МРНТИ 20.19.27 https://doi.org/10.58805/kazutb.v.3.20-153

МАТЕМАТИЧЕСКИЙ АППАРАТ ДЛЯ АНАЛИЗА НАУЧНЫХ ТЕКСТОВ: ТЕОРИЯ ВЕРОЯТНОСТЕЙ БАЙЕСА И ЕЕ РЕАЛИЗАЦИЯ

С.А. Алтынбек[1], Г.Ж. Шуйтенов[2], У.К. Турусбекова[2*], В.К. Кубекова[1]

[1]Казахский университет технологии и бизнеса, Астана, Казахстан,

[2]Esil University, Астана, Казахстан,

e-mail: umut.t@mail.ru

В настоящей статье рассматривается математический аппарат, а именно теория вероятностей Байеса, и его применение для анализа научных методов текстов. Основной целью исследования является выбор оптимальных алгоритмов для разработки будущей интеллектуальной системы параллельного анализа неструктурированных данных. Для достижения этой цели авторы обзора изучают распределенный фреймворк Apache Spark. Они проводят анализ возможностей и функциональности этого фреймворка и предлагают оптимальные алгоритмы для анализа неструктурированных данных на основе теории вероятностей Байеса. Такой подход позволяет эффективно анализировать большие объемы текстовой информации, выделять и классифицировать ее по различным параметрам. Статья также описывает преимущества использования Apache Spark для параллельного анализа данных. Фреймворк обеспечивает высокую скорость обработки и эффективное использование ресурсов, что делает его подходящим выбором для анализа больших объемов неструктурированной информации. В заключение, авторы статьи делают вывод о том, что использование математического аппарата теории вероятностей Байеса и распределенного фреймворка Apache Spark позволяет разработать интеллектуальную систему параллельного анализа неструктурированных данных, обеспечивая эффективность и точность анализа текстовой информации.

Ключевые слова: параллельный анализ**,** теория вероятностей, теория вероятностей Байерса, научный текст, большие данные, неструктурированные данные, Apache Spark, распределенные вычисления, математический аппарат.

MATHEMATICAL APPARATUS FOR THE ANALYSIS OF SCIENTIFIC TEXTS: BAYESIAN PROBABILITY THEORY AND ITS IMPLEMENTATION

S.A. Altynbek[1], G.Zh. Shuitenov[2], U.K. Turusbekova[2*], V.K. Kubekova[1]

[1]Kazakh University of Technology and Business, Astana, Kazakhstan,

[2]Esil University, Astana, Republic of Kazakhstan,

e-mail: umut.t@mail.ru

This article discusses the mathematical apparatus, namely the Bayesian probability theory, and its application for the analysis of scientific methods of texts. The main purpose of the study is to select optimal algorithms for the development of a future intelligent system for parallel analysis of unstructured data. To achieve this goal, the authors of the review are studying the Apache Spark distributed framework. They analyze the capabilities and functionality of this framework and propose optimal algorithms for analyzing unstructured data based on Bayes probability theory. This approach makes it possible to effectively analyze large amounts of textual information, isolate and classify it according to various parameters. The article also describes the advantages of using Apache Spark for parallel data analysis. The framework provides high processing speed and efficient use of resources, which makes it a suitable choice for analyzing large volumes of unstructured information. In conclusion, the authors of the article conclude that the use of the mathematical apparatus of Bayes probability theory and the Apache Spark distributed framework makes it possible to develop an intelligent system for parallel analysis of unstructured data, ensuring the efficiency and accuracy of text information analysis.

Keywords: Parallel analysis, probability theory, Bayers probability theory, scientific text, big data, unstructured data, Apache Spark, distributed computing, mathematical apparatus.

