Лабораторная работа 4. Кластерный анализ

1. Цели

Приобрести навыки построения модели для кластеризации

2. Задачи

1. Построить модель для кластеризации методом k-средних при обучения без учителя
2. Интерпретировать результат работы модели

3. Теоретические сведения

Методические указания для решения поставленного задания

3.1. Кластеризация

Кластерный анализ – множество алгоритмов, используемых для классификации, в результате работы которых образуются группы объектов имеющие некоторые сходства. Например, с помощью модели кластеризации для 4 кластеров можно построить Бостонскую матрицу, которая поможет задать стратегию управления бизнесом. В данной работе модель кластеризации модель поделит компании на неперспективные, малоперспективные, перспективные и состоявшиеся

Задача кластеризации не имеет эталонного решения, так как:

1. Не существует однозначно наилучшего критерия качества кластеризации
2. Число кластеров при обучении без учителя не известно и определяется субъективно
3. Результат кластеризации также субъективен

Задача кластерного анализа – организация данных в наглядные структуры. Для решения этой задачи существуют следующие методы:

1. Древовидная кластеризация (англ. Tree clustering)
2. Метод k-средних (англ. K means clustering)
3. Двухвходовое объединение (англ. Two-way joining)

Метод k-средних – кластеризация на основе прототипов. Это означает, что каждый кластер представлен прототипом, который может быть либо центроидом (средним) подобных точек с непрерывными признаками, либо медоидом (наиболее представительной или наиболее часто встречающейся точкой) в случае категориальных признаков.

При неудачном выборе числа кластеров или неподходящем задании начального положения центроидов кластеров результат кластеризации может быть сомнительным. Кроме того, этот метод чувствителен к качеству данных: выбросы, аномальные наблюдения, шум в данных могут также ухудшить итоговый результат.

Алгоритм метода k-средних:

1. Выбрать количество кластеров
2. Произвольным образом расположить в пространстве данных центроиды
3. Определить ближайший центроид для каждой точки набора данных
4. Для полученных кластеров найти новое положение центроида

Пункты 3 и 4 повторяются либо фиксированное количество раз, либо до момента, когда смещение новых центров кластеров относительно предыдущей итерации будет меньше какого-либо порога

Для кластеризации объектов с непрерывными признаками обычно используют евклидово расстояние – алгоритм минимизирует суммарное квадратичное отклонение точек кластеров

от центров этих кластеров, для этого используется следующее уравнение, где $x_i^{(j)}$ – наблюдение с номером i, которое отнесено к кластеру j, c_j – центроид кластера j, k – число кластеров, n – число наблюдений

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left(x_i^{(j)} - c_j \right)^2$$

При кластеризации методом k-средних количество кластеров чаще всего оценивают с помощью «метода локтя» (англ. Elbow method) – интуитивной и довольно грубой эвристики.

На графике откладывается некоторая величина, характеризующая качество кластеризации, например, внутри-кластерная сумма расстояний J для разного количества кластеров. В прикладных программах для реализации метода k-средних нужная величина – это искажение или инерция, например в методе `KMeans` библиотеки `sklearn` – это значение атрибута `inertia_`). Оптимальное количество кластеров соответствует значению k, после которого величина J перестает резко падать. Лучший вариант $J = 0$ достигается при количестве кластеров, совпадающих с числом наблюдений. Но в действительности обычно нужны стабильные кластеры с большим количеством наблюдений, для которых можно выявить закономерности.

3.2. Построение модели с помощью `scikit-learn`

Пример построения модели будет приведён ниже, демонстрация будет производится на наборе Unicorn Companies Dataset (рус. Набор данных о компаниях-единорогах)

3.2.1. Подключение библиотек

Подключение библиотек

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.cluster import KMeans
```

3.2.2. Работа с набором данных

Кластеризация будет проводится на части набора, где компании относятся либо к сфере `Artificial intelligence`, либо `Data management & analytics`, также будут удалены столбцы, не несущие полезной информации. Будут заменены отсутствующие значения, исправлена разметка данных для того, чтобы их могла воспринимать модель. Столбцы `Country` и `Financial Stage` будут факторизованы, для них создадутся вспомогательные таблицы, для сопоставления числовых и строковых значений

```
PATH = "unicorn.csv"
INDUSTRIES = ["Artificial intelligence", "Data management & analytics"]
COLUMNS_FOR_FACTORISATION = ["Country", "Financial Stage"]
NUMERIC_COLUMNS = ["Total Raised", "Valuation", "Founded Year", "Financial Stage", "Investors
Count", "Deal Terms"]
NONE_REPLACEMENT = {
    "Total Raised": "$0M",
    "Founded Year": "2000",
    "Deal Terms": "0",
    "Investors Count": "0"
}

dataset = pd.read_csv(PATH)
```

```
factorization_table = {}

dataset = dataset.loc[
    dataset['Industry'].isin(
        INDUSTRIES
    )
]

dataset = dataset[
    [
        "Company",
        "Country",
        "Founded Year",
        "Valuation ($B)",
        "Total Raised",
        "Financial Stage",
        "Investors Count",
        "Deal Terms",
    ]
]

dataset.rename(
    columns = {"Valuation ($B)": "Valuation"},
    inplace=True,
)

