индексирование» первоначально понималось присвоение документам/запросам тематических индексов, отражающих некие атрибуты их классификации. С развитием направление приобрело смысл процесса своеобразного «перевода» описаний документов/запросов с естественного языка на формализованный, когда эти описания представляют собой перечни ключевых слов и словосочетаний, отражающие их тематическое содержание. Такая форма получила называние поискового образа описаний; при этом поисковый образ запроса представляет собой логическую конструкцию, где слова и словосочетания соединены при помощи логических и синтаксических операторов.
* Извлечение ключевых понятий. Первичной целью извлечения ключевых понятий является идентификация фактов и отношений в тексте. В большинстве случаев такими понятиями являются имена существительные и нарицательные: имена и фа­милии людей, названия организаций и др. Алгоритмы извлечения понятий могут использовать словари, чтобы идентифицировать некоторые термины и лингвистические шаблоны для определения других.
* Автоматическое аннотирование. Автоматическое аннотирование (позволяет сократить текст, сохраняя его смысл. Решение этой задачи обычно регулируется пользовате­лем при помощи определения количества извлекаемых предложений или процентом извлекаемого текста по отношению ко всему тексту. Результат включает в себя наиболее значимые предложения в тексте.
* обобщение документов,
* моделирование отношений между сущностями и т.д.

Основные проблемы понимания смысла текста, которые приходится преодолевать в системах Text Mining, заключаются в следующем [5]:

1) Сложность обеспечения знания системой контекста и проблемной области и обучения этому системы. Например, из предложения «мальчик вошел в дом с желтым ранцем» можно извлечь как представление о мальчике с желтым ранцем, так и о доме с желтым ранцем;

2) Различия в форме передачи синтаксиса предложения в разных языках. Например, если синтаксическая роль слова (подлежащее, сказуемое, определение и т.д.) в английской речи во многом определяется положением слова в предложении относительно других слов, то в русском предложении существует свободный порядок слов и для выявления синтаксической роли слова служат его морфологические признаки (например, окончания слов), служебные слова и знаки препинания;

3) Проблема равнозначности. Предложения «дикая кошка нежится на солнышке» и «рысь нежится на солнышке» могут означать одно и то же, но могут иметь и разный смысл.

4) Наличие в тексте новых для компьютера слов. Самообучаемая система должна уметь «интуитивно» определить (может быть, и неправильно, но с возможностью в дальнейшем исправить себя) лексическую роль, морфологическую форму нового слова, попробовать вписать его в существующую структуру знаний, наделить его какими-то атрибутами или выяснить все это в диалоге с оператором. Если же система не самообучаема, то она просто потеряет какое-то количество информации;

5) Проблема совместимости новой информации с уже накопленными знаниями. Новая информация может каким-то образом противоречить уже накопленной информации. Необходимо реализовать механизм, определяющий, в каких случаях нужно отвергнуть старую информацию, а в каких — новую;

6) Проблема временных противоречий. Так, в предложении «я надеялся, что поеду в отпуск» глагол в прошедшей форме «надеялся» сочетается с глаголом будущего времени «поеду»;

7) Проблема предложений с пропущенными фактически, но неявно существующими (благодаря контексту) словами. Например, в предложении «я передам документы тебе, а ты — на кафедру» во второй части опущен глагол «передашь» и существительное «документы». Однако понятно, что документы нужно передать на кафедру, а не в другие руки.

* 1. **Обзор технологий решения задачи**

Для решения задач анализа текстовых данных применяются инструменты языка Python.

Python – это универсальный современный язык программирования высокого уровня, к преимуществам которого относят высокую производительность программных решений и структурированный, хорошо читаемый код.  Синтаксис Python максимально облегчен, что позволяет выучить его за сравнительно короткое время. Ядро имеет очень удобную структуру, а широкий перечень встроенных библиотек позволяет применять внушительный набор полезных функций и возможностей. Python может использоваться для написания прикладных приложений, а также разработки WEB-сервисов [6].

Python может поддерживать широкий перечень стилей разработки приложений, в том числе, очень удобен для работы с ООП и функционального программирования.

Особенности программирования на Python [9]:

1. Простой язык, легкий и доступный в изучении.

У Python читаемый синтаксис. Гораздо проще читать и писать программы на Python по сравнению с другими языками, такими как: C++, Java, C#. Python делает программирование интересным и позволяет сфокусироваться на решении, а не синтаксисе.

1. Бесплатный и с открытым кодом.

Можно свободно использовать и распространять программное обеспечение, написанное на Python, даже для коммерческого использования. Вносить изменения в исходный код Python. Над Python работает большое сообщество, постоянно совершенствуя язык в каждой новой версии.

1. Портативность.

Можно перемещать и запускать программы на Python из одной платформы на другую без каких-либо изменений. Код работает практически на всех платформах, включая Windows, Mac OS X и Linux.

1. Масштабируемый и встраиваемый.

Пользователь может с легкостью комбинировать фрагменты кода на C/C++ и других языках вместе с кодом Python. Это повысит производительность приложения, а также дает возможность написания скриптов, создание которых на других языках требует больше настроек и времени.

1. Высокоуровневый, интерпретируемый язык.

В отличии от C/C++, пользователю не нужно беспокоиться о таких сложных задачах, как “сборка мусора” или управление памятью. Так же, когда идет запуск кода Python, происходит автоматическое преобразование кода в язык, который понимает компьютер. Не нужно думать об операциях более низкого уровня.

1. Стандартные библиотеки для решения общих задач.

Python укомплектован рядом стандартных библиотек, что облегчает жизнь программиста, так как нет необходимости писать весь код самостоятельно.

1. Объектно-ориентированный.

Объектно-ориентированное программирование (ООП) помогает решить сложную проблему интуитивно.

Имеется множество комбинаторных и вероятностных подходов к вычислительному анализу естественных языков.

Рассмотрим некоторые из них:

1. Векторная модель.

Сущность метрики этой модели — учёт частотных характеристик слов в документе и коллекции документов, точнее, вес некоторого слова пропорционален количеству употребления этого слова в документе, и обратно пропорционален частоте употребления слова в других документах коллекции [8].

TF — это частотность, которая измеряет, насколько часто термин встречается в документе. Логично предположить, что в длинных документах термин может встретиться в больших количествах, чем в коротких, поэтому абсолютные числа здесь не подходят. Поэтому применяют относительные — делят количество раз, когда нужный термин встретился в тексте, на общее количество слов в тексте.

IDF — это обратная частотность документов. Она измеряет непосредственно важность термина. Он считается как логарифм от общего количества документов, делённого на количество документов, в которых встречается термин х.

Таким образом, мера TF-IDF является произведением двух сомножителей

Это простой и удобный способ оценить важность термина для какого-либо документа относительно всех остальных документов. Принцип такой — если слово встречается в каком-либо документе часто, при этом встречаясь редко во всех остальных документах — это слово имеет большую значимость для того самого документа [8].

1. Наивный Байесовский классификатор.

Наивный Байесовский классификатор – вероятностный классификатор, который делает предсказания, основываясь на списке определенных классов [11]. Он основан на теореме Байеса, которая позволяет рассчитать вероятность некоторого события, на основании факта, что произошло некоторое другое событие. Формула Байеса:

где:

– вероятность, что документ «d» принадлежит классу «c»;

*–* вероятность встретить документ «d» среди всех документов класса «c»;

– безусловная вероятность встретить документ класса «с» в корпусе документов;

– безусловная вероятность документа «d» в корпусе документов.

Конечной задачей является определение к какому классу принадлежит документ, поэтому необходимо получить не саму вероятность, а наиболее вероятный класс. Для этого НБА производит оценку апостериорного максимума. Другими словами, рассчитывается вероятность для всех классов и выбирается тот, который обладает максимальной вероятностью [11].

Безусловная вероятность документа является константой, следовательно, не может повлиять на результат, поэтому знаменатель исключается. Формула принимает вид:

В натуральном языке вероятность появления слова сильно зависит от контекста. Алгоритм Байеса использует подход «bag of words model». В этой модели документ представляется в виде мультимножества его слов, игнорируя грамматику и даже порядок слов, но сохраняя множественность, т.е. документ представляется набором отдельных слов, у которых вероятность их появления не зависит друг от друга. Именно поэтому классификатор называют наивным.

Оценка вероятностей и осуществляется на обучающей выборке. Вероятность класса выражается как:

где:

– общее количество документов;

– количество документов, принадлежащих классу «с» в обучающей выборке.

Оценка вероятности слова в классе определяется как:

где:

– количество документов класса «с», где встречается слово.

Наивная Байесовская классификация выполняется без особых сложностей и с учетом оптимизации довольно быстро. Она требует меньший объем обучающих данных, чем другие подобные алгоритмы и лучше работает с категорийными признаками. Эти особенности следует учитывать при выборе классификатора для своих исследований [11].

1. Булева модель.

Булева модель стала использоваться в информационно-поисковых системах достаточно давно. Это одна из старейших моделей поиска. Основным достоинством такой модели является ее простота, способность работать с большими объемами информации и высокая скорость выполнения поисковых запросов. По этой причине на основе булевой модели построено большое количество поисковых систем [7].

В булевой модели запросы пользователей представляют собой логические выражения, в которых слова связаны операторами AND, OR и NOT. Для того чтобы документ был найден, в нем должны содержаться все слова, связанные оператором AND, или хотя бы одно из слов, связанных оператором OR. Не трудно заметить, что при сложных запросах, состоящих из нескольких слов, и большом количестве документов в поисковой базе может наблюдаться некий дисбаланс результатов поиска [7]:

* список результатов поиска при использовании оператора AND может оказаться слишком коротким, так как из результатов поиска исключаются все документы, в которых отсутствует хотя бы одно из слов запроса;
* список результатов поиска при использовании оператора OR может оказаться слишком большим, так как в результаты поиска включаются все документы, в которых встречается хотя бы одно из слов запроса.

Кроме того, у булевой модели есть существенный недостаток – в ней нет возможности установить веса термов (слов) и, соответственно, нельзя провести ранжирование результатов поиска. По сути, при поиске документы делятся на две группы – соответствующие и несоответствующие запросу. Так, при использовании оператора AND документы, не содержащие по крайней мере одного из слов запроса, являются столь же несоответствующими запросу, как и документы, не содержащие ни одного из слов запроса. Аналогично при использовании оператора OR: документы, содержащие одно из слов запроса, в равной степени соответствуют запросу, как и документы, содержащие все слова запроса.

* 1. **Задача представления и обработки данных связанных с рекомендацией на научное мероприятие**

Нередко пользователь, с желанием оказаться на научном мероприятии со своей статьей, сталкивается с проблемой незнания, на какую конференцию и на какое направление подать заявку. На таких популярных сервисах для организаций конференций, как «Ломоносов» (рис.1), «Конференции.ru» (рис.3), «ПММ Воронежского Государственного Университета» (рис.2) и других, нет возможности сразу же получить рекомендацию по конференции.

