ФГБОУ ВО

Уфимский университет науки и технологий

Отчет по прохождению программы повышения квалификации по направлению «Технологии обработки количественных данных»

Тема: «Интеллектуальный анализ данных о поставщиках»

Выполнил: Шарипова А. М.

Уфа 2023 г.

**Введение**

«Проведение эффективного аудита поставщика – сложный процесс, требующий большого внимания к деталям, начиная от выбора объекта для проведения аудита, заканчивая анализом полученных результатов после. Предприятие, инициирующее аудит, получает преимущества в виде улучшения отношений с поставщиками, сокращение затрат на цепочке поставок, повышение качества поставок, рост удовлетворенности клиентов, в то время как проверяемая организация – возможность увидеть «слабые» места в производстве, получить рекомендации и улучшить качество поставляемой продукции».

В настоящее время все больше компаний сталкивается с необходимостью проходить аудит своих поставщиков. Это связано с увеличением рисков, связанных с качеством продукции и услуг, а также со стремлением компаний к сокращению расходов.

Одним из наиболее эффективных способов проведения аудита является дистанционный формат. В этом случае аудиторы могут проводить проверку, не выезжая на место, что позволяет существенно сократить затраты на проведение процедуры.

Кроме того, дистанционный формат более быстр и удобен для проведения. Например, аудитор может проводить проверку в любое удобное время, не привязываясь к рабочему графику сотрудников компании поставщика. Также в данном формате легче выявить потенциальные риски, связанные с качеством продукции и услуг.

В целом, использование дистанционного формата проведения аудита поставщиков является актуальным и эффективным подходом, который позволяет существенно снизить затраты на проведение процедуры, одновременно обеспечивая высокое качество проверки.

Целью работы является повышение качества процесса выбора формата проведения аудита.

Задачами работы являются:

* разработка нейронной сети для определения формата проведения аудита;
* построение дерево решений для формирования правил классификации.

Объектом исследования является процесс проведения аудита поставщиков.

Предметом исследования является модели и алгоритмы процесса проведения аудита поставщиков.

При выполнении работы использовались методы системного анализа, функционального моделирования и программирования.

Научная новизна заключается в сформированных правилах классификации, позволяющих определить формат проведения аудита.

В ходе работы выполнен интеллектуальный анализ данных в среде *Deductor*, который позволил определить соответствующий формат проведения аудита для проблемных поставщиков. Разработана нейронная сеть и построено дерево решений.

# **1 Проведение аудита поставщиков**

Проверка поставщика – это процесс подтверждения способности поставщика поставлять продукцию, отвечающую определенным требованиям.

Основанием для принятия решения о проверке поставщика может быть:

* большое количество ремонтных работ на поставляемую продукцию;
* увеличение несоответствий, обнаруживаемых при входном контроле и т.д.

Цели аудита включают:

* подтверждение возможности поставщика производить и доставлять продукцию, соответствующую установленным требованиям организации;
* получение объективной информации об условиях разработки, производства и испытаний покупной продукции и услуг поставщика, а также информации о функционировании и управляемости бизнес-процессов и процессов производства, влияющих на качество и надежность продукции поставщика;
* контроль выполнения договорных требований;
* выявление рисков несогласованности процессов, влияющих на качество и надежность продукции;
* определение эффективности корректирующих действий по результатам предыдущих проверок [1].

По формату проведения аудита выделяют [2] очный, то есть с выездом на территорию поставщика, и дистанционный.

При дистанционном аудите проверка проходит при помощи обмена информации при помощи информационно-коммуникационных технологий, систем электронного документооборота и др.

В настоящее время дистанционный формат проведения аудита имеет много преимуществ:

* сокращение командировочных расходов, так как не требуется выезжать на территорию поставщика;
* сокращение временных затрат, так как для подготовки к дистанционному аудиту требуется меньше времени;
* отсутствие затрат на замещение сотрудников, так как работники могут не покидать рабочее место.

Несмотря на разного рода достоинства дистанционного аудита, ранее многие компании рассматривали возможность проведения лишь очных аудитов. В настоящее время наблюдается другая ситуация: компании начали активно внедрять удаленный формат аудита.

Рассмотрим особенности процесса проведения дистанционного аудита поставщика.

Во-первых, необходимо тщательно выбирать поставщика на данный способ проверки. Если поставщик имеет большое количество рекламационных актов за прошедший год, заметен рост по сравнению с предыдущим отчетным периодом, представители поставщика не принимают меры по сокращению брака или они неэффективны, то проведение дистанционного аудита будет неправильно. В данном случае, стоит отдать предпочтение очному аудиту.

Во-вторых, в процесс аудита будут привлекаться службы, работающие в области информационных технологий. Передача и хранение большого количества данных требует от руководителя аудита обеспечения их надежной защиты и уничтожения с целью предотвращения несанкционированного доступа.

В-третьих, будут внесены изменения в стандарте организации на проведение аудита поставщика. Если предприятие готово к проведению дистанционных аудитов, то оно должно включить требования в стандарт организации, регламентирующий порядок организации таких проверок.

Модель проведения аудита представлены на рисунке 1.1.

Выбор поставщика на аудит

Формирование план-графика

Организация аудита

Проведение аудита

Список проблемных поставщиков

План-график аудита

План аудита

Отчет по аудиту

Справка по качеству

Рисунок 1.1 – Модель проведения аудита поставщика

Таким образом, обоснована необходимость в проведении аудита поставщиков и проведения анализа данных о поставщиках.

**2 Интеллектуальный анализ данных о поставщиках**

**2.1 Обоснование выбора среды разработки**

Программа *Deductor* – это инновационная среда для разработки и реализации интеллектуальных систем, которая предоставляет полный спектр инструментов для анализа данных, машинного обучения и решения задач искусственного интеллекта. Эта среда призвана ускорять и упрощать процесс создания и развертывания интеллектуальных приложений, благодаря своей гибкости, скорости, удобной интерфейсной схеме, а также мощным и интеллектуальным алгоритмам [10].

Среда *Deductor* предоставляет набор инструментов для работы с разными типами данных, в том числе текстовыми, числовыми, временными рядами, геопространственными данными и т.д., и позволяет использовать различные методы и алгоритмы для их анализа. Это включает в себя множество интегрированных библиотек, таких как *pandas*, *NumPy*, *scikit-learn*, *TensorFlow* и *Keras* [13].

Среда *Deductor* также обладает статистическими и графическими инструментами для анализа данных, визуализации результатов и проведения статистических тестов. Она предоставляет возможность быстро и просто отбирать наиболее важные признаки, которые могут способствовать лучшему обучению моделей машинного обучения. Среда *Deductor* также предоставляет возможность проводить предварительную обработку данных, такую как заполнение пропусков, нормализацию данных и т.д.

