Московский Государственный Университет им. М. В. Ломоносова
Механико-математический факультет
Кафедра вычислительной математики

Перевозников Александр Владимирович

Группа 510

Дипломная работа

Поиск дескрипторов молекулярных графов в задаче «Структура-свойство» на основе k-NN классификаторов

Научный руководитель
д. ф.-м. н., профессор
Кумсков М. И.

Москва, 2010

В работе реализован новый подход решению задачи «структура-свойство» (QSAR-задачи) с использованием кластер анализа и эволюционного алгоритма МГУА на основе k-NN с ограничением радиуса.

Проведены вычислительные эксперименты на выборках молекул. На выборке глюкозидов проведено сравнение качества прогноза с разбиением на кластеры и без них, получен высокий процент успешных предсказаний. Так же исследовалась выборка молекул амбровых одорантов, на ней проведено сравнение эволюционного алгоритма МГУА на k-NN с обычным методом k ближайших соседей на дескрипторах и на факторах. Так же рассматривалась выборка токсичных соединений, на которой, проверенна устойчивость качества прогноза.

Проект реализован в системе MatlabR2006b. При этом использовались как основной пакет программ.

Оглавление.

Введение.

Задача «Структура-свойство» или QSAR , что является сокращением от английского Quantitative Structure Activity Relationships, - это одно из наиболее интенсивно развивающихся в наши дни направлений молекулярного моделирования. Задача «Структура-свойство» заключается в поиске взаимозависимостей между структурами химических соединений и их свойствами с помощью построения математических моделей. Ее основателем во многих литературных источниках считается американский ученый Корвин Ганч, хотя в ее создание существенный вклад внесли и такие ученые, как Ч.Овертон, Г.Мейер, Г.Фюнер, С.Фри, Дж.Вильсон, Т.Фуджита и др.

Современный подход к решению этой задачи заключается в построении дескрипторов – численно выраженных свойств химической структуры и последующей работы с ними. Структура соединений описана с помощью молекулярных графов, потом с помощью них вычисляются дескрипторы. Далее на основе этих дескрипторов с использованием различных алгоритмов строятся модели, которые последствии используются для предсказания свойств новых молекул. Это позволяет существенно уменьшить затраты на синтез новых соединений, т. е. создаются только те молекулы, у которых согласно прогнозу имеется нужное свойство.

В методологии QSAR выделяют две подзадачи: прямую и обратную. Прямая заключается в определении по структуре свойств молекулы. Обратной QSAR задачей является создание структур химических соединений, которые будут обладать заданными свойствами.

В данной работе рассмотрено решение прямой задачи «Структура-свойство». Основное внимание уделено отбору значимых дескрипторов.

Целью данной работы являлась разработка и исследование эффективности метода построения классифицирующей функции, основанного на эволюционном отборе дескрипторов с помощью k-NN, в задаче обнаружения функциональной зависимости «структура-свойство».

Научная новизна работы состоит в следующем:

1. Предложен новый метод построения классифицирующей функции в задаче «структура - свойство», основанный на эволюционном отборе дескрипторов с помощью k-NN.

2. Реализован алгоритм анализа матрицы «молекула - дескриптор» в задаче «структура-свойство», проведена оценка вычислительной сложности алгоритма и проверенна его эффективность .

3. Подтверждена практическая значимость подхода в серии вычислительных экспериментов по прогнозированию биологической активности органических соединений.

Практическая значимость работы состоит в том, что разработанный алгоритм решения QSAR-задачи могут быть использован для решения прикладных задач предсказания физико-химической или биологической активности веществ по их структуре. Это может позволить отказаться от дорогостоящих и длительных исследований свойств большого числа молекул. Анализировать можно только те соединения, которые разработанная модель считает активными.

Материалы диплома докладывались и обсуждались на Международной научной конференции "Компьютерные науки и информационные технологии" (2009 г.), 14-ой Всероссийской конференции «Математические методы распознавания образов» ММРО-2009 (2009 г.). Полученные результаты также обсуждались на научных семинарах механико-математического факультета МГУ им. М.В. Ломоносова и Института Органической Химии им. Н.Д.Зелинского РАН. По материалам диплома опубликованы 3 научные работы. Основные результаты содержатся в следующих статьях:

1. Прохоров Е.И., Перевозников А.В., Воропаев И.Д., Кумсков М.И., Пономарёва Л.А. Поиск представления молекул и методы прогнозирования активности в задаче «структура–свойство» // Доклады 14-ой Всероссийской конференции «Математические методы распознавания образов» ММРО-2009. – М: МАКС Пресс. – 2009. - С. 589-591/

2. Деветьяров Д.А., Кумсков М.И., Апрышко Г.Н., Носеевич Ф.М., Прохоров Е.И., Перевозников А.В., Пермяков Е.А. Сравнительный анализ применения нечетких дескрипторов при решении задачи «структура-свойство» // Доклады 14-ой Всероссийской конференции «Математические методы распознавания образов» ММРО-2009. – М: МАКС Пресс. – 2009. – С. 511-514.

3. Прохоров Е.И., Перевозников А.В., Пономарева Л.А. Кумсков М.И. Нейронная сеть как инструмент реализации кусочно-линейного классификатора при массовом скрининге молекул в задаче «структура-свойство» // Нейрокомпьютеры: разработка, применение. – № 3. – С. 39-45.

Работа поддержана Российским Фондом Фундаментальных Исследований (РФФИ) по гранту №07-07-00282.

В первой главе работы рассмотрим общую постановку задачи Структура-свойство для молекулярных графов и дадим основные определения и описания молекулярных графов. Во второй главе описан алгоритм построения моделей функциональной зависимости. В третьей главе приведем полученные результаты, их сравнения.