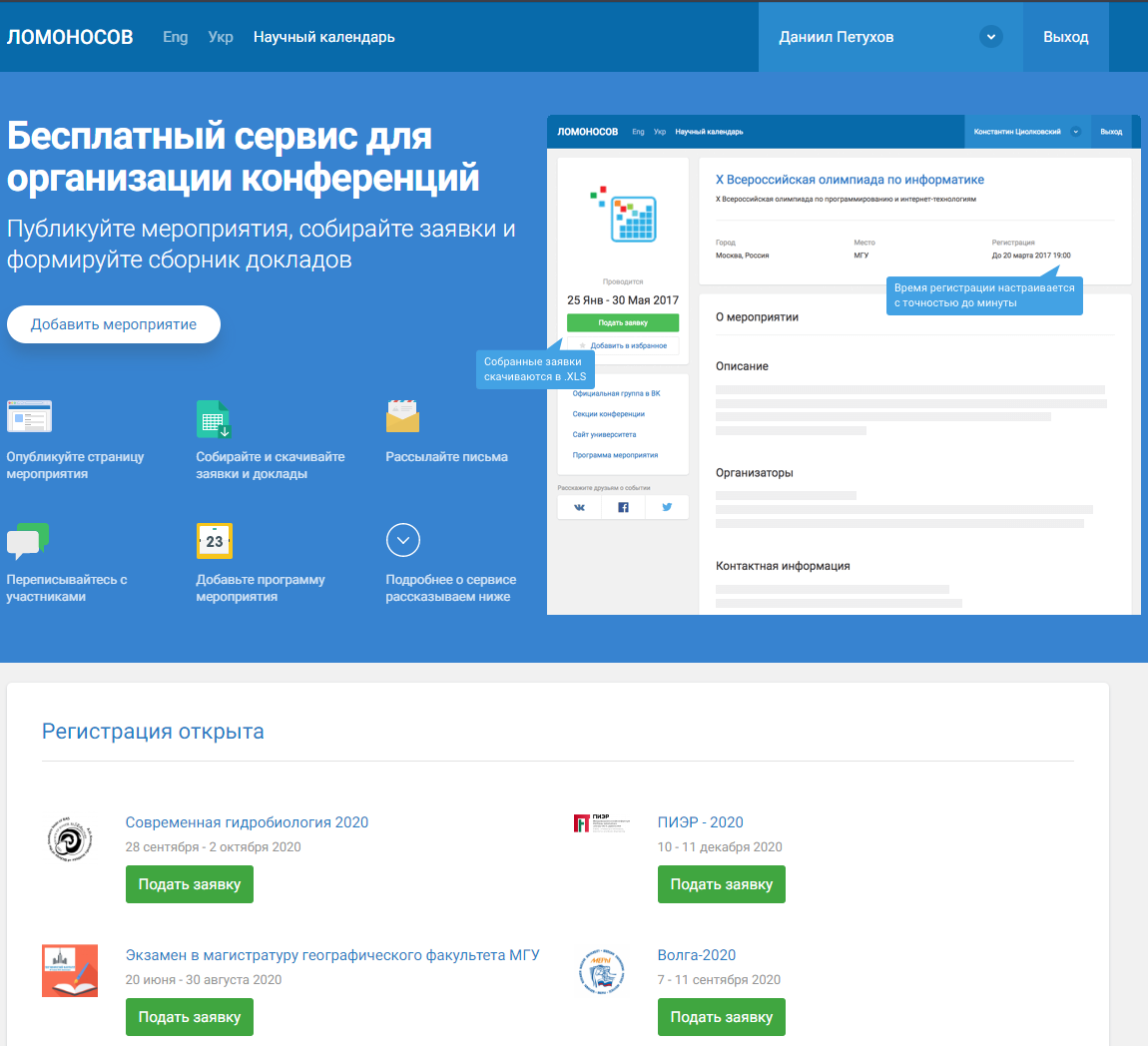


Рисунок 1 – Главная страница сайта «Ломоносов»

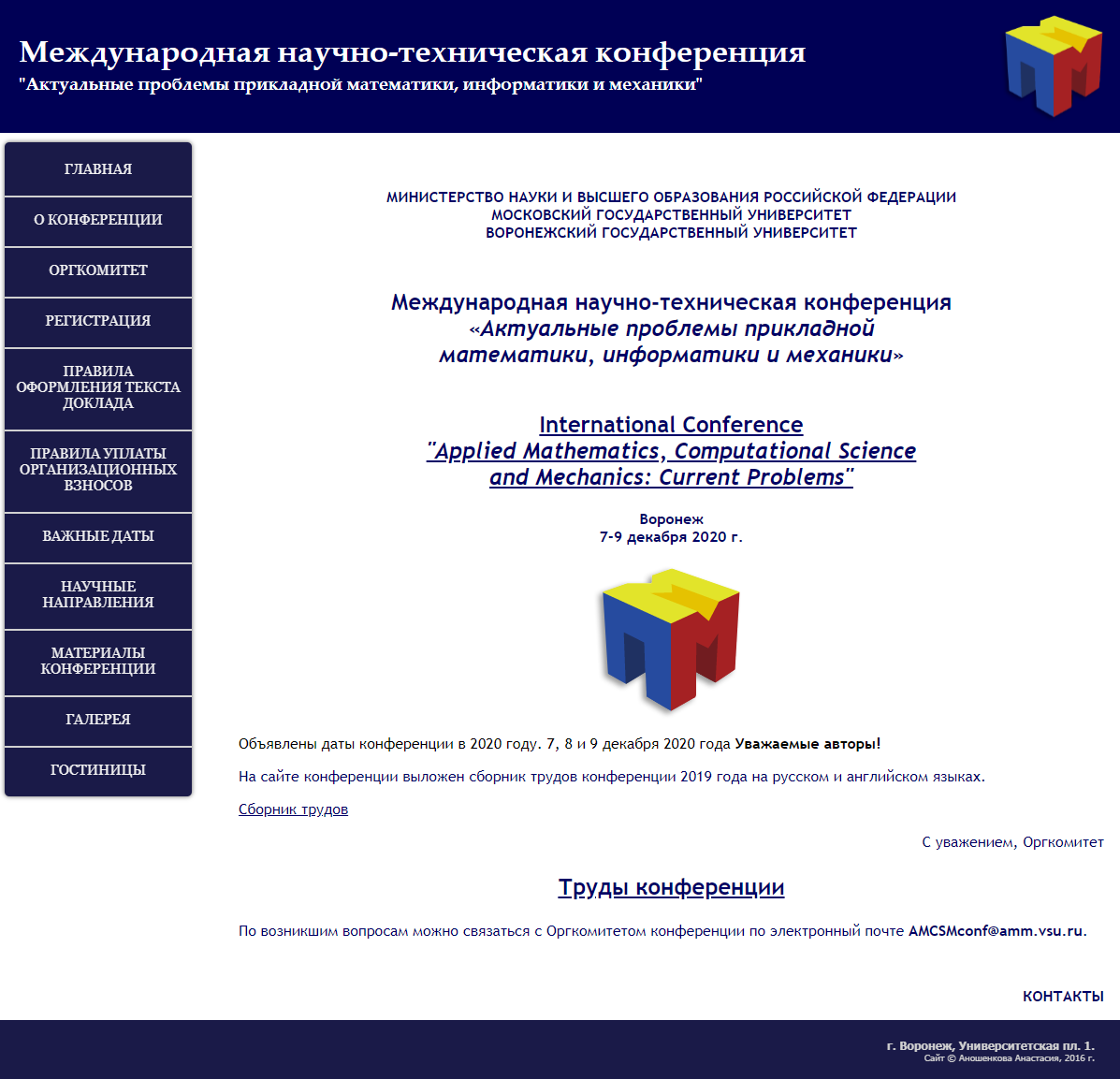


Рисунок 2 – Главная страница сайта «ПММ Воронежского Государственного Университета»

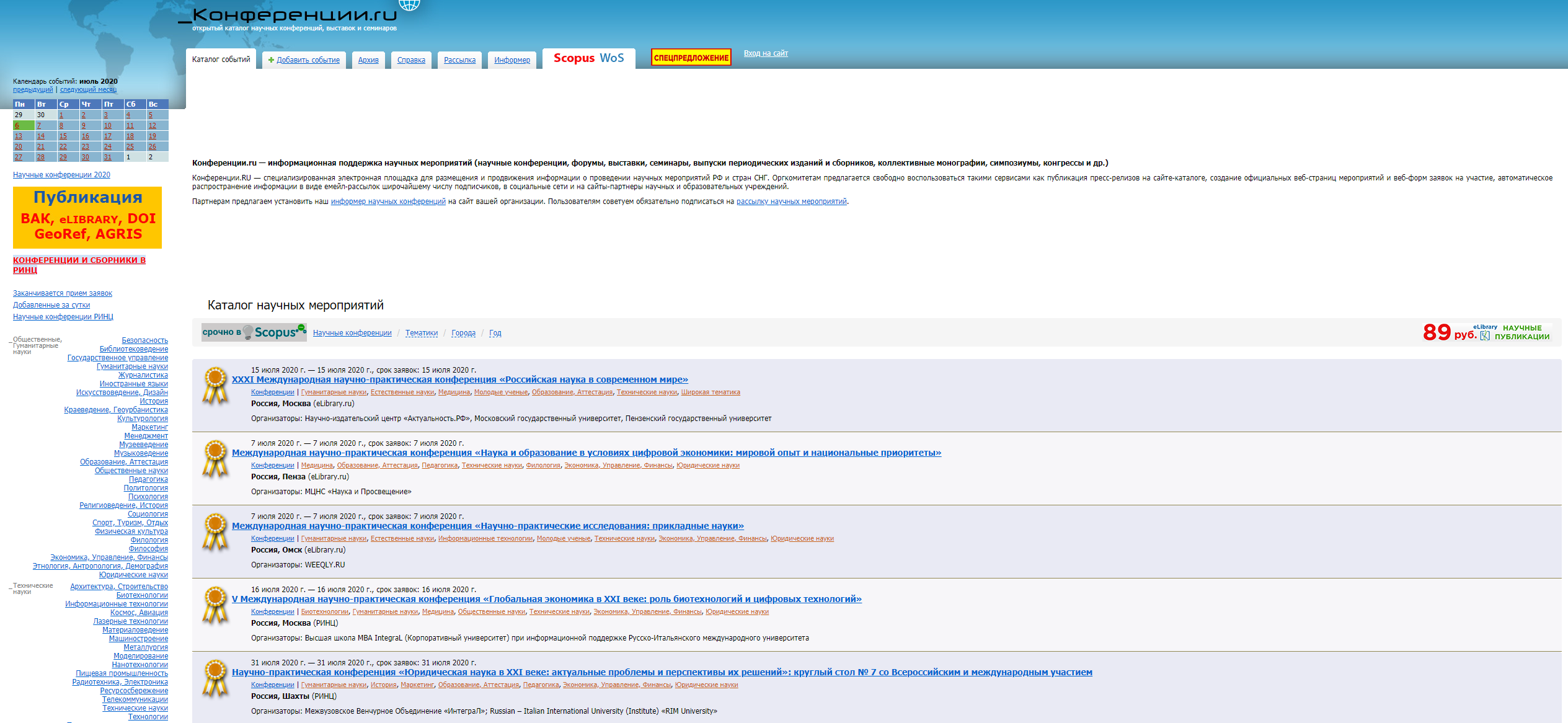


Рисунок 3 – Главная страница сайта «Конференции.ru»

Исходя из этого, была составлена задача написать свой программный продукт для создания рекомендаций на научное мероприятие заинтересованному пользователю, по ключевым словам, или анализу загруженной участником статьи. Так же дополнением к этому были составлены задачи реализации поиска пользователей и аналогичных статей по ключевым словам.

Реализовать это возможно с помощью такого языка программирования, как Python, который был рассмотрен раннее, и программы для создания графического интерфейса Python Qt Designer (рис.4).