Основной функционал *Deductor* связан с разработкой и обучением различных моделей машинного обучения. В этой среде доступны множество интеллектуальных алгоритмов, такие как: регрессия, классификация, кластеризация, деревья решений, ансамблирование, нейронные сети и т.д. Кроме того, *Deductor* предлагает гибкий интерфейс для настройки параметров моделей и управления обучением. Среда *Deductor* может автоматически подбирать гиперпараметры модели, используя механизм оптимизации, что упрощает этот сложный процесс.

Среда *Deductor* позволяет экспортировать обученные модели в различные форматы, такие как *Python*, *Java*, *C*++, а также создавать web-приложения на базе моделей, которые развертываются на веб-сервере. Этот механизм упрощает развертывание моделей машинного обучения в продакшн-системах.

*Deductor* поддерживает совместную работу с другими инструментами, такими как *R*, *Python*, *MATLAB*, *Excel* и другими, что позволяет экспортировать данные и модели между различными средами и интегрировать их в комплексные аналитические решения.

Наконец, *Deductor* предоставляет обширную документацию и реализует гибкую систему поддержки пользователей, которые могут получить помощь от разработчиков инструментария через форумы и онлайновые сервисы поддержки.

Основные преимущества *Deductor* включают:

1. Широкий выбор методов классификации. *Deductor* предлагает несколько методов классификации, таких как *Support vector machine* (*SVM*), *Naive Bayes*, *Decision trees*, *k-nearest neighbors* (*k*-*NN*), *Random Forest*;
2. Интерфейс пользователя. *Deductor* имеет простой и удобный интерфейс, облегчающий работу начинающим пользователям. В некоторых случаях, программа может быть полезна для быстрого построения и обучения классификатора;
3. Поддержка многопроцессорных систем. *Deductor* позволяет распараллеливание вычислений, что делает возможным построение более сложных моделей классификации в кратчайшие сроки;
4. Удобная система представления результатов. *Deductor* имеет графический интерфейс, чтобы пользователю было проще ориентироваться в полученных результатах, например, находить метрики, которые отображают качество модели.

Таким образом, *Deductor* – это мощная среда, которая обладает всем необходимым функционалом для решения комплексных задач в области машинного обучения и анализа данных. *Deductor* позволяет проводить предварительную обработку и визуализацию данных, выбирать наилучшие модели машинного обучения и оптимизировать их параметры, а также проводить анализ результатов и принимать на их основе решения. Более того, Deductor имеет простой и удобный интерфейс, что делает его доступным как для специалистов в области анализа данных, так и для новичков. Все это делает *Deductor* незаменимым инструментом для работы с данными и достижения высоких результатов в анализе и прогнозировании.

Нейронные сети часто применяют для решения задач классификации. Они могут использоваться во многих областях, например, в медицине, финансах, маркетинге, обработке изображений и анализе текста.

Для решения задач классификации нейронные сети могут быть оптимизированы для использования в различных средах. Например, сверточные нейронные сети могут быть использованы для обработки изображений, поскольку они могут распознавать образцы, характерные для конкретных типов объектов. Рекуррентные нейронные сети могут использоваться для анализа текста, поскольку они могут учитывать контекст и последовательность слов.

Для решения задач классификации нейронные сети могут быть обучены на большом объеме данных, что позволяет им выявлять характеристики различных классов и классифицировать новые объекты в соответствии с выявленными признаками. В зависимости от типа задачи можно выбрать разные архитектуры нейронных сетей и различные методы оптимизации параметров.

Применение нейронных сетей для решения задач классификации позволяет повысить точность и скорость решения задач, а также получить более точные результаты в сравнении с традиционными методами.

**2.2 Разработка нейронной сети для выбора формата проведения аудита**

Разработка нейронной сети выполняется в пять этапов.

На первом этапе определяются входные переменные нейронной сети. В качествен входных переменных выбраны критерии оценки поставщиков. Предложено использовать следующие критерии.

1. Географическое расположение. Если поставщик находится в другом городе, регионе или даже стране, то проведение очного аудита может быть затруднительным и нерентабельным. В таком случае целесообразно провести дистанционный аудит, используя средства связи и удаленный доступ к информации.
2. Техническая оснащенность. Если поставщик обладает современными информационными технологиями и системами управления, то для проведения аудита можно использовать возможности удаленного доступа к данным и программам. В противном случае, будет необходимо проводить очный аудит.
3. Категория риска. Если поставщик является критическим для бизнеса организации, то его аудит следует проводить в очной форме, чтобы получить максимально полное представление о его деятельности. Если же поставщик не является ключевым для бизнеса, то можно провести дистанционный аудит с использованием минимально необходимых данных.
4. Сложность процессов. Если поставщик выполняет простые и стандартизированные процессы, то можно провести дистанционный аудит, ориентированный на проверку соответствия установленным стандартам и требованиям. Если же поставщик выполняет сложные и уникальные процессы, то следует проводить очный аудит для более глубокой проверки процессов и их эффективности.

Таким образом, выбор формата аудита зависит от многих факторов. (поямснить что это выходная переменная)

При формировании обучающей выборки применялись следующие оценки по критериям, представленные в таблице 1.1.

Таблица 2.1 – Оценки критериев

|  |  |  |
| --- | --- | --- |
| № | Название критерия | Оценки |
| 1 | Географическое расположение | Критерий представлен в числовом виде как расстояние между двумя городами  (АО «УАПО» и городом поставщика) |
| 2 | Техническая оснащенность | 1 – низкая оснащенность, 2 – средняя оснащенность, 3 – высокая оснащенность |
| 3 | Категория риска | 1 – низкий риск, 2 – средний риск,  3 – высокий риск |
| 4 | Сложность процессов | 1 – простые и стандартизированные процессы, 2 – сочетание разных процессов,  3 – сложные и уникальные процессы |

На втором этапе по выделенным критериям был подготовлен файл «Поставщики.txt», с данными для загрузки в программу *Deductor*(приложение А).

В качестве обучающего выбора данных выступает база данных, включающая записи с описанием характеристик поставщика; часть поставщиков рекомендована на проведение аудита в дистанционном и очном форматах. Характеристики поставщика: «Географическое расположение», «Техническая оснащенность», «Категория риска», «Сложность процессов». На основе этих данных необходимо построить нейросетевую модель, которая сможет дать ответ, какой формат проведения аудита подходит поставщику. Задача относится к группе задач классификации, т.е. обучения с учителем.

Перед построением нейросетевой модели проводится процедура импортирования, в которой столбец с номером класса нужно указать как дискретный вид данных. Затем запускается «Мастер обработки», выбирается метод обработки данных – «Нейронная сеть».

На третьем этапе построения модели проводится настройка назначения столбцов. На рисунке 2.1 показана настройка назначений исходных столбцов данных. Выходной столбец – столбец «*Form*» с дискретным значением типом данных, остальные столбцы – входные, столбец «№» – информационное.