Глава 1. Общая постановка задачи «структура-свойство» для молекулярных графов.

1.1 Основные понятия и определения, постановка QSAR задачи.

Меченый молекулярный граф $G = \{E, V\}$ – помеченный граф, вершины которого интерпретируются как атомы молекулы, а ребра – как валентные связи между парами атомов. Метки вершин и ребер (числа или символы) кодируют атомы и связи различной химической природы. В качестве меток вершин могут быть использованы любые характеристики соответствующих атомов (например, трехмерные координаты, символ химического элемента, заряд ядра, поляризуемость, атомный вес, атомный радиус и др.), а в качестве меток ребер – любые характеристики соответствующих связей (кратность, длины, порядки связей, полученные из квантово-химических расчетов, и т.д.).

Задача «структура-свойство»(QSAR) Пусть задана *обучающая (или эталонная) выборка* - база данных из N химических соединений, которые обладают следующими двумя свойствами:

1) i-ое соединение представлено меченым молекулярным графом G_i, имеющим укладку в трехмерном пространстве (т.е., для каждой вершины в качестве меток заданы ее трехмерные координаты);
2) либо i-ое соединение к C_i - одному из K классов активности (например, «активных», «слабоактивных», «неактивных» веществ) согласно исследуемому свойству, либо для него задано численное значение исследуемого свойства A_i.

Необходимо построить классифицирующую функцию F, получающую в качестве аргумента произвольный молекулярный граф с метками того же типа, и «наилучшим образом» относящую это соединение к одному из классов активности, либо «наилучшим образом» предсказывающую численное значение исследуемого свойства.

Функционал качества $\varphi(F)$ позволяет отределить какая из классифицирующих функций наилучшая. Например, в качестве функционала качества можно использовать процент верно классифицированных функцией F молекул из обучающей выборки:

$$\varphi(F) = 1 - \frac{\sum_{i=1}^{N} \varepsilon_i}{N}, \text{ где } \varepsilon_i = \begin{cases} 0, \textit{если } F(G_i) = C_i \\ 1, \textit{в противном случае} \end{cases},$$

или, в случае, когда функция должна предсказывать численное значение свойства,

$$\varphi(F) = 1 - \frac{\sum_{i=1}^{N} (F(G_i) - A_i)^2}{\sum_{i=1}^{N} A_i^2}$$

Дескриптором будем называть какое-либо свойство, численное значение которого может быть вычислено для произвольного молекулярного графа G.

Алфавитом дескрипторов будем называть множество всех дескрипторов, используемых для анализа обучающей выборки, обозначенных различными символьными метками.

Вектором признаков молекулярного графа *G* будем называть вектор $\bar{x} = (x_1, .., x_M) \in R^M$, где x_j- значение *j*-ого дескриптора, вычисленное для *G*, алфавит дескрипторов состоит из *M* элементов - В алфавит дескрипторов состоит из *M* элементов.

Матрицей «молекула-признак» (матрицей признаков) для рассматриваемой обучающей выборки будем называть матрицу размера N x M, в i-ой строке которой стоит вектор признаков $\bar{x}_i = (x_{i1}, .., x_{iM})$ i-ого соединения.

1.2 Основные этапы решения QSAR-задачи

Задача «структура-свойство» разбивается на два этапа: выбор описания молекулярных графов и поиск функциональной зависимости. На первом, исходя из формата молекулярных графов (типа меток вершин и ребер), выбирается алфавит дескрипторов. На основе этого алфавита строится отображение из множества молекулярных графов в признаковое

пространство R^M и формируется матрица «молекула-признак» для обучающей выборки. На втором, в результате анализа матрицы «структура-свойство» на признаковом пространстве, строится модель функциональной зависимости - классифицирующая функция F с наилучшей прогностической способностью, т.е. с наибольшим значением функционала качества *φ(F)*.

1.2.1 Различные описания молекулярных графов

Рассмотрим два основных типа дескрипторов, на основе которых чаще всего строятся вектора признаков для молекулярных графов. Сразу отметим, что в нашей работе проводиться анализ матриц, построенных преимущественно из дескрипторов второго типа.

1. Топологические дескрипторы**.** Данные дескрипторы строятся по *топологической матрице.*

   Топологическая матрица $A = (a_{ij})$ меченого молекулярного графа строится следующим образом: в ней элемент a_{ii} – это метка *i*-ой вершины, a_{ij} – ребра (*i, j*). Очевидно, что при изменении нумерации вершин графа получается, вообще говоря, другая матрица. Это является существенным недостатком описания химических структур в терминах матриц. Эта проблема привела к термину «инвариант графа».

   Инвариант графа – это такое число (или функция, если метка графа - символы), вычисляемое по матрице графа А, которое не зависит от нумерации вершин графа.

   Молекулярный граф называется простым, если выполнены следующие условия: метки вершин равны нулю и если атомы связаны химической связью, то им сопоставляется ребро с меткой «1», в противном случае метка ребра равна нулю. То есть простой граф отображает только наличие связей между вершинами.

   Топологическими индексами (ТИ) называют инварианты простых графов. Часто это определение переносится и на инварианты меченых графов, которые могут отражать не только топологию молекулы, но и элементы электронного и пространственного строения. Топологические индексы и широко

используются как дескрипторы при решении задачи «структура-свойство». Популярность данного подхода к описанию молекулярной структуры связана с простотой и быстротой вычисления ТИ, возможностью учитывать при их построении элементы электронного и пространственного строения, я также наличием огромного количества удачных корреляций вида «ТИ – свойство». Однако такой подход имеет и очевидный недостаток: он не позволяет различать разные конфигурации молекул и не учитывает их конформационные особенности.

Ниже приведены примеры наиболее популярных ТИ.