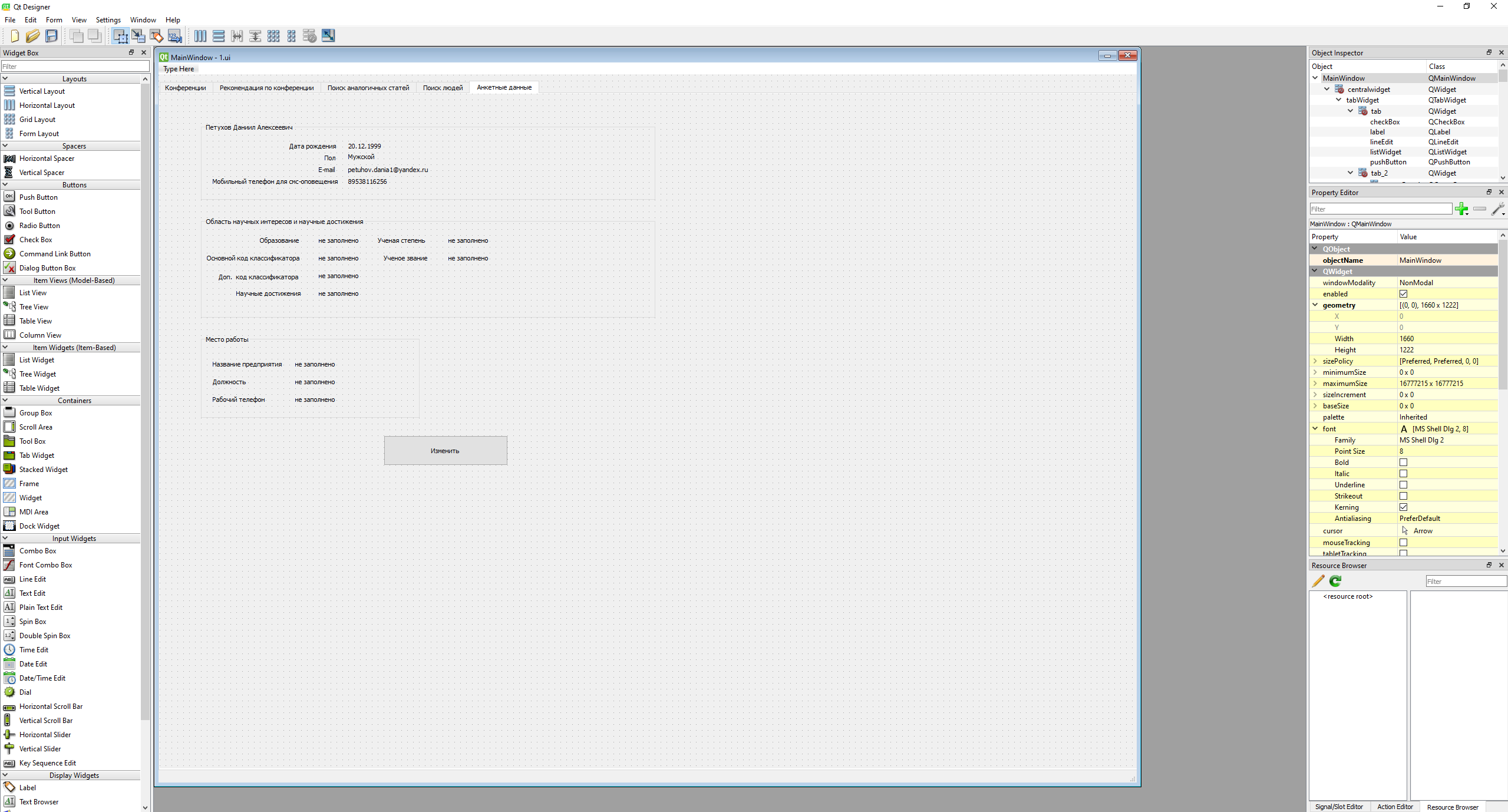


Рисунок 4 – Рабочий стол программы Qt Designer

Qt Designer – одна из частей фреймворка Qt. Это кроссплатформенный инструментарий разработчика прикладного программного обеспечения, широко используемый для создания графических интерфейсов [12].

Таким образом, для создания программного продукта потребуется две программы: QtDesigner, для создания интерфейса, и Python, для программной реализации поставленных задач.

# **Глава 2. Реализация формирования рекомендации по научному мероприятию**

## **2.1 Математические результаты решения задачи**

Для реализации каждой выделенной задачи можно выделить ряд этапов, позволяющих описать каждое действие системы. В целях лучшего понимания всех этапов была построена блок-схема.

Блок-схема – это схематичное представление процесса, системы или компьютерного алгоритма, в котором шаги изображаются в виде блоков различной формы, соединенных между собой стрелками.

В написанном программном продукте используется 3 способа анализа введенных текстовых данных:

1. С помощью векторной модели.
2. С помощью наивного Байесовского классификатора.
3. С помощью булевой модели.

Для их реализации потребовались множество инструментов языка программирования Python (рис.5).

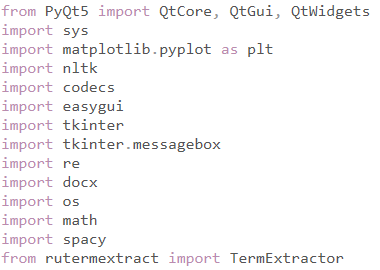


Рисунок 5 – Импорт нужных библиотек

Рассмотрим некоторые из основных инструментов.

rutermextract – позволяет извлекать ключевые слова из русского текста

docx – используется для открытия и прочитывания документов формата «.docx»

nltk – пакет библиотек и программ для символьной и статистической обработки естественного языка

Рассмотрим этапы каждого анализа введенных текстовых данных, используемых в программе.

Этапы работы алгоритма булевой модели для рекомендации научного мероприятия (рис.10):

* на первом этапе удаляются все стоп-слова из введенного пользователем текста (рис.6) и происходит стемминг (рис.7);

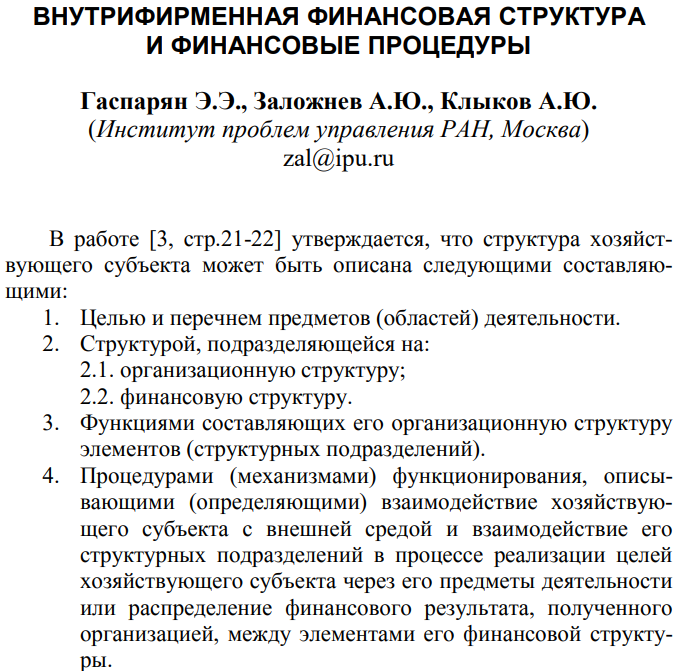


Рисунок 6 – Фрагмент текста

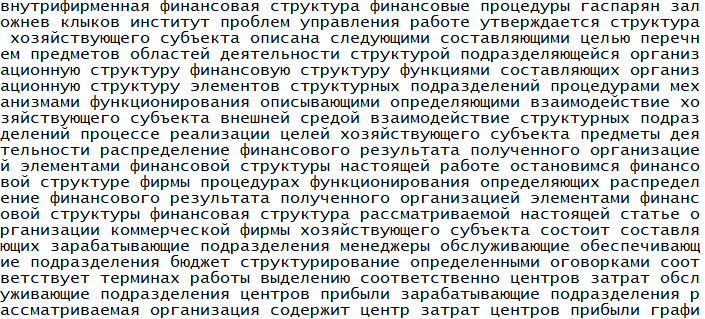


Рисунок 7 – Фрагмент текста после первого этапа

* на втором этапе извлекаются ключевые слова (рис.8);



Рисунок 8 – Ключевые слова

* на третьем этапе происходит сравнение ключевых слов, извлеченных из входящего текста с ключевыми словами конкретного направления конференции;
* на четвёртом этапе полученное количество совпадений сравнивается с заданными критериями;
* если критерий соблюден, то происходит вывод рекомендации пользователю (рис.9), иначе система переходит на следующее направление конференции и возвращается на третий этап;

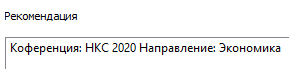


Рисунок 9 – Вывод рекомендации пользователю

* если после вывода рекомендации конференции для анализа остались, то система переходит на следующую конференцию и возвращается на третий этап, иначе алгоритм заканчивается;

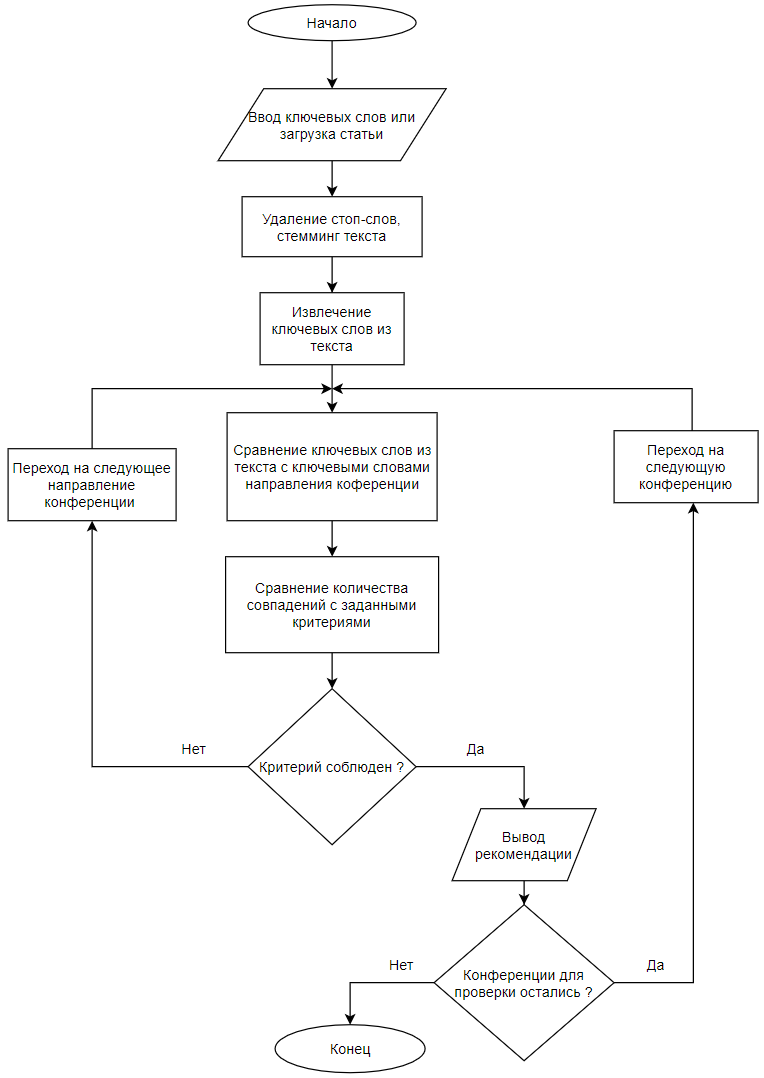


Рисунок 10 ­­­­– Блок-схема алгоритма булевой модели для рекомендации конференции

Этапы работы алгоритма наивного Байесовского классификатора для рекомендации конференции (рис.17):

* на первом этапе удаляются все стоп-слова из введенного пользователем текста (рис.11) и происходит стемминг (рис.12);