Способ разбиения исходного набора данных на обучающее и тестовое множество соответствует заданному по умолчанию.

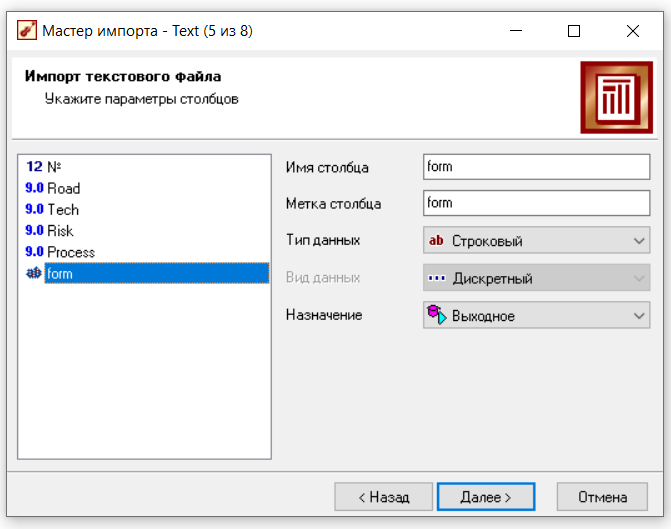


Рисунок 2.1 – Настройка назначений столбцов

На четвертом этапе проводится настройка структуры ИНС (рисунок 2.2): количество нейронов входного слоя указывается равным 100 (количество входных переменных), выходного слоя – 1 (количество выходных переменных). Устанавливается 2 скрытых слоя по 4 нейрона в каждом слое, активационная функция – сигмоида с крутизной, равной единице.

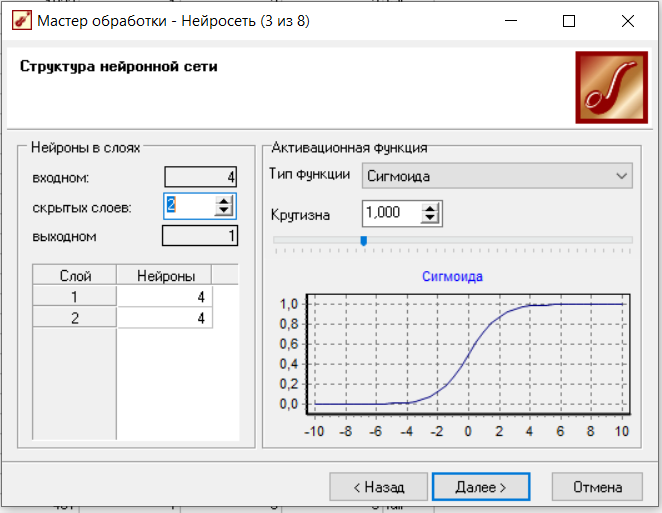


Рисунок 2.2 – Структура нейронной сети

На пятом этапе проводится настройка параметров обучения нейронной сети (рисунок 2.3). В качестве алгоритма обучения выбирается алгоритм обратного распространения ошибки (*Back-Propagation*).

При настройке параметров остановки обучения устанавливаются следующие значения: пример будет считаться распознанным, если ошибка меньше 0,05, условие остановки обучения – при достижении эпохи, равной 1000.

Запускается процесс обучения, фрагмент которого проиллюстрирован на рисунке 1.4. Показано, что на эпохе №1000 на обучающем и на тестовом множествах распознано 100% примеров.

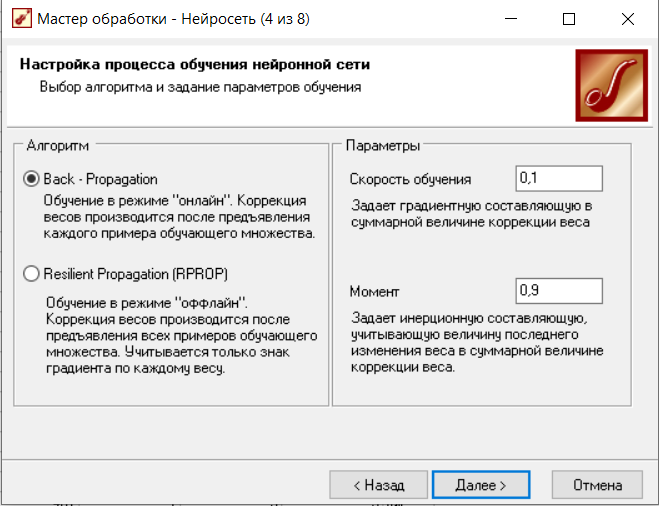


Рисунок 2.3 – Настройка процесса обучения нейронной сети

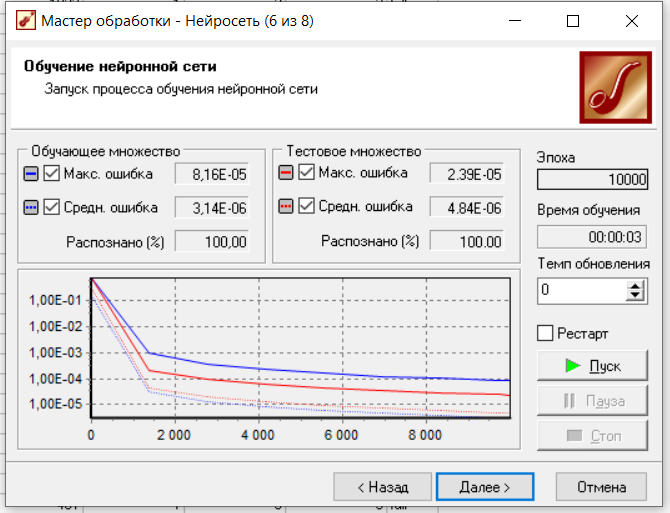


Рисунок 2.4 – Обучение нейронной сети

При выборе способа отображения обработки данных для решаемой задачи утанавливаются следующие визуализаторы: «Граф нейронной сети», анализ «Что-если», «Таблица сопряженности».

На рисунке 2.5 представлен граф нейронной сети, построенной в соответствии с ранее заданной структурой (рисунок 1.2): входной слой (4 нейрона), 2 скрытых слоя (по 4 нейрона в каждом слое) и выходной слой (1 нейрон).

