УДК 004.042, 004.89 Вестник УрФО № 2(52) / 2024, с. 5–12

Крыжановская Ю. А., Курченкова Т. В.

10.14529/secur240201

ПРИМЕНЕНИЕ ДЕРЕВЬЕВ КЛАССИФИКАЦИИ ДЛЯ РЕАЛИЗАЦИИ СКОРИНГОВОЙ СИСТЕМЫ

В статье обсуждаются результаты разработки системы оценивания кредитоспособности заявителя. На основе анализа существующих моделей выбран подход применения деревьев классификации, преимуществам которого являются минимальное количество ручной конфигурации, возможность обработки данных с числовыми и категориальными атрибутами, а также наглядная интерпретация причин принятия того или иного решения. Модель классификации обучалась на общедоступном наборе данных, предоставляющем данные по 20 параметрам. Для хранения данных приложения была спроектирована база данных. Разработанное приложение основано на клиент-серверной архитектуре и реализует паттерн MVC.

Ключевые слова: *скоринговая система, деревья классификации, база данных, модель классификации, принятие решений, клиент-серверное приложение.*

Kryzhanovskaya Yu. A., Kurchenkova T. V.

CLASSIFICATION TREESBASED SCORING SYSTEM DEVELOPMENT

The article discusses the developed system for assessing the borrower's creditworthiness. Based on the existing models analysis, the approach of using classification trees was chosen. It's advantages are the minimum manual configuration amount, the ability to process data with numerical and categorical attributes, as well as visual interpretation reasons for making a particular decision. The classification model was trained on a publicly available dataset providing data on 20 parameters. A database was designed to store application data. The developed application is based on client-server architecture and implements the MVC pattern.

Keywords: *scoring system, classification trees, database, classification model, decision making, client-server application.*

Введение

На протяжении истории существования банковской системы недобросовестные заемщики были большой проблемой, приносящей значительные убытки. И даже опытные кредитные эксперты [1] далеко не всегда могут с достаточной точностью оценить потенциального заемщика. Для решения этой проблемы применяются системы, способные на основании информации о предыдущих кредитных историях делать гораздо более точные прогнозы [2], тем самым превентивно решая потенциальные проблемы с недобросовестным клиентом. Также использование автоматизированных систем позволяет не предъявлять высоких требований к квалификации кредитного инспектора и влияет на скорость рассмотрения заявок.

Целью данной работы является разработка клиент-серверного приложения кредитного скоринга на основе деревьев классификации, дающего оценку заемщика по данным из его анкеты. Основное назначение приложения – быстрая оценка заемщика и сокращение времени на изучение документов.

Приложение должно обладать такими функциональными возможностями, как возможность построения модели принятия решений на основе подготовленного датасета, ввод данных о клиенте в анкету, сохранение переданных данных в базе данных, вынесение решения о предоставлении кредита на основе переданных в анкете данных, автоматическое перестроение модели через определенный системным администратором промежуток времени на основе сохраненных в БД анкет.

Анализ существующих моделей классификации

Для создания системы кредитного скоринга необходимого выбрать математическую модель её реализации. При выборе модели для скоринговых систем оценки кредитного риска заемщиков важно учитывать различные факторы, такие, как размер и качество данных, сложность данных, требуемая скорость и точность предсказаний, а также стоимость и время разработки. На первом этапе работы были проанализированы различные подходы к данной проблеме [3, 4].

Один из популярных и хорошо изученных подходов – метод ближайших соседей [5]. По сравнению с другими подходами характеризуется относительно простой реализацией и изученностью. Однако, если число признаков, на которых строится классификация, велико, метод плохо работает, а для признаков может быть сложным подобрать веса и определить какие из них являются важными.

Еще один применяемый для построения скоринговых систем подходов – логистическая регрессия [6]. Это модель хорошо подходит для бинарных классификаций, а с многоклассовыми задачами менее эффективна. Также важна подготовка данных, причем для стабильности получаемого результата может потребоваться большой объем выборки.

