**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Кафедра “фундаментальная информатика и информационные технологии”**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Алгоритмы и структуры данных»**

**на тему «обезличивание синтетических данных»**

**Вариант – 5**

**Студент гр. 22Б16-пу \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Олизько С.С.**

**Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дик А.Г.**

**Санкт-Петербург**

**2023 г**

Цель работы...................................................................................................................3

Задача ........................................................................................................................... 3

Теоретическая часть .....................................................................................................3

Алгоритм метода ...........................................................................................................4

Блок-схема программы .................................................................................................6

Описание программы ...................................................................................................7

Рекомендации пользователя ......................................................................................10

Рекомендации для программиста ..............................................................................11

Контрольный пример ...................................................................................................12

Вывод ............................................................................................................................13

Источники......................................................................................................................13

**Цель работы:**

Целью проекта является освоить и понять методы и принципы обезличивания синтетических данных.

**Задача:**

Создать алгоритм и разработать программу, которые позволят обезличивать, вычислять показатель к-анонимности, считать количество независимых строк относительно квази-идентификаторов синтетических данных покупок железнодорожных билетов.

**Теоритическая часть:**

В этой работе производится обезличивание данных с использованием различных методов, таких как локальное обобщение, удаление атрибутов, локальное подавление и маскеризация. Эти методы направлены на обеспечение высокой к-анонимности данных и уменьшение риска разглашения личных идентифицирующих сведений.

К-анонимность - это свойство анонимизации данных, которое обеспечивает, что в результирующем наборе данных каждая строка не может быть однозначно идентифицирована, а только как один из *k* эквивалентных элементов.

Для вычисления показателей к-анонимности датасета следует сгруппировать одинаковые строки и посчитать количество элементов в каждой группе. Показатель к-анонимности датасета определяется количеством элементов в наименьшей по размеру группе.

Квази-идентификаторы - это атрибуты в наборе данных, которые могут быть использованы для идентификации отдельных субъектов, но не обязательно уникальные. Для вычисления количества независимых строк согласно квази-идентификаторам, необходимо:

1. Определить атрибуты, которые могут служить квази-идентификаторами (например, ФИО, номера паспортов).
2. Подсчитать количество уникальных комбинаций значений этих атрибутов в данных.

Используемые методы анонимизации:

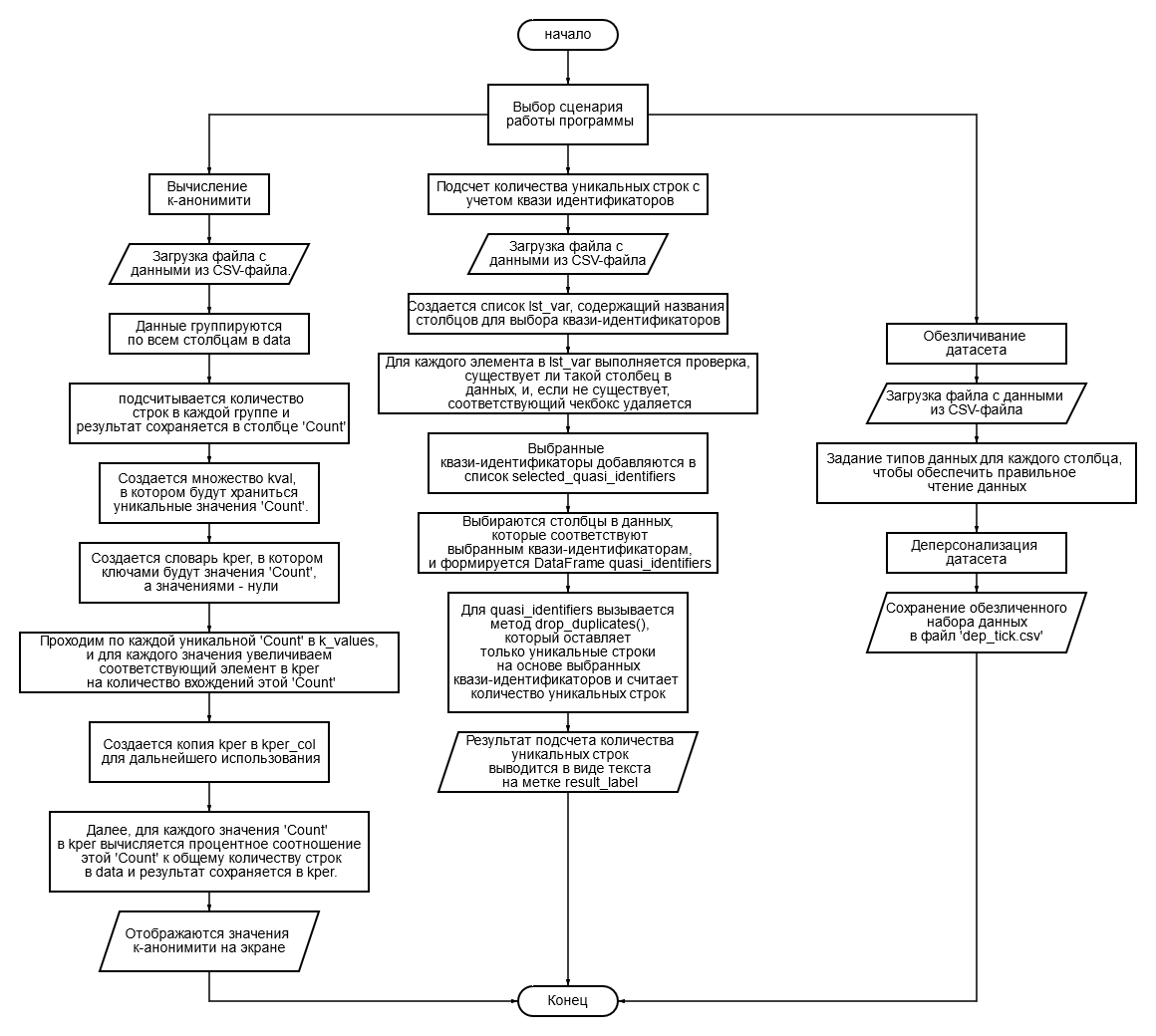
* *Локальное обобщение* - замена детализированных данных на обобщенные значения, такие как сезоны, диапазоны значений и т.д., для соблюдения к-анонимности.
* *Удаление атрибутов* - удаление чувствительных атрибутов, которые могут использоваться для идентификации, из набора данных.
* *Локальное подавление* – удаление уникальных, выбивающихся из датасета строк с количеством вхождений меньше заданного значения k.
* *Маскеризация* – скрытие чувствительных атрибутов частично или полностью, чтобы сделать их менее идентифицируемыми.

Использование этих методов в комбинации позволяет достичь желаемых значений к-анонимности и обезопасить личные данные от утечки информации. Однако, необходимо учитывать потери данных и соблюдать баланс между обезличиванием и сохранением полезной информации.