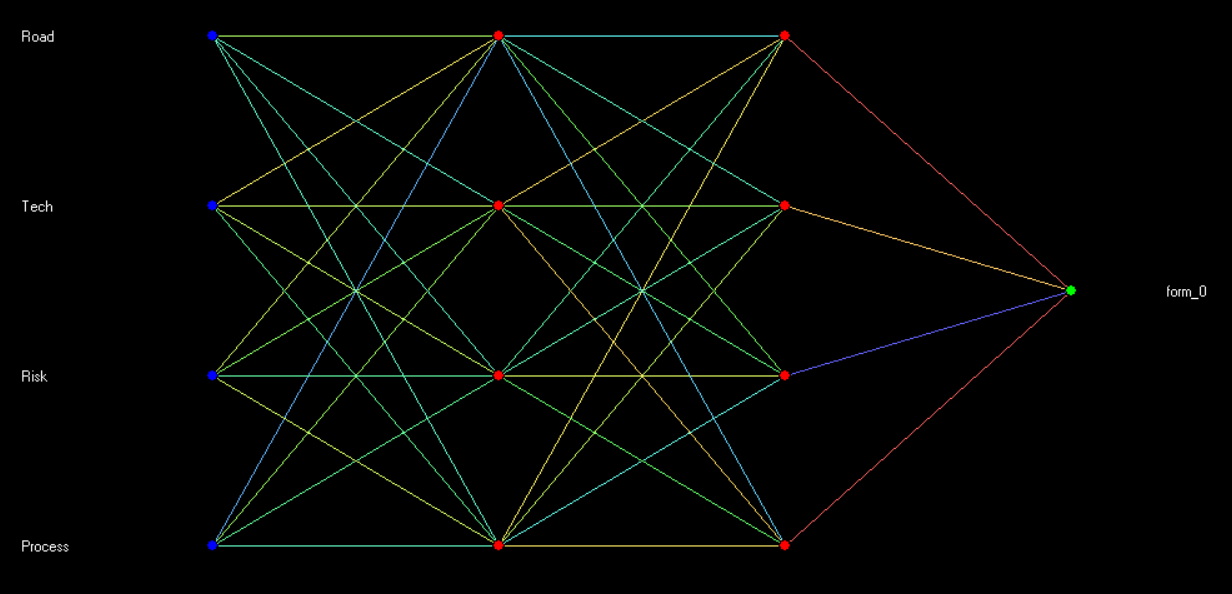


Рисунок 2.5 – Фрагмент графа нейросети

Качество построения классификационной модели отражается в таблице сопряженности (рисунок 2.6). В ячейке главной диагонали таблицы сопряженности записываются значения числа примеров, которые были правильно распознаны для разных меток классов: 19 поставщиков, которые рекомендованы на проведение аудита в дистанционной формате (класс «*dist*»), и 81 поставщиков, которые рекомендованы на проведение аудита в очном формате. Неверно классифицированные примеры отсутствуют.

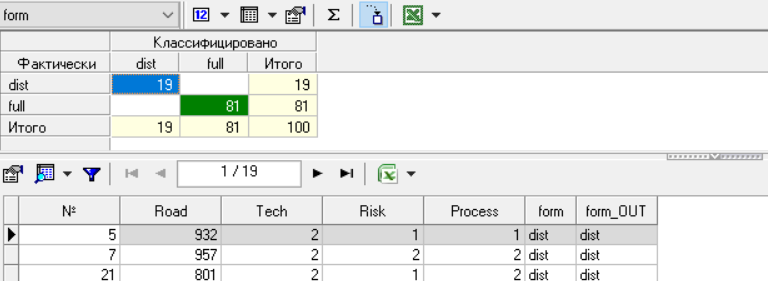


Рисунок 2.6 – Таблица сопряженности

Исследовние поведения построенной классификационной нейросетевой модели проводится в визуализаторе «что-если». Исходные данные о некотором объекте вводятся в соответствующие входные поля, а выходное поле рассчитывается.

На рисунке 2.7 представлен фрагмент эксперимента, при котором исходые данные характеристики поставщиков формируют в нейросетевой модели результаты в виде рекомендаций «дистанционный»/»очный».

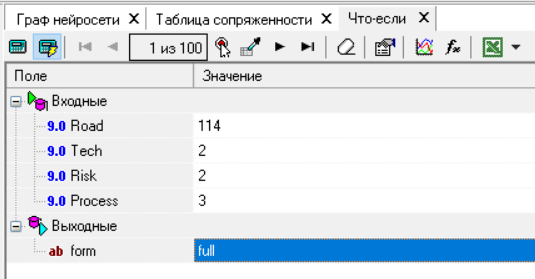
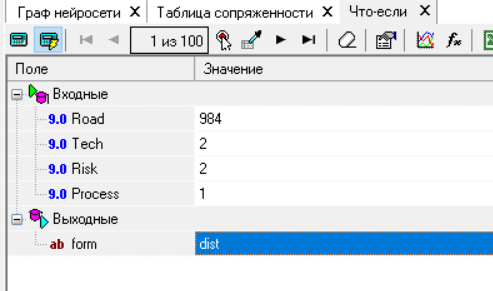
 

Рисунок 2.7 – Фрагмент анализа «что-если»

Многократное проведение экспериментов с различными исходными данными позволяет, во-первых, выявить значимые и незначимые признаки во входных полях (изменения значений незначимых признаков оказывает очень слабое влияние на результат классификации, либо не влияет на него вовсе), а во-вторых, определить диапазоны изменения набора значений входных признаков, приводящих к тому или иному результату классификации.

Построенная нейросетевая классификационная модель в неявном виде содержит правила классификации поставщиков к заданным форматам и при появлении нового объекта позволит сформировать решение о подходящем поставщику формату аудита (очный или дистанционный). Для получения класиифкации правил в явном виде далее строится дерево решений.

**2.3 Построение дерева решений для формирования правил классификации**

Дерево решений – это тип алгоритма машинного обучения, который используется в задачах классификации или прогнозирования. Он разбивает данные на подмножества на основе определенных критериев (например, признаков), которые выбираются таким образом, чтобы максимизировать различие между классами или снизить ошибку прогнозирования. Построенное дерево используется для предсказания класса или значения целевой переменной.

Данный тип алгоритма поможет выявить правила для формирования правил классификации поставщиков по 2-м форматам аудита.