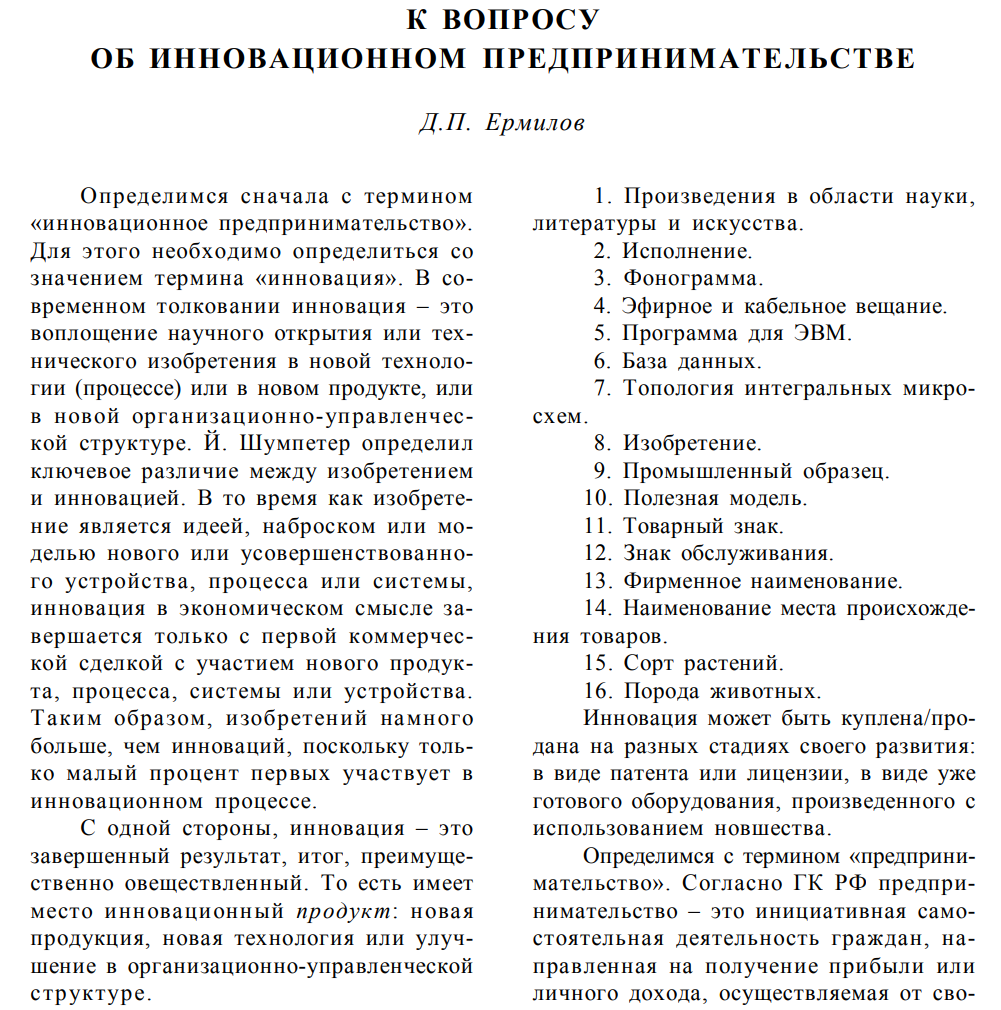


Рисунок 11 – Фрагмент текста

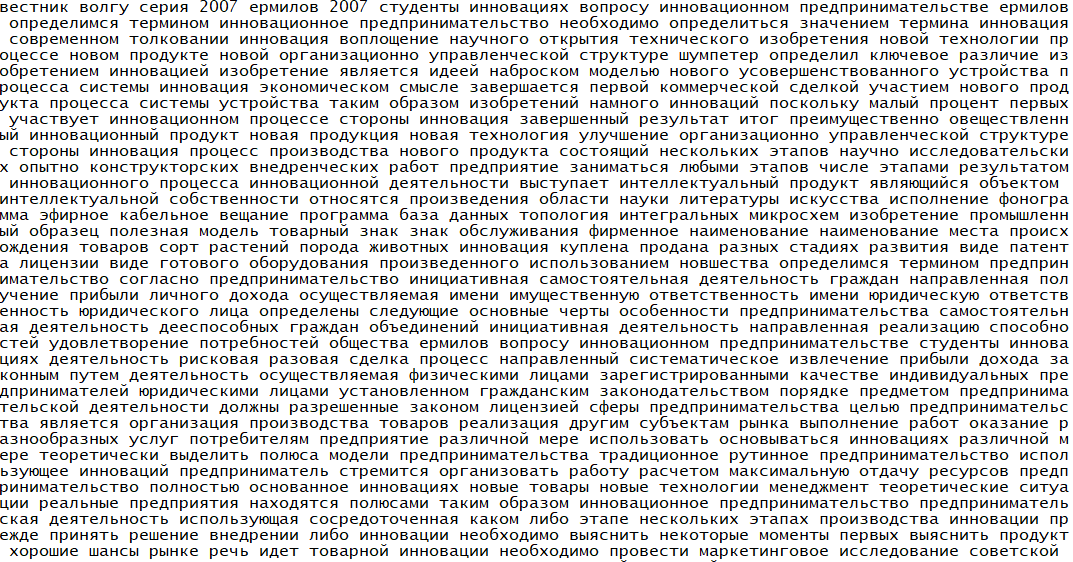


Рисунок 12 – Фрагмент текста после первого этапа

* на втором этапе извлекаются ключевые слова (рис.13);

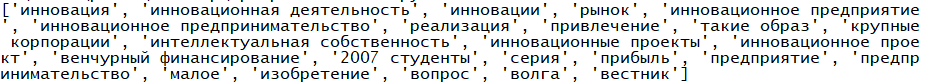


Рисунок 13 – Ключевые слова

* на третьем этапе происходит расчет вероятности p (x, Ck) (рис.14);

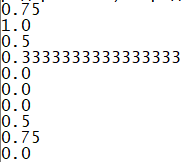


Рисунок 14 – Вероятности p (x, Ck)

* на четвёртом этапе рассчитывается вероятность p (Ck, x);
* далее, на пятом этапе результат записывается в отдельный массив (рис.15);



Рисунок 15 – Вероятности p (Ck, x)

* если после расчета вероятности p (Ck, x) конференции для анализа еще остались, то происходит переход на следующую конференцию и система возвращается на третий этап, иначе программа ищет наибольшую вероятность в массиве и, исходя из этого, выводит рекомендацию пользователю (рис.16);



Рисунок 16 – Вывод рекомендации пользователю

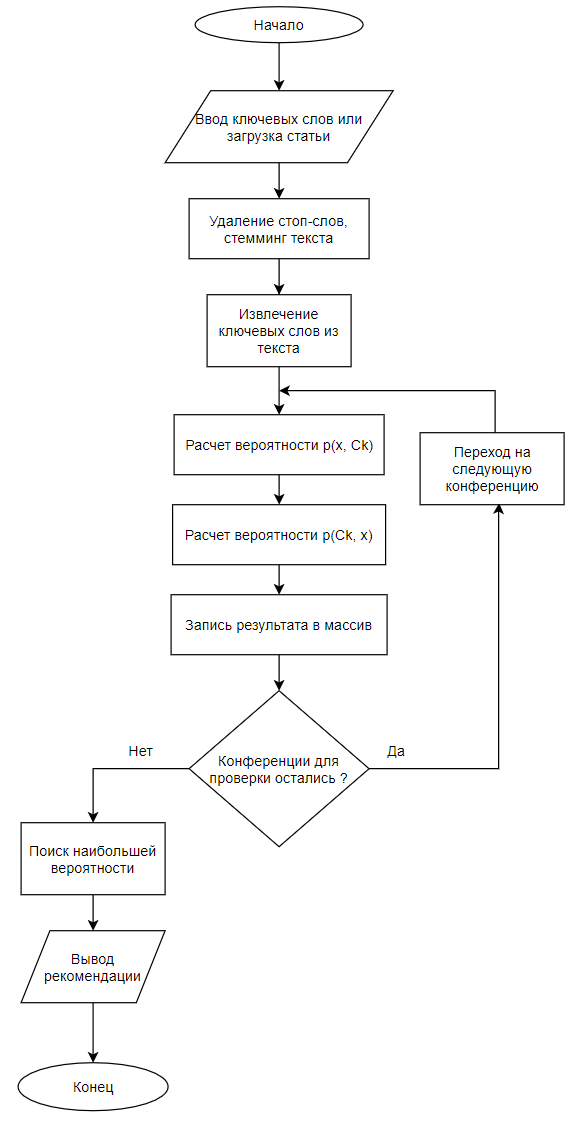


Рисунок 17 – Блок-схема алгоритма наивного Байесовского классификатора для рекомендации конференции

Этапы работы алгоритма булевой модели для рекомендации научного мероприятия (рис.23):

* на первом этапе удаляются все стоп-слова из введенного пользователем текста (рис.18) и происходит стемминг (рис.19);

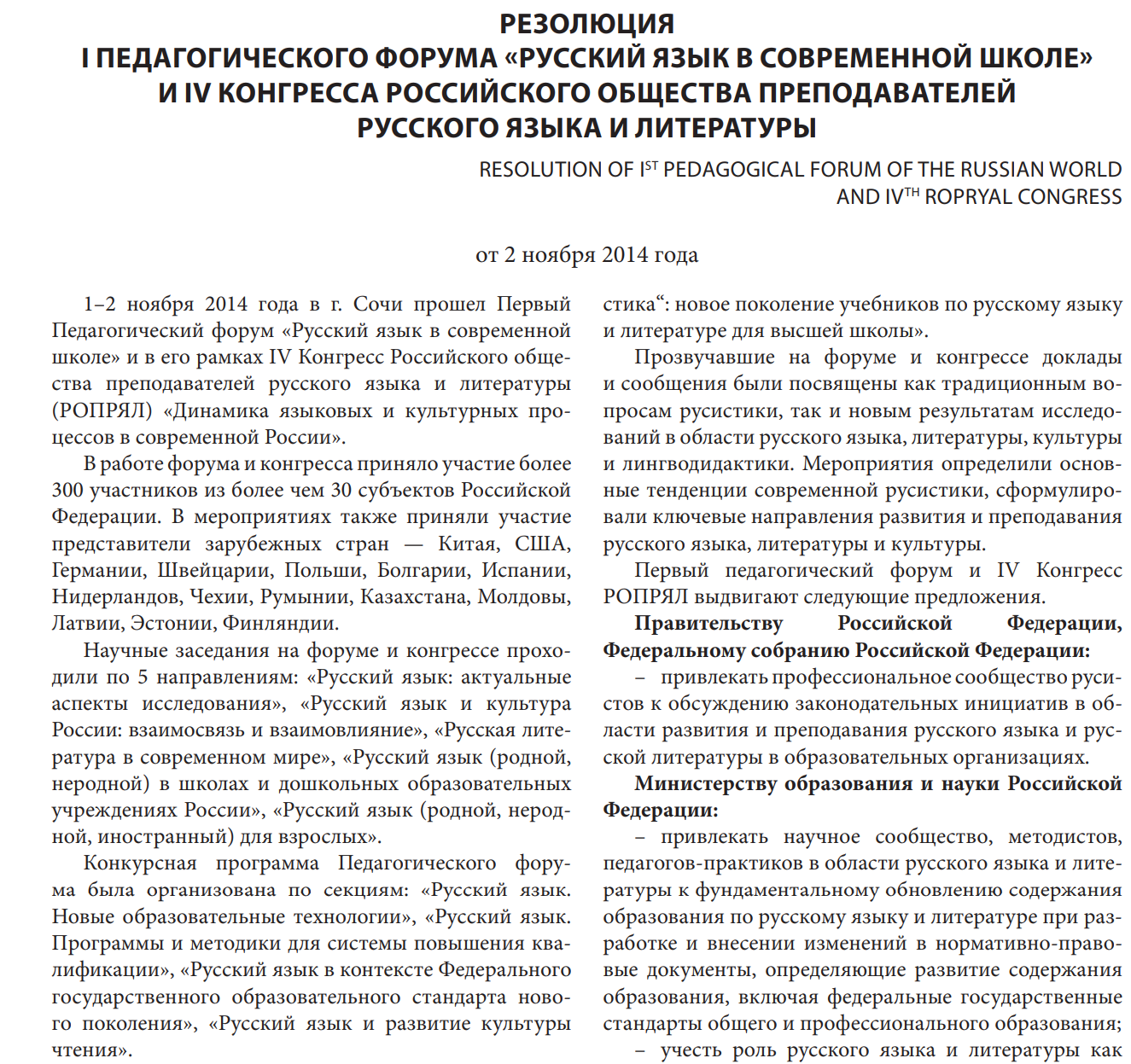


Рисунок 18 – Фрагмент текста



Рисунок 19 – Фрагмент текста после первого этапа

* на втором этапе рассчитываются TF и IDF (рис.20);