ҒЫЛЫМИ МӘТІНДЕРДІ ТАЛДАУДЫҢ МАТЕМАТИКАЛЫҚ АППАРАТЫ: БАЙЕС ЫҚТИМАЛДЫҚТАР ТЕОРИЯСЫ ЖӘНЕ ОНЫ ЖҮЗЕГЕ АСЫРУ

Ғ.Ж. Шүйтенов[1], С.А. Алтынбек[2], У.К. Турусбекова[2*], В.К. Кубекова[1]

[1]Қазақ технология және бизнес университеті, Астана, Қазақстан,

[2]Esil University, Астана, Қазақстан,

e-mail: umut.t@mail.ru

Бұл мақалада математикалық аппарат, атап айтқанда Байестің ықтималдықтар теориясы және оны мәтіндердің ғылыми әдістерін талдау үшін қолдану қарастырылады. Зерттеудің негізгі мақсаты құрылымдалмаған деректерді параллельді талдаудың болашақ интеллектуалды жүйесін әзірлеу үшін оңтайлы алгоритмдерді таңдау болып табылады. Осы мақсатқа жету үшін авторлар Apache Spark таратылған фраммаларын зерттейді. Олар осы фраммалардың мүмкіндіктері мен функционалдығын талдайды және Байес ықтималдық теориясына негізделген құрылымдалмаған деректерді талдаудың оңтайлы алгоритмдерін ұсынады. Бұл тәсіл мәтіндік ақпараттың үлкен көлемін тиімді талдауға, оны әртүрлі параметрлер бойынша бөлуге және жіктеуге мүмкіндік береді. Мақалада сонымен қатар деректерді параллельді талдау үшін Apache Spark қолданудың артықшылықтары сипатталған. Фрамма жоғары өңдеу жылдамдығын және ресурстарды тиімді пайдалануды қамтамасыз етеді, бұл құрылымдалмаған ақпараттың үлкен көлемін талдау үшін қолайлы таңдау жасайды. Мақала авторлары Байес ықтималдықтар теориясының математикалық аппаратын және Apache Spark таратылған фраммаларын пайдалану мәтіндік ақпаратты талдаудың тиімділігі мен дәлдігін қамтамасыз ете отырып, құрылымдалмаған деректерді параллель талдаудың интеллектуалды жүйесін жасауға мүмкіндік береді деген қорытындыға келеді.

Түйін сөздер: параллельді талдау,ықтималдықтар теориясы, Байерс ықтималдық теориясы, ғылыми мәтін, үлкен деректер, құрылымдалмаған деректер, Apache Spark, бөлінген есептеулер, математикалық аппарат.

Введение. Анализ научных текстов требует использования математического аппарата, который помогает в извлечении, обработке и интерпретации информации из текста. Ниже приведены некоторые математические методы и инструменты, которые часто используются в анализе научных текстов:

Методы обработки текстов:

- Мешок слов (Bag of Words): Этот метод преобразует текстовые документы в векторы, представляющие частоту встречаемости слов в документах.
- Tf-idf (Term Frequency-Inverse Document Frequency): Этот метод вычисляет важность слова в документе, учитывая его частоту встречаемости в документе и общую частоту встречаемости в корпусе документов.
- Word Embeddings: Это векторные представления слов, которые учитывают смысл и контекст слова в предложении или документе. Методы, такие как Word2Vec и GloVe, используются для создания таких представлений.
- Latent Dirichlet Allocation (LDA): Это статистическая модель, используемая для выявления скрытых тем или тематических структур в наборе документов. LDA помогает определить, какие слова обычно соседствуют в тексте, чтобы выявить основные темы, описывающие содержание документов.
- Графовый анализ: Методы графового анализа могут быть применены для анализа научных текстов, чтобы определить ключевые слова, авторов, журналы или статьи, которые являются наиболее важными или влиятельными в конкретной области исследования.
- Регрессионный анализ: Метод регрессионного анализа может быть использован для исследования взаимосвязей между различными переменными, такими как число цитирований статьи и ее содержание или характеристики автора и его вклада в научную область.

Материалы и методы. Это только некоторые из математических методов, которые могут быть применены для анализа научных текстов. Выбор конкретного метода определяется целями и вопросами исследования, а также доступностью и характеристиками данных. Рассмотрим, например, Байесовскую классификацию - это статистический метод классификации объектов (в нашем случае, текстовых документов). Метод основан на теореме Байеса, которая позволяет пересчитывать вероятность

наступления события при наличии определенных условий. В контексте анализа текста, Байесовская классификация может использоваться для определения того, какой категории принадлежит текстовый документ. Например, мы можем классифицировать электронное письмо как «спам» или «не спам» [1].