Индекс Рандича χ[4]:

Определяется по следующей формуле: $x = \sum(v_i\ v_j) - 1/2$, v_i - степень i-ой вершины графа; суммирование проводится по всем ребрам графа.

Индекс x был обобщен Киром и Холлом, которые ввели определение так называемого индекса связности m-ого порядка (индекс Кира-Холла) $mxv = \sum(\delta_i\delta_j\delta_k) - 1/2$, m ≥ 1, δ = (Z_i^v - h_i) / (Z_i - Z_i^v - 1), где Z_i – общее число, а Z_i^v - число валентных электронов i-ого атома; h_i – число атомов водорода, связанных с -ым атомом; суммирование происходит по всем цепочкам, состоящим из m ребер графа; i, j, k – номера вершин, образующих соответствующую цепочку. В [4] показано, что индекс Рандича может быть использован для предсказания ряда физико-химических свойств соединений.

Индекс Винера W[1]:

Определяется по следующей формуле: $W = 0.5\sum d_{ij}$,

где d_{ij}– кратчайшее расстояние между i-ой и j-ой вершинами. Матрица D = (d_{ij}) называется матрицей топологических расстояний. В [1] показано, что индекс Винера сильно коррелирует с температурой кипения алканов.

Индекс Хосойя Z:

Определяется формулой: $Z = \Sigma p_k$, где p_k – число способов выбрать в графе k ребер ($k \geq 1$) так, что никакие два из них не являются смежными.

Индекс Балабана:

(для ациклических графов (деревьев)) определяется по следующей формуле: $B = \Sigma r_i^2$, где r_i - число вершин дерева, отсекаемых на i-ом шаге процедуры обрезки деревьев. Данная процедура состоит из последовательного удаления вершин степени 1 исходного дерева и инцедентных им ребер.

2. Фрагментарные (структурные) дескрипторы.

Использование данного типа дескрипторов основано на выделении в молекулах структурных фрагментов. Структурные дескрипторы были впервые использованы в [5].

Структурным фрагментом молекулярного графа называется группа вершин с заданными условиями на их метки или метки их связей.

После выделения фрагментов каждому фрагменту сопоставляется структурный дескриптор, значение которого соответствует либо наличию или отсутствию данного фрагмента в молекулярном графе, либо количеству повторений фрагмента. В первом случае получаем дескриптор, принимающий логические значения, во втором - целые неотрицательные.[7,8]

Структурные фрагменты и соответствующие им дескрипторы подразделяют на два типа:

2D-дескрипторы

В данном случае не учитываются значения валентных углов и евклидовых расстояний между атомами, не учитывается трехмерная структура фрагмента, важны только связи между атомами. Структурные 2D-фрагменты обычно имеют вид цепочек связанных атомов с определенными метками вершин и ребер, образующих данную цепочку.

3D-дескрипторы

Дескрипторы этого типа учитывают трехмерную структуру фрагмента и обычно представляют собой множество вершин с заданными условиями на расстояния между ними и на их метки.

Известной моделью биологической активности является пространственный треугольник, вершины которого являются особыми точками и имеют заданные локальные физико-химические свойства[6]

Этап построения структурных дескрипторов разделяется на 2 крупных шага:

1) Этап формирования особых точек
2) Этап формирования алфавита дескрипторов и построения матрицы признаков

1 Этап формирования особых точек.

Особенность данной работы состоит в том, что дескрипторы строятся не по исходным молекулярным графам, а производным из них трехмерным графам, в вершинах которых располагаются не атомы молекул, а так называемые особые (критические) точки. Только затем по полученному трехмерному меченому молекулярному графу вычисляются значения структурных 3D-дескрипторов. В качестве особых точек так же можно использовать любые другие элементы структуры (атомы, связи, двойки, тройки и цепочки атомов и т.д. [2,3]).

Приведем описание построения особых точек:

1 Расчет пространственной структуры и заряда молекулярных графов.

На данном этапе использовался пакет Gaussian - программный комплекс для проведения неэмпирических и полуэмпирических квантово-химических расчетов. На вход данной программы подавались вышеописанные .mol и .sdf – файлы. В результате,

получали файлы с расширением .out, содержащие информацию о многих квантово-химических характеристиках молекул, в том числе и тех, которые были задействованы далее.

2 Расчет триангулированных молекулярных поверхностей.

Целью данного этапа является построение триангулированной молекулярной поверхности для каждой молекулы выборки. Такая поверхность «обволакивает» атомы молекулы и характеризует взаимодействие молекулы с другими соединениями. Было решено использовать описание молекулярной поверхности, а не структуру атомов молекулы, так как активность соединения или наличие запаха определяется именно взаимодействием молекулы с другими соединениями.

Строится описание молекулярной поверхности с помощью программы-редактора DS ViewerPro, предоставляющего ряд инструментов «молекулярной визуализации». На вход подаются данные из ранее полученного .out-файла. Сначала для каждого атома вычисляется вандерваальсов радиус. Затем вокруг каждого атома строится шар данного радиуса и рассматривается объединение построенных шаров. Полученная область является основой молекулярной поверхности. Чтобы получить саму поверхность, необходимо «сгладить» объединение шаров. С этой целью по вандерваальсовой поверхности прокатывается шар определенного радиуса. В качестве дескриптора молекулы можно так же использовать объем и площадь ее молекулярной и вандерваальсовой поверхности.[12,13,14]