**Алгоритм метода:**

1. Выбор сценария работы программы
   1. Подсчет к-анонимности
      1. Загрузка файла с данными из CSV-файла.
      2. Данные группируются по всем столбцам в data с помощью data.groupby(data.columns.tolist()). Это позволяет сгруппировать данные по всем столбцам, создав группы, в которых строки имеют одинаковые значения во всех столбцах.
      3. С помощью метода .size() подсчитывается количество строк в каждой группе и результат сохраняется в столбце 'Count'.
      4. Создается множество kval, в котором будут храниться уникальные значения 'Count'.
      5. Создается словарь kper, в котором ключами будут значения 'Count', а значениями - нули.
      6. Проходим по каждой уникальной 'Count' в k\_values, и для каждого значения увеличиваем соответствующий элемент в kper на количество вхождений этой 'Count'.
      7. Создается копия kper в kper\_col для дальнейшего использования.
      8. Далее, для каждого значения 'Count' в kper вычисляется процентное соотношение этой 'Count' к общему количеству строк в data и результат сохраняется в kper.
      9. В интерфейсе приложения обновляется текст на метке percentage\_label, чтобы отобразить текст "К-анонимность в %".
      10. Содержимое текстового виджета result\_text очищается с помощью result\_text.delete(1.0, tk.END).
      11. Затем для каждой 'Count' в kper значения сортируются, и информация о каждом v вставляется в текстовый виджет result\_text, включая значение 'Count', процентное соотношение и количество строк с этой 'Count'.
   2. Подсчет количества уникальных строк с учетом квази идентификаторов
      1. Загрузка файла с данными из CSV-файла.
      2. Создается список lst\_var, содержащий названия столбцов для выбора квази-идентификаторов.
      3. Для каждого элемента в lst\_var выполняется проверка, существует ли такой столбец в данных, и, если не существует, соответствующий чекбокс удаляется.
      4. Выбранные квази-идентификаторы добавляются в список selected\_quasi\_identifiers.
      5. Выбираются столбцы в данных, которые соответствуют выбранным квази-идентификаторам, и формируется DataFrame quasi\_identifiers.
      6. Для quasi\_identifiers вызывается метод drop\_duplicates(), который оставляет только уникальные строки на основе выбранных квази-идентификаторов и считает количество уникальных строк.
      7. Результат подсчета количества уникальных строк выводится в виде текста на метке result\_label.
   3. Обезличивание датасета
      1. Загрузка файла с данными из CSV-файла.
      2. Задание типов данных для каждого столбца, чтобы обеспечить правильное чтение данных.
      3. Замена ФИО на пол с использованием функции replace\_names\_with\_gender.
      4. Маскирование паспортных данных, оставляя только первые две цифры номера паспорта, с помощью функции mask\_num\_data.
      5. Преобразование дат отъезда и приезда в сезоны года с использованием функции local\_generalization\_date.
      6. Удаление атрибута "Выбор вагона и места" с помощью функции remove\_attributes.
      7. Замена цен на категории (низкая, средняя, высокая) с использованием функции local\_generalization\_price.
      8. Маскирование номера карты оплаты, оставляя только первые две цифры, с помощью mask\_num\_data.
      9. Применение локального подавления строк с количеством вхождений меньше 10, используя функцию local\_suppression.
      10. Сохранение обезличенного набора данных в файл 'dep\_tick.csv'.

**Блок схема программы:**



*Рис 2: блок-схема*

**Описание программы:**

Программная реализация написана на языке python 3.10.4, с использованием библиотек: random, tkinter, pandas, datetime В программе использовались 13 функций и 13 структур данных.

В таблице ниже представлено описание функций:

*Таблица 1: функции*

|  |  |  |
| --- | --- | --- |
| Имя функции | Тип возвращаемого значения | Описание функции |
| calculate\_k\_anonymity | None | Запрашивает у пользователя выбор CSV-файла, загружает его данные, вычисляет K-анонимность и отображает результаты в графическом интерфейсе. Выводит количество строк с одинаковыми значениями для всех столбцов и их процентное соотношение. |
| calculate\_unique\_rows | None | Запрашивает у пользователя выбор CSV-файла, загружает его данные, и позволяет пользователю выбрать столбцы (квази-идентификаторы), для которых будет рассчитываться количество уникальных строк. Отображает количество уникальных строк для выбранных столбцов в графическом интерфейсе. |
| replace\_names\_with\_initials | DataFrame | Заменяет значения в столбце, содержащем ФИО, на инициалы. Функция разбивает ФИО на отдельные слова, затем создает список инициалов и заменяет исходные ФИО на инициалы. |
| mask\_num\_data | DataFrame | Маскирует часть данных в столбце, содержащем паспортные данные. Функция заменяет определенное количество символов в паспортных данных на символ '\*'. |
| round\_price\_to\_thousands | DataFrame | Округляет значения в указанном столбце до ближайшей тысячи. |
| remove\_attributes | DataFrame | Удаляет указанные столбцы из набора данных. |
| replace\_names\_with\_gender | DataFrame | Определяет пол по именам в столбце с ФИО и заменяет ФИО на пол. Функция также может удалить столбец с полами, если он больше не нужен. |
| date\_to\_season | Строка | Преобразует дату в сезон (Весна, Лето, Осень, Зима) на основе месяца. |
| local\_generalization\_date | DataFrame | Применяет функцию date\_to\_season к столбцам с датами приезда и отъезда для локальной обобщения данных. |
| price\_to\_category | Строка | Классифицирует цену в категории (Низкая, Средняя, Высокая) на основе заданных диапазонов. |
| local\_generalization\_price | DataFrame | Применяет функцию price\_to\_category к столбцу с ценой для локальной обобщения данных. |
| local\_suppression | DataFrame | Осуществляет локальное подавление данных, оставляя только те строки, у которых количество вхождений >= n. |
| dataset\_depersonalization | None | Обезличивает набор данных, применяя набор трансформаций, включая замену ФИО на пол, маскирование части паспортных данных, классификацию цены, обобщение дат, удаление ненужных атрибутов и локальное подавление данных. Результат сохраняется в новый CSV-файл. |