Обсуждение и результаты. Байесовский классификатор предполагает, что каждый класс имеет свой набор особенностей (features), которые могут быть использованы для их идентификации. Примерами особенностей могут быть частота встречаемости слов или наличие определенных слов в документе. Классификатор использует эти особенности, чтобы определить, к какому классу принадлежит данный документ. При обучении Байесовского классификатора, используется набор документов, которые уже отнесены к определенным классам. На основе этих данных классификатор «учится» определять, какие особенности (features) наиболее характерны для каждого класса. Затем классификатор может использоваться для классификации новых документов на основе того, какие особенности (features) в них содержатся. Байесовский классификатор является эффективным инструментом для классификации текстовых документов, и часто используется в почтовых фильтрах для обнаружения спама. Он также может использоваться для определения настроений или чувств в текстовых сообщениях или комментариях на социальных сетях [2].

Теория вероятностей Байеса может быть использована для анализа текстов и принятия решений на основе статистических данных. К примеру, можно использовать теорию Байеса для определения темы текста. Если у нас есть набор текстов разных тематик и мы хотим классифицировать новый текст на соответствие той или иной тематике, то мы можем использовать формулу Байеса. Для этого нам необходимо предварительно посчитать вероятности каждого слова для каждой темы с помощью обучающей выборки. Затем, используя формулу Байеса, мы можем рассчитать вероятность того, что данный текст относится к какой-то из тем.

Также, теория вероятностей Байеса может быть использована для анализа тональности текстов. Например, мы можем научить нашу систему различать тексты с положительной и отрицательной тональностью. Для этого нам необходимо обучить модель на наборе текстов с различными тональностями. Затем, используя формулу Байеса, мы можем определить вероятность того, что новый текст имеет положительную или отрицательную тональность. Таким образом, теория вероятностей Байеса является мощным инструментом для анализа текстов и принятия решений на основе статистических данных. Теория вероятностей Байеса является основополагающим инструментом для статистического анализа и принятия решений. Она позволяет рассчитывать вероятность наступления определенных событий на основе предварительной информации или предыдущего опыта.

Формула вероятностей Байеса может быть записана следующим образом:

$$P(A|B) = (P(B|A) * P(A)) / P(B)$$

где:

$P(A|B)$ - вероятность события A, при условии наступления события B

$P(B|A)$ - вероятность наступления события B, при условии наступления события A

$P(A)$ - вероятность наступления события A

$P(B)$ - вероятность наступления события B [3].

Формула Байеса позволяет обновлять вероятности событий на основе новой информации, таким образом, можно получить более точные оценки вероятностей.

Эта формула широко используется в различных областях, таких как медицина, финансы, машинное обучение и другие, где требуется принятие решений на основе статистической информации. Для создания математической модели на основе формулы вероятностей Байеса для анализа научных текстов Вам понадобится:

- Создать корпус текстов - коллекцию документов, которые будут использоваться для анализа.
- Отобрать существенные для анализа признаки. Это могут быть, например, ключевые слова, термины или фразы, характеризующие объект, явление или процесс, описываемые в тексте.
- Определить априорные вероятности. Априорные вероятности описывают вероятности появления определенной темы или категории в тексте.
- Вычислить условные вероятности. Условные вероятности определяют вероятность того, что текст отнесен к определенной категории при наличии определенного признака.
- Вычислить совместные вероятности. Совместная вероятность - это вероятность того, что текст содержит некоторый набор признаков одновременно.

- Использовать формулу Байеса для вычисления окончательных вероятностей.
- Конечная модель должна быть настроена на определение категории текста на основе входных данных и априорных вероятностей [4].

Теория вероятностей Байеса и Apache Spark - это две разные концепции, которые можно использовать вместе для решения различных задач анализа данных. Теория вероятностей Байеса является статистическим подходом, который позволяет обновлять вероятности на основе новой информации. Она основана на формуле Байеса, которая позволяет рассчитывать вероятности событий, учитывая априорную информацию и новые наблюдения. Apache Spark, с другой стороны, является распределенной вычислительной системой, предназначенной для обработки больших объемов данных. Она предоставляет удобные инструменты для распределенного анализа данных, машинного обучения и обработки потоков данных. Apache Spark может быть использован для реализации алгоритмов, основанных на теории вероятностей Байеса. Например, вы можете использовать Apache Spark для обработки и анализа больших текстовых наборов данных, применяя методы Байесовской классификации или фильтрации спама. Apache Spark также предоставляет библиотеки и инструменты для обработки и анализа данных, которые могут быть полезны при работе с теорией вероятностей Байеса. Например, библиотека MLlib в Apache Spark предоставляет различные алгоритмы машинного обучения, которые могут быть использованы для решения задач классификации или кластеризации на основе вероятностей [5].