В скоринговых системах применим и нечетко-множественный подход [7]. К преимуществам нечеткой модели можно отнести способность оперировать одновременно количественными и качественными характеристиками, к недостаткам – сложность интерпретации результатов и трудности в обновлении и адаптации, а также худшую производительность.

Также в скоринговых системах возможно применение генетических алгоритмов [8]. В этом случае генерируется начальное множество скоринговых функций, к которому применяются операции скрещивания и мутации с выводом из рассмотрения наименее пригодных функций. Генетические алгоритмы применимы в системах оценки кредитоспособности заемщиков благодаря гибкости и надежности, однако требуют значительных вычислительных ресурсов и времени для нахождения оптимальных решений, а также тщательной настройки параметров.

Еще один метод построения скоринговой модели – нейронная сеть [9]. Эта модель может обрабатывать сложные данные и выявлять нелинейные зависимости, но требует больших вычислительных ресурсов и времени обучения.

Также возможно применение модели, основанной на использовании деревьев классификации [10]. Такая модель позволяет отнести объект – потенциального заемщика – к одному из заранее известных классов, например, к классу «Одобрить заявку» или к классу «Отклонить заявку». Это позволяет, основываясь на значениях признаков, разделить данные на подмножества и найти лучшие правила классификации. В результате получается представление правил в виде иерархической структуры, где каждому объекту соответствует единственный узел-решение. Модель допускает адаптацию к меняющимся условиям путем перестроения дерева.

Алгоритм классификации

Для реализации скоринговой системы, представленной в данной работе, из перечисленных моделей был выбран алгоритм дерева решений, так как, как сказано выше, дерево может перестраиваться при добавлении новых данных и игнорировать несущественные признаки. К преимуществам алгоритма можно отнести и минимальное количество ручной. Также он удобен подходит для случает обработки данных со смешанными атрибутами (числовыми и категориальными). Еще одним достоинством является наглядная интерпретация причин принятия того или иного решения с помощью дерева.

Одной из ключевых составляющих при построении дерева решения является энтропия Шеннона (информационная энтропия) — мера неопределённости, связанной со случайной величиной. Эквивалентно, энтропия Шеннона – мера информационной упорядоченность, однородности множества.

Построение дерева начинается с задания множества объектов, предикатов и минимального числа объектов для поиска разбиения. Каждый объект имеет набор атрибутов со значениями. Предикаты являются логическими условиями (больше, меньше, равно и т. д.). При построении рекурсивно вычисляется наилучшее разбиение исходного множества. Для каждого объекта множества ищется такое сочетание атрибута и предиката, которое приведет к уменьшению среднего значения энтропии Шеннона. Критерием остановки поиска разбиения является либо близкое к нулю значение энтропии, либо размер оставшегося после разбиения множества, меньший заданного минимального. Из полученного множества путем подсчета наиболее часто встречающегося класса объекта создается лист дерева с полученным классом.

После завершения построения производится оптимизация полученного дерева. Рекурсивно вверх, начиная с нижних узлов дерева, если оба листа относятся к одному классу, то они сливаются в 1 лист на место родительского узла. Чтобы определить к какому классу относится объект при помощи дерева классификации, следует рекурсивно спускаться по дереву. Направление при этом выбирается исходя из значений предикатов, применяемых к классифицируемому объекту. С точки зрения интерпретации результатов, каждый путь от корня дерева до листа отражает объяснение того, почему объект был отнесён к указанному классу. Проблема переобучения может быть решена при помощи леса решений, который включает несколько деревьев. При этом результат классификации определяется «голосованием», при котором ответом выбирается тот класс, за которой проголосовало большее число деревьев.

В представленной работе для прогнозирования строится лес из нечетного количества деревьев. При классификации обрабатываемого объекта каждое дерево выносит свое решение о том, к какому классу будет отнесен объект. Определить, с какой вероятностью объект принадлежит конкретному классу, можно исходя из количества проголосовавших за этот результат деревьев. При этом каждое дерево обучается на своём случайном подмножестве исходной обучающей выборки данных, что необходимо для уменьшения числа ошибочных классификаций. Слишком большое количество деревьев в лесе не имеет практической целесообразности, так как увеличение количества деревьев больше, чем до 11, практически не увеличивает качество классификации.

