|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА - Российский технологический университет****"**  **РТУ МИРЭА** |
| Институт информационных технологий (ИТ) |
| Кафедра инструментального и прикладного программного обеспечения (ИиППО) |

|  |
| --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1** |
| **по дисциплине** |
| **«**Программное обеспечение интеллектуальных систем»  *Изучение технологии анализа данных* |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
| Выполнил студент группы ИКБО-02-17 | *Крамаренко З.В.* |
| Приняла | *Зорина Н.В* |

|  |  |  |
| --- | --- | --- |
| Работа выполнена | «\_\_\_» \_\_\_\_\_\_\_\_ 2020 г. |  |
|  |  |  |
| «Зачтено» | «\_\_\_» \_\_\_\_\_\_\_\_ 2020г. |  |

**Цель**: изучение технологий анализа данных методами поиска ассоциативных правил, построения деревьев и кластеризации.

**Задание 1:** Реализовать поток работ, выполняющего решение задачи анализа рыночной корзины и поиска ассоциативных правил. Данный поток должен выполнять следующую последовательность действий: загрузить данные из текстового файла, преобразовать загруженные данные в специализированный тип для дальнейшего анализа, найти частые наборы и ассоциативные правила, вывести результаты.

**Ход работы:**

Реализация велась на языке программирования Python при помощи дистрибутива Anaconda в Jupyter Notebook.

Для начала найдем свободный dataset и выгрузим в удобный нам формате xlsx. Далее создадим блокнот с ассоциативными правилами AssocRule.

Импортируем наиболее популярные и необходимые нам библиотеки pandas и numpy

Далее загрузим данные из нашего dataset и проверим результат



Рисунок 1.1 – Результат 2 действия AssocRule

Далее построим промежуточную бинарную матрицу, где 1 будет обозначаться «покупка совершена», а 0 – «покупка не совершена». Проверим результат на первых 5 строчках.

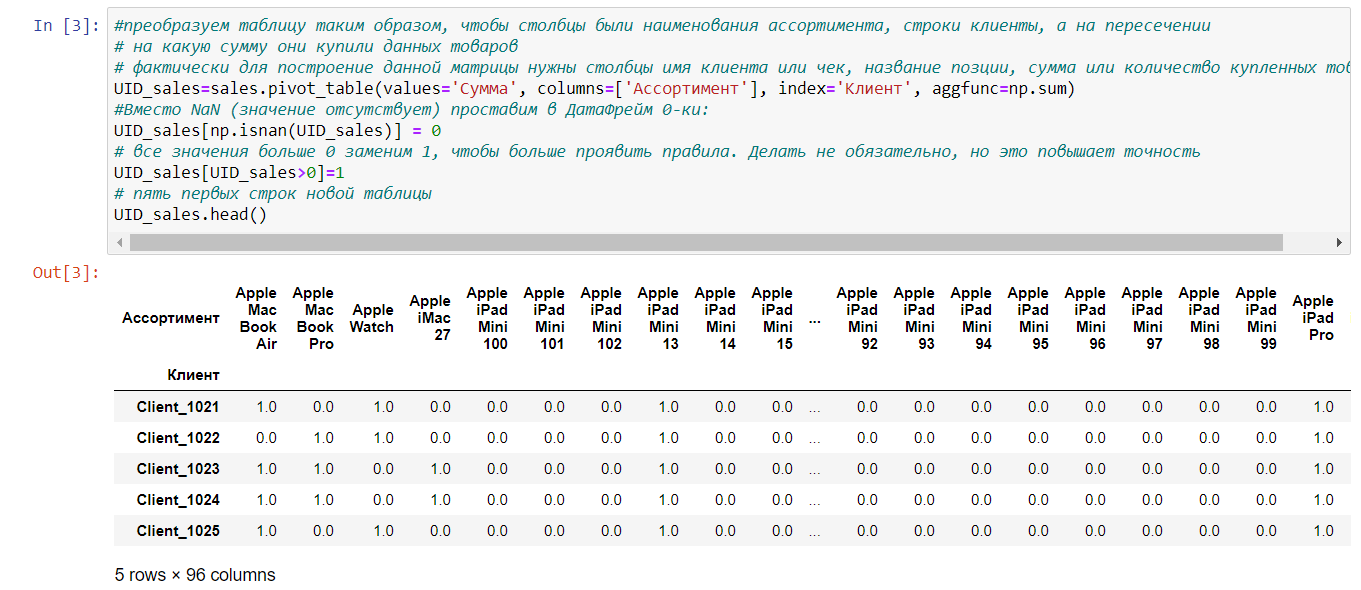


Рисунок 1.2 – Результат 3 действия AssocRule

Далее создадим список транзакций, т.е. построим описание наборов покупок (что было в чеке) по каждому клиенту.