Описание структур данных:

*Таблица 2: структуры данных*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя структуры | Тип структуры | Тип хранимых данных | Описание |
| data | DataFrame | Various data types | Data structure representing a table of data loaded from a CSV file. |
| k\_values | DataFrame | Various data types | Data structure representing the results of counting the number of rows with identical values in all columns. |
| kval | set | Various data types | Set containing unique values of 'Count' obtained from the k\_values structure. |
| kper | dict | Int | Dictionary where keys are unique 'Count' values, and values represent the count of occurrences of each 'Count' value. |
| kper\_col | dict | Int | A copy of the kper structure used to store the original values of kper. |
| selected\_quasi\_identifiers | list | Bool | List of boolean values representing the user's selection for specific columns (quasi-identifiers) in the calculate\_unique\_rows function. |
| quasi\_identifiers | DataFrame | Various data types | Data structure representing a subset of data from the dataset that corresponds to the selected quasi-identifiers. |
| unique\_rows | int | Int | The number of unique rows in the quasi\_identifiers structure after removing duplicates. |
| name\_parts | Series | String | Series of data representing split full names into separate words in the data structure. |
| initials\_list | list | String | List of strings representing initials for each full name in the name\_parts structure. |
| counts | Series | Bool | Series of data representing boolean values (True/False) for each row in the dataset, indicating whether duplicates of that row exist. |
| data\_types | dict | Data types | Dictionary used to specify data types for columns when reading a CSV file in the dataset\_depersonalization function. |
| data | DataFrame | Various data types | Dataset loaded from a CSV file and used for depersonalization in the dataset\_depersonalization function. |

**Рекомендации для пользователя:**

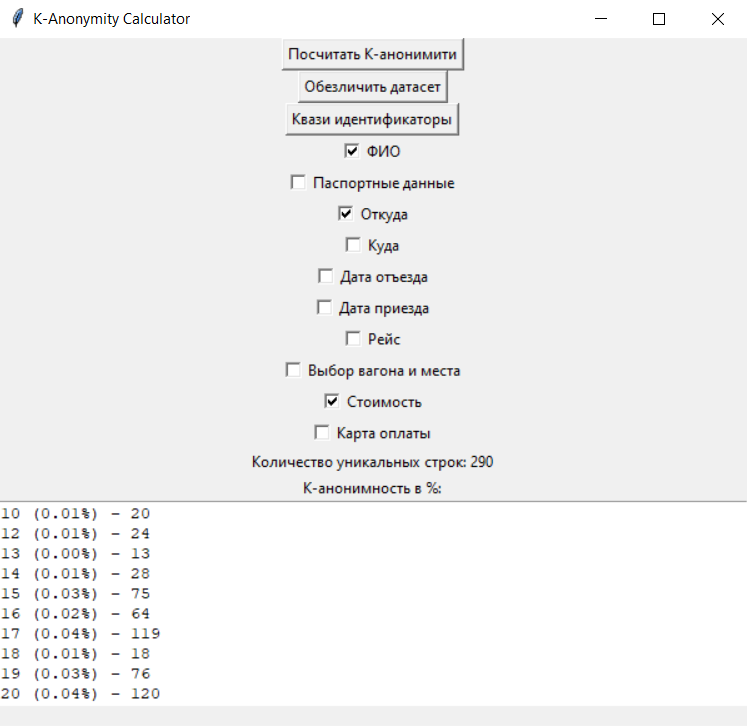
1. **Запуск программы**: Для запуска программы удостоверьтесь, что у вас установлен Python и необходимые библиотеки: tkinter, pandas, random и datetime. Запустите код, и вам будет доступен графический интерфейс, который позволяет вам выполнять различные операции с данными.
2. **Работа с K-анонимностью**: Для вычисления K-анонимности выберите опцию "Посчитать К-анонимити". Выберите CSV файл, содержащий ваши данные. Программа выведет информацию о степени K-анонимности и проценте анонимных записей.
3. **Обезличивание датасета**: Вы можете обезличить датасет, чтобы уменьшить риск раскрытия личных данных. Для этого выберите опцию "Обезличить датасет". Выберите CSV файл, содержащий данные, и программа выполнит различные операции обезличивания.
4. **Ввод и вывод данных**: Сгенерированные и обезличенные данные будут сохранены в файл dep\_tick.csv. Обратите внимание на правильность выбора параметров перед запуском обезличивания.

