Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт математики и механики им. Н. И. Лобачевского

Кафедра теории функций и приближений

Направление подготовки: 01.03.01 – «Математика»

Профиль: Математика в цифровой экономике

КУРСОВАЯ РАБОТА

Применение AI при решении задач биоинформатики

Студент 3 курса

группы 05-204

«__» _____ 2025 г. _____ Ф. И. Фаррахов

Научный руководитель:

доцент, к.ф.м.н.

«__» _____ 2025 г. _____ Р. К. Губайдуллина

Казань – 2025

СОДЕРЖАНИЕ

Введение

Современная наука о веществах и материалах немыслима без использования методов компьютерного моделирования и анализа больших массивов данных. С развитием компьютерных технологий и растущим объемом информации, получаемой при изучении химических соединений, возникла необходимость в создании инструментов, способных систематизировать и интерпретировать эти данные. Одним из таких инструментов стала кластеризация.

В химии кластеризация играет особую роль, поскольку позволяет исследовать химическое пространство - совокупность всех возможных химических соединений. Объем этого пространства невообразимо велик: количество потенциально существующих молекул больше 10^{60}. Очевидно, что при таком количестве каждое отдельное соединение не поддается прямому изучения и сравнению, поэтому в современной химии стали активно внедряться и применяться методы, способные выявлять закономерности и структурировать данные - именно это и обеспечивает кластеризация.

Кластеризация химических веществ имеет широкое практическое значение. В фармацевтической промышленности она используется для поиска новых лекарственных соединений, когда необходимо выделить группы структурно похожих молекул и определить среди них перспективные кандидаты. В области материаловедения кластеризация помогает идентифицировать вещества с аналогичными свойствами, что ускоряет поиск новых катализаторов, полимеров или наноматериалов. Кроме того, этот подход применяется в экологической химии и токсикологии для выявления групп веществ, обладающих сходным воздействием на живые организмы или окружающую среду.

Таким образом, кластеризация химических веществ является мощным инструментом, позволяющим систематизировать огромные массивы химической информации, выявлять скрытые взаимосвязи между структурой и свойствами соединений, а также прогнозировать поведение новых веществ.

Цель работы - рассмотреть основные методы кластеризации, особенности представления химических данных, а также применить методы машинного обучения к задаче группирования веществ по заданным признакам и по химическому представлению формулы вещества.

В этой работе были рассмотрены основы кластерного анализа, методы кластеризации и их применение к конкретным химическим данным. На основе набора соединений, представленных в формате SMILES, были рассчитаны молекулярные дескрипторы, проведена их обработка, нормализация и снижение размерности с помощью алгоритмов PCA и UMAP. Затем были применены различные алгоритмы кластеризации (K-Means, DBSCAN и др.), выполнена их оценка по метрикам качества и проведен анализ полученных результатов.

1. Теоретическая часть

1.1. Понятие кластеризации

Кластеризация (или кластерный анализ) - это совокупность методов многомерной статистики и машинного обучения, предназначенных для автоматического разделения набора объектов на группы (кластеры) таким образом, чтобы объекты внутри кластера были максимально похожи между собой и существенно отличались между разными кластерами.

Кластеризацию следует отличать от классификации, хотя обе эти процедуры очень похожи и относятся к методам анализа данных.

Основное различие заключается в том, что:

- Классификация - это метод обучения с учителем, при котором заранее известны категории (классы), и задача алгоритма состоит в отнесении новых объектов к одному из предопределенных классов на основе обучающей выборки.

  Пример: классификация химических веществ на "токсичные" и "нетоксичные" в соответствии с предварительно помеченными данными.
- Кластеризация, напротив, относится к методам обучения без учителя: алгоритм не имеет информацией о предопределенных группах и самостоятельно выявляет естественные объединения в данных, основываясь на внутренней структуре и степени сходства объектов.

  Пример: автоматическое выделение групп молекул со схожей структурой или физико-химическими параметрами без предварительного знания их типа или функции.

Кластеризации имеет множество практических применений, но, если обобщить, то кластеризация позволяет:

- находить естественные группы или скрытые закономерности в наборах данных;
- упрощать последующую обработку информации за счёт представления множества объектов через ограниченное число типичных групп;
- визуализировать многомерных данных;
- формировать основу для классификации, прогнозирования и моделирования, когда заранее неизвестны категории или структура данных.

В общем виде процесс проведения кластерного анализа включает следующие основные этапы:

1. Формирование выборки объектов для кластеризации
2. Выбор набора признаков (переменных), по которым будет оцениваться степень сходства объектов; при необходимости выполняется нормализация данных для устранения влияния масштаба измерений.
3. Вычисление значений меры сходства между объектами.
4. Применение выбранного алгоритма кластеризации для создания групп сходных объектов (кластеров).
5. Интерпретация и визуализация полученных результатов, то есть анализ структуры сформированных кластеров и их характеристик.

1.2. Метрики определения схожести объектов

Чтобы определить сходство объектов нужно, во-первых, выделить вектор характеристик для каждого объекта - как правило, это набор числовых значений (рост, вес, координаты, ...). Впрочем, существуют и алгоритмы, работающие с качественными характеристиками (цвет, форма, статус, ...)

При большом количестве характеристик, процесс кластеризации происходит довольно медленно, и его результаты не всегда приемлемы. Поэтому такую размерность характеристик нужно стараться сокращать, оставляя наиболее важные свойства объектов.

После отбора характеристик выполняется их нормализация, что необходимо для обеспечения сопоставимости значений различных параметров. Нормализация заключается в приведении каждого вектора признаков к единому масштабу, например, к диапазону [0;1] или [-1;1], в зависимости от специфики задачи. Это позволяет избежать доминирования признаков с большими числовыми диапазонами и улучшить качество кластеризации.

