Отчёт по практической работе «Поиск ассоциативных правил»

Корсун КЭ-401

Задание 1. Доработайте программу из задания Поиск частых наборов, чтобы она также выполняла поиск ассоциативных правил. Список результирующих правил должен выдаваться в удобочитаемом виде (антецедент→консеквент) с указанием поддержки и достоверности каждого правила. Дополнительные параметры программы: порог достоверности, способ упорядочивания результирующего списка наборов (по убыванию значения поддержки или лексикографическое).

Для выполнения задания была использована функция association_rules библиотеки mlxtend. Получившийся список правил был отсортирован по значения поддержки. Реализация отображена в листинге 1.

Листинг 1 – Нахождение ассоциативных правил

```
df_accidents = read_file('accidents.dat') # Чтение файла
frequent_itemsets = apriori(df_accidents, min_support=0.5) #
Поиск частых наборов с помощью алгоритма Apriori
rules = association_rules(frequent_itemsets,
metric="confidence", min_threshold=0.7)
rules.style.hide(subset=["lift", "leverage", "conviction"], axis
= 1)
sorted_frequent_itemsets = rules.sort_values(by = 'support',
ascending=False) # Сортировка результатов по порогу поддержки
sorted_frequent_itemsets.style.hide(subset=["lift", "leverage",
"conviction"], axis = 1)
```

Задание 2. Проведите эксперименты на наборах из задания 1. В экспериментах зафиксируйте значение пороговое значение поддержки (например, 10%), варьируйте пороговое значение достоверности (например, от 70% до 95% с шагом 5%).

Задание 3. Выполните визуализацию результатов экспериментов в виде следующих диаграмм:

- сравнение быстродействия на фиксированном наборе данных при изменяемом пороге достоверности;

- общее количество найденных правил на фиксированном наборе данных при изменяемом пороге достоверности.

Для выполнения заданий была доработана функция find_result для проведения экспериментов над наборами данных и нахождения результата этих экспериментов. Для визуализации результатов также была доработана функция visualize_results. Реализации функции представлены в листингах 2 и 3.

Листинг 2 – Реализация функции find_result

```
def find_result(df): # Функция экспериментирования над наборами данных
  min_conf = [] # Порог достоверности для каждого эксперимента
  rules_count = [] # Количество правил
  process_time = [] # Время работы
  for i in range(0,6):
    start = time.time()
    if len(df) <= 1000:
      min_support = 0.01
    else:
      min_support = 0.5
    min_c = i*0.05+0.7 # Шаг порога достоверности
    frequent_itemsets = apriori(df, min_support = min_support) # Запуск алгоритма Apriori
    rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=min_c) # Поиск правил
    end = time.time()
    min_conf.append(min_c*100)
    rules_count.append(rules['antecedents'].count())
    process_time.append(end-start)
  result = pd.DataFrame(dict(min_conf = min_conf, rules_count = rules_count, process_time = process_time))
  return result # Возврат результатов эксперимента над набором данных
```

Листинг 3 – Реализация функции visualize_results

```
def visualize_results(result): # Функция визуализации результатов с помощью библиотеки Plotly
  fig = px.bar(result, x = 'min_conf', y = 'process_time', labels = {'min_conf': 'Порог достоверности, %', 'process_time': 'Время выполнения, с'})
  fig.show()
  fig = px.bar(result, x = 'min_conf', y = 'rules_count', labels = {'min_conf': 'Порог достоверности, %', 'rules_count': 'Количество правил'})
 fig.show()
```

В эксперименте над набором данных ритейла использовались пороговые значение поддержки 1%, для данных ДТП использовалось значение 50%. Пороговое значение достоверности для обоих наборов варьировалось от 70% до 95% с шагом 5%. Для визуализации результатов использовалась библиотека Plotly.

Результаты экспериментов изображены на рисунках 1 и 2.