**Рекомендации для программиста:**

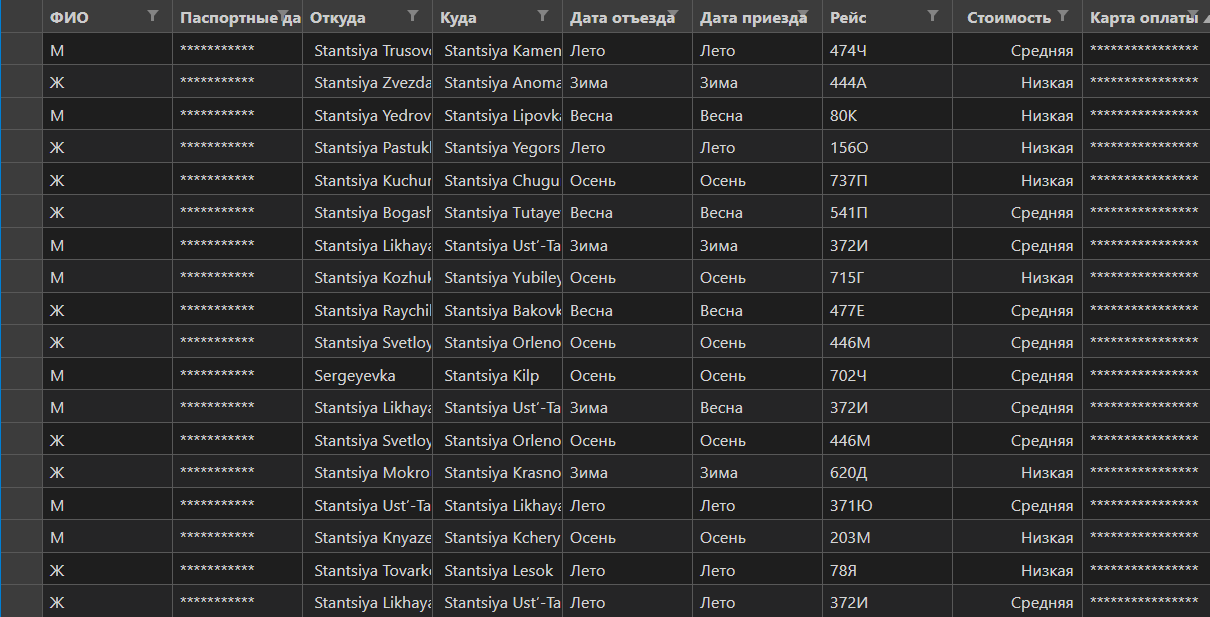
Для внесения изменений в код программы, вам потребуется устройство с установленным Python.

**Исходный код программы доступен по ссылке ниже:**

https://github.com/StephanOlizko?tab=repositories

**Контрольный пример:**  
  
****

*Рис 5: пример окна программы*

**

*Рис 6: пример вывода программы*

**Вывод:**

В рамках данной работы были исследованы основы обезличивания синтетических данных. Был разработан алгоритм вычисления показателя к-анонимити, обезличивания и нахождения количества независимых строк относительно квази-идентификаторов, а также написана соответствующая программа.

**Источники:**

1. <https://docs.python.org/3/library/random.html>
2. <https://docs.python.org/3/library/tkinter.html>
3. <https://pandas.pydata.org/>
4. <https://docs.python.org/3/library/datetime.html>