После того как мы нашли вектора характеристик необходимо выбрать функцию для определения степени сходства двух объекта, называемую мерой расстояний (или метрикой)

Существуют множество метрик для вычисления схожести объектов. Обозначив u, v объекты, между которыми вычисляется расстояние $\mathrm{d}(\mathrm{u},\mathrm{v})$ и u_i, v_i - их координаты, опишем основные из существующих метрик:

1. Евклидово расстояние

Наиболее распространенная функция расстояния, являющаяся геометрическим расстоянием в многомерном пространстве:

$$\mathrm{d}(u,v) = \sqrt{\sum_{i}^{n}(u_i - v_i)^2}$$

2. Квадрат евклидова расстояния

Применяется для придания большего веса более отдаленным друг от друга объектам. Вычисляется следующим образом:

$$d(u,v) = \sum_{i}^{n}(u_i - v_i)^2$$

3. Расстояние городских кварталов или манхэттенское расстояние

Вычисляется как средняя разность по координатам и чаще всего приводит к результатам аналогичным обычному расстоянию Евклида

$$\mathrm{d}(u,v) = \sum_{i}^{n} |u_i - v_i|$$

4. Степенное расстояние

Применяется в случае, когда необходимо увеличить или уменьшить вес, относящийся к размерности, для которой соответствующие объекты сильно отличаются

$$\mathrm{d}(u,v) = \left(\sum_{i}^{n} (u_i - v_i)^p \right)^{1/q}$$

где q и p – параметры, определяемые пользователем

5. Расстояние Чебышева

Это расстояние используется, когда нужно определить два объекта как «различные», если они различаются по какой-либо одной координат

$$\mathrm{d}(u,v) = max(|u_i - v_i|)$$

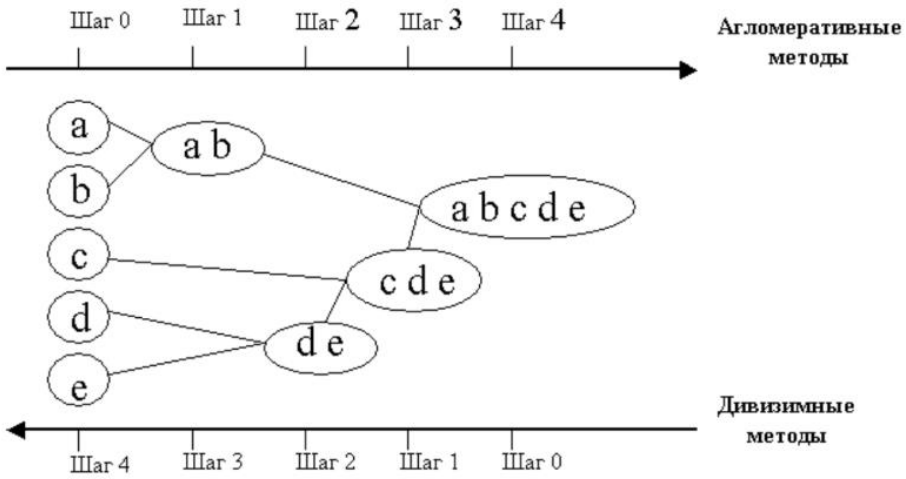
1.3. Методы кластеризации

Методы кластерного анализа можно разделить на две группы - иерархические и неиерархические. Каждая из них включает множество подходов и алгоритмов, отличающихся способом формирования кластеров и критериями их объединения или разделения.

Иерархические методы кластеризации

Суть иерархической кластеризации состоит в последовательном объединении меньших кластеров в большие, или разделении больших кластеров

на меньшие. Следовательно, эти методы можно разделить на две группы: агломеративные и дивизивные



а) Агломеративные методы.

Эти алгоритмы работают по принципу «снизу вверх». На начальном этапе каждый объект рассматривается как отдельный кластер. Затем последовательно объединяются наиболее близкие (похожие) объекты или группы объектов, в результате чего количество кластеров постепенно уменьшается. Процесс продолжается до тех пор, пока все элементы не будут объединены в один общий кластер.

б) Дивизивные (делимые) методы.

Эти методы являются логической противоположностью агломеративным методам. В начале работы алгоритма все объекты принадлежат одному кластеру, который на последующих шагах делится на меньшие кластеры, В результате образуется последовательность расщепляющих групп.

Иерархические методы кластеризации различаются правилами построения кластеров, которые определяются выбранным критерием "схожести" между объектами.

Когда каждый объект представляет собой отдельный кластер, расстояния между этими объектами определяются выбранной мерой. Возникает следующий вопрос как определить расстояния между кластерами? Существуют различные правила, называемые методами объединения или связи для двух кластеров.

Метод ближайшего соседа. В данном способе расстояние между двумя кластерами определяется расстоянием между двумя наиболее близкими объектами в различных кластерах.

Метод наиболее удаленных соседей. Расстояние между кластерами определяется наибольшим расстоянием между любыми двумя объектами в различных кластерах (т.е. “наиболее удаленными соседями”). Данный способ, как правило, работает довольно хорошо, если кластеры имеют форму близкую к сферической. Если же кластеры являются «цепочечными» или имеют удлиненную форму, то этот метод непригоден.

Метод невзвешенного попарного среднего. В качестве расстояния между двумя кластерами берется среднее расстояние между всеми парами объектов в них.

Метод взвешенного попарного среднего. Этот метод схож с методом невзвешенного попарного среднего, однако в нем, при вычислении расстояний учитывается размер соответствующих кластеров, (т.е. число объектов, содержащихся в них), и используется в качестве весового коэффициента. В связи с этим данный метод необходимо использовать, если ожидаются кластеры, значительно отличающиеся по размерам.

Метод минимизация внутрикластерной дисперсии (Ward’s method). В отличие от других методов, которые определяют расстояние между кластерами по внешним признакам – ближайшие или самые дальние точки, среднее всех точек, метод Уорда делает иначе. Он объединяет те кластеры, чье слияние минимально увеличит разброс точек внутри кластеров (внутрикластерную дисперсию). Является наиболее популярным при иерархической кластеризации, т.к. по сравнению с другими наименее болезненно увеличивает дисперсию при объединении двух кластеров.

Неиерархические методы кластеризации

Неиерархические методы (или плоские) представляют собой класс алгоритмов кластеризации, в которых данные разделяются на заранее заданное число кластеров без построения иерархической структуры. В отличие от иерархических подходов, которые формируют древовидное представление вложенных групп, неиерархические алгоритмы сразу создают фиксированное разбиение множества объектов на несколько непересекающихся подмножеств.

Такие методы широко применяются при анализе больших наборов данных, где важна вычислительная эффективность и возможность уточнять параметры модели.

Наиболее известными среди них являются алгоритм K-Means и DBSCAN

Алгоритм K-Means (метод k-средних)

$$e^2(X) = \sum_{j=1}^{k} \sum_{i=1}^{n_j} \left\| x_i^{(j)} - c_j \right\|^2 ,$$

Один из наиболее распространённых и простых неиерархических методов Он относится к числу чётких алгоритмов, поскольку каждый объект однозначно относится к одному кластеру. Основная идея заключается в том, чтобы минимизировать суммарное квадратичное отклонение объектов от центров (средних значений) своих кластеров:

где X – множество объектов x^j , x_i^j - их координаты, c_j – «центр масс» кластера j (считая массу каждой точки равной единице).