#Создадим функцию для создания списка транзакций:

def transaction\_list(df):

    list\_external=[]

    for i in range(df.shape[0]):

        list\_internal=[]

        data=df.iloc[i]

        index=data[data>0]

        for element in index.index:

            list\_internal.append(element)

        list\_external.append(list\_internal)

    return list\_external

# используем функцию, чтобы преобразовать массив в список транзакций

transactions=transaction\_list(UID\_sales[:1000])

Проверим работу выводом списка по первому клиенту

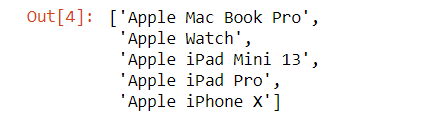


Рисунок 1.3 – Результат 4 действия AssocRule

Далее нам необходимо установить пакет pymining и импортировать дополнительные библиотеки для поиска ассоциативных правил.

pip install pymining

# импортируем дополнительные библиотеки

from pymining import itemmining, assocrules, perftesting

#подготовливаем список стандартными функциями

relim\_input = itemmining.get\_relim\_input(transactions)

item\_sets = itemmining.relim(relim\_input, min\_support=1)

#item\_sets

Далее пропишем требования для поиска: #min\_support = 5, т.е столько раз должна было повториться ассоциативное правило, min\_confidence = 0.6 - минимальный порог вероятности 60%.

# запускаем расчет ассоциативным правил

# внимание, локальная система может не справиться

#min\_support=5 - минимум 5 реализаций правила, min\_confidence=0.6 - минимальный порог вероятности

rules = assocrules.mine\_assoc\_rules(item\_sets, min\_support=5, min\_confidence=0.6)

rules

Запустим расчет



Рисунок 1.4 – Часть результата 7 действия AssocRule

Для более удобного формата записи ассоциативных правил создадим функцию для записи в виде матрицы.

#Создадим функцию для записи ассоциативных правил:

def write\_rules2(rul):

    retMass=[]

    for el in rul:

        basis=''

        for iterator in iter(el[0]):

            basis=basis+iterator+'-'

        conclusion=''

        for iterator in iter(el[1]):

            conclusion=conclusion+iterator+'-'

        retMass.append([basis, conclusion, str(el[2]), str(el[3])])

    return retMass

В заключении создадим правило для преобразования выходного массива в DataFrame и сохраним данные в Excel файл.

#созраним правила

rul1=write\_rules2(rules)

#преобразуем выходной массив в DataFrame и сохраним в Excel

df\_rules=pd.DataFrame(rul1, columns=('Посыл', 'Следствие', 'Поддержка', 'Достоверность'))

#имя файла AssocRules.xlsx

writer = pd.ExcelWriter('AssResult.xlsx')

df\_rules.to\_excel(writer,'AR')

writer.save()

На выходе получим Excel файл со всеми правила.

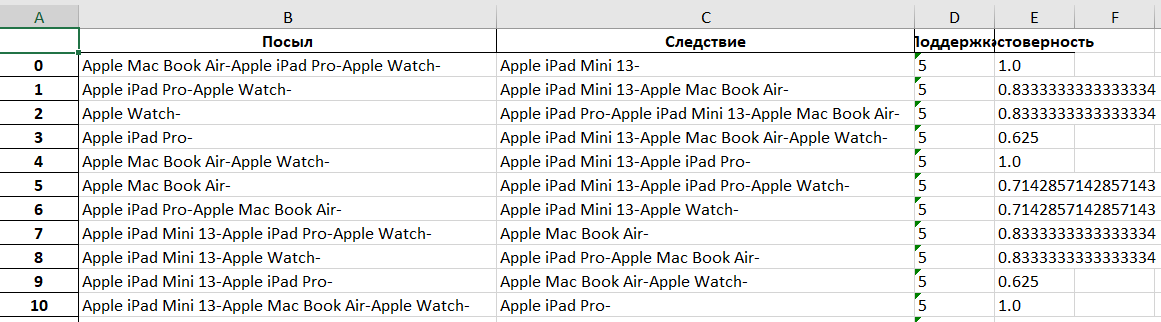


Рисунок 1.5 – 10 первых записей из файла

Для лучшего понимания результата создадим визуальный аналитический отчет в Power BI. Для этого загрузим xlsx файл в PBI и постоим диаграммы.