Средства реализации

В качества языка программирования в работе используется Java 17.0.3. Выбор обусловлен большим количеством фреймворков и библиотек для разработки web-приложений, а также хорошей скоростью работы. В частности, Spring Framework предлагает функцию внедрения зависимостей, которая позволяет объектам определять свои собственные зависимости, которые контейнер Spring позже вводит в них. Также в работе используется Spring Boot, упрощающий разработку web-приложений благодаря возможности автоматической конфигурации с помощью предустановленных зависимостей, которые не нужно настраивать вручную. Spring Data — дополнительный удобный механизм для взаимодействия с сущностями базы данных, организации их в репозитории, извлечения и изменения данных.

В качестве системы управления базами данных (СУБД) была выбрана PostgreSQL. СУБД отличается высокой надёжностью и хорошей производительностью, поддерживает транзакции (ACID), поддерживает хранение больших двоичных объектов, а репликация реализована встроенными механизмами. Также PostgreSQL часто используется в связке

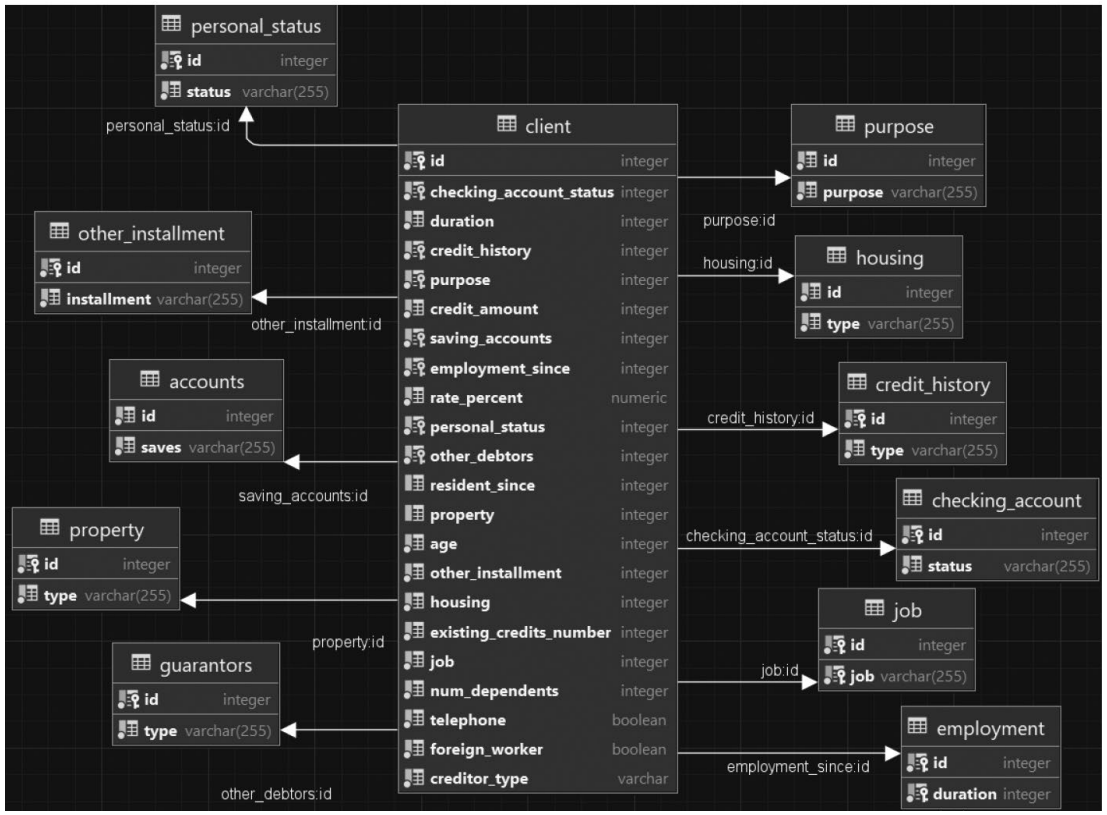

Рис. 1. Схема БД

с языком программирования Java, благодаря чему существует большая база знаний по совместной работе этих продуктов.