Метод k-средних считается наиболее популярным в этой категории ввиду того, что алгоритм разбивает заданное множество объектов на указанное число кластеров, расположенных на как можно большем расстоянии друг от друга. Работа этого метода разбивается на несколько этапов:

1) Инициализация: случайным образом выбираются k начальных «центров масс»

2) Распределение объектов: каждый объект относится к тому кластеру, центр которого находится на минимальном расстоянии.
3) Пересчёт центров: для каждого кластера вычисляется новое положение центра как среднее значение координат всех входящих в него объектов.
4) Проверка условия сходимости: если критерий остановки не выполнен, процесс распределения и пересчёта повторяется.

В качестве критерия остановки работы алгоритма как правило используют минимальное изменение среднеквадратической ошибки. Также работа алгоритма завершается, если на шаге 2 не было объектов, сменивших свой кластер.

Алгоритм DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN относится к методам кластеризации на основе плотности. В отличие от K-Means, он не требует заранее задавать число кластеров и способен выявлять кластеры произвольной формы, а также отделять шумовые точки, которые не принадлежат ни одной группе.

Основные параметры DBSCAN:

- ε - радиус окрестности
- min_samples - минимальное количество точек, необходимое для образования кластера.

Алгоритм работы DBSCAN:

1) Для каждой точки определяется её ε-окрестность - множество всех объектов, расстояние до которых не превышает заданного значения ε
2) Если число точек в окрестности ≥ min_samples, то данная точка считается ядровой и становится основой нового кластера.

3) Все точки, находящиеся в окрестности ядра, присоединяются к кластеру. Если среди них есть другие ядровые точки, их окрестности также добавляются — кластер «растёт».
4) Процесс продолжается, пока все точки, достижимые из исходного ядра, не будут включены в кластер.
5) Точки, которые не входят ни в одну ε-окрестность ядра, считаются шумом или аномальными наблюдениями.

Результатом работы DBSCAN является разбиение множества объектов на кластеры различной плотности и форму, а также выделение точек, не принадлежащих ни одной группе.

2. Практическая часть

В этой части будут рассмотрены основные методы кластеризации на конкретном примере

Для работы был использован язык программирования Python 3.12.4, и все библиотеки, примененные далее, используются для выбранного языка

2.1. Анализ и подготовка данных для кластеризации

Для проведения кластерного анализа использовался набор данных, содержащий информацию о химических соединениях и их физических, химических и биологических свойствах

In [7]:
```
# Загрузка таблицы
file_path = 'data/SMILES_Big_Data_Set.csv'
df = chunk_read_csv(file_path, chunk_size=200)
df
```

Out[7]:

	SMILES	pIC50	mol	num_atoms	logP
0	O=S(=O)(Nc1cccc(-c2cnc3ccccc3n2)c1)c1cccs1	4.26	<rdkit.Chem.rdchem.Mol object at 0x7f59df45bc30>	25	4.15910
1	O=c1cc(-c2nc(-c3ccc(-c4cn(CCP(=O)(O)O)nn4)cc3)...	4.34	<rdkit.Chem.rdchem.Mol object at 0x7f59a320c9e0>	36	3.67430
2	NC(=O)c1ccc2c(c1)nc(C1CCC(O)CC1)n2CCCO	4.53	<rdkit.Chem.rdchem.Mol object at 0x7f59a320cac0>	23	1.53610
3	NCCCn1c(C2CCNCC2)nc2cc(C(N)=O)ccc21	4.56	<rdkit.Chem.rdchem.Mol object at 0x7f59a320cba0>	22	0.95100

Данные были представлены в формате таблицы (.csv) и содержали такие данные:

SMILES – текстовое представление химической структуры (Simplified Molecular Input Line Entry System)

pIC50 - это показатель, используемый в фармакологии и разработке лекарств для оценки эффективности соединения в ингибировании конкретной биологической мишени или фермента

mol - это представление молекулы в библиотеке RDKit, которая является широко используемым программным обеспечением для работы с молекулярными данными. В данном случае это означает, что столбец «mol» содержит объекты RDKit Mol.

num_atoms - количество атомов в молекуле. В контексте хемоинформатики и анализа молекулярных данных атрибут «num_atoms» представляет собой количество атомов в молекуле. Атомы — это фундаментальные частицы, из которых состоят молекулы. Это могут быть такие элементы, как углерод (C), водород (H), кислород (O), азот (N) и многие другие. Количество атомов в молекуле — важное свойство, которое позволяет судить о её размере, сложности и возможных химических взаимодействиях.

logP - это показатель того, как молекула взаимодействует с различными растворителями. Он указывает на то, что молекула предпочитает неполярную (маслянистую) или полярную (водорастворимую) среду. Более высокое значение logP означает, что молекула с большей вероятностью растворится в масле и с меньшей вероятностью растворится в воде. Это свойство важно при разработке лекарств, так как оно влияет на способность молекулы всасываться и распределяться в организме

Таким образом, каждая строка представляет отдельное химическое соединение с набором его структурных и физико-химических свойств.

Дальше из данных были убраны пустые значения pIC50 (всего их было 1050)

Проверка на уникальность

```
duplicates = df[df.duplicated(subset=['SMILES'], keep=False)]
duplicates_sorted = duplicates.sort_values(by='SMILES')
print(f"Кол-во неуникальных SMILES: {duplicates_sorted['SMILES'].unique().shape[0]}")
duplicates_sorted.shape
```

```
Кол-во неуникальных SMILES: 179
(393, 5)
```

Потом для чистоты данных были убраны ненужные дубликаты веществ. Покажем, как это было:

Всего в этой таблице 395 веществ, имеющих дубликаты SMILES с другими значениями в остальных столбцах

Давайте посмотрим, какие значения различаются в этих столбцах

```
SMILES с противоречиями в pIC50: 153
SMILES с противоречиями в num_atoms: 0
SMILES с противоречиями в logP: 0
SMILES с противоречиями в mol: 179
```

Из этого вывода следует, что в данных дублируются только значения в столбцах pIC50 и mol и видно, что дубликатов с столбце mol больше.