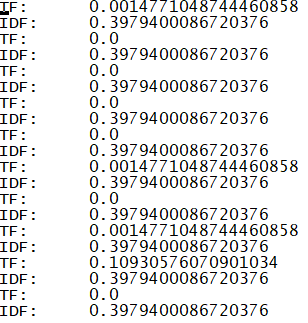


Рисунок 20 – Значения TF и IDF

* на третьем этапе происходит расчет уровня соответствия документа запросу (рис.21);

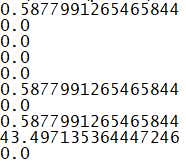


Рисунок 21 – Уровни соответствия документа запросу

* на четвёртом этапе полученный уровень соответствия документа запросу сравнивается с заданными критериями;
* если критерий соблюден, то происходит вывод рекомендации пользователю (рис.22), иначе система переходит на следующее направление конференции и возвращается на третий этап;

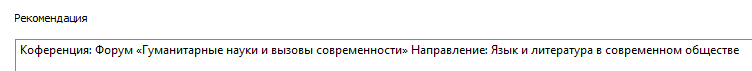


Рисунок 22 – Вывод рекомендации пользователю

* если после вывода рекомендации конференции для анализа остались, то система переходит на следующую конференцию и возвращается на второй этап, иначе алгоритм заканчивается;

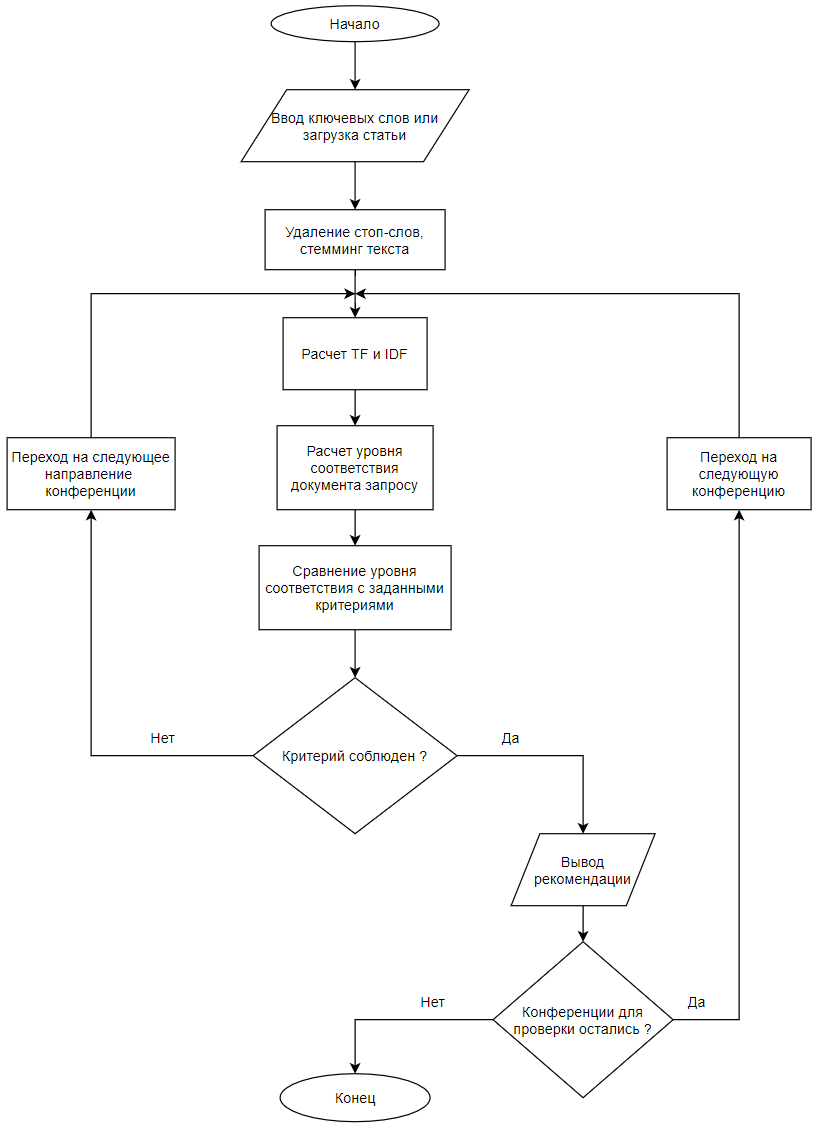


Рисунок 23 – Блок-схема алгоритма векторной модели для рекомендации конференции

Этапы работы алгоритма булевой модели для рекомендации пользователей системы (рис.27):

* на первом этапе удаляются все стоп-слова из введенного пользователем текста (рис.24) и происходит стемминг (рис.25);

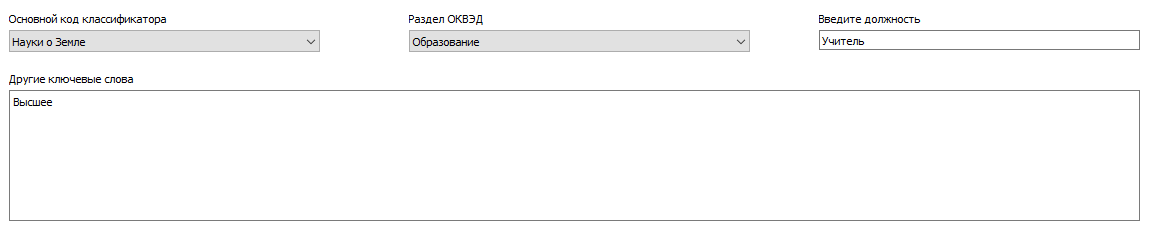


Рисунок 24 – Ввод ключевых слов



Рисунок 25 – Ключевые слова после первого этапа

* на втором этапе происходит сравнение ключевых слов, извлеченных из входящего текста с ключевыми словами конкретного пользователя;
* на третьем этапе полученное количество совпадений сравнивается с заданными критериями;
* если критерий соблюден, то происходит вывод рекомендации пользователю (рис.26), иначе система переходит к следующему пользователю и возвращается на второй этап;



Рисунок 26 – Вывод рекомендации пользователю

* если после вывода рекомендации пользователи для анализа остались, то система переходит к следующему пользователю и возвращается на второй этап, иначе алгоритм заканчивается;

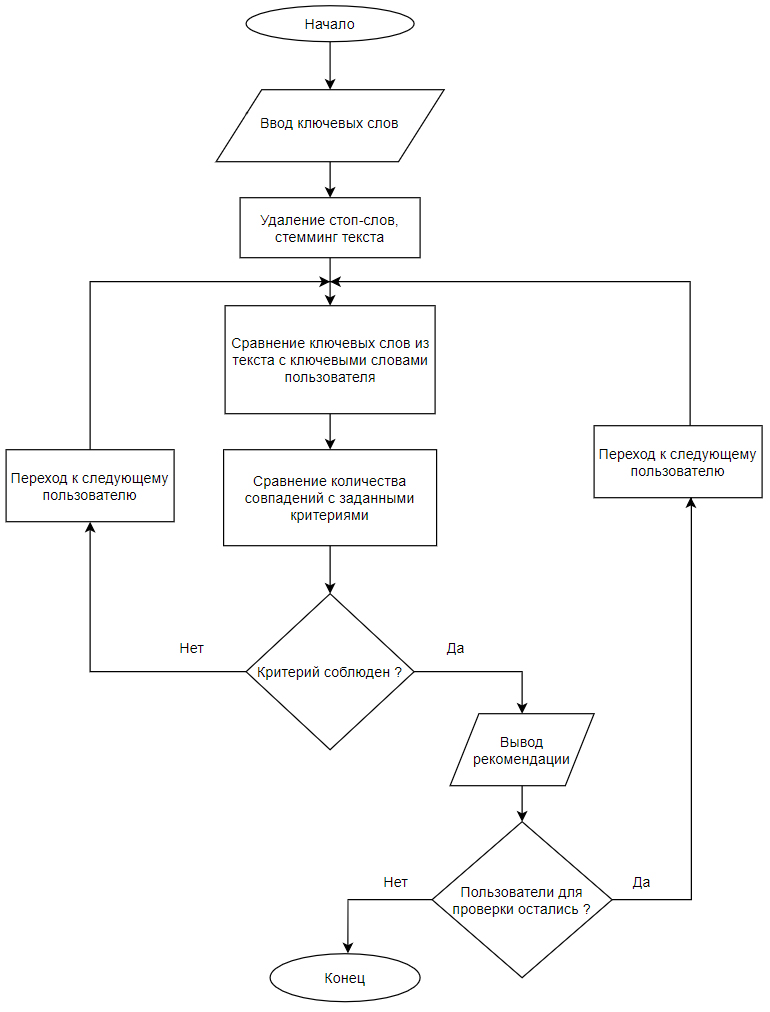


Рисунок 27 – Блок-схема алгоритма булевой модели для рекомендации пользователей

Этапы работы алгоритма булевой модели для рекомендации статей (рис.33):

* на первом этапе удаляются все стоп-слова из введенного пользователем текста (рис.28) и происходит стемминг (рис.29);

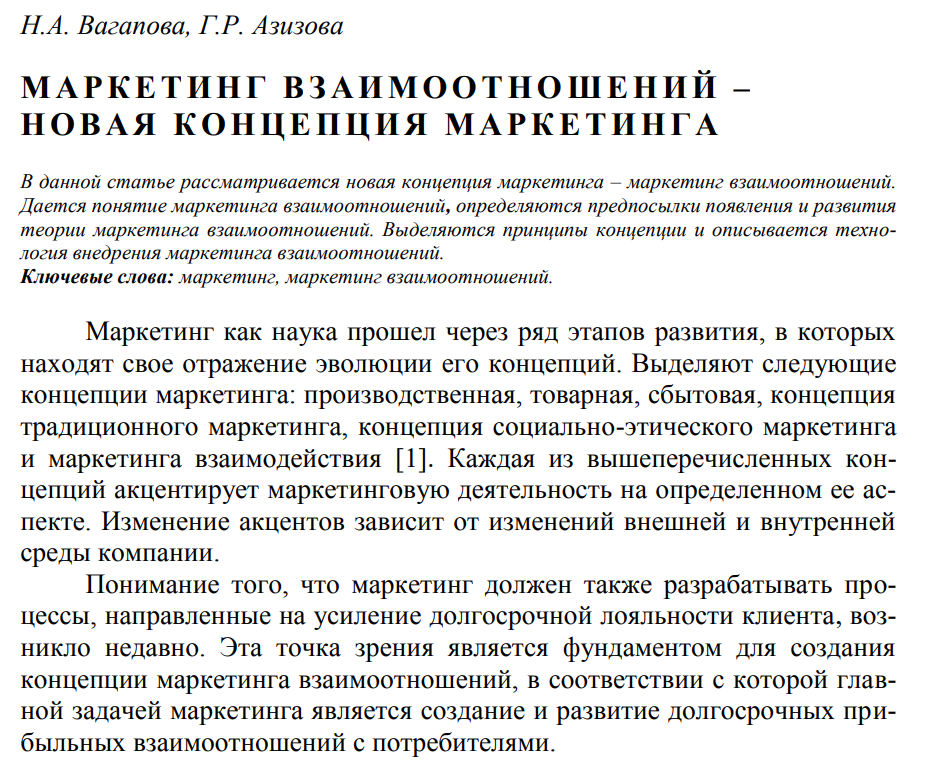


Рисунок 28 – Фрагмент текста



Рисунок 29 – Фрагмент текста после первого этапа

* на втором этапе извлекаются ключевые слова из текста (рис.30);