Таким образом, используя Apache Spark, вы можете эффективно работать с большими объемами данных и применять методы теории вероятностей Байеса для анализа и принятия решений на основе статистических данных.

Этот фреймворк позволяет эффективно работать с большими объемами данных, производить различные операции над текстом и добиться точных результатов. Независимо от того, нужно ли вам провести анализ отзывов, собрать статистику по тексту или построить модель машинного обучения, Apache Spark - ваш идеальный выбор. Apache Spark - это высокопроизводительный фреймворк для обработки данных, который широко используется в современной аналитике данных. Этот инструмент позволяет разработчикам и аналитикам эффективно обрабатывать большие объемы данных, включая текстовые файлы. Обработка текстовых данных является одним из наиболее важных применений Apache Spark. Этот инструмент предлагает мощные средства для анализа текста, включая возможность работы с большими наборами данных.

Одним из наиболее часто используемых методов обработки текстов является токенизация. Токенизация - это процесс разделения набора текстовых данных на отдельные слова или токены. Apache Spark предоставляет удобные средства для выполнения этой операции. Например, с использованием метода split() можно разделить строку на отдельные слова, указав разделитель, как пробел или запятую. Кроме токенизации, Apache Spark также предлагает ряд других методов для обработки текста, таких как удаление стоп-слов, преобразование регистра, удаление пунктуации и многое другое. Эти методы позволяют очистить и подготовить текстовые данные перед дальнейшим анализом.

Еще одной важной возможностью Apache Spark является построение статистики по тексту. Например, можно вычислить частоту встречаемости слов в тексте, а также найти наиболее часто встречаемые слова. Для этого можно воспользоваться методом count() для подсчета количества вхождений каждого слова в тексте. Важным аспектом обработки текстовых данных является работа с большими объемами данных. Apache Spark позволяет эффективно обрабатывать такие данные, благодаря своей распределенной архитектуре и возможности параллельного выполнения операций. Это позволяет существенно ускорить процесс обработки текста и справиться с большими объемами данных. Еще одним полезным инструментом Apache Spark для обработки текста является машинное обучение. С помощью этого фреймворка можно строить модели машинного обучения, которые позволяют классифицировать и анализировать текстовые данные. Например, можно обучить модель на наборе текстовых данных и использовать ее для классификации новых текстов. Одним из примеров использования Apache Spark для обработки текстов является анализ отзывов пользователей. Например, можно провести анализ тональности отзывов, определить настроение пользователей по их текстовым комментариям. Это может быть полезно для оценки качества продукта или услуги. В заключение, Apache Spark предоставляет мощные возможности для обработки текстовых данных [6-7].

Apache Spark - это мощная распределенная вычислительная система с открытым исходным кодом, которая предоставляет высокоуровневые API для

крупномасштабной обработки данных в режиме реального времени. Python API для Apache Spark называется PySpark, и он позволяет пользователям выполнять обработку данных в распределенной среде с использованием Python [8]. PySpark позволяет пользователям интерактивно анализировать свои данные и предоставляет для этой цели оболочку PySpark [1]. Он также предлагает исчерпывающий справочник по API для всех модулей, классов, функций и методов PySpark, включая Spark SQL, Pandas API на Spark, структурированную потоковую передачу, MLlib (на основе DataFrame), MLlib (на основе RDD) и Spark Core. В дополнение к Python API, Apache Spark также предоставляет высокоуровневые API на Java, Scala и R, что делает его универсальным выбором для задач обработки данных и аналитики [9-10]. Ключевые функции Apache Spark включают пакетную и потоковую обработку данных, возможность унифицировать обработку данных в пакетном режиме и потоковую передачу в режиме реального времени с использованием нескольких языков, включая Python, SQL, Scala, Java или R, а также возможность выполнять быстрые и распределенные запросы ANSI SQL для создания информационных панелей и специальных отчетов.