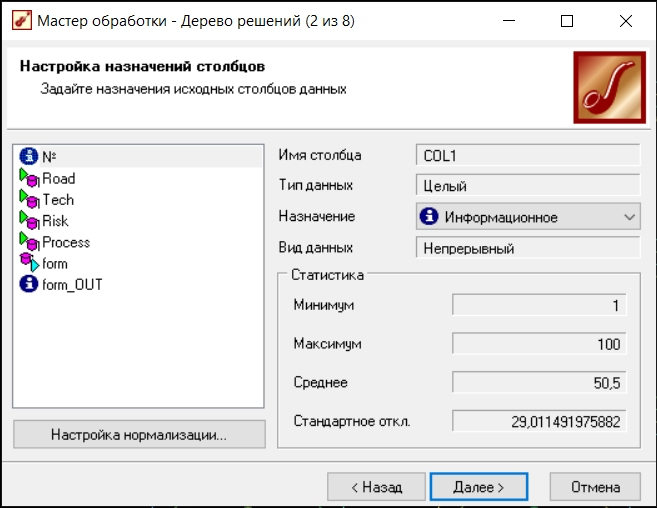
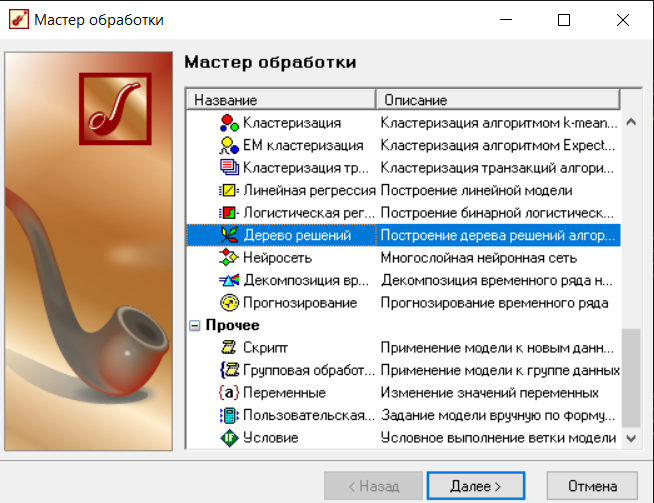
Мастер обработки «Дерево решений» в программе *Deductor* позволяет создавать модели прогнозирования и классификации данных на основе алгоритма дерева решений.

Мастер обработки «Дерево решений» в *Deductor* позволяет задать параметры алгоритма, такие как глубина дерева, количество листьев, пороговые значения для разбиения данных и другие. Кроме того, можно задать веса для классов, чтобы учесть дисбаланс между классами в данных.

После обучения модели можно произвести ее применение на новых данных и оценить качество прогнозирования при помощи метрик, таких как точность, полнота, *F*1-мера и другие. Также можно визуализировать полученное дерево и проанализировать, какие признаки оказывают наибольшее влияние на прогнозирование.

Мастер обработки «Дерево решений» является одним из мощных инструментов для анализа данных в *Deductor*. Он позволяет создавать прогностические и классификационные модели на основе алгоритма дерева решений, которые могут быть использованы для прогнозирования и определения наилучших стратегий принятия решений.

Настройка Мастера обработки в *Deductor* представлена на рисунке 2.8.

****

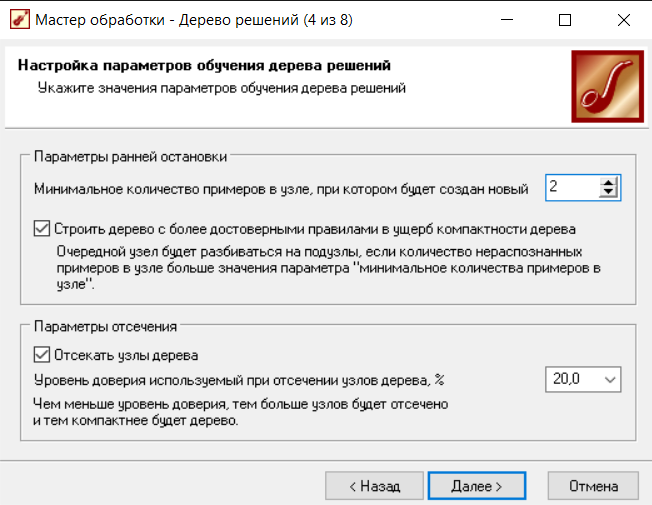
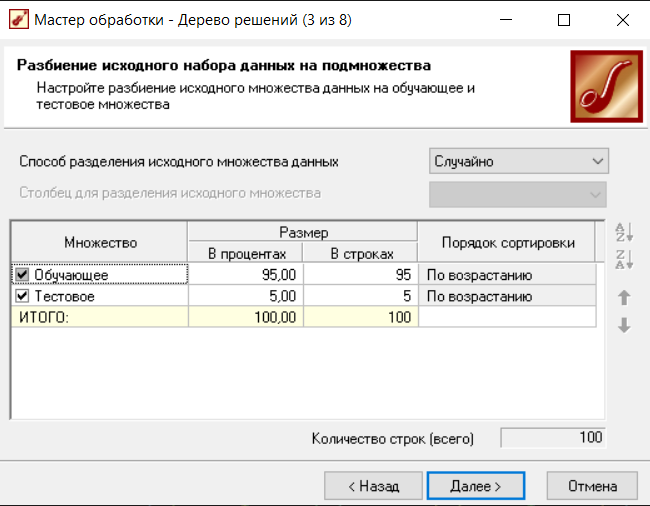
****

Рисунок 2.8 – Настройка дерева решений

Полученное дерево решений представлено на рисунке 2.9.

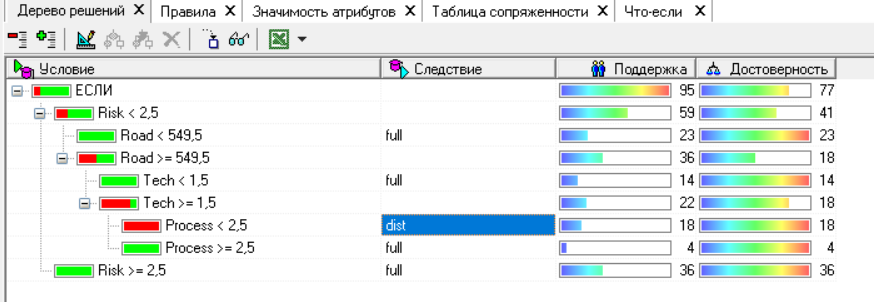
****

Рисунок 2.9 – Дерево решений

Схематичное представление дерева решения показана на рисунке 2.10.

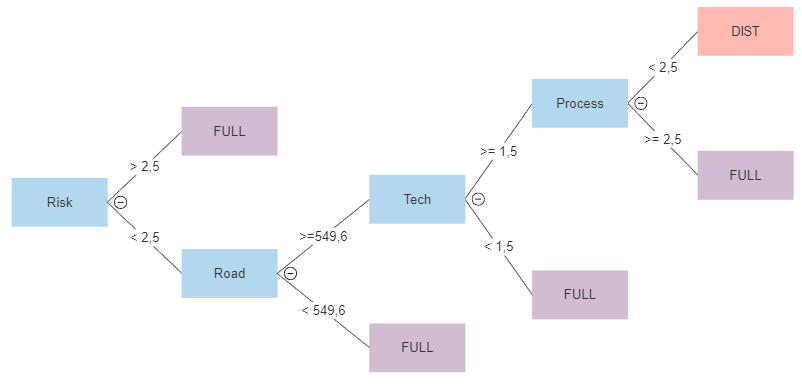
****

Рисунок 2.10 – Схематичное представление дерева решений

Полученные правила приведены на рисунке 2.11.