Для начала создадим ТОПы по Посылам и Следствиям, чтобы понять какие товары были наиболее популярны и чаще всего встречались в АП.

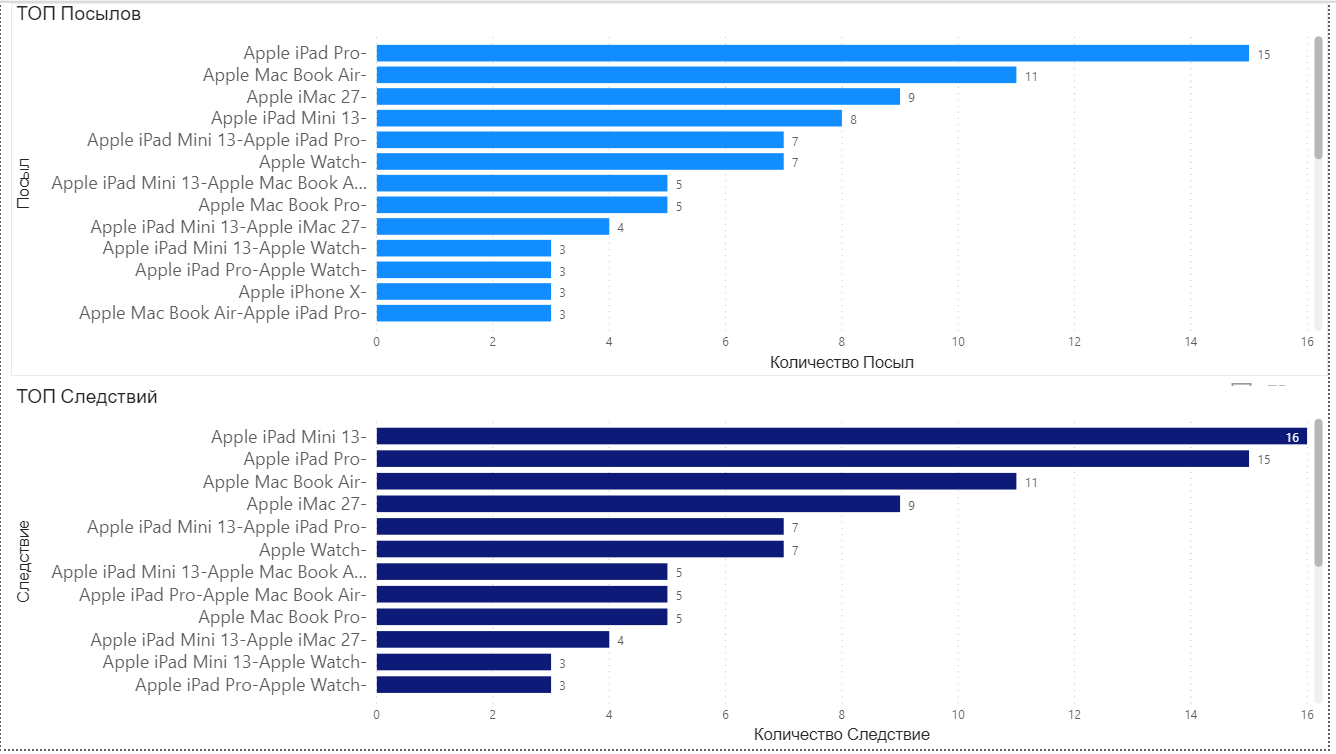


Рисунок 1.6 – Диаграммы ТОПов

Далее создадим интерактивные диаграмму по всей таблице сгруппированную по посылам, в значение поместим итоговое кол-во поддержек. Также высчитаем процентное соотношение относительной всей группы.