Структура базы данных

Для скоринговой системы была разработана база данных (БД), которая включает 12 таблиц:

- client содержит основную анкетную информацию о клиенте, которая используется для переобучения модели и оценки возможности предоставления кредита;
- checking_account хранит статусы расчетного счета;
- purpose – существующие цели кредита;
- credit_history – хранение статусов кредитной истории клиента;
- accounts хранит в себе статусы сберегательного счета;
- personal_status – социальные статусы и пол клиента;
- other_installment – статусы расчетного счета;
- property отражает существующие типы собственности;
- guarantors содержит возможные типы прочих должников и поручителей;
- housing хранит существующие виды владения жильем;
- job – типы рабочей квалификации клиента;
- employment – существующие типы стажа работы.

Структура БД показана на рис. 1.

Обучение модели

Одним из важнейших этапов создания модели классификации, определяющим качество последующих предсказаний, является подбор качественной обучающей выборки данных. Подобранный датасет должен содержать достаточное количество данных, быть применимым в реальной системе и учитывать ключевые характеристики заемщика.

Для обучения был подобран общедоступный датасет из репозитория машинного обучения UCI (University of California, Irvine). Датасет содержит 5000 обезличенных реальных кредитных историй за 2008 год и классификацию того, считается ли заявитель хорошим или плохим. Он предоставляет данные по 20 параметрам как числовым, так и категориальным, которые нашли отражение в структуре таблицы client.

Структура приложения

Разрабатываемое приложение основано на клиент-серверной архитектуре и реализует паттерн MVC. На стороне клиента происходит получение запрошенных данных, а также ввод данных пользователем в систему. Сервер получает запросы с клиентской части и реагирует на них. Осуществляются запросы к базе данных с последующей необходимой обработкой над полученной выборкой данных. Впоследствии эти данных заполняются в представление и передаются в виде ответа на клиентскую часть. База данных позволяет сохранять данные в ПЗУ, тем самым предотвращая их потерю и не засоряя оперативную память сервера.

Работа приложения начинается с http-запроса пользователя. Spring перенаправляет выполнение поступающих http-запросов на соответствующие методы контроллеров. Классы контроллеров содержат реализации service-классов. В сервисном классе производится обращение к деревьям решений для классификации переданной анкеты, а также запись анкеты в базу данных.

На начальном этапе разработки приложения были определены основные доменные сущности и классы. Модуль классификации состоит из классов, представленных на рис. 2.