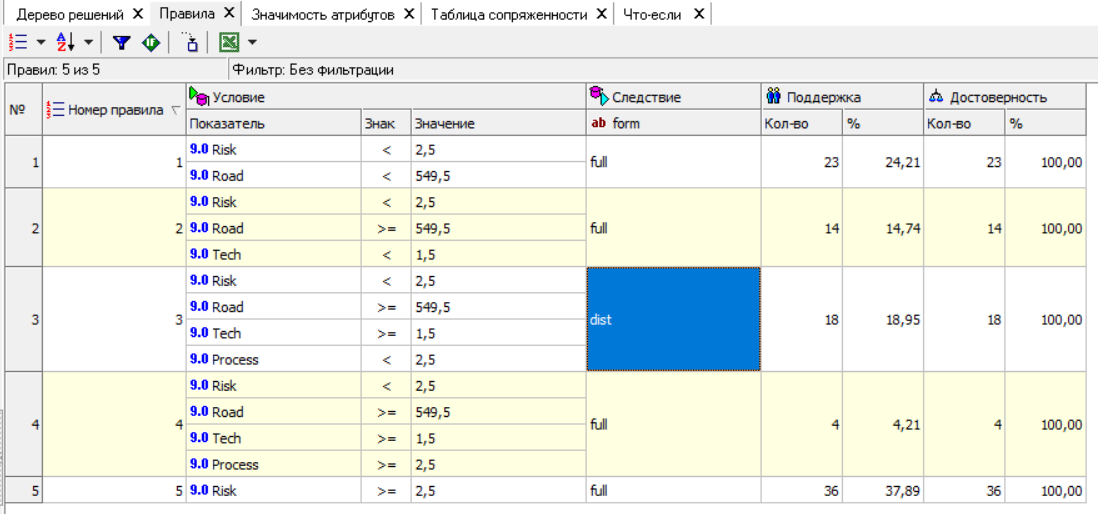


Рисунок 2.11 – Правила дерева решений

Распределение значимости атрибутов приведено на рисунке 2.12.

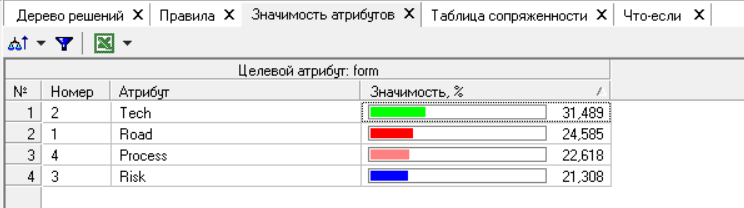


Рисунок 2.12 – Значимость атрибутов

Самым значимым атрибутом является *Tech* (Техническая оснащенность), вторым по важности программа определила критерий *Road* (Дорога), третий – критерий *Process* (Сложность процессов), четвертый – критерий *Risk* (Риск).

Таблица сопряженности приведена на рисунке 2.13.

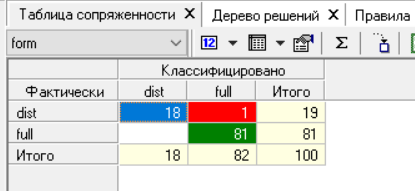
****

Рисунок 2.13 – Таблица сопряженности

Неправильно определился лишь один результат (фактически он относится к дистанционному формату, но классифицировался как очный), на рисунке 2.13 он отмечен красным.

Фрагменты выполнения анализа «что-если» представлены на рисунке 2.14.

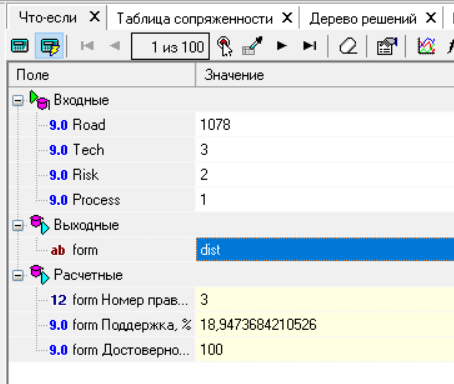
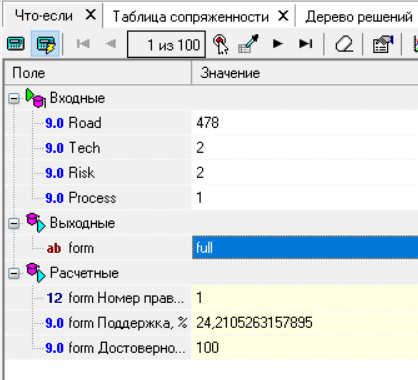
****

Рисунок 2.14 – Фрагмент анализа «что-если»

После того, как дерево решений построено на обучающем наборе данных и принято решение о его работоспособности (процент правильно распознанных примеров на обучающем множестве достаточно велик), можно приступать к практической работе с деревом — классификации новых объектов.

Новый объект, который требуется классифицировать, поступает сначала в корневой узел дерева, а затем перемещается по узлам, в каждом из которых проверяется соответствие значения атрибута правилу в данном узле, после чего объект перенаправляется в один из узлов-потомков. Процесс продолжается до тех пор, пока объект не окажется в листе. После этого ему присваивается метка класса, ассоциированного с данным листом.

По полученным данным в программе *Deductor* были проанализированы поставщики, которые отмечены в справке по качеству за 2022 год. Данная справка дополнена столбцом «Рекомендуемый формат проведения аудита».

Полученные результаты являются основой для поддержки принятия решения при определении формата проведения аудита.

# **3 Разработка программного обеспечения для подсчета подходящего формата проведения аудита**

В качестве среды реализации программного обеспечения выбран объектно-ориентированный язык программирования C# в силу его следующих особенностей [7]:

- статистическая типизация;

- поддержка полиморфизма;

- поддержка перегрузки операторов;

- доступность делегации, атрибутов, событий и др. [6].

Язык программирования С# популярен за счет своей нетипичный конструкции и специфичного синтаксиса, позволяющего в сокращенные сроки проектировать функциональные и производительные приложения. Так С# используют при создания любой программы: продвинутый бизнес-приложения, видеоигры, функциональные веб-приложения, приложения для macOS, Windows, мобильные программы для iOS и Android.ро [12].

С# популярен среди российских компаний:

- Сбербанк – для создания 3D-графики и программ виртуальной реальности;

- Ozon – для создания складских и логистических систем;

- «Яндекс» – для создания приложений для автоматизации продаж [9].

Язык C# является одним из наиболее популярных в программировании во всем мире и обладает преимуществами, которые делают С# предпочтительным для разработки разнообразных приложений:

1. Язык программирования C# относится к объектно-ориентированныму языку, что означает, что он позволяет создавать объекты, классы и абстрактные типы данных, включающие их свойства (атрибуты) и методы, которые определяют, что эти объекты могут делать.