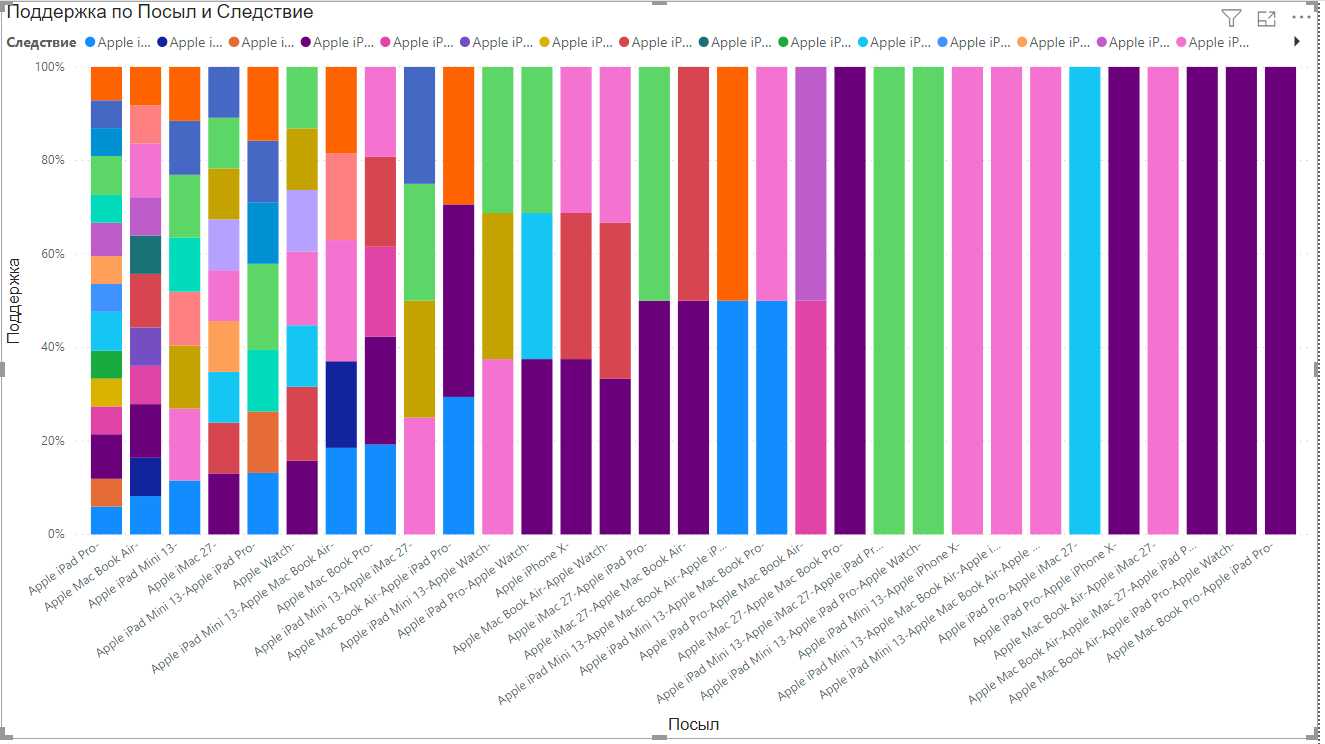


Рисунок 1.7 – Визуализация таблицы

В заключении на основании ранее построенных диаграмм выберем ТОП 3 Посылов и Следствий и создадим интерактивный аналитический отчет.

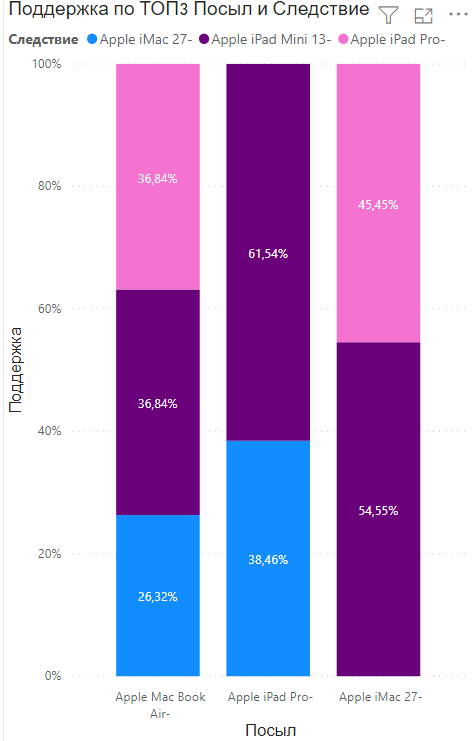


Рисунок 1.8 – Визуализация АП по ТОП 3

Также создадим диаграммы «Количество Посылов и Следствий по Достоверности» и «Количество Следствий и Посылов по Поддержка» для понимания итоговой вероятности появления АП и кол-ву повторений.