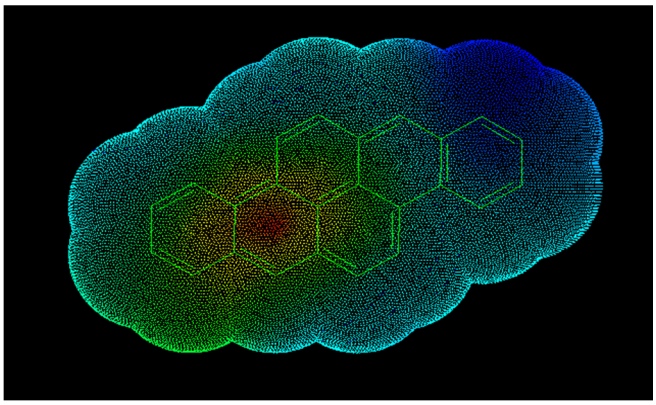


Рис. 3. Молекулярная поверхность

Программа DS ViewerPro предоставляет возможность получить триангуляцию молекулярной поверхности. Данная триангуляция сохраняется в файле с расширением .wrl. Из файла .wrl сохраняем следующие данные:

- матрицу *Triangle Mesh Definition* - содержит координаты триангулированной поверхности
- матрицу *Indexed Face Set* – содержит описание триангуляции в виде троек точек, составляющих треугольники.

Из файла .out для дальнейшего использования сохраняем:

- матрицу Symbolic z-matrix – содержит координаты атомов самой молекулы;
- матрицу Mulliken Atomic Charges – содержит заряд на атомах.

3 Нахождение геометрических локальных экстремумов.

На триангулизированной молекулярной поверхности выделяем особые (критические) точки. Определяем их как геометрические локальные экстремумы в терминах ближайших и наиболее отдаленных от определенных групп атомов точек на поверхности и находим их геометрическими переборными методами. При этом выделяем два типа: локальные минимумы и максимумы.

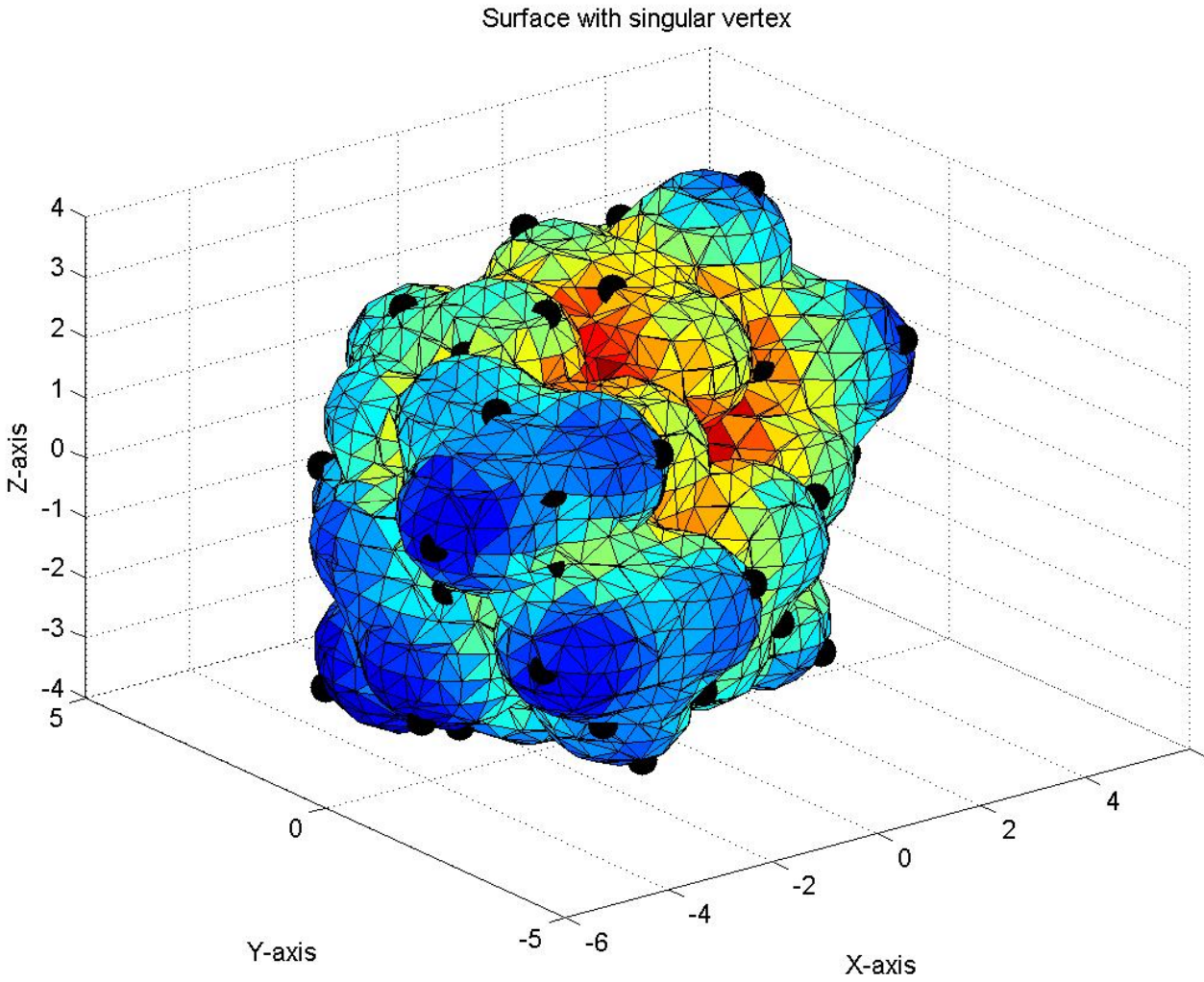


Рис. 4. Особые точки на молекулярной поверхности.

4 Расчет электростатического заряда на триангулированной молекулярной поверхности.

Для каждой особой точки поверхности вычисляем значение электростатического потенциала. Сначала рассчитываем электростатический потенциал в данной точке, порожденный зарядом отдельного атома. Затем все значения потенциала суммируем.