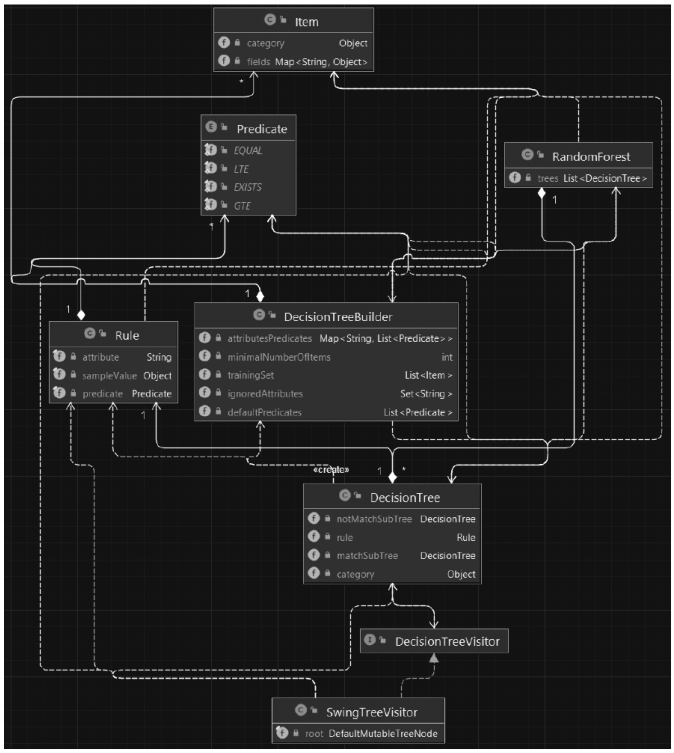

Рис. 2. UML диаграмма классов модуля классификации

Класс Item предназначен для хранения значения атрибутов классифицируемого объекта. Для большей гибкости он содержит в себе ассоциативный массив, ключом в котором является название атрибута, а значением – значение атрибута. Это позволяет легко добавлять и убирать поля при необходимости, не затрагивая остальные компоненты системы.

Класс Predicate хранит в себе логические выражения, представленные в виде полей класса. Каждое поле содержит в себе переопределенный метод «Predicate», принимающий 2 сравниваемых значения и возвращающий логический результат его вычисления. Таким образом, достигается инкапсуляция работы с логическими выражениями в отдельных объектах, благодаря чему можно легко настраивать необходимый набор предикатов для каждого атрибута объекта.

Класс Rule хранит сочетание атрибута объекта, его значения и предиката. Именно это сочетание определяет искомое в процессе построения дерева разбиение множества на подмножества, удовлетворяющее и не удовлетворяющее этому правилу. Впоследствии оно сохраняется в узле дерева, и классифицируемый объект проходит проверку на соответствие этому правилу.

Класс DecisionTree непосредственно отвечает за хранение и построения дерева классификации. Для этого в нем определены поля того же типа matchSubTree и notMatchSubTree, что делает структуру данных рекурсивной. Поле rule предназначено для хранения правила типа «Rule» и инициализировано, только если текущий объект является узлом дерева. Поле category отображает класс объекта и заполнено только когда текущий объект является листом дерева.

Класс DecisionTreeBuilder является вспомогательным для построения дерева. Он хранит в себе параметры, которые нужно только для построения и не должны сохраняться в самом дереве.

Класс RandomForest предназначен для хранения леса классификации. Он хранит в себе список деревьев и методы для разбиения

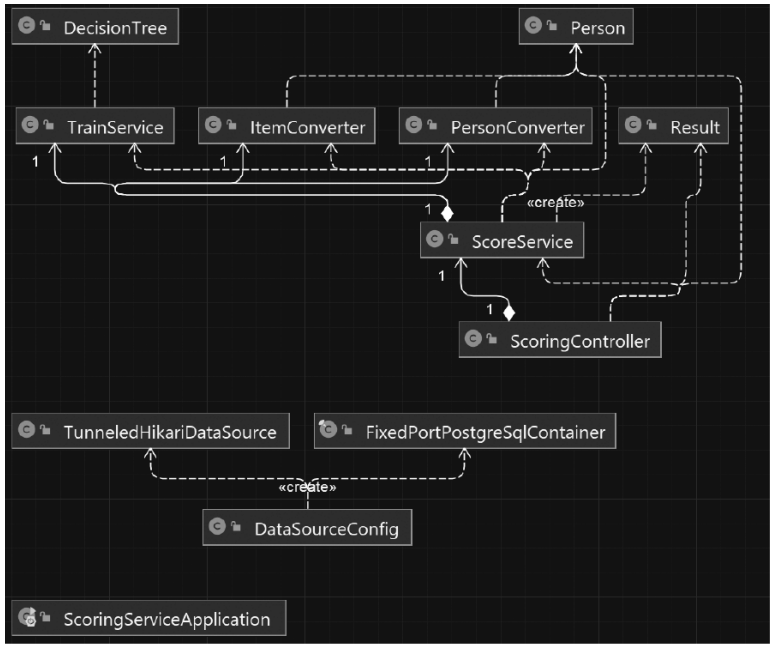

Рис. 3. UML диаграмма классов модуля web части приложения

исходного датасета на случайные обучающие подмножества.