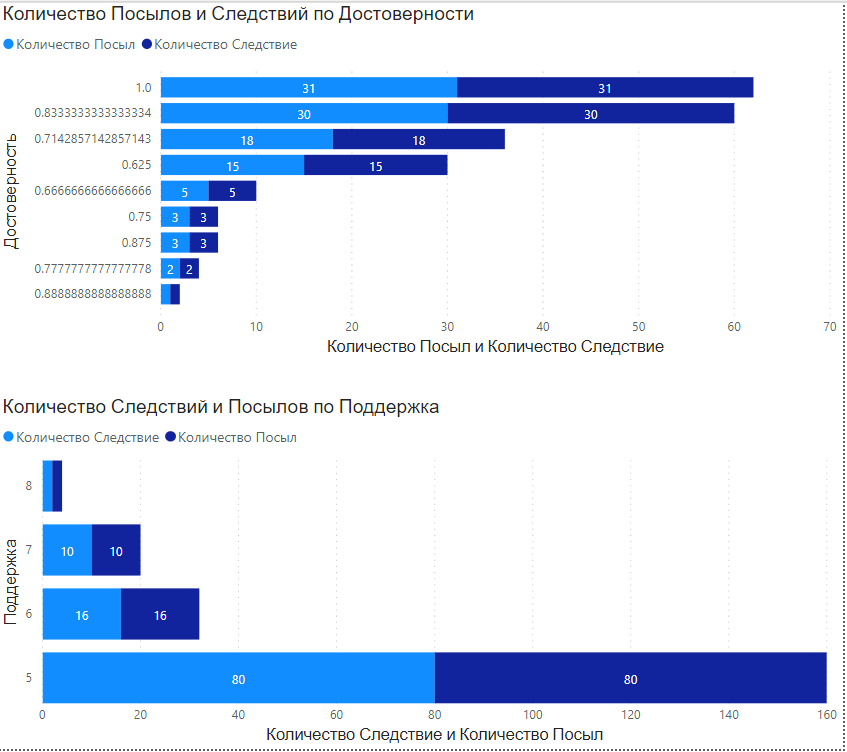


Рисунок 1.9 – Диаграммы

*Для более наглядного понимания можно открыть отчет в PBI и посмотреть отчет в интерактивном режиме.*

**Итог**: В заключении мы получили, что при покупке Apple Mac Book Air покупатели с вероятность 26,32% купят Apple iMac 27, с вероятность 36,84% - Apple iPas Mini 13 и Apple iPad Pro.

При покупке Apple iPad Pro покупатели с вероятность 38,46% купят Apple iMac 27, с вероятность 61,54% - Apple iPas Mini 13.

При покупке Apple iMac 27 покупатели с вероятность 54,55% купят Apple iPas Mini 13, с вероятность 45,45% - Apple iPad Pro

**Задание 2:** Реализовать поток работ, выполняющего решение задачи классификации посредством построения дерева решений. Данный поток должен выполнять следующую последовательность действий: загрузить данные из текстового файла, построить дерево решений, вывести результаты.

**Ход работы:**

Реализация велась на языке программирования Python при помощи дистрибутива Anaconda в Jupyter Notebook.

Для начала найдем свободный dataset и выгрузим в удобный нам формате xlsx. Далее создадим блокнот с ассоциативными правилами.

Импортируем наиболее популярные и необходимые нам библиотеки. Далее выгрузим наш dataset и выведем первые строки для проверки корректности работы.

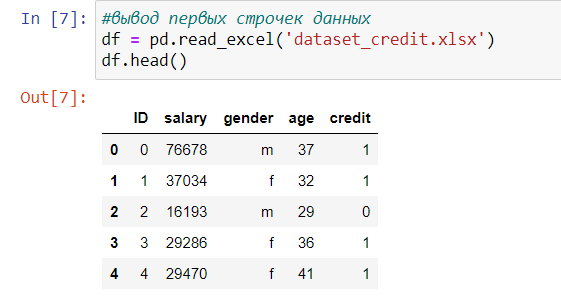


Рисунок 2.1 – Выгрузка dataset

Далее для дальнейшего анализа приведем все показатели к числовому виду. И сформируем матрицу признаков с результирующим столбцом: в качестве признаков будут выступать все показатели, кроме credit, credit будет результирующим столбцом, т.е мы будем анализировать по какому правилу выдавался кредит, согласно данному dataset.