Вот пример реализации наивного алгоритма Байеса для классификации текста на Python:

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split

# Load the text data and corresponding labels
data = pd.read_csv('text_data.csv')
text = data['text']
labels = data['label']

# Split the data into training and testing sets
text_train, text_test, labels_train, labels_test =
train_test_split(text, labels, test_size=0.2, random_state=42)

# Create a bag-of-words representation of the text data
vectorizer = CountVectorizer()
vectorizer.fit(text_train)
text_train_vectorized = vectorizer.transform(text_train)
text_test_vectorized = vectorizer.transform(text_test)

# Train a Naive Bayes classifier
classifier = MultinomialNB()
classifier.fit(text_train_vectorized, labels_train)

# Make predictions on the test set
predictions = classifier.predict(text_test_vectorized)

# Evaluate the accuracy of the classifier
accuracy = (predictions == labels_test).mean()
print("Accuracy: ", accuracy)
```

В этом примере мы сначала загружаем текстовые данные и соответствующие метки из CSV-файла. Затем мы разделяем данные на обучающий и тестовый наборы, используя функцию train_test_split из scikit-learn. Далее мы используем CountVectorizer для преобразования текстовых данных в представление в виде набора слов. Представление пакета слов представляет каждый документ в виде вектора частот слов. Мы подгоняем векторизатор к обучающим данным и преобразуем как обучающие, так и тестовые данные в их векторизованные формы.

Затем мы создаем экземпляр многочленного наивного байесовского классификатора из scikit-learn и обучаем его на векторизованных обучающих данных.

Наконец, мы используем обученный классификатор для составления прогнозов на основе векторизованных тестовых данных и вычисления точности классификатора путем сравнения прогнозов с истинными метками.

Вот другой алгоритм Байерса для классификации символов в тексте:

- Подготовьте обучающий набор данных, который будет содержать тексты с уже известным классом символов.
- Посчитайте вероятность встречи каждого символа в каждом классе на основе обучающего набора данных.
- Разделите каждую вероятность на общее количество символов каждого класса, чтобы получить условную вероятность встречи символа для каждого класса.
- Затем подготовьте тестовый текст для классификации.
- Разбейте текст на отдельные символы.
- Для каждого символа, воспользуйтесь условными вероятностями из обучающего набора данных, чтобы определить вероятность вхождения каждого символа в каждый класс.
- Перемножьте все вероятности для каждого символа для каждого класса, чтобы получить окончательную оценку вероятности класса для данного текста.
- Выберите класс с наибольшей оценкой вероятности и присвойте текст этому классу [11-12].

Это основной шаги алгоритма Байерса для классификации символов в тексте. Однако для более точной классификации может потребоваться дополнительная обработка данных и учет других факторов, таких как взаимодействие символов и контекст.

Конкретно для классификации символов в тексте, покажем пример реализации алгоритма Наивного Байеса в Python с использованием библиотеки Scikit-learn:

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer

# обучающий набор данных
X_train = ['абвгде', 'вгдеёжз', 'бвгеёж']
y_train = ['категория1', 'категория2', 'категория1']

# тестовый набор данных
X_test = ['ажбоб']

# создаем объект "мешок слов"
vect = CountVectorizer(analyzer='char')

# преобразуем обучающий и тестовый наборы данных в матрицы признаков
X_train_vect = vect.fit_transform(X_train)
X_test_vect = vect.transform(X_test)

# создаем объект классификатора
clf = MultinomialNB()

# обучаем классификатор на обучающем наборе данных
clf.fit(X_train_vect, y_train)

# используем классификатор для предсказания категории тестового текста
prediction = clf.predict(X_test_vect)

# выводим результаты
print(prediction)
```

В этом примере в качестве обучающего набора данных используются три строки символов, размеченные на две категории. Далее текст из тестового набора преобразуется в матрицу признаков с помощью объекта "мешок слов". Затем создается и обучается объект классификатора Наивного Байеса. Наконец, классификатор используется для предсказания категории тестового текста, и результаты выводятся на экран. Обратите внимание, что для этого примера текст преобразуется не посимвольно, а по словам (поэтому используется CountVectorizer с параметром analyzer='char'). Если вы хотите работать с отдельными символами, можно использовать CountVectorizer без параметра analyzer, что позволит преобразовывать текст посимвольно.