Класс SwingTreeVisitor предназначен для обхода дерева с последующим его выводом на экран.

Общая структура приложения представлена на рис. 3.

На диаграмме видна структура web части приложения. Класс ScoringController принимает приходящие запросы. Классы ScoreService и TrainService являются сервисными и отвечают за классификацию поступающей анкеты и переобучение модели соответственно. Классы Person и Result отображают модели клиента и результата классификации.

Также реализованы классы конвертеров, для перевода поступающих данных в формат, обрабатываемый деревьями классификации. Также в конфигурационных классах заданы настройки подключения к базе данных и создание докер-образа приложения.

Общее описание приложения

Перед стартом приложения в конфигурационном файле application.properties необходимо указать путь к файлу с обучающим датасетом, количество генерируемых деревьев в лесе, а также настройки для подключения к базе данных. Далее при старте генерируется деревья классификации на основе переданного датасета. При желании, в конфигурационном файле можно включить отображение всех построенных деревьев, которые различаются между собой, поскольку обучаются на случайных подмножествах исходного датасета.

Для ввода данных предусмотрена анкета, находящаяся на главной странице приложения. При нажатии на кнопку «Получить оценку» на сервер отправляется HTTP POST запрос с переданными данными на адрес /score. На сервере запрос принимает контроллер, где он валидируется. Далее, в сервисном слое, вызывается метод, обрабатывающий данные клиента через деревья классификации и возвращающий количество решений «За» и «Против», на основе которых формируется окончательный результат. После этого анкета с принятым решением сохраняется в базе данных. Далее результат возвращается контроллером и отображается на странице ввода.

Для переобучения предусмотрен HTTP PATCH метод /retrain, с помощью которого можно переобучить модель, дополнив исходный датасет накопленными данными из БД.

Заключение

В результате проделанной была работы спроектирована база данных и реализовано приложение, позволяющее с помощью деревьев классификации выносить решение о возможности предоставления заемщику кредита, исходя из данных анкеты. Функциональность приложения может быть расширена, также есть возможность масштабирования. Приложение было протестировано с использованием 200 записей датасета, не применявшихся при обучении модели, и показало результат классификации, совпадающий с находящимся в датасете на 85%, что позволяет сделать вывод о применимости реализованного подхода в реальной системе.

Литература

1. Hutchins J. The US Farm credit system and agricultural development: Evidence from an early expansion, 1920–1940. American Journal of Agricultural Economics. 2023. Vol. 105. No. 1. P. 3-26. DOI: 10.1111/ajae.12290/.

2. Pushkareva L.V., Galochkina O.A., Bezgacheva O.L. Current trends in the banking system of Russia. Espacios. 2019. Vol. 40. No. 4. P. 22-29.

3. Addo, P., Guegan, D., & Hassani, B. (2018). Credit Risk Analysis Using Machine and Deep Learning Models. Risks, 6(2), 38. doi: 10.3390/risks6020038

4. Munkhdalai, L., Munkhdalai, T., Namsrai, O., Lee, J., & Ryu, K. (2019). An Empirical Comparison of Machine-Learning Methods on Bank Client Credit Assessments. Sustainability, 11(3), 699. doi: 10.3390/su11030699

5. Алексеева В. А. Применение метода ближайших соседей при моделировании кредитных рисков / В. А. Алексеева, Р. И. Калимулинна // Вестник Ульяновского государственного технического университета. – 2014. - № 3. – С. 54-56

6. Сорокин А. С. Построение скоринговых карт с использованием модели логистической регрессии / А. С. Сорокин // НАУКОВЕДЕНИЕ. – 2014. - № 2.

7. Волкова Е.С., Гисин В.Б., Соловьев В.И. Методы теории нечетких множеств в кредитном скоринге // Финансы и кредит. – 2017. – Т. 23, № 35. – С. 2088 – 2106. https://doi.org/10.24891/fc.23.35.2088

8. Ong C. S., Huang J. J., Tzeng G. H. Building credit scoring models using genetic programming // Expert Systems with Applications. – 2005. – Vol. 29. – [1] 1. – P. 41-47.