Рисунок 30 – ключевые слова из текста

* на третьем этапе извлекаются ключевые слова из статьи для сравнения (рис.31);

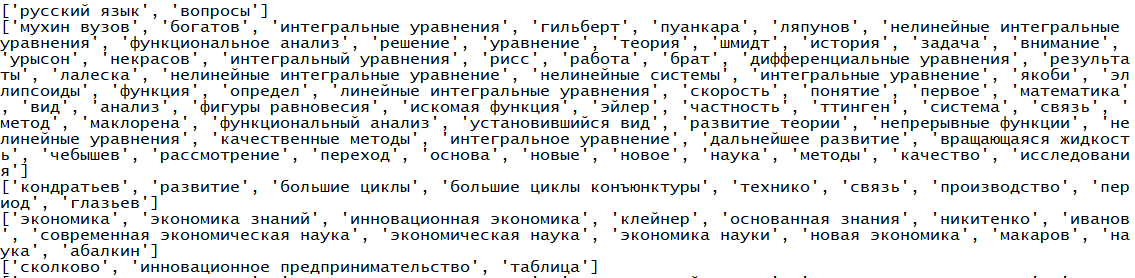


Рисунок 31 – Ключевые слова нескольких статей для анализа

* на четвёртом этапе происходит сравнение ключевых слов, извлеченных из входящего текста, с ключевыми словами статьи;
* на пятом этапе полученное количество совпадений сравнивается с заданными критериями;
* если критерий соблюден, то происходит вывод рекомендации пользователю (рис.32), иначе система переходит к следующей статье и возвращается на третий этап;

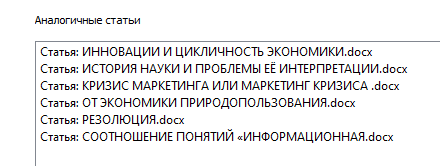


Рисунок 32 – Вывод аналогичных статей

* если после вывода рекомендации статьи для анализа остались, то система переходит к следующей статье и возвращается на третий этап, иначе алгоритм заканчивается;

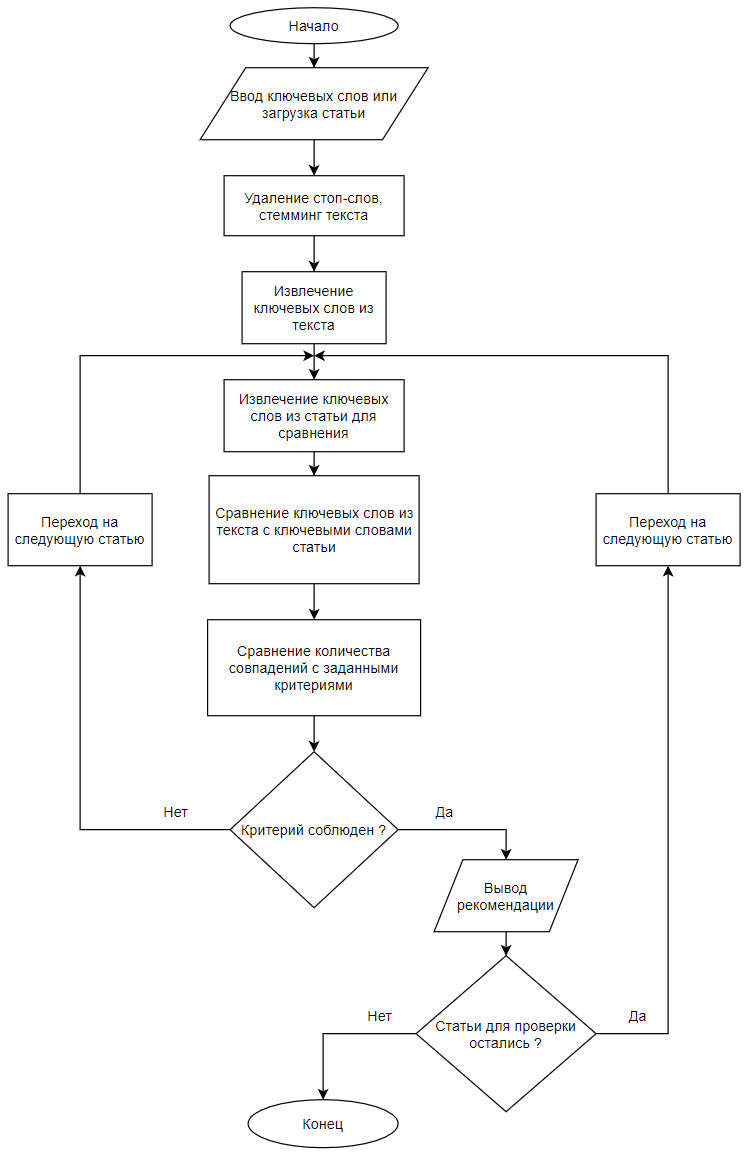


Рисунок 33 – Блок-схема алгоритма булевой модели для рекомендации статей

## **2.2 Программная реализация**

При входе в систему пользователя встречает окно авторизации (рис.34), где он вводит свой логин и пароль для входа. Если же пользователь не зарегистрирован в системе, он может это сделать, нажав на кнопку «Зарегистрироваться»

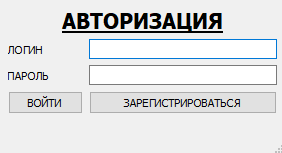


Рисунок 34 – Окно авторизации

В окне регистрации пользователь должен ввести свои данные в трех разных категориях, это «Общая информация», «Научная квалификация» и «Производственная квалификация».

Во вкладке «Общая информация» (рис.35) пользователь заполняет свой логин и пароль, ФИО, дату рождения, телефон, e-mail и почтовый индекс. После заполнения нужно нажать на кнопку «Сохранить» для сохранения данных в БД и переходить к следующей категории.

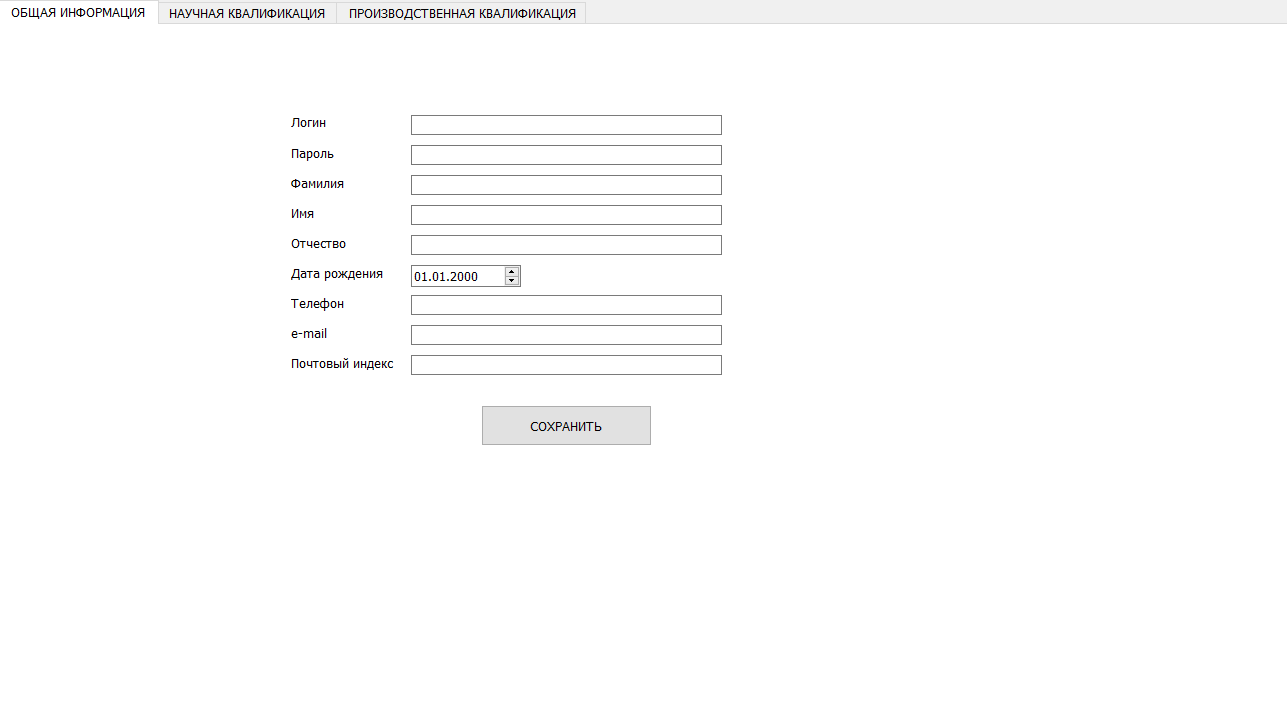


Рисунок 35 – Вкладка «Общая информация» в окне регистрации

В «Научной квалификации» (рис.36) пользователь выбирает свой уровень образования, пишет свои научные достижения, если они есть, выбирает основной код классификатора и дополнительный, если есть необходимость, а также свое ученое звание и ученую степень.

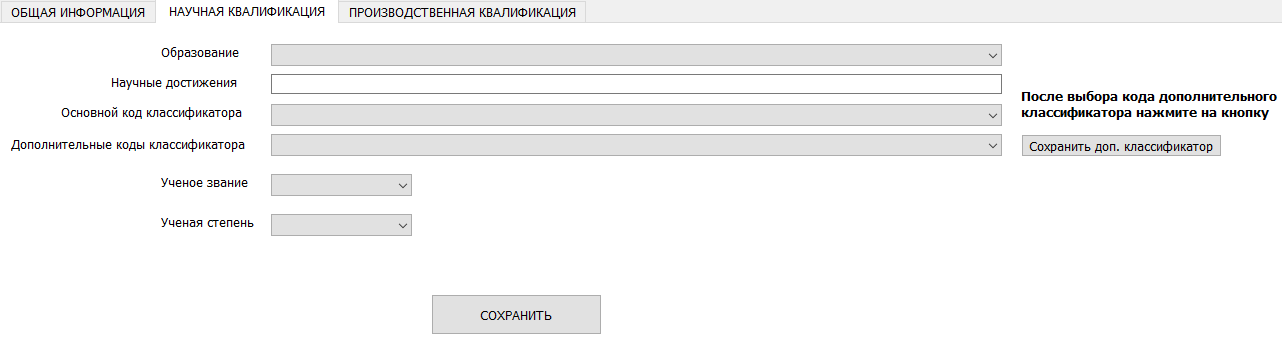


Рисунок 36 - Вкладка «Научная квалификация» в окне регистрации

Во вкладке «Производственная квалификация» (рис.37) выбирает раздел ОКВЭД (Общероссийский классификатор видов экономической деятельности), пишет свою должность и рабочий телефон организации, где он работает.