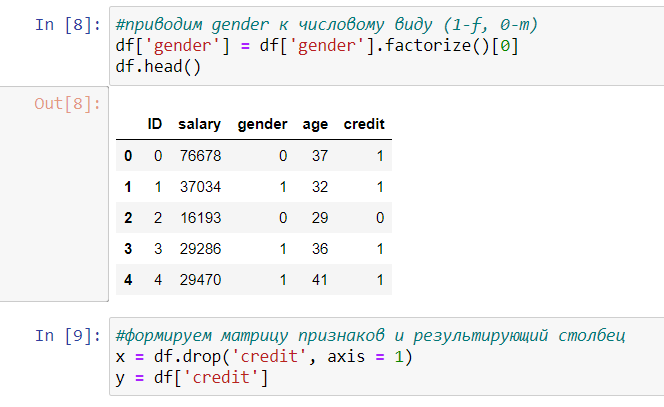


Рисунок 2.2 – Результаты работы вышеописанных шагов

После разобьем нашу выгрузку на train и test выборки в соотношении 80%/20%. Это делается для улучшения качества работы алгоритма, чтобы не происходил так называем «подгон» алгоритма по данные. Создадим дерево решений и обучим его.

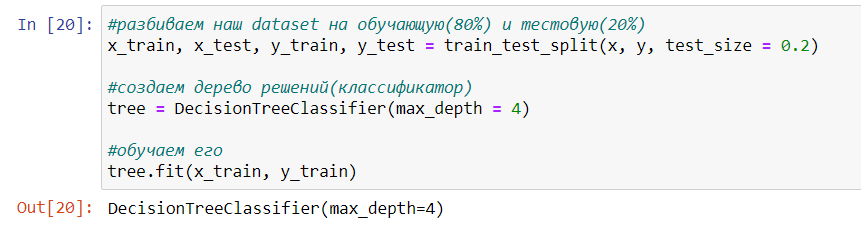


Рисунок 2.3 – Создание и обучение дерева решений

Далее делаем предсказания и проверяем работу нашего алгоритма (долю правильных ответов). Как видно из ответов у нас очень высокая вероятность получения точного/правильного ответа.

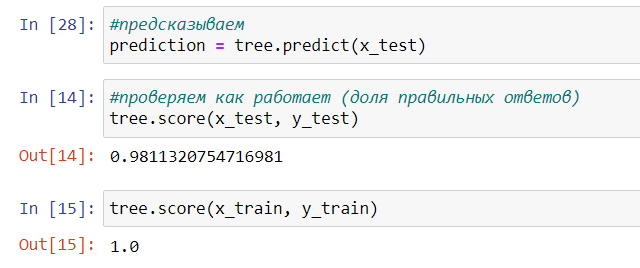


Рисунок 2.4 – Предсказание и проверка работоспособности алгоритма

И наконец записываем наше дерево в dot файл, после преобразуем его в png файл и выводим на панель.