5 Классификация особых точек.

Рассматриваем множество всех полученных значений электростатического потенциала и делим интервал, содержащий это множество, на несколько интервалов разбиения. Количество интервалов разбиения и их границы являются параметрами алгоритма (параметры дискретизации), которые можно менять, чем мы и воспользуемся. В нашем случае мы выделили три интервала: интервал «близких к нулю» значений (абсолютное значение потенциала меньше заданного порогового значения) и интервалы положительных и отрицательных значений (абсолютное значение потенциала больше порогового значения).

В результате каждая особая точка получила две характеристики:

- Локальный максимум или локальный минимум (2 варианта);
- Интервал значения электростатического потенциала в точке (3 варианта).

Комбинациями данных характеристик получаем 6 типов особых точек. Согласно данным типам особым точкам и присваиваются символьные метки. При реализации в качестве меток использовались числа от 1 до 6.

Таким образом, по молекулярному графу соединений были построены молекулярные графы, в вершинах которых располагаются не атомы, а особые точки с дополнительными атрибутами – символьными метками. Заметим, что ребра у сформированных графов отсутствуют.

2 Этап формирования дескрипторов и построения матрицы «структура-свойство».

После получения описания соединений выборки в виде молекулярных графов особых точек необходимо построить алфавит дескрипторов, адаптированный к данной задаче, а затем по нему сформировать матрицу «структура-свойство». Таким образом, на данном этапе задача состоит в выборе набора дескрипторов.

Известной моделью биологической активности является пространственный треугольник, у которого вершины имеют заданные локальные физико-химические свойства, а стороны треугольника задаются интервалами расстояний. Если существует 3D-конформация молекулы, «содержащая» такой треугольник, то считается, что она будет обладать заданным биологическим свойством. Более сложным вариантом такой модели является пространственная пирамида с заданными свойствами «вершин» и «ребер». Исходя из этого, будем пытаться построить алфавит дескрипторов таким образом, чтобы представить взаимное расположение пар, троек, четверок особых точек молекулярной поверхности.

Для построения матрицы «молекула-признак» предлагается использовать структурные 3D-дескрипторы. Для вычисления их численных значений введем понятие соответствия структурного дескриптора D и химической функциональной группы G. Значением структурного дескриптора будет количество фрагментов (химических функциональных групп) молекулярного графа, соответствующих данному дескриптору.

Химической функциональной группой (фрагментом) в молекулярном графе будем называть некоторое непустое подмножество множества особых точек этого молекулярного графа.

Пусть $\{A_1, A_2, \dots, A_l\}$ множество всех символьных меток особых точек, причем они упорядочены в алфавитном порядке так, что $A_1 < A_2 < \dots < A_l$. Обозначим AD = $\{A_1, A_2, \dots, A_l\}$ - алфавит дескрипторов первого уровня и положим, что дескриптору A_i соответствуют те и только химические функциональные группы G, которые состоят ровно из одной особой точки и G $= \{A_i\}$.

Далее, пусть d_{max} - максимум по всей выборке всех возможных евклидовых расстояний между особыми точками одной молекулы. Введем на отрезке $[0, d_{max}]$ P интервалов расстояний. Количество интервалов P и их границы являются

параметрами дискретизации данного алгоритма и могут варьироваться для нахождения лучшего описания молекулярных графов. В реализации данной работы P = 3, интервал $[0, d_{max}]$ делится на 3 равных отрезка.

Сформируем алфавит дескрипторов второго уровня (т.е., дескрипторов для пар особых точек) AD^2 как множество символьных строк (A_i, A_j, c), где $A_i, A_j \in AD, A_i \leq A_j, c = 1, .., P$. Положим, что дескриптор $D_2 \in AD^2$ соответствует фрагменту G, если и только если G состоит из двух особых точек с метками T_1 и T_2, $T_1 \leq T_2$ таких, что $T_1 = A_i$, $T_2 = A_j$, и расстояние между особыми точками $d(T_1,T_2)$ принадлежит интервалу под номером c. Аналогичным образом строятся дескрипторы болие высокого уровня.

Кроме вышеописанных структурных 3D-дескрипторов, при формировании матрицы также использовался ряд скалярных дескрипторов – общих химико-физических характеристик молекул. Среди них - молярный вес, объем, рефракция, поверхностное натяжение, плотность, диэлектрическая постоянная, поляризуемость и пр. Данные свойства рассчитаны в редакторе ChemSketch - пакете программ создания, визуализации, расчета параметров химических структур, а также их каталогизации.

1.2.2 Этап поиска функциональной зависимости.

Напомним, что после формирования матрицы «структура-свойство» для обучающей выборки необходимо построить классифицирующую функцию $F(x_1, x_2, \ldots, x_M)$, где $(x_1, x_2, \ldots, x_M)$ – вектор признаков молекулярного графа. Причем, построенная функция должна обеспечивать лучшее значение функционала качества.

Обычно вид классифицирующей функции F заранее задается (например, функция может быть линейной, квадратичной и др.) и зависит от ряда параметров, которые определяются по обучающей выборке соединений. Чаще всего в качестве F используется линейная функция. Получаемое уравнение называют линейной

регрессионной моделью. В нашей работе на этапе поиска функциональной зависимости использовался Метод Группового Учета

Аргументов на основе k ближайших соседей, расскажем о нем подробнее в третьей главе.

Следует отметить, что в задаче «структура-свойство» число дескрипторов M, как правило, значительно превышает число молекул в обучающей выборке (M >> N), что затрудняет анализ матрицы «молекула-признак». Для того чтобы сократить число дескрипторов, необходимо рассматривать лишь наиболее информативные из них, т.е. те, которые потенциально будут значимы при построении классифицирующей функции на признаковом пространстве. Это можно проделать как на этапе описания (например, при эволюционном формировании дескрипторов), так и на этапе анализа матрицы признаков. В нашей работе отбираются наиболее информативные дескрипторы путем взаимодействия этих двух этапов – использования результатов этапа анализа на этапе описания.