Теперь поближе как выглядят эти вещества:

```
df[df['SMILES'] == 'CC(=O)N1CCC(Nc2ncccc2-c2cnc3[nH]ccc3n2)C1']
```

	SMILES	pIC50	mol	num_atoms	logP
1865	CC(=O)N1CCC(Nc2ncccc2-c2cnc3[nH]ccc3n2)C1	10.03	<rdkit.Chem.rdchem.Mol object at 0x7f59a2966a40>	24	2.0526
1889	CC(=O)N1CCC(Nc2ncccc2-c2cnc3[nH]ccc3n2)C1	10.40	<rdkit.Chem.rdchem.Mol object at 0x7f59a29674c0>	24	2.0526

У этого вещества все свойства совпадают, кроме значений в столбцах pIC50 и mol

Посмотрим теперь на другое вещества, у которых отличаются значения в столбце mol

```
# Тут совпадает все, кроме значений в столбце 'mol'
duplicates_sorted[duplicates_sorted['SMILES'] == 'CCCC(=O)O']
```

	SMILES	pIC50	mol	num_atoms	logP
3479	CCCC(=O)O	0.13	<rdkit.Chem.rdchem.Mol object at 0x7f59a2993290>	6	0.8711
2966	CCCC(=O)O	0.13	<rdkit.Chem.rdchem.Mol object at 0x7f59a2985070>	6	0.8711

Т.к. столбец mol в дальнейшем никак не участвует в кластеризации, то мы можем удалить лишние дубликаты

Для проведения дальнейших вычислений химические структуры необходимо было преобразовать из текстового формата SMILES в объект молекул библиотеки RDKit. Эти объекты позволять нам вычислить химические дескрипторы, такие как молекулярная масса, число водородных доноров и акцепторов и др. (с помощью библиотек RDKit и mordred)

```
# Преобразование SMILES в RDKit Mol объекты
df["nmol"]            = df["SMILES"].apply(Chem.MolFromSmiles)
df["MolecularWeight"] = df["nmol"].apply(lambda mol: Descriptors.MolWt(mol))
df["HBD"]             = df["nmol"].apply(lambda mol: rdMolDescriptors.CalcNumHBD(mol))
df["HBA"]             = df["nmol"].apply(lambda mol: rdMolDescriptors.CalcNumHBA(mol))
df["TPSA"]            = df["nmol"].apply(lambda mol: Descriptors.TPSA(mol))
df["RotatableBonds"]  = df["nmol"].apply(lambda mol: Descriptors.NumRotatableBonds(mol))
df["NumRings"]        = df["nmol"].apply(lambda mol: Descriptors.RingCount(mol))
```

nmol	MolecularWeight	HBD	HBA	TPSA	RotatableBonds	NumRings
<rdkit.Chem.rdchem.Mol object at 0x00000299C0C...	367.455	1	5	71.95	4	4
<rdkit.Chem.rdchem.Mol object at 0x00000299C0C...	506.434	4	6	149.78	7	5
<rdkit.Chem.rdchem.Mol object at 0x00000299C0C...	317.389	3	5	101.37	5	3

```
# Вычисление дескрипторов с помощью mordred

chunk_size = 500
num_chunks = len(df) // chunk_size + 1
mordred_descs = []
for i in range(0, num_chunks):
    mordred_descs.append(calc.pandas(df.nmol[chunk_size*i : chunk_size*(i + 1)], quiet=True, ipynb=True))
    print(f"Посчитан чанк {i} из {num_chunks}, размер чанка {chunk_size}")
mordred_desc = pd.concat(mordred_descs)
```

```
mordred_desc
```

	nAcid	nBase	SpAbs_A	SpMax_A	SpDiam_A	SpAD_A	SpMAD_A	LogEE_A	VE1_A	VE2_A	...	SRW10	TSRW10
0	0	0	32.948198	2.414628	4.821014	32.948198	1.317928	4.168672	4.719023	0.188761	...	10.193055	73.227130
1	2	0	46.394088	2.482610	4.833785	46.394088	1.288725	4.525955	4.720563	0.131127	...	10.490746	88.575845
2	0	0	29.582533	2.502688	4.837103	29.582533	1.286197	4.066543	3.989500	0.173457	...	10.051520	71.685248
3	0	2	28.780389	2.500563	4.830733	28.780389	1.308200	4.022884	3.920684	0.178213	...	9.990995	70.411294
4	0	0	27.864278	2.395853	4.791705	27.864278	1.326870	3.974928	4.048018	0.192763	...	9.874419	54.743813
...	...	...	...	...	...	...	...	...	...	...	...	...	...
15000	0	0	31.739988	2.341548	4.683096	31.739988	1.322500	4.092013	4.062478	0.169270	...	9.790487	57.865372
15001	0	0	24.242034	2.158461	4.316922	24.242034	1.275897	3.796717	3.156898	0.166153	...	8.837681	49.767777
15002	0	0	21.685508	2.158449	4.316897	21.685508	1.275618	3.688767	3.145350	0.185021	...	8.761707	47.292351
15003	0	0	29.801196	2.458468	4.785373	29.801196	1.354600	4.051069	4.297256	0.195330	...	10.034560	72.337257
15004	0	1	15.816713	2.292674	4.386637	15.816713	1.216670	3.446765	3.080930	0.236995	...	8.873048	56.507269

15005 rows × 1088 columns

В результате проделанных шагов исходные данные были очищенны от дубликатов и пропусков, также были вычислили новые свойства. Однако, их оказалось слишком много (больше 1000), большинство из которых только усложнят вычисления без улучшения качества кластеризации. Поэтому далее, для подготовки данных к кластеризации, было проведено сокращение размерности. Для этого использовались такие методы, как удаление высококоррелированных, низкодисперсных (т.е. почти констант) признаков, метод PCA и UMAP. Вкратце про методы PCA и UMAP

Метод главных компонент (PCA):

PCA – это линейный метод снижения размерности. Он преобразует исходные данные в новую систему координат (главные компоненты):

Первая главная компонента - это направление, вдоль которого разброс данных максимален. Она содержит максимальную информацию о различиях в данных. Вторая главная компонента перпендикулярна первой и содержит вторую по значимости часть вариативности, и т. д. Каждая следующая компонента объясняет всё меньшую часть общей дисперсии данных.

Метод UMAP (Uniform Manifold Approximation and Projection):

При снижении размерности UMAP сначала выполняет построение взвешенного графа, соединяя ребрами только те объекты, которые являются ближайшими соседями. Множество из ребер графа — это нечёткое множество с функцией принадлежности, она определяется как вероятность существования ребра между двумя вершинами. Затем алгоритм создает граф в низкоразмерном пространстве и приближает его к исходному, минимизируя сумму дивергенций Кульбака-Лейблера для каждого ребра из множеств.