Выводы. В статье рассмотрена теория вероятностей Байеса и ее применение в анализе научных методов текстов. Авторы обзора провели исследование и изучили распределенный фреймворк Apache Spark для достижения данной цели. Теория вероятностей Байеса является одной из основных теорий в статистике, которая позволяет определить вероятность наступления события на основе имеющихся данных. Применение этой теории в анализе научных методов текстов позволяет получить более точные результаты и улучшить качество исследования. Изучение распределенного фреймворка Apache Spark в данном контексте имеет большое значение. Этот фреймворк предоставляет возможность эффективной обработки больших объемов данных и распределенных вычислений, что является необходимым для анализа научных методов текстов. Результаты исследования показали, что использование теории вероятностей Байеса и распределенного фреймворка Apache Spark позволяет значительно улучшить анализ научных методов текстов. Это может быть полезно не только для исследователей и ученых, но и для разработчиков и промышленных предприятий, которые заинтересованы в извлечении знаний из текстовых данных. Теория вероятностей Байеса позволяет учитывать предыдущую информацию и делать статистические выводы на основе вероятностей. Это особенно полезно в анализе научных методов, где часто требуется учитывать различные факторы и уровни неопределенности. Применение распределенного фреймворка Apache Spark позволяет эффективно обрабатывать большие объемы данных. Это особенно важно в анализе научных методов текстов и данных, где существует большое количество информации, которую необходимо обработать. Apache Spark позволяет распараллеливать анализ и расчеты на кластере, что ускоряет процесс и позволяет работать с большими объемами данных. Статья также рассмотрела примеры применения теории вероятностей Байеса и Apache Spark в анализе научных методов текстов и данных. Например, авторы рассмотрели использование Байесовского анализа для предсказания успешности дизайнов, а также использование Apache Spark для анализа и классификации большого объема научных статей. В целом, статья подчеркнула значимость и преимущества применения теории вероятностей Байеса и распределенного фреймворка Apache Spark в анализе научных методов текстов и данных. Эти методы позволяют учитывать неопределенность, работать с большими объемами данных и эффективно проводить анализ и классификацию.

В целом, статья показала значимость и преимущества применения теории вероятностей Байеса и распределенного фреймворка Apache Spark в анализе научных методов текстов. Дальнейшие исследования в этой области могут привести к еще более точным и эффективным методам анализа текстовых данных.

Научно-исследовательская работа выполняется в рамках ГФ Министерством науки и высшего образования Республики Казахстан AR19677733 по теме «Разработка интеллектуальной распределенной системы параллельного анализа научных текстов» на 2023-2025 гг.

Литература

1. Колмогоров А.Н. Теория вероятностей и математическая статистика. - М.: Наука, 1986. - 535 с.

2. Barber D. Bayesian Reasoning and Machine Learning.- http://web4.cs.ucl.ac.uk

3. Ветров Д.П., Кропотов Д.А. Байесовские методы машинного обучения: Пособие.- 2017.- - 67 с.

4. Conrady S., Jouffe L. Introduction to Bayesian Networks & BayesiaLab. -

https://library.bayesia.com/download/attachments/10092794/Bayesian_Networks_Intro_v16.pdf

5. Abdymanapov, S., Muratbekov, M., Altynbek, S., Barlybayev, A. Fuzzy expert system of information security risk assessment on the example of analysis Learning Management Systems. // IEEE Access. - 2021. -99(1-1)- pp. 156556-156565. DOI: 10.1109/ACCESS.2021.3129488.

6. Boranbayev, A., Shuitenov, G., Boranbayev, S. The Method of Analysis of Data from Social Networks Using Rapidminer //Advances in Intelligent Systems and Computing. - 2020. - 1229 AISC.- pp. 667-673.

7. Boranbayev, A., Shuitenov, G., Boranbayev, S. The method of data analysis from social networks using apache Hadoop // Advances in Intelligent Systems and Computing. - 2018. - 558.- pp. 281-288.

8. Пол Дейтел, Харви Дейтел Python: Искусственный интеллект, большие данные и облачные вычисления. - Издательство: ЛитРес, 2020. - 864 с.- ISBN 978-5-4461-1432-0

9. Altynbek, S., Begehr, H. A pair of rational double sequences. // Georgian Mathematical Journal. - 2022, 29(2).- pp. 163-166. https://doi.org/10.1515/gmj-2021-2119.