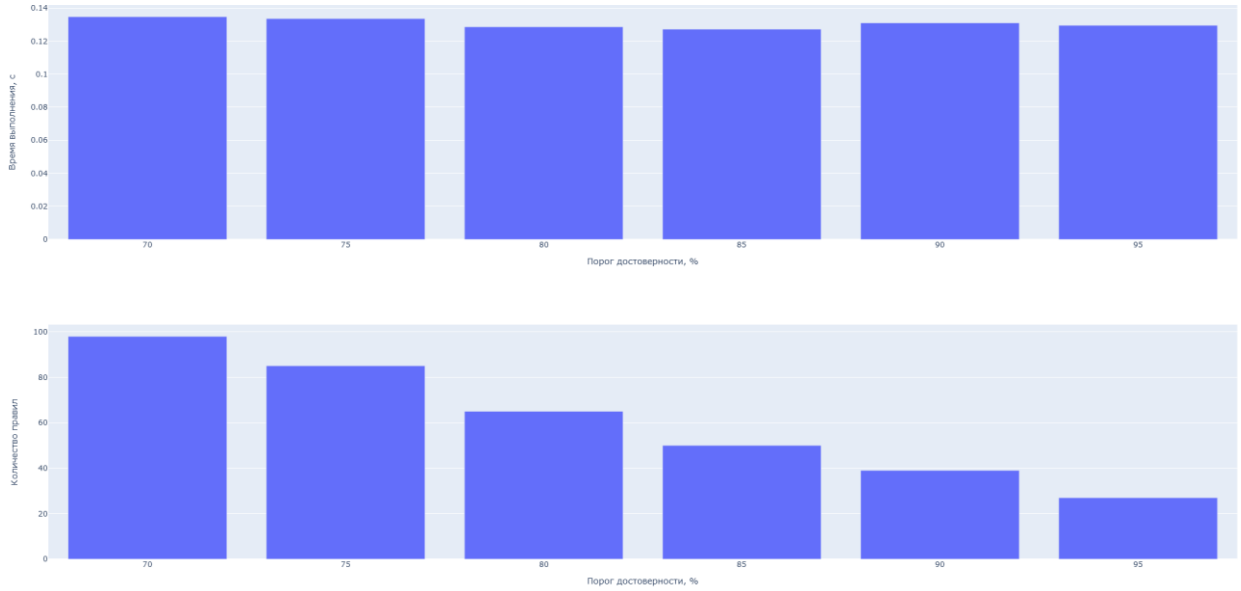

Рисунок 1 – Результаты эксперимента над набором данных ритейла

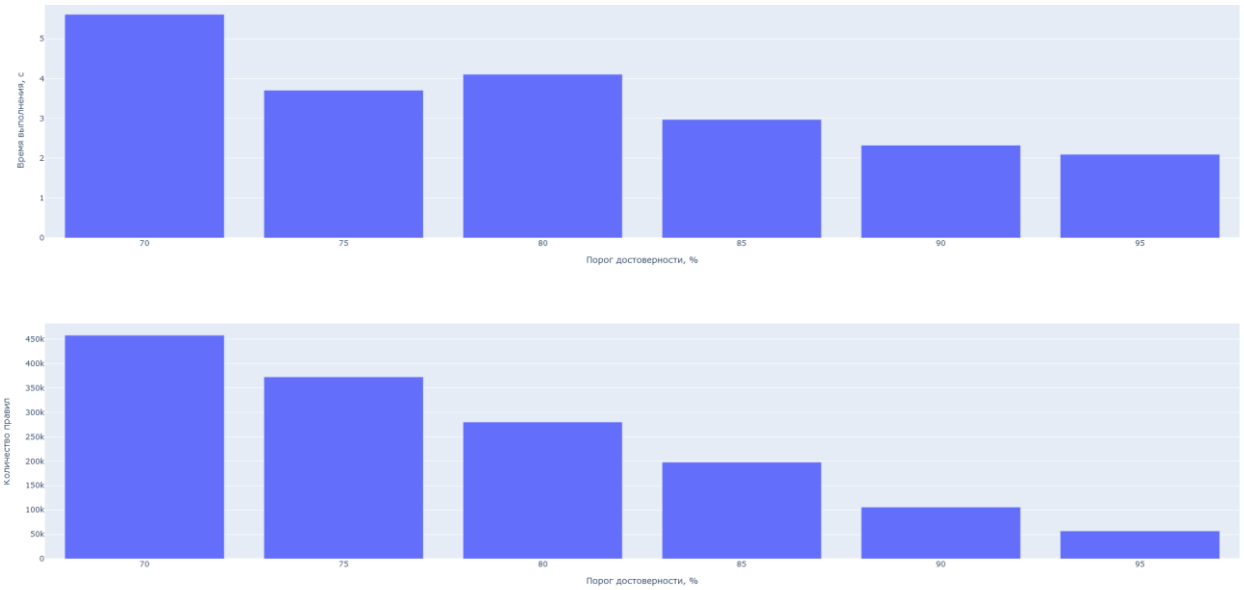

Рисунок 2 – Результаты эксперимента над набором данных ДТП

По графикам для первого набора данных можно сделать вывод, что при увеличении минимального порога достоверности количество правил линейно убывает. Время работы программы практически не отличается для разных значений порога достоверности.

Исходя из графиков для набора данных о ДТП можно сделать вывод, что значении порога поддержки 50% при уменьшении порога достоверности количество правил также линейно убывает. Время работы программы зависит от значения порога достоверности, чем ниже значение, тем дольше работает алгоритм.

Задание 4. Подготовьте список правил, в которых антецедент и консеквент суммарно включают в себя не более семи объектов (разумное количество). Проанализируйте и изложите содержательный смысл полученного результата.

Для решения задания был разработан код, находящийся в листинге 4.

Листинг 3 – Список правил из количества объектов не более 7

```
rules["antecedent_len"] = rules["antecedents"].apply(lambda x:
len(x)) # Количество объектов в антецеденте
rules["consequent_len"] = rules["consequents"].apply(lambda x:
len(x)) # Количество объектов в консеквенте
rules[(rules['antecedent_len'] + rules["consequent_len"] <= 7)]
```

В данный список попадают те объекты, которые связаны с не более чем шестью другими объектами и часто формирующие транзакции с не более, чем семью объектами.

Ссылка на репозиторий с исходными кодами: https://github.com/Vlater1/TAIP