9. Кадиев А.Д., Чибисова А.В. Нейросетевые методы решения задачи кредитного скоринга. Математическое моделирование и численные методы, 2022, № 4, с. 81–92.

10. Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research //European Journal of Operational Research. – 2015. – Vol. 247. – [1] 1. – P. 124-136.

References

1. Hutchins J. The US Farm credit system and agricultural development: Evidence from an early expansion, 1920–1940. American Journal of Agricultural Economics. 2023. Vol. 105. No. 1. P. 3-26. DOI: 10.1111/ajae.12290/.

2. Pushkareva L.V., Galochkina O.A., Bezgacheva O.L. Current trends in the banking system of Russia. Espacios. 2019. Vol. 40. No. 4. P. 22-29.

3. Addo, P., Guegan, D., & Hassani, B. (2018). Credit Risk Analysis Using Machine and Deep Learning Models. Risks, 6(2), 38. doi: 10.3390/risks6020038

4. Munkhdalai, L., Munkhdalai, T., Namsrai, O., Lee, J., & Ryu, K. (2019). An Empirical Comparison of Machine-Learning Methods on Bank Client Credit Assessments. Sustainability, 11(3), 699. doi: 10.3390/su11030699

5. Alekseyeva V. A. Primeneniye metoda blizhayshikh sosedey pri modelirovanii kreditnykh riskov / V. A. Alekseyeva, R. I. Kalimulinna // Vestnik Ul'yanovskogo gosudarstvennogo tekhnicheskogo universiteta. – 2014. - № 3. – S. 54-56

6. Sorokin A. S. Postroyeniye skoringovykh kart s ispol'zovaniyem modeli logisticheskoy regressii / A. S. Sorokin // NAUKOVEDENIYe. – 2014. - № 2.

7. Volkova Ye.S., Gisin V.B., Solov'yev V.I. Metody teorii nechetkikh mnozhestv v kreditnom skoringe // Finansy i kredit. – 2017. – T. 23, № 35. – S. 2088 – 2106. https://doi.org/10.24891/fc.23.35.2088

8. Ong C. S., Huang J. J., Tzeng G. H. Building credit scoring models using genetic programming // Expert Systems with Applications. – 2005. – Vol. 29. – [1] 1. – P. 41-47.

9. Kadiyev A.D., Chibisova A.V. Neyrosetevyye metody resheniya zadachi kreditnogo skoringa. Matematicheskoye modelirovaniye i chislennyye metody, 2022, № 4, s. 81–92.

10. Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research //European Journal of Operational Research. – 2015. – Vol. 247. – [1] 1. – P. 124-136.

КРЫЖАНОВСКАЯ Юлиана Александровна, старший преподаватель, кафедра Кибербезопасности информационных систем, факультет Прикладной математики, информатики и механики, федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный университет». 394018, г. Воронеж, Университетская площадь, 1. E-mail: jak@mail.ru

КУРЧЕНКОВА Татьяна Викторовна, кандидат технических наук, доцент, доцент кафедры Программного обеспечения и администрирования информационных систем, факультет Прикладной математики, информатики и механики, федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный университет». 394018, г. Воронеж, Университетская площадь, 1. E-mail: tatyana36136@mail.ru

KRYZHANOVSKAYA Yuliana Alexandrovna, Senior Lecturer, Department of Cybersecurity of Information Systems, Faculty of Applied Mathematics, Computer Science and Mechanics, Federal State Budgetary Educational Institution of Higher Education «Voronezh State University». 394018, Voronezh, University Square, 1. E-mail: jak@mail.ru

KURCHENKOVA Tatyana Viktorovna, Candidate of Technical Sciences, Associate Professor, Associate Professor of the Department of Software and Administration of Information Systems, Faculty of Applied Mathematics, Computer Science and Mechanics, Federal State Budgetary Educational Institution of Higher Education «Voronezh State University». 394018, Voronezh, University Square, 1. E-mail: tatyana36136@mail.ru