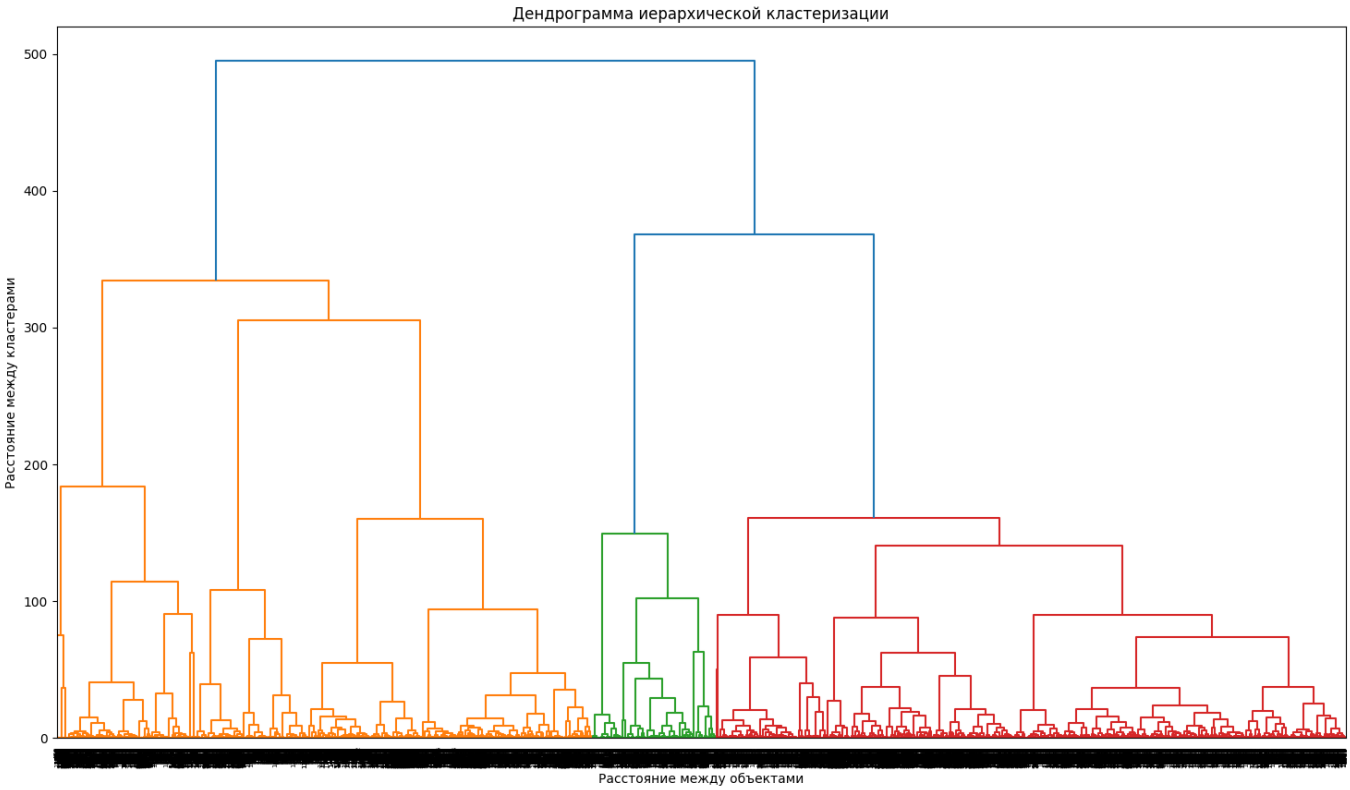
2.2. Кластеризация

Иерархическая кластеризация

После анализа и предобработки исходных данных была получена обучающая выборка размерности (15005, 10). Эти данные теперь можно использовать для кластеризации. Для выполнения этой задачи удобно применять библиотеку SciPy. В этой библиотеке реализован модуль cluster.hierarchy, представляющим набор средств для иерархической кластеризации.

Для выполнения кластеризации использовалась функция linkage(X, method, metric). Данная функция принимает следующие параметры:

- X – данные для кластеризации
- method - метод вычисления расстояния между кластерами (single, complete, average, weighted, centroid, ward)
- metric - метрика для вычисления расстояния между точками. По умолчанию – Евклидова



И как понять, что получилось при этой кластеризации? Для этого была построена дендрограмма, отображающая процесс разбиения множества веществ на кластеры. Дальше, для более детального анализа, получим метки для заданного количества кластеров и оценим качество полученного разбиения. Чтобы оценить качество, были использованы разные метрики качества

Для этого были использованы разные метрики качества

1. **Силуэтный коэффициент (Silhouette Score):**

   Оценивает, насколько объект похож на свой кластер и отличается от других.

   Интерпретация:

   - Значения от -1 до 1. Чем ближе к 1, тем лучше.
   - Если ≈ 0, объект находиться на границе кластеров.
   - Отрицательные значения – признак неудачной кластеризации

   Замечание:

Чем больше размер кластера и при этом меньше число всех кластеров, тем точнее оценка, т.к. элемент будет очень похож на свой кластер и отличен от других, потому что других кластеров вообще нет или их очень мало