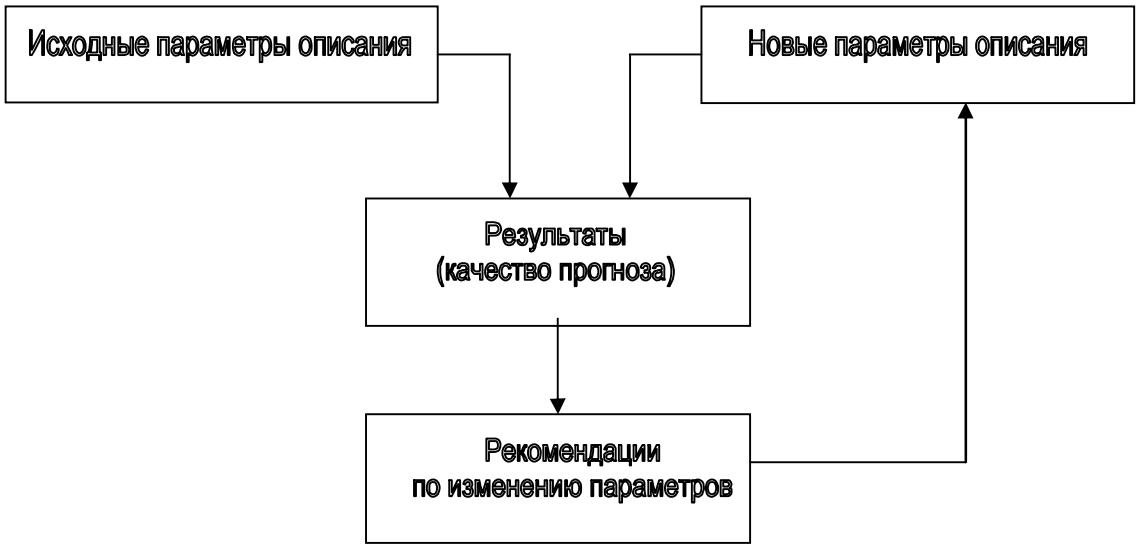
1.2.3 Обратная связь между этапом поиска функциональной зависимости и этапом построения дескрипторов.

Как и в любой QSAR-задаче, в нашем случае было необходимо найти способ описания молекулярных графов, а потом по построенной матрице «структура-свойство» построить некоторую модель классификации (классифицирующую функцию) с лучшим качеством прогноза.

В результате нам хотелось получить прогнозирующую модель, которую можно интерпретировать в терминах молекулярных поверхностей. Поэтому, с одной стороны, мы стремимся задействовать как можно более простые дескрипторы; а с другой - хотим получить как можно более точную прогностическую оценку.

Таким образом, происходит поиск описания, адаптированный по сложности к свойствам, с последующим построением моделей функциональной зависимости. Здесь приведено подробное описание предшествующего ему *этапа описания молекулярных*

графов. На данном этапе используются параметры детализации, от которых зависит конечный результат (качество прогноза). В силу чего реализуется *обратная связь* между этапами описания и поиска функциональной зависимости: по результатам построенной прогнозирующей модели делаются выводы об оптимальных значениях *параметров детализации* и весь алгоритм запускается с новыми, предположительно лучшими по качеству прогноза, параметрами. Этот процесс повторяется до тех пор, пока не добьемся адекватного прогностического качества построенной модели.



Вывод.

Для получений различных дескрипторов, используются следующие параметры:

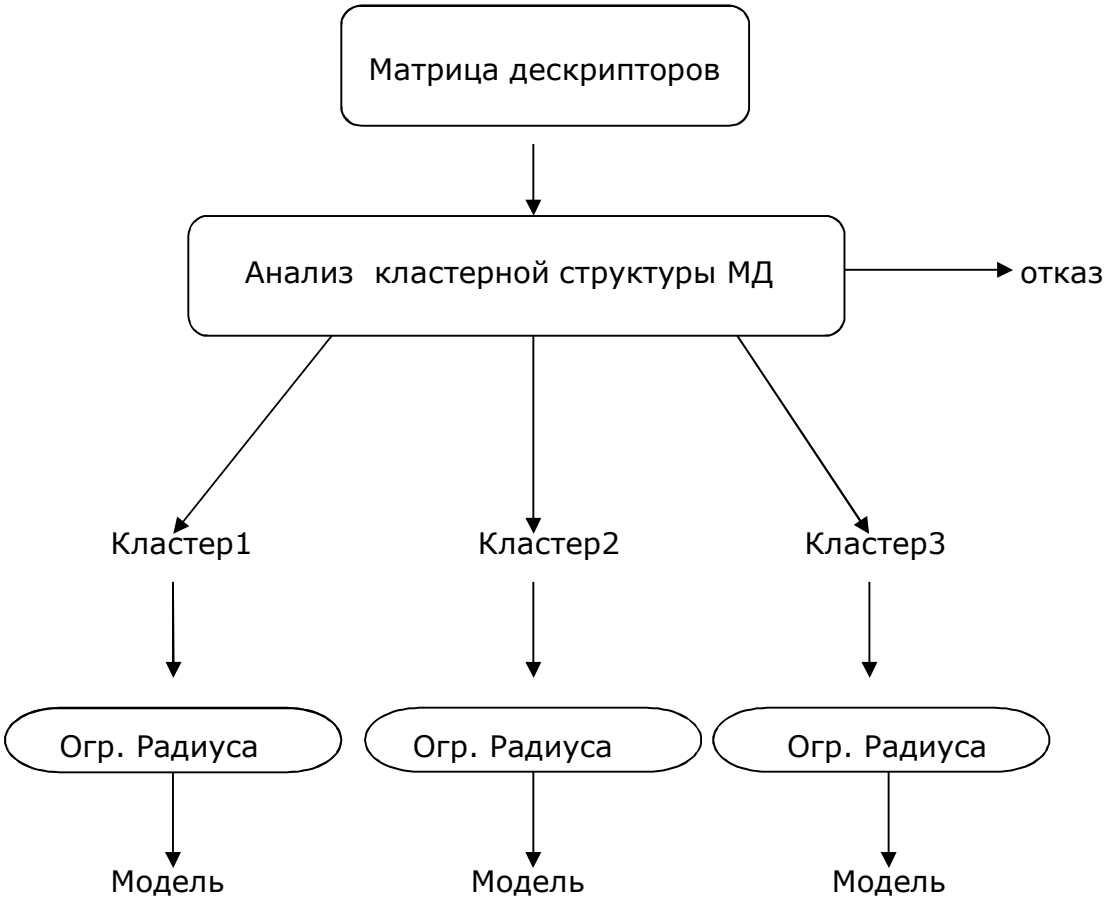
1. Типы особых точек.
2. Различная дискретизация расстояния между особыми точками
3. Уровень алфавита дескрипторов.