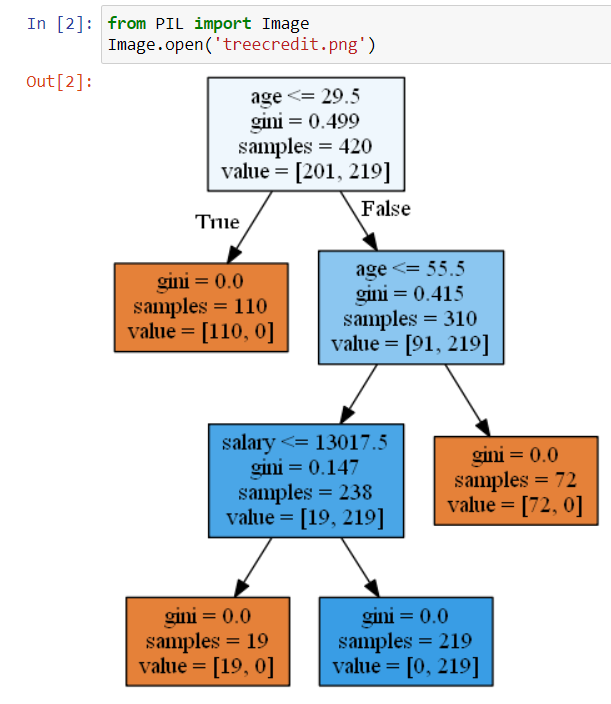


Рисунок 2.5 – Дерево решений

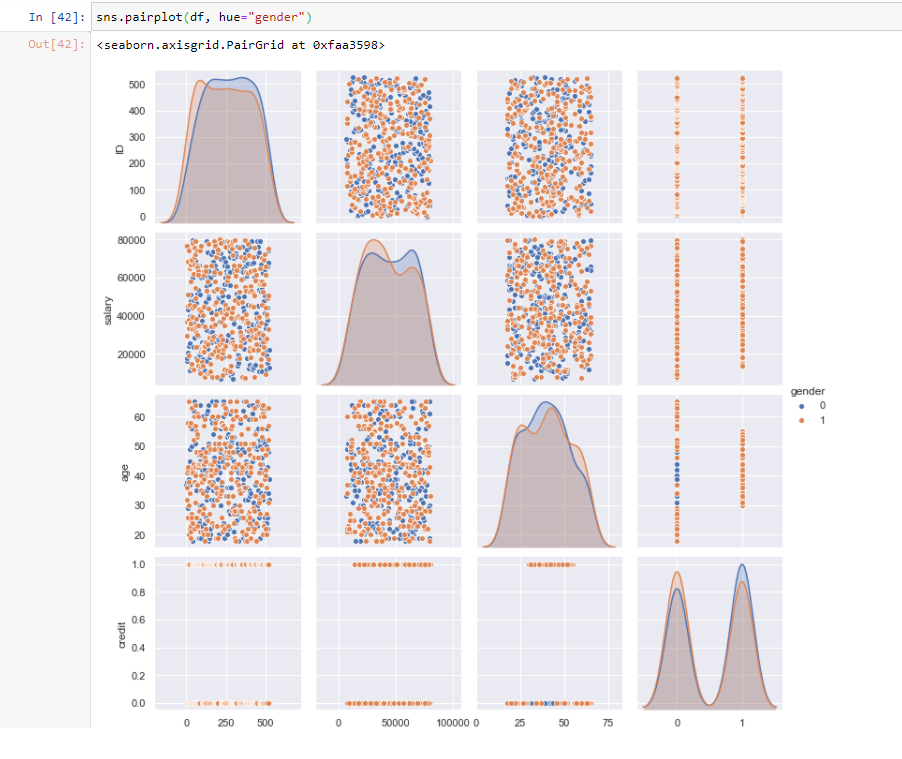
**Итог**: с помощью графа можно понять, как дерево разделяет данные на классы. Деревья решений представляют быстрый и понятный способ для решения задач классификации и регрессии. Деревья вычисляют значения для каждого поля, с помощью которых можно разделить данные на классы. При обучении деревья стремятся выбрать такое разделение, чтобы минимизировать энтропию, то есть так, чтобы каждое разделение было наиболее информативно и лучше разделяло данные. Из полученного дерева можно сделать вывод: ***в нашем примере кредит выдается лицам независимо от пола, у кого зарплата больше 13017.5 и возраст в диапазоне от 29.5 до 55.5 (с учетом погрешности), а если быть точнее, то в возрасте от 30 до 56.***

**Задание 3:** Реализовать поток работ, выполняющего решение задачи кластеризации. Данный поток должен выполнять следующую последовательность действий: загрузить данные из текстового файла, построить дерево решений, вывести результаты.

**Ход работы:**

Для решения задачи воспользуемся dataset и деревом решений из предыдущего задания.

В качестве алгоритма выберем K-Means. Сначала подключим необходимые нам библиотеки, а после попробуем разбить данные на кластеры, по признаку “gender”.