2. Язык C# имеет отличную интеграцию с .NET Framework, что обеспечивает возможность доступа к множеству библиотек и ресурсов, которые разработчики могут использовать для создания приложений, использующих уникальные и продвинутые функции.

3. Язык C# позволяет объединять элементы различных языков, таких как C++, Java и C#, что способствует созданию более гибких решений для разработки.

4. Язык C# имеет сильную типизацию и проверку на этапе компиляции, что позволяет снизить количество отклонений в процессе создания приложений, а также повысить производительность приложения.

5. Кроме того, C# поддерживает лямбда-выражения и LINQ-запросы, что облегчает написание кода и ускоряет процесс разработки.

6. Язык C# также обеспечивает множество инструментов для автоматизации тестирования и отладки кода, что позволяет быстро находить и исправлять ошибки.

7. Язык C# имеет возможность использования промышленных стандартов, таких как XML, SOAP, JSON и многих других.

8. Язык C# позволяет проектировать приложения для множества платформ, включая Android, Windows, Linux, iOS и многие другие.

9. Язык C# также обладает большой сообществом разработчиков и обширной документацией, что является важным преимуществом при работе с ним.

10. Наконец, C# – это один из наиболее быстрорастущих языков программирования, и его известность растет по мере того, как компании и разработчики всё больше используют этот язык для создания своих проектов.

В целом, C# предоставляет множество возможностей для проектирования мощных, продвинутых и устойчивых приложений, и является одним из лучших выборов, доступных для современного разработчика.

Популярная среда разработок для программирования – это Visual Studio.

Среда *Visual Studio* – это интегрированная среда разработки от Microsoft, предназначенная для создания различных приложений и программного обеспечения. Идеально подходит для работы с .NET-платформами и языками программирования C#, F#, VB.NET, C++ и т.д. [11].

В *Visual Studio* предоставляется широкий набор инструментов для работы с кодом, стандартных компонентов и расширений от сторонних производителей. В окне редактора можно легко переключаться между кодом, XAML разметкой и дизайнером графического интерфейса.

Программа *Visual Studio* поддерживает множество функций, таких как автоматическое завершение кода, отладка, профилирование, развертывание, тестирование, создание отчетов и т.д. Плюсом является также наличие интегрированных инструментов управления контрольными системами (например, *GIT*).

Для пользователей, знакомых с Visual Studio, процесс разработки может стать достаточно простым и быстрым благодаря доступности шаблонных проектов и использованию инструментов для генерации кода.

Подсчет подходящего формата проведения аудита представлена на рисунке (рисунок 3.1). На странице представлено предупреждение о том, что программа сделает подсчет только при заполнении всех полей, а при некорректности введенных данных – выйдет информационное сообщение (рисунок 3.2).