for column in dataset.columns:
    if column in NONE_REPLACEMENT.keys():
        dataset[column].replace("None", NONE_REPLACEMENT[column], inplace=True)

    if column == "Valuation":
        for index in dataset.index:
            dataset.at[index, column] = float(dataset.at[index, column][1:])

    if column == "Total Raised":
        for index in dataset.index:
            value = dataset.at[index, column][1:]
            if value[-1] == "B":
                dataset.at[index, column] = value[:-1]
            elif value[-1] == "M":
                dataset.at[index, column] = float(value[:-1]) / 1000
            elif value[-1] == "K":
                dataset.at[index, column] = float(value[:-1]) / 1000000

    if column in COLUMNS_FOR_FACTORISATION:
        dataset[column], table = pd.factorize(dataset[column])
        factorization_table[column] = pd.DataFrame(
            columns=[column],
            data=table
        )

    if column in NUMERIC_COLUMNS:
        dataset[column] = pd.to_numeric(dataset[column])

dataset.index = [index for index in range(len(dataset))]

dataset
```

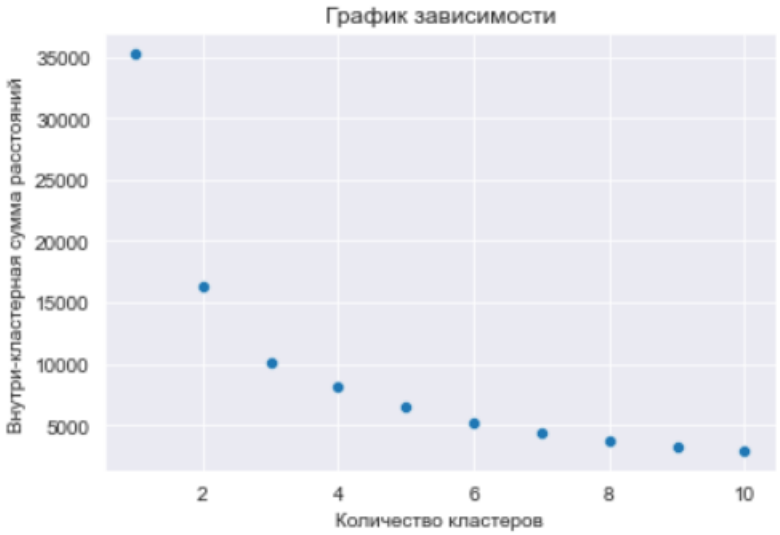
3.2.3. Построение модели

Для выбора нужного количества кластеров будет построен график зависимости инерции от количества кластеров.

```
inertia = []

for i in range(1, 11):
    k_means = KMeans(n_clusters=i, init= 'k-means++')
    k_means.fit(
        dataset.drop(
            "Company",
            axis=1,
        )
```

```
    )
    inertia.append(k_means.inertia_)

sns.set_style('darkgrid')
sns.scatterplot(
    x=[x for x in range(1, 11)],
    y=inertia,
)

plt.title('График зависимости')
plt.xlabel('Количество кластеров')
plt.ylabel('Внутри-кластерная сумма расстояний')
```

График зависимости



Оптимальное количество кластеров, на основе графика – 4

Код для построения модели, делящей данные на 4 кластера. Также код предоставляет средние значения кластеров и количество вхождений в них

```
CLUSTERS = 4

model = KMeans(
    n_clusters=CLUSTERS
)

model.fit(
    dataset.drop(
            "Company",
            axis=1,
        )
)

clusters = pd.DataFrame(
    columns=dataset.columns.drop("Company"),
    data=model.cluster_centers_
)

clusters["Amount"] = np.unique(
    model.labels_,
    return_counts=True
)[1]

clusters
```

Получена следующая таблица (таблица будет меняться каждый раз, даже на одинаковых данных, из-за случайного выбора начальных значений центроидов)

	Country	Founded Year	Valuation	Total Raised	Financial Stage	Investors Count	Deal Terms	Amount
0	0.000000	2012.000000	140.000000	7.440000	0.000000	28.000000	8.000000	1
1	2.214286	2013.869048	2.312143	0.378626	1.035714	11.595238	2.857143	84
2	0.750000	2013.250000	5.100625	0.968912	1.000000	31.937500	4.125000	16
3	4.727273	2000.000000	2.581818	0.467004	1.000000	7.181818	2.545455	11

Вспомогательные таблицы с данными до факторизации вызываются командой

```
factorization_table["Название столбца"]
```

4. Задание

Выбрать с сайта kaggle.com набор данных в формате .csv, пригодный для построения модели кластеризации, загрузить и подготовить его к дальнейшей обработке. Если набор уже имеет метки классов – удалить метки. Наборы данных не должны повторяться внутри группы. Задание индивидуальное. Требования:

1. Построить модель кластеризации методом k-средних
2. Определить оптимальное количество кластеров, обосновать своё решение
3. Получить центроиды кластеров и количество вхождений в кластеры
4. Интерпретировать каждый кластер
5. Указать какие знания можно получить из набора
6. Сохранить IPython Notebook

4.1. Продвинутое задание

Построить вторую модель, без использования средств scikit-learn