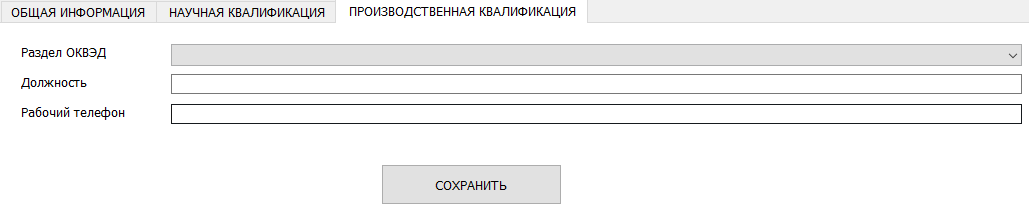


Рисунок 37 - Вкладка «Производственная квалификация» в окне регистрации

После входа в свой аккаунт пользователя встречает окно с пятью вкладками – «Конференции», «Рекомендация по конференции», «Поиск аналогичных статей», «Поиск людей» и «Анкетные данные».

Во вкладке «Конференции» (рис.38) отображаются все научные мероприятия, поле для поиска конференций и кнопка «Подать заявку»

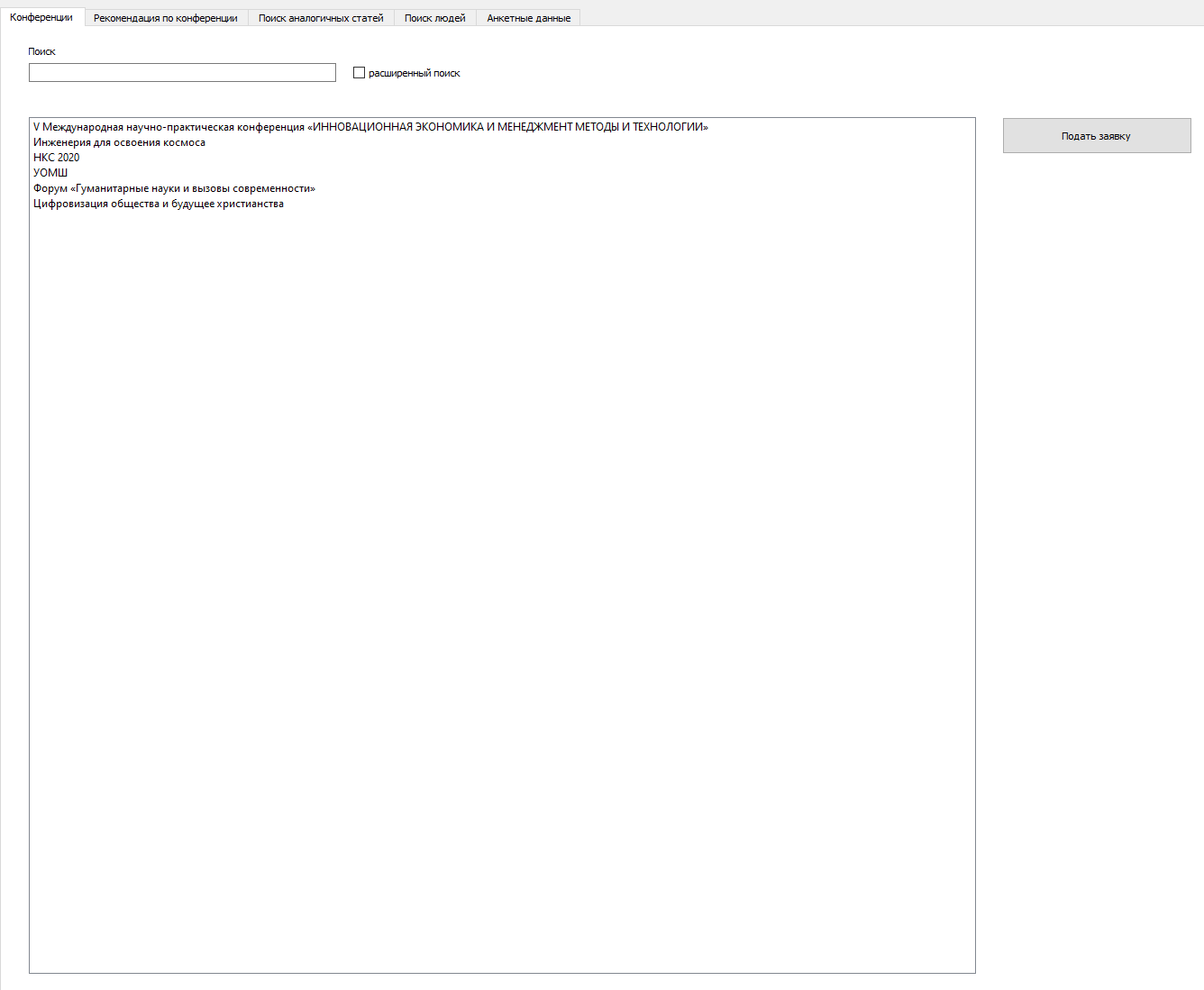


Рисунок 38 – Вкладка «Конференции» в основном окне системы

В «Рекомендации по конференции» пользователь может получить рекомендацию от системы на какое-либо научное мероприятие, путем анализа его статьи или введенных ключевых слов одним из тремя способов. Кнопкой «Открыть статью» пользователь может выбрать свой доклад, для дальнейшего анализа. Ниже находится поле, где можно написать свои ключевые слова или, если участник загрузил статью, она будет отображаться здесь.

Кнопки «1», «2» и «3» – это способы анализа введенного текста. Первый способ основан на алгоритме с векторной моделью. Второй способ основан на алгоритме с булевой моделью. Третий способ основан на алгоритме с байесовским классификатором.

В поле «Рекомендация» (рис.39) выводятся все рекомендации, сделанные системой.

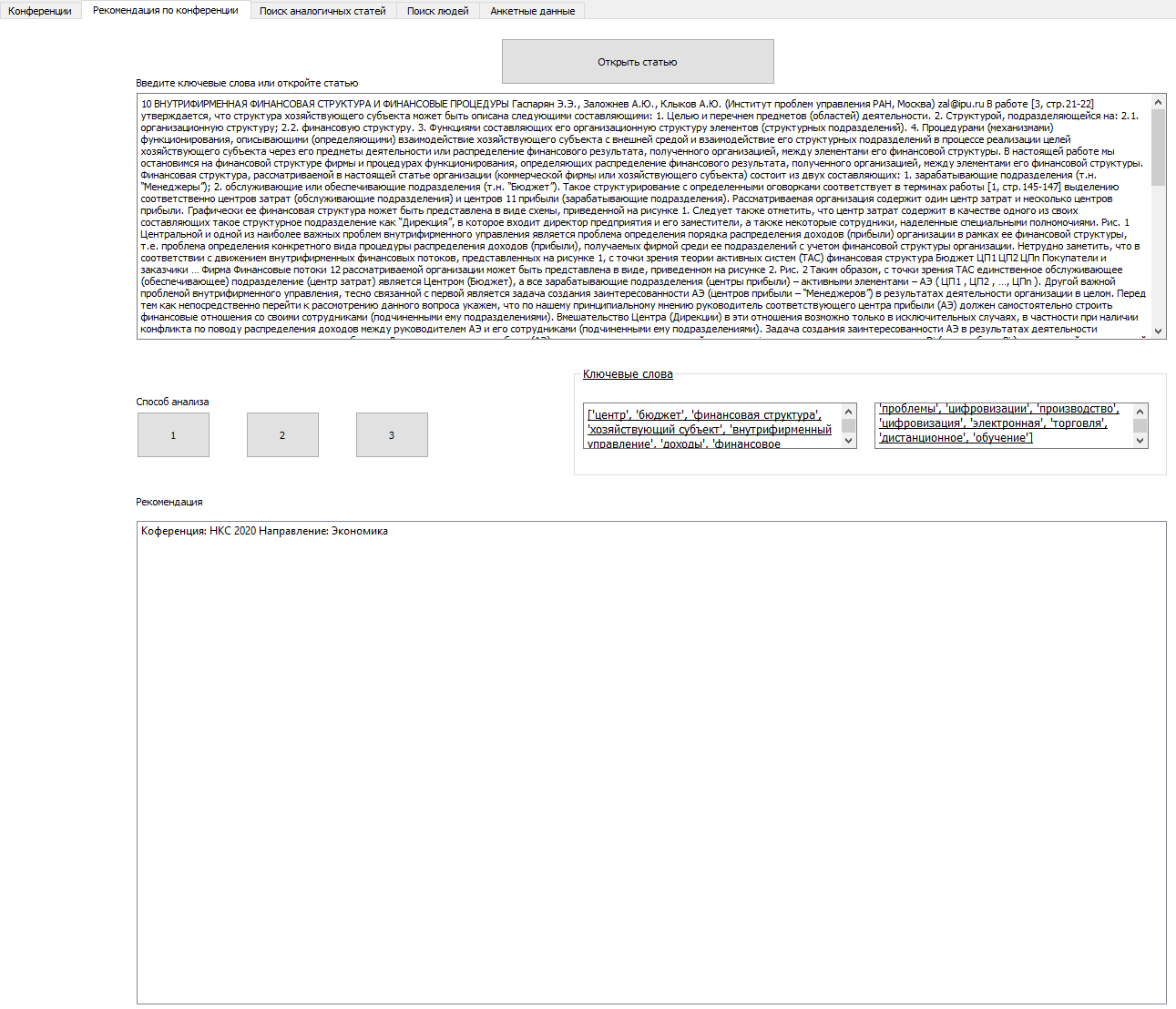


Рисунок 39 – Вкладка «Рекомендация по конференции» в основном окне системы

Во вкладке «Поиск аналогичных статей» (рис.40) пользователь может найти интересующие его статьи введя ключевые слова или найти похожую загрузив свой доклад. Здесь так же, как и в «Рекомендациях по конференции» есть кнопка «Открыть статью», с помощью которой пользователь загружает свой доклад, и поле для ввода ключевых слов или вывода текста загруженной статьи. С помощью кнопки «Поиск» система подбирает участнику нужные ему статьи и выводит их в поле ниже, где пользователь может выбрать интересующий доклад и с помощью кнопки «Скачать статью» скачать его.

Для поиска статей был выбран алгоритм с булевой моделью.