2. **Индекс Дэвиса-Болдуина (Davies-Bouldin Index):**

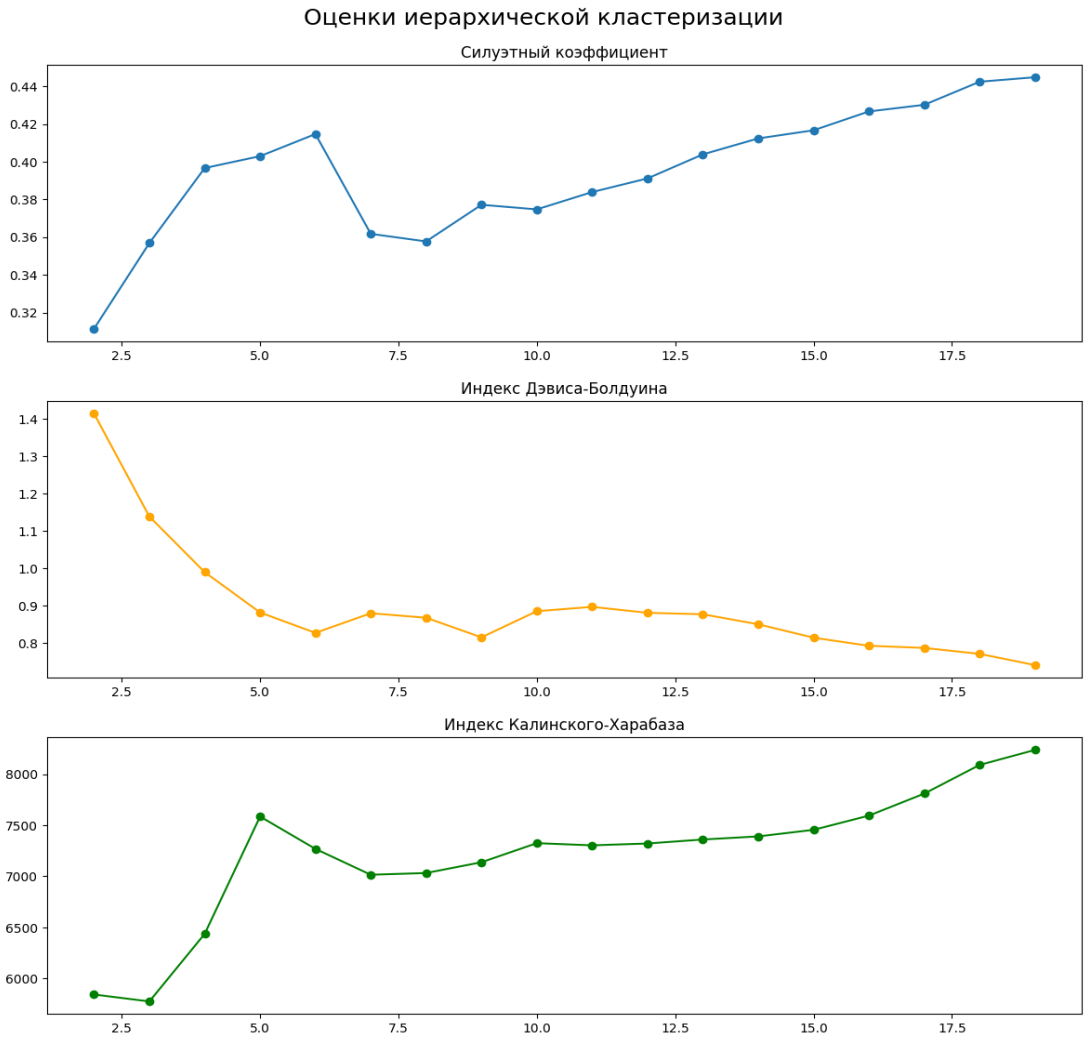
   Среднее сходство между кластерами. Чем значение ближе к 0, тем лучше.

   Замечание:

   Чем меньше элементов в кластере (кластер из одного элемента является самым оптимальным), тем точнее оценка, т.к. в таком кластере содержится меньше элементов, которые могут отклонить "среднее значение”

3. **Индекс Калинского-Харабаза (Calinski-Harabasz Index):**

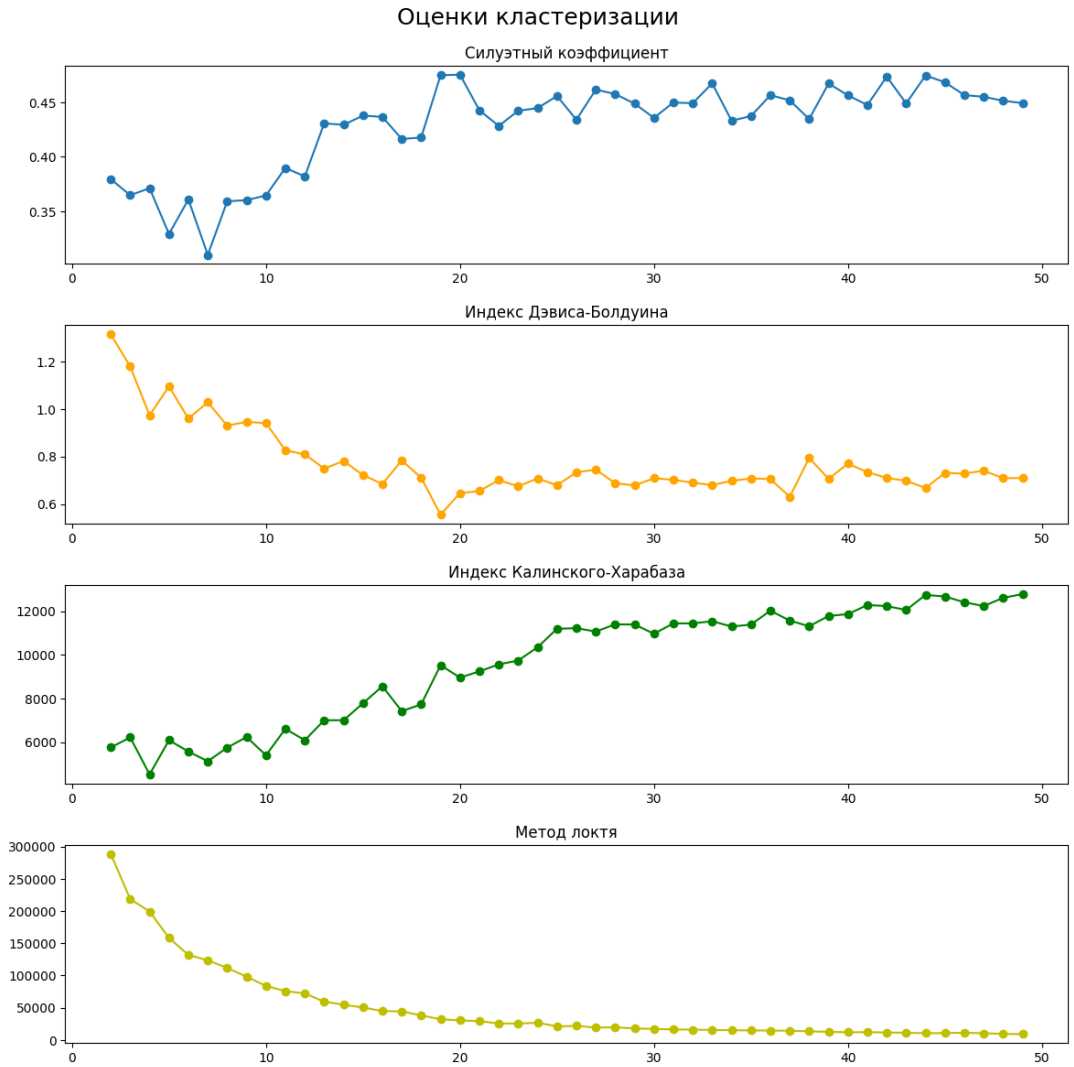
   Отношения дисперсии между кластерами к дисперсии внутри кластеров. Чем выше значение, тем лучше



По этим графикам можно увидеть, что кластеризация была не особо удачной для количества кластеров от 2 до 20, т.к. силуэтный коэффициент меньше 0.5 (< 0.25 – очень плохое разделение сравнимо со случайным распределением, 0.25 – 0.50 – слабое распределение кластеры пересекаются, 0.50 – 0.75 – хорошее, 0.75 – 1.0 – отличное разделение), индекс Дэвиса-Болдуина больше 0.5 (> 1 – кластеры сильно размыты, 0.5 – 1 – кластеры могут пересекаться, < 0.5 – хорошая кластеризация).