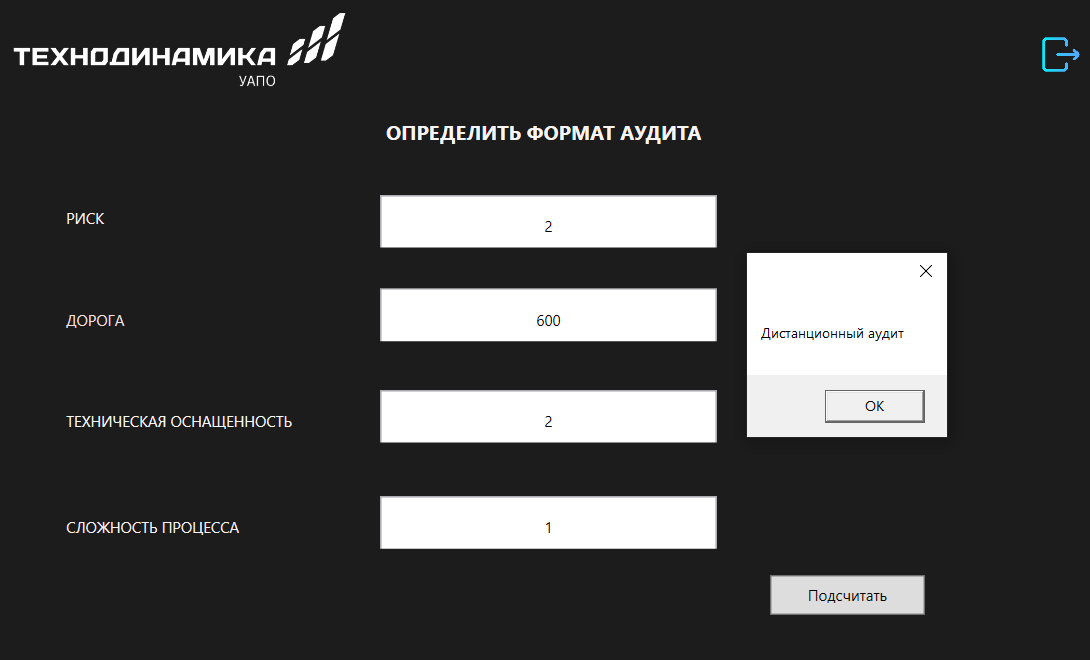


Рисунок 3.1 – Подбор формата проведения аудита

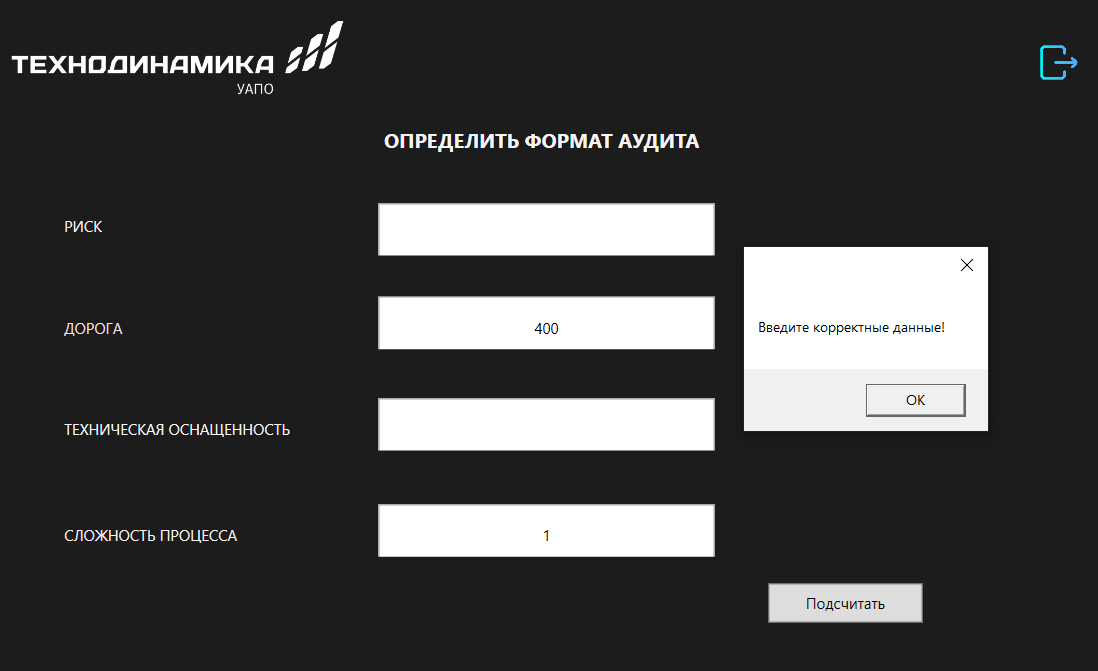


Рисунок 3.2 – Сообщение о необходимости заполнения всех полей

# **Заключение**

Обоснована актуальность проведения аудита поставщиков, необходимость внедрения дистанционного формата аудита, проведения анализа данных о поставщиках.

Выполнен интеллектуальный анализ данных о поставщиках в среде *Deductor*, который позволил определить соответствующий формат проведения аудита для проблемных поставщиков. Разработана нейронная сеть и построено дерево решений, позволяющие определить соответствующий формат проведения аудита для проблемных поставщиков.

Обоснован выбор среды реализации программного обеспечения. Разработано программное обеспечение в среде разработки для программирования *Visual Studio* на языке *С*#.

**СПИСОК ЛИТЕРАТУРЫ**

1. ГОСТ Р ИСО 19011-2021. Оценка соответствия. Руководящие указания по проведению аудита систем менеджмента: дата введения 2021-07-01. – М.: Изд-во стандартов, 2021. – 49 с.
2. ГОСТ Р 59424-2021. Руководящие указания по дистанционному проведению анализа состояния производства и аудита систем менеджмента: дата введения 2021-04-09. – М.: Стандартинформ, 2021. – 19 с.
3. Брезгин В.И. Моделирование бизнес-процессов с *ALLFusion Process Modeler* 4.1: Ч. 2: учеб. пособие / В.И. Брезгин, К.Э. Аронсон. – Екатеринбург: Изд-во Урал. ун-та, 2015. – 52 с.
4. Волкова В.Н. Моделирование систем и процессов: учебник для академического бакалавриата / В.Н. Волкова, Г. В. Горелова, В. Н. Козлов; под ред. В. Н. Волковой, В. Н. Козлова. – М.: Изд-во Юрайт, 2015. – 449 с.
5. Горлушкина Н.Н. Системный анализ и моделирование информационных процессов и систем: учеб. пособие / Н.Н. Горлушкина. – СПб: Университет ИТМО, 2016. – 120 с.
6. Марченко А.Л. Основы программирования на *С*# 2.0: учеб. пособие / А.Л. Марченко. – М.: ИНТУИТ, Вузовское образование, 2017. – 552 с.
7. Осипов Н.А. Разработка *Windows* приложений на *C*#: учеб. пособие / Н.А. Осипов. – СПб.: Университет ИТМО, 2012. – 74 с.
8. Советов Б.Я. Моделирование систем: учебник для акад. бакалавриата / Б. Я. Советов, С. А. Яковлев. – 7-е изд. – М.: Изд-во Юрайт, 2019. – 343 с.
9. Алексеев Г.С. Автоматизированное тестирование на языке программирования C# в интегрированной среде разработки *MICROSOFT VISUAL STUDIO* / Г.С. Алексеев // Научное сообщество студентов столетия. – 2018. – №1(60). – С. 52–58.
10. Вахрушев В.И. Нейросетевое моделирование оценки рыночной стоимости недвижимости с использованием *Deductor* / В.И. Вахрушев, Л.В. Курзаева, В.А. Дьяков // Современная техника и технологии. – 2016. – №11-2 (63). – С.190–194.
11. Кистанов Е.С. Технологии мобильных приложений в бизнесе / Е.С. Кистанов, Т.В. Алексеева, О.А. Страхов // Прикладная информатика. – 2019. – №2. – С. 16–31.
12. Куприенко В.М. Разработка *Windows*-приложений на языке C# / В.М. Куприенко, Н.В. Нечитайло // Современные инновации в науке и технике. – 2014. – №2. – С.340–343.
13. Фешина Е.В. Возможности системы *Deductor* как инструмента анализа данных / Е.В. Фешина, А.П. Овчарова, В.Р. Лабинцева // *COLLOQUIUM-JOURNAL*. – 2018. – №12-6(23). – С.49–51.

# **ПРИЛОЖЕНИЕ А**

(*справочное*)

**Файл «Поставщики» для загрузки в программу *Deductor***

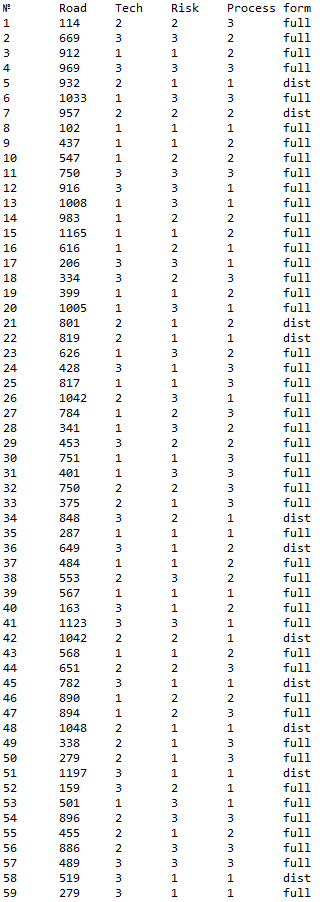


Рисунок А.1 – Файл с данными по поставщикам для проведения анализа данных

# **ПРИЛОЖЕНИЕ Б**

(*справочное*)

**Фрагмент кода функции**

private void Button\_Click(object sender, RoutedEventArgs e)

{

try

{

int num1 = int.Parse(tbx1.Text);

if (num1 > 3)

{

MessageBox.Show("Введите значение от 1 до 3");

tbx1.Text = "0";

return;

}

int num2 = int.Parse(tbx2.Text);

int num3 = int.Parse(tbx3.Text);

int num4 = int.Parse(tbx4.Text);

if (num1 > 2)

{

MessageBox.Show("Очный аудит");

}

else

{

if (num2 < 549)

{

MessageBox.Show("Очный аудит");

}

else

{

if (num3 < 2)

{

MessageBox.Show("Очный аудит");

}

else

{

if (num4 > 2)

{

MessageBox.Show("Очный аудит");

}

else

MessageBox.Show("Дистанционный аудит");

}

}

}

}

catch

{

MessageBox.Show("Введите корректные данные!");

}

}