Эти параметры с учетом обратной связи позволяют найти наиболее подходящий для каждой выборки способ описания.

Глава 2. Построение моделей функциональной зависимости

2.1 Общая схема алгоритма

Итак на предыдущем этапе мы получили матрицу дескрипторов. Теперь необходимо определить функциональную зависимость между дескрипторами и активностью. Сначала с помощью алгоритма минимального покрывающего дерева разобьем выборку на кластеры, далее уменьшим количество дескрипторов выкинув совпадающие, выберем ограничение радиуса в котором следует искать ближайших соседей, потом Методом Группового Учета Аргументов на основе k ближайших соседей(МГУА на k-NN) строим модель. Изобразим схему работы на рисунке.



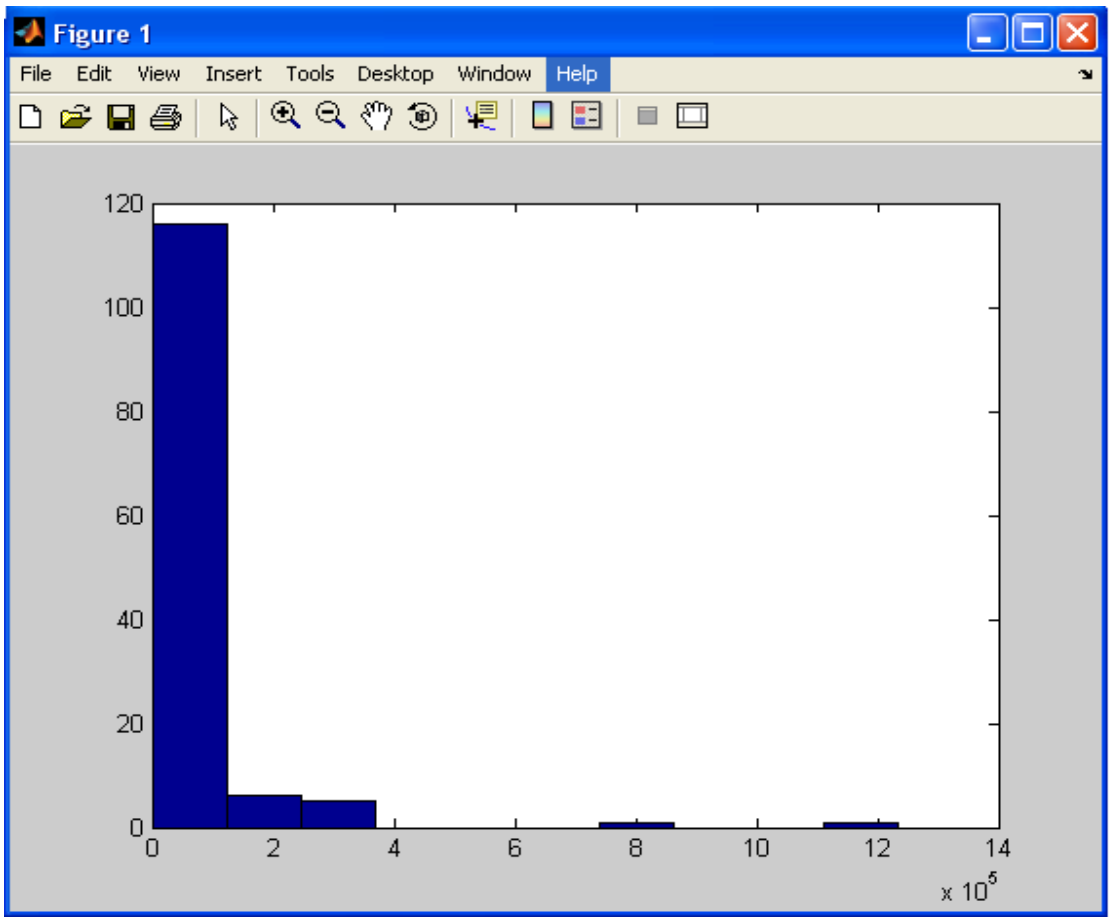
2.2 **Анализ кластерной структуры МД.**

С помощью матрицы дескрипторов сопоставим каждой молекуле точку в евклидовом пространстве R^M (координатами служат значения дескрипторов). Найдем две ближайшие друг к другу точки и соединим их ребром. Далее найдем точку ближайшую к ним и добавим ее в дерево. Продолжаем это до тех пор пока точки не закончатся. Итак, мы получили дерево. Осталось разбить выборку на кластеры и удалить выбросы, для этого по очереди отрезаем самые длинные ребра, здесь возможны два случая:

1. В результате образовалась компонента связанности, состоящая их 1-5 вершин, ее будем называть выбросом.
2. Обе компоненты содержат больше 5 вершин, тогда будем называть их кластерами.