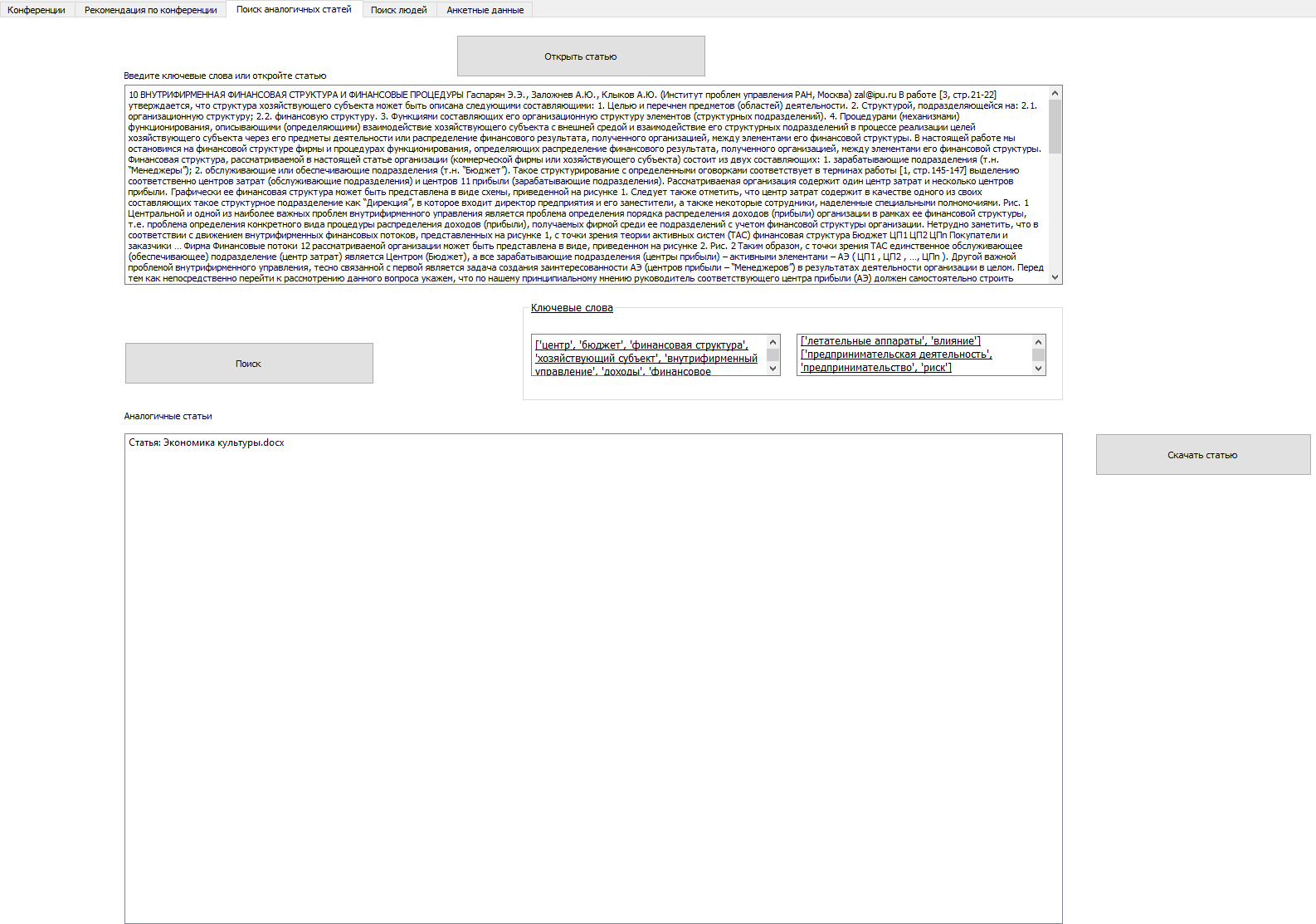


Рисунок 40 – Вкладка «Поиск аналогичных статей» в основном окне системы

В «Поиске людей» (рис.41) пользователь может найти интересующих ему людей, зарегистрированных в системе, выбрав основной код классификатора, раздел ОКВЭД, должность и при необходимости другие ключевые слова для поиска. Нажав на кнопку «Поиск», система выведет в поле «Рекомендуемые люди» всех пользователей, которых удалось найти. Выбрав одного из них можно открыть его профиль м помощью кнопки «Открыть профиль»

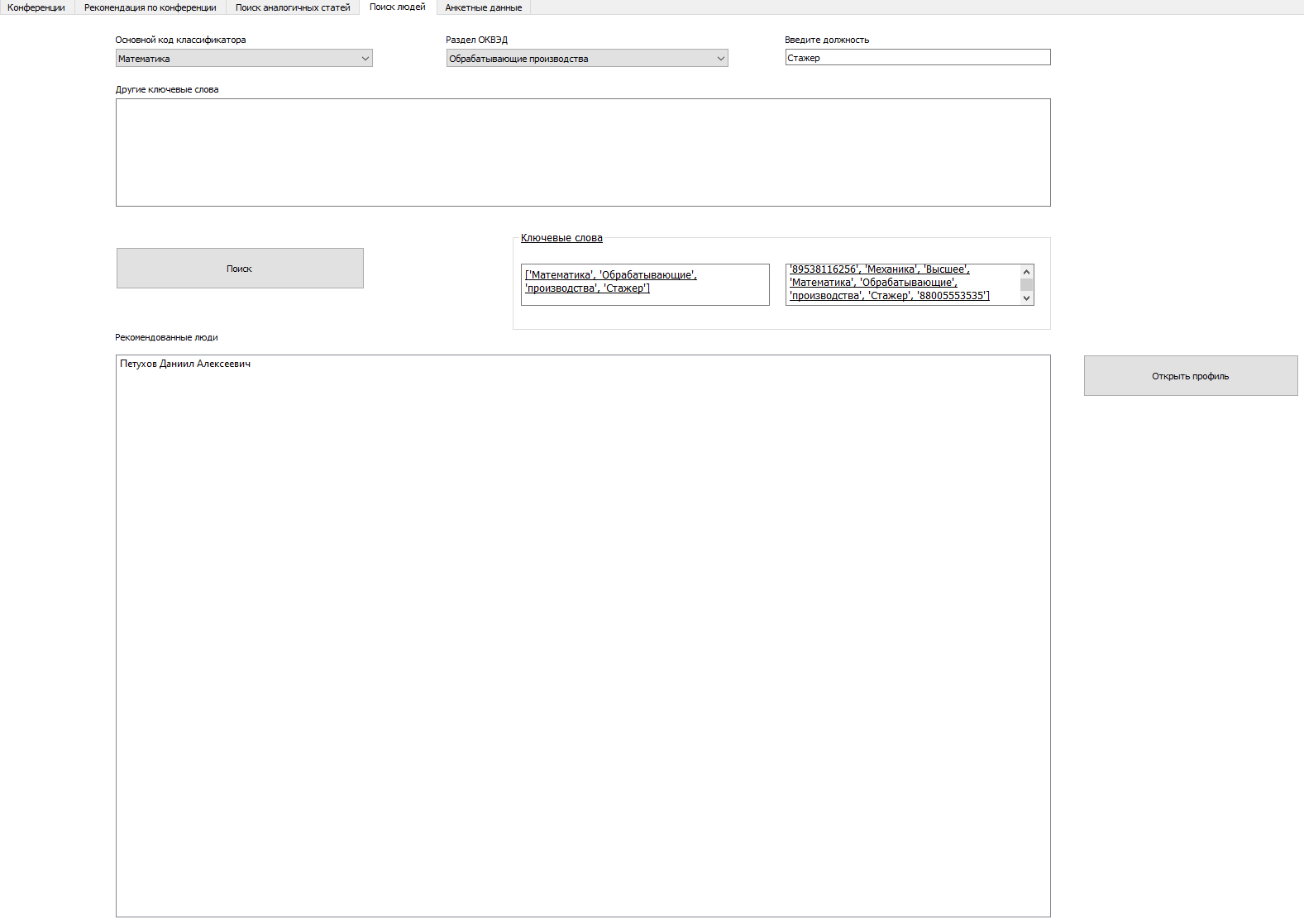


Рисунок 41 – Вкладка «Поиск людей» в основном окне системы

Во вкладке «Анкетные данные» (рис.42) пользователь может увидеть все данные, которые он заполнял при регистрации. Здесь же он может изменить их, нажав на кнопку «Изменить»

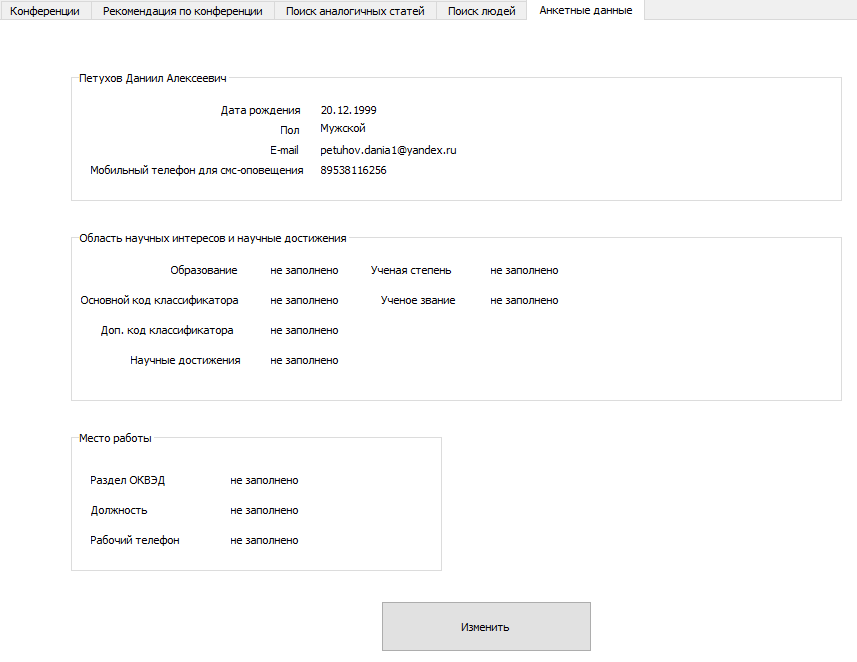


Рисунок 42 – Вкладка «Анкетные данные» в основном окне системы

# **Заключение**

Основной целью курсовой работы является реализация системы обработки полнотекстовых документов, осуществляющая методы интеллектуального анализа данных на примере формирования рекомендаций по научному мероприятию.

В ходе курсовой работы выделены задачи интеллектуальной обработки текстовых данных, проведен обзор технологий, предназначенных для решения отдельных задач по обработке полнотекстовых документов, описана задача представления и обработки данных по рекомендациям на научное мероприятие.

В работе подробно описаны этапы обработки текстовых данных для всех методов, используемых в программе, построены блок-схемы алгоритмов системы.

Описана программная реализация предоставления рекомендаций на научное мероприятие с помощью языка Python.

В результате выполнения данных задач была спроектирована система предоставления рекомендаций на научное мероприятие.

# **Список литературы**

1. Feinerer I. Text Mining Infrastructure in R. Journal of Statistical Software / I. Feinerer – N.Y.: McGraw-Hill Education, 2008. – 54 p.
2. Feldman R., The Text Mining Handbook / R. Feldman, J. Sanger – Cambridge University Press, 2006. – 423 p.
3. Kamber M. Data Mining: Concepts and Techniques / M. Kamber – N.Y.: Apress, 2006. – 800 p.
4. Weiss S.M., Fundamentals of Predictive Text Mining / S.M. Weiss, N. Indurkhya, T. Zhang – Berlin: Spinger, 2010. 231 p.
5. Барсегян А.А. Технологии анализа данных. Data Mining, Visual Mining, Text Mining, OLAP / А.А. Барсегян. – СПб.: БХВ-Петербург, 2007. 384 с.
6. Доусон М. Программируем на Python / М. Доусон. – СПб.: Питер, 2016. – 416 с.
7. Опарин Г.А., Новопашин А.П. Булево моделирование планирования действий в распределенных вычислительных системах / Г.А. Опарин, А.П. Новопашин – М.: НОУ ИНТУИТ, 2004. – 108 с
8. Распопин Н.А. Модели и методы оптимизации сбора и обработки информации / Н. А. Распопин. – М.: ДМК Пресс, 2012. – 162 с.
9. Саммерфильд М. Python на практике / М. Саммерфильд. – М.: ДМК Пресс, 2014. – 338 с.
10. Фомин В.В., Осочкин А.А. Классификация текстов на основе частотного и морфологического анализов с применением алгоритмов Data-mining / В.В. Фомин, А.А. Осочкин. – СПб.: Символ-Плюс, 2016. 158 с.
11. 6 простых шагов для освоения наивного байесовского алгоритма: [Электронный ресурс], 2015. URL: http://datareview.info/article/6-prostyih-shagov-dlya-osvoeniya-naivnogo-bayesovskogo-algoritma-s-primerom-koda-na-python/ (дата обращения: 13.06.2020)
12. Базовое приложение с QtCreator и QtDesigner [Электронный ресурс], 2011. URL: <https://riptutorial.com/ru/qt/example/29087/базовое-приложение-с-qtcreator-и-qtdesigner> (дата обращения: 13.06.2020)