10. A. Barlybayev; Z. Kaderkeyeva; G. Bekmanova; A. Sharipbay; A. Omarbekova; S. Altynbek. Intelligent System for Evaluating the Level of Formation of Professional Competencies of Students.//IEEE Access. -2020.- 8.- pp.58829-58835, 9027836. DOI: 10.1109/ACCESS.2020.2979277

11. https://spark.apache.org/ Дата обращения -17.07.2023

12. https://wiki5.ru/wiki/Letter_frequency/ Дата обращения-17.07.2023

References

1. Колмогоров А.Н. Теория вероятностей и математическая статистика. - М.: Наука, 1986. - 535 с.

2. Barber D. Bayesian Reasoning and Machine Learning.- http://web4.cs.ucl.ac.uk

3. Ветров Д.П., Кропотов Д.А. Байесовские методы машинного обучения: Пособие.- 2017.- - 67 с.

4. Conrady S., Jouffe L. Introduction to Bayesian Networks & BayesiaLab. -

https://library.bayesia.com/download/attachments/10092794/Bayesian_Networks_Intro_v16.pdf

5. Abdymanapov, S., Muratbekov, M., Altynbek, S., Barlybayev, A. Fuzzy expert system of information security risk assessment on the example of analysis Learning Management Systems. // IEEE Access. - 2021. -99(1-1)- pp. 156556-156565. DOI: 10.1109/ACCESS.2021.3129488.

6. Boranbayev, A., Shuitenov, G., Boranbayev, S. The Method of Analysis of Data from Social Networks Using Rapidminer //Advances in Intelligent Systems and Computing. - 2020. - 1229 AISC.- pp. 667-673.

7. Boranbayev, A., Shuitenov, G., Boranbayev, S. The method of data analysis from social networks using apache Hadoop // Advances in Intelligent Systems and Computing. - 2018. - 558.- pp. 281-288.

8. Пол Дейтел, Харви Дейтел Python: Искусственный интеллект, большие данные и облачные вычисления. - Издательство: ЛитРес, 2020. - 864 с.- ISBN 978-5-4461-1432-0

9. Altynbek, S., Begehr, H. A pair of rational double sequences. // Georgian Mathematical Journal. - 2022, 29(2).- pp. 163-166. https://doi.org/10.1515/gmj-2021-2119.

10. A. Barlybayev; Z. Kaderkeyeva; G. Bekmanova; A. Sharipbay; A. Omarbekova; S. Altynbek. Intelligent System for Evaluating the Level of Formation of Professional Competencies of Students.//IEEE Access. -2020.- 8.- pp.58829-58835, 9027836. DOI: 10.1109/ACCESS.2020.2979277

11. https://spark.apache.org/ Дата обращения -17.07.2023

12. https://wiki5.ru/wiki/Letter_frequency/ Дата обращения-17.07.2023

Сведения об авторах

Алтынбек С.А. - PhD, проректор по науке, Казахский университет технологии и бизнеса, Астана, Казахстан, e-mail: serik_aa@bk.ru;

Шуйтенов Габит Жумабаевич - проректор по цифровизации, Esil University, Астана, Казахстан, г., e-mail: g.shuitenov@mail.ru;

Турусбекова У.К. - PhD, и.о. доцента, Esil University, Астана, Казахстан, e-mail: umut.t@mail.ru;

Кубекова В.К. - магистр, старший преподаватель, Казахский университет технологии и бизнеса, Астана, Казахстан, e-mail: kubekova.venera@list.ru.

Information about the authors

Altynbek S.A. - PhD, Vice-Rector for Science, Kazakh University of Technology and Business, Astana, Kazakhstan, e-mail: serik_aa@bk.ru;

Shuitenov G.Zh. - Vice-Rector for Digitalization, Esil University, Astana, Kazakhstan, e-mail: g.shuitenov@mail.ru;

Turusbekova U.K. - PhD, Acting Associate Professor, Esil University, Astana,Kazakhstan, e-mail: umut.t@mail.ru;

Kubekova V. K. - Master's degree, senior lecturer, Kazakh University of Technology and Business, Astana, Kazakhstan, e-mail: kubekova.venera@list.ru