Выбросы удаляем, каждый кластер теперь рассматривается как отдельная выборка. Так же важным вопросом является то сколько ребер обрезать. Есть несколько подходов:

1. Обрезаем ребра пока не получим нужное число кластеров.
2. Посмотреть на гистограмму длин ребер и обрезать самые длинные ребра. Между длинными и средними, как правило, есть зазор (смотри рисунок).



Теперь остановимся на преимуществах использования данного алгоритма:

1. Выбросы. Наш метод предсказания основан на предположении, что близкие друг к другу молекулы имеют сходное значение активности. Отсюда следует необходимость удалять такие молекулы из выборки.
2. Кластеры. Если имеется несколько удаленных друг от друга скоплений молекул, то мы предполагаем, что на значения активности молекул из разных скоплений влияют различные дескрипторы.
3. Производительность. Основной алгоритм (МГУА на k-NN) имеет й квадратичны порядок сложности по количеству молекул, следовательно разбиение на кластеры существенно ускоряет работу (если поделить выборку пополам то в 2 раза), тем более что вычисления на разных кластерах могут идти параллельно(в 4 раз).

2.3 **Ограничение радиуса.**

Как было сказано выше, наш метод предсказания основан на предположении, что близкие друг к другу молекулы имеют сходное значение активности, но на предыдущем шаге мы убрали выбросы основываясь на значении всех дескрипторов – это слишком грубая оценка. Необходимо учесть еще и следующие факторы:

1. Предсказываться значение активности будет не на всех дескрипторах, а только на комбинации из 3-5 лучших. На них могут оказаться свои выбросы.
2. При отборе дескрипторов будет оптимизироваться число соседей, и может оказаться, что у некоторых молекул мало близких молекул и у других наоборот, поэтому нужно не допустить, чтобы отдаленные молекулы влияли на результат.

Учитывая приведенные выше факторы, приходим к выводу о необходимости применение ограничения радиуса. Перейдем к вопросу как его выбрать. Способы, приведенные в предыдущем параграфе, нам не подходят, так как мы не сможем останавливать вычисления на каждом шаге и смотреть гистограмму. Поэтому количество ребер, которые будут обрезаны, решено зафиксировать(1% от числа молекул). Альтернативный подход осуществлять перебор не по количеству соседей, а по радиусу учитывай все молекулы, которые попадают в круг с данным радиусом.

2.4 **Прогнозирование активности с помощью k ближайших соседей (k-NN).**

Пусть каждой молекуле сопоставлена точка в евклидовом пространстве R^M (координатами служат значения дескрипторов). Дальше возможно два случая:

1. <u>Активность дискретна.</u>

   Пусть для простоты надо классифицировать молекулы на два класса (C_1, C_2) и у нас есть обучающая выборка C. Для того чтобы определить какому классу относится новая молекула x нужно из C найти k ближайших к ней точек (означим это множество C_x). Пусть m_1 количество точек из $C_x \cap C_1$, а m_2 из $C_x \cap C_2$. Если $m_1 > m_2$, то a принадлежит первому классу. $m_2 > m_1$ – второму.

2. <u>Активность недискретна.</u>

Пусть у нас есть обучающая выборка C. Для того чтобы активность новой молекулы x нужно из C найти k ближайших к ней точек. Сложить их значения их активности и поделить на k.

Как правило, имеется только тестовая выборка в этом случае, активность каждой молекулы по очереди считается не известной и предсказывается по остальным.

Более подробно о k-NN методе можно посмотреть в [15,16].

Надо отметить, что в Матлабе был реализовано прогнозирование активности с помощью k ближайших соседей, но эта реализация не подходила для нашего метода решения QSAR задачи. Отметим преимущества нашего алгоритма:

1. Не нужно запускать алгоритм насколько раз, чтобы получить прогноз для различного числа соседей. Значение прогноза выдается сразу для всех k от 1 до n, практически без дополнительных затрат на вычисления. В результате обеспечивается подбор оптимального числа соседей в n раз быстрее, чем на стандартном алгоритме.
2. Есть возможность ввести ограничение по радиусу.
3. Работает не только с дискретной активностью.

2.5 **Основные идеи эволюционного алгоритма МГУА.**

Пусть заданы:

- матрица «объект-признак» *X*, состоящая из *N* строк и *M* столбцов;
- *результирующий вектор* - вектор *y*, который необходимо «предсказать»
- класс функций *C* для построения классифицирующей функции;
- *функционал качества φ* (*F*, *X*, *y*), *F ϵ C*;
- количество отбираемых на каждом этапе наилучших функций *Q*;
- условие остановки алгоритма (максимальное число используемых переменных в классификаторе или предельный уровень функционала качества).

Тогда эволюционный алгоритм МГУА состоит из следующих этапов:

1) Перебором строим всевозможные функции $F \in C$ от одной переменной (столбцов матрицы X).

2) Оцениваем множество всех построенных на данный момент функций при помощи функционала качества φ и выбираем из них Q наилучших (данный набор функций, отобранных на определенном этапе, называется *селекцией*), причем для каждого шага число Q может иметь отдельное значение.

3) Перебором присоединяем к каждой из отобранных функций новую переменную (в рамках класса функций C), или функцию от одной переменной из числа отобранных (зависит от реализации), определенным методом формируем классифицирующие функции новой селекции.