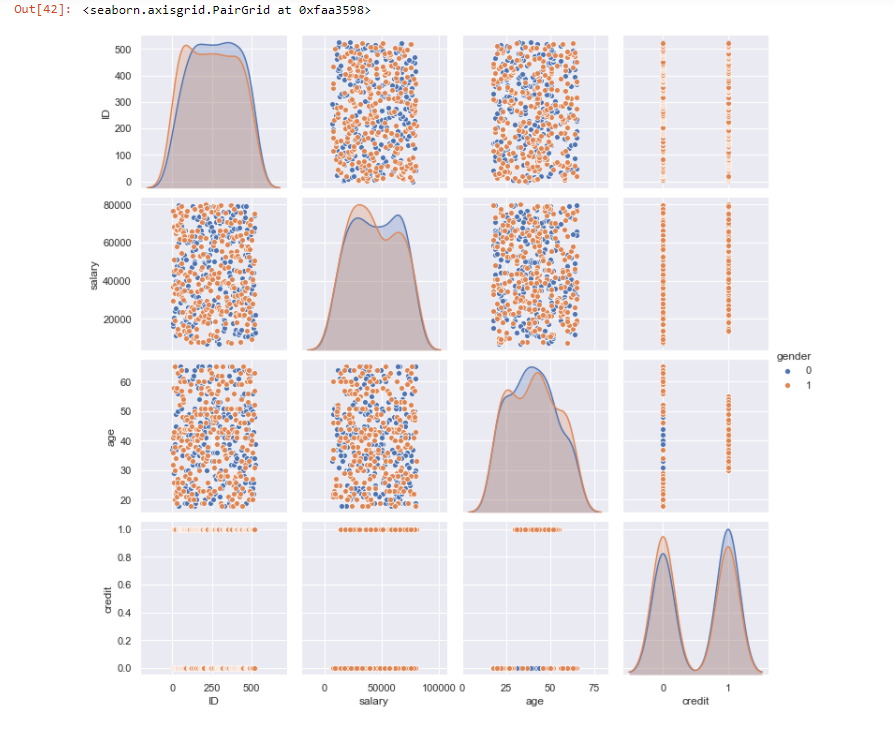


Рисунок 3.1. – Кластеризация по признаку “gender”

Отображен парный график, в котором сравниваются все колонки исходных данных. Цветом обозначен кластер (0-мужчина, 1-женщина). Например, из данного последнего графика (credit-gender) мы можем увидеть, что мужчинам выдали больше кредитов, чем женщинам, а также, что женщинам чаще отказывали в выдаче кредитов, чем мужчинам.

Или, например, разбить данные на кластеры по результирующему столбцу. Если посмотреть на графики (credit-age), то можно увидеть, что наши графики «терпят» небольшую погрешность: согласно графикам, правило, по которому выдается кредит – это люди в возрасте от 25 до 56. Если вспомнить наше дерево решений, то наш возраст был в диапазоне от 29.5 до 55.5.

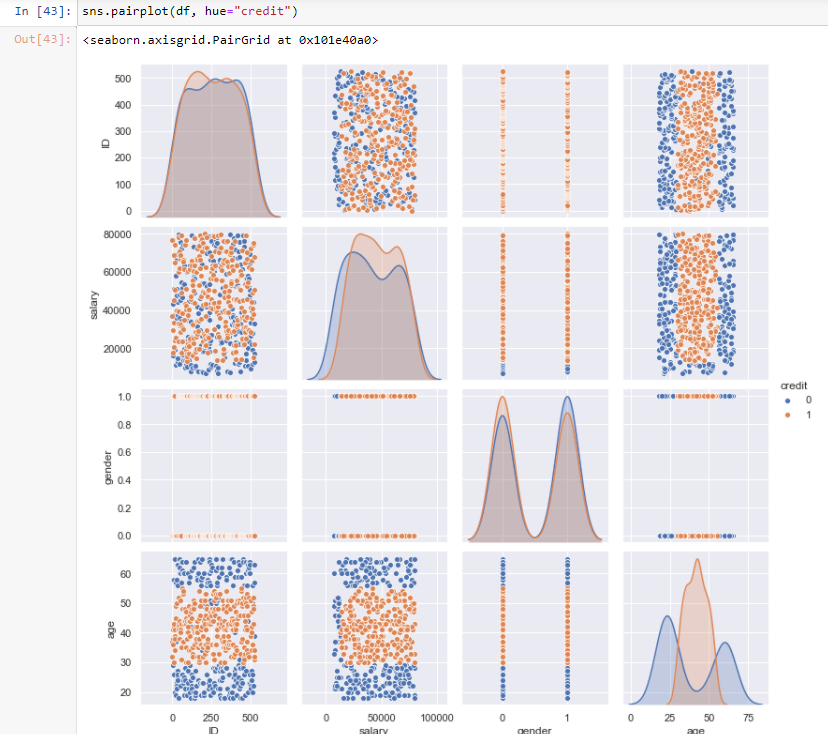


Рисунок 3.2 - Кластеризация по результирующему столбцу

***Исходные код, dataset и файлы PBI по ссылке:*** <https://github.com/ZlataKr/POIS/tree/master/Lab1>

**Вывод**: С помощью различных методов анализа данных можно выявить зависимости в данных. Например, с помощью поиска ассоциативных правил можно выяснить, какие факторы зависят друг от друга и найти зависимости. С помощью деревьев решений можно узнать, по каким правилам можно разделять или предсказывать данные. С помощью кластеризации можно выявить группы схожих объектов. Также благодаря графикам можно делать различные интересующие нас выводы.

**Список литературы**

1. Себастьян Рашка. Python и машинное обучение / - М.: Изд-во Вильямс, 2019
2. Петер Флах. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных / - М.: Изд-во ДМК Пресс, 2019
3. IBM. Обработка данных и машинное обучение / URL: <https://www.ibm.com/ru-ru/analytics/machine-learning>