Алгоритм K-Means

Для этого алгоритма сделаем тоже самое: получим метки для заданного количества кластеров и оценим качество полученного разбиения



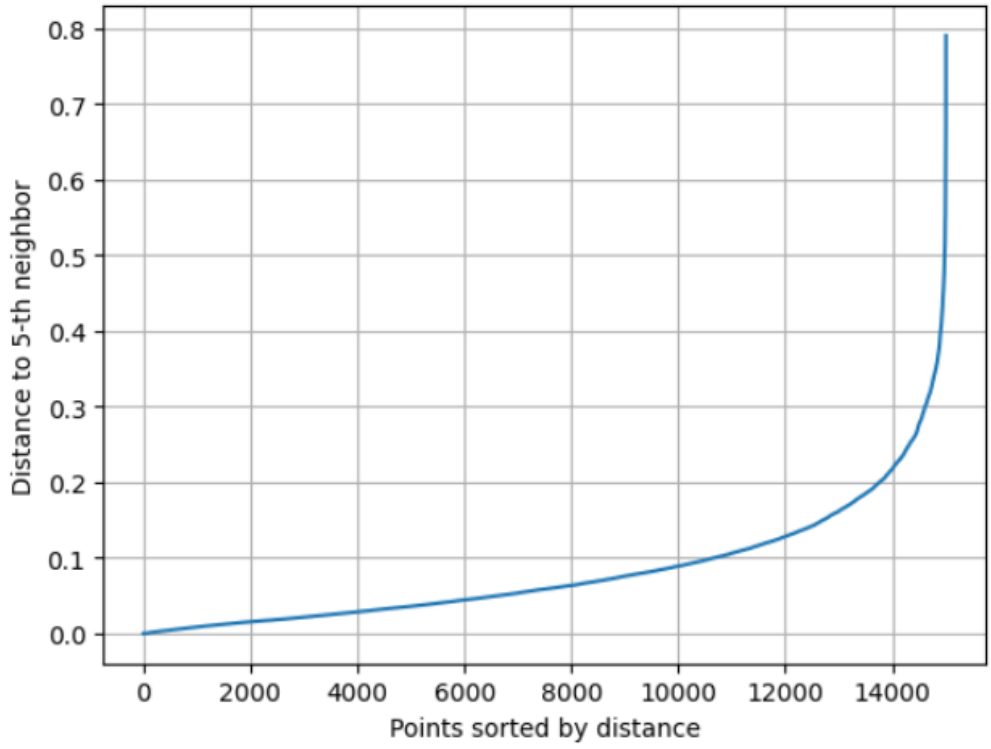
По графикам видно, что кластеризация опять вышла неудачной. Силуэтный коэффициент также меньше 0.5, а индекс Дэвиса-Болдуина больше 0.5, но зато ближе к этому значению, чем в иерархической кластеризации.

Алгоритм DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Для получения наилучшего результата необходимо выбрать параметр ε - радиус окрестности. Для этого нужно вычислить на каком расстоянии расположены вещества друг от друга. Воспользуемся функцией NearestNeighbors из модуля sklearn.neighbors

```
k = 5  # обычно = min_samples
nbrs = NearestNeighbors(n_neighbors=k).fit(X)
distances, indices = nbrs.kneighbors(X)
k_distances = np.sort(distances[:, -1])  # расстояние до k-го соседа, отсортированное

plt.plot(k_distances)
plt.ylabel(f"Distance to {k}-th neighbor")
plt.xlabel("Points sorted by distance")
plt.grid(True)
plt.show()
```



На графике видно, что расстояние плавно растет до отметки 0.2, а после резко взлетает вверх. Значение 0.2 на оси ординат можно понимать как некую границу между плотными и разреженными точками.

```
# Кластеризация
dbscan = DBSCAN(eps=0.2, min_samples=5)
clusters_db = dbscan.fit_predict(X)

# Оценка качества (игнорируем шумовые точки)
if len(np.unique(clusters_db)) > 1:
    sil = silhouette_score(X[clusters_db != -1], clusters_db[clusters_db != -1])
    db = davies_bouldin_score(X[clusters_db != -1], clusters_db[clusters_db != -1])
    ch = calinski_harabasz_score(X[clusters_db != -1], clusters_db[clusters_db != -1])
    print("Силуэтный коэффициент:", sil, "\nИндекс Дэвиса-Болдуина:", db, "\nИндекс Ка
else:
    print("Все данные отнесены к шуму или одному кластеру.")
```

```
Силуэтный коэффициент: 0.39376113
Индекс Дэвиса-Болдуина: 0.6524564878566164
Индекс Калинского-Харабаза: 8195.371344938525
Кол-во кластеров: 200
```

Анализируя полученный результат, можно сказать, что алгоритм DBSCAN оказался не лучше предыдущих методов. Если же чуть поменять параметры, например, min_samples (параметр, который определяет минимальное количество соседей, необходимых для того, чтобы точка считалась «ядром» кластера) взять равным 20, то получим следующее:

```
Силуэтный коэффициент: 0.6109816
Индекс Дэвиса-Болдуина: 0.43424417945213256
Индекс Калинского-Харабаза: 25497.810456721403
Кол-во кластеров: 183
```

Уже получился неплохой результат, в разы лучше того, что было до этого. Однако, если посчитать количество элементов в каждом кластере, то окажется, что очень много элементов отнесены к шуму.

```
# Посчитает кол-во элементов в каждом кластере
counts = np.bincount(clusters_db + 1, minlength=len(np.unique(clusters_db)))
counts
```

```
array([3876,  207,  325,  153,  205,   29,   39,  316,  166,   39,   29,
         58,  172,   43,   92,   79,  124,  133,   27,  106,  136,   26,
        142,   26,   65,   56,   64,   37,   38,   16,  105,   71,   99,
         93,  253,   65,   84,  216,   51,   39,  130,   63,   72,  429,
         45,   50,   76,   24,   28,   44,   74,   54,   34,   25,   33,
         42,   21,   25,   34,  156,  108,   33,   31,   50,   50,  325,
         26,   40,  255,   50,  198,   24,   53,   74,   27,   36,   59,
         29,   32,   20,   43,   35,   38,   21,   66,   24,   50,   35,
         96,   23,   30,   29,  136,   61,   33,   31,   23,   57,   20,
         30,   26,   46,   26,   51,   54,   24,   23,   30,   49,   34,
         92,   43,   96,   32,  145,   35,   43,   37,   24,   35,   35,
         19,   82,   22,   23,   77,  112,   80,   28,   36,   33,   20,
         35,   30,   54,   28,   39,   68,   37,   28,   75,   41,   31,
         25,   29,   58,   36,   21,   23,   20,   20,   31,   53,   51,
         58,   40,   50,   34,   39,   55,   47,   29,   27,   35,   23,
         22,   27,   26,   22,   20,   21,   32,   21,   26,   33,   30,
         20,   21,   37,   20,   26,   20,   39], dtype=int64)
```

В результате получено, что параметр min_samples оказывает сильное влияние на качество кластеризации и долю точек, отнесённых к шуму.

При низких значениях min_samples алгоритм становится менее требовательным к плотности данных. В результате формируется больше кластеров, включая те, которые содержат малое число объектов и могут быть статистически незначимыми. Это приводит к ухудшению качества кластеризации.

Напротив, при увеличении min_samples DBSCAN начинает требовать более высокую плотность для формирования кластера. Это приводит к тому, что количество кластеров уменьшается, но при этом значительная часть точек не удовлетворяет условию плотности и классифицируется как шум.

В итоге наблюдается, что при малом min_samples ухудшается структурная устойчивость и однородность кластеров, а при большом - резко возрастает доля шумовых точек. При компромиссном значении получаются плохо интерпретируемые результаты со слабыми оценками.

Заключение

В данной курсовой работе была рассмотрена задача кластеризации химических соединений на основе их структурных и физико-химических признаков. Цель исследования заключалась в изучении теоретических и практических аспектов применения методов машинного обучения для группирования веществ по их свойствам и химическому представлению в формате SMILES.

В теоретической части были изучены основы кластерного анализа, а также основные подходы к решению задач кластеризации. В частности, рассмотрены два ключевых типа кластерных методов – иерархические и неиерархические. К иерархическим методам был отнесен метод агломеративной кластеризации, а неиерархическим – алгоритмы K-Means и DBSCAN. Для каждого подхода проанализированы их принципы работы и математические основы. Дополнительно были изучены основные метрики оценки качества кластеризации, включая силуэтный коэффициент, индекс Дэвиса-Болдуина и индекс Калинского-Харабаза, что позволило более объективно оценить результаты практических экспериментов.

В практической части работы была проведена комплексная обработка набора данных химических веществ. На первом этапе были устранены пропуски, дублирующиеся записи и вычислены молекулярные дескрипторы с использованием библиотек RDKit и Mordred. После этого выполнена стандартизация и снижение размерности данных с помощью методов PCA и UMAP, что позволило представить сложные многомерные химические данные в более удобном для анализа виде. Далее были применены различные алгоритмы кластеризации (K-Means, DBSCAN, иерархическая кластеризация), а результаты их работы были сопоставлены по ряду метрик.

Результаты экспериментов показали, что применение рассмотренных методов не привело к формированию чётко интерпретируемых кластеров. Полученные значения метрик качества в большинстве случаев находились чуть ниже среднего уровня, что свидетельствует о слабой выраженности кластерной структуры. Это можно объяснить тем, что данные распределены достаточно равномерно, без явно выраженных групп, и расстояния между объектами в пространстве признаков равны друг к другу.

Несмотря на то, что полученные результаты нельзя считать полностью удовлетворительными, исследование имеет потенциал применения подобных подходов в задачах анализа и генерации химических соединений. Поскольку использованный набор данных предположительно взят из проекта ReLeaSE (Reinforcement Learning for Structural Evolution), где молекулы генерируются на основе заданных свойств, приведенную кластеризацию можно рассматривать как дополнение к этому проекту. Такой подход может быть использован для поиска аналогов веществ определенного кластера. Например, при выделении кластера лекарств можно определить диапазоны или закономерности свойств, характерные для данного класса соединений, и использовать эти данные для генерации новых кандидатов с аналогичным действием (т.е. новых кандидатов - лекарств), но более эффективных.

Таким образом, проведённое исследование не только позволило изучить и применить различные методы кластеризации, но и заложило основу для дальнейших исследований, направленных на создание и поиск новых химических соединений с требуемыми свойствами.

Список источников и литературы

1. Ершов К. С. Анализ и классификация алгоритмов кластеризации / К. С. Ершов, Т. Н. Романова. // Новые информационные технологии в автоматизированных системах. – 2016. – №19. – С. 274-279.
2. Котов А., Красильников Н. Кластеризация данных [Электронный ресурс]. – Режим доступа: http://logic.pdmi.ras.ru/~yura/internet/02ia-seminar-note.pdf
3. Мандель, И. Д. Кластерный анализ / И. Д. Мандель. – М.: Финансы и статика, 1988. – 176 с.
4. Воронцов, К. В. Лекции по алгоритмам кластеризации и многомерного шкалирования / К. В. Воронцов. — М.: МГУ, 2007 — 18 с.
5. Миркин, Б. Г. Методы кластер – анализа для поддержки принятия решений: обзор: препринт WP7/2011/03/ Б.Г. Миркин. — М.: Изд. Дом Национального исследовательского университета «Высшая школа экономики», 2011 — 88 с.
6. Müller, A. C. Introduction to Machine Learning with Python: A Guide for Data Scientists / A. C. Müller, S. Guido. – O'Reilly Media, 2016. – 394 p.
7. Scikit-learn: Clustering in Python. — [Электронный ресурс]. — Режим доступа: https://scikit-learn.org/stable/modules/clustering.html
8. VanderPlas, J. Python Data Science Handbook: Essential Tools for Working with Data / J. VanderPlas. – O'Reilly Media, 2016. – 548 p.
9. Blanco-Silva, F. J. Learning SciPy for Numerical and Scientific Computing / F. J. Blanco-Silva. – Packt publishing, 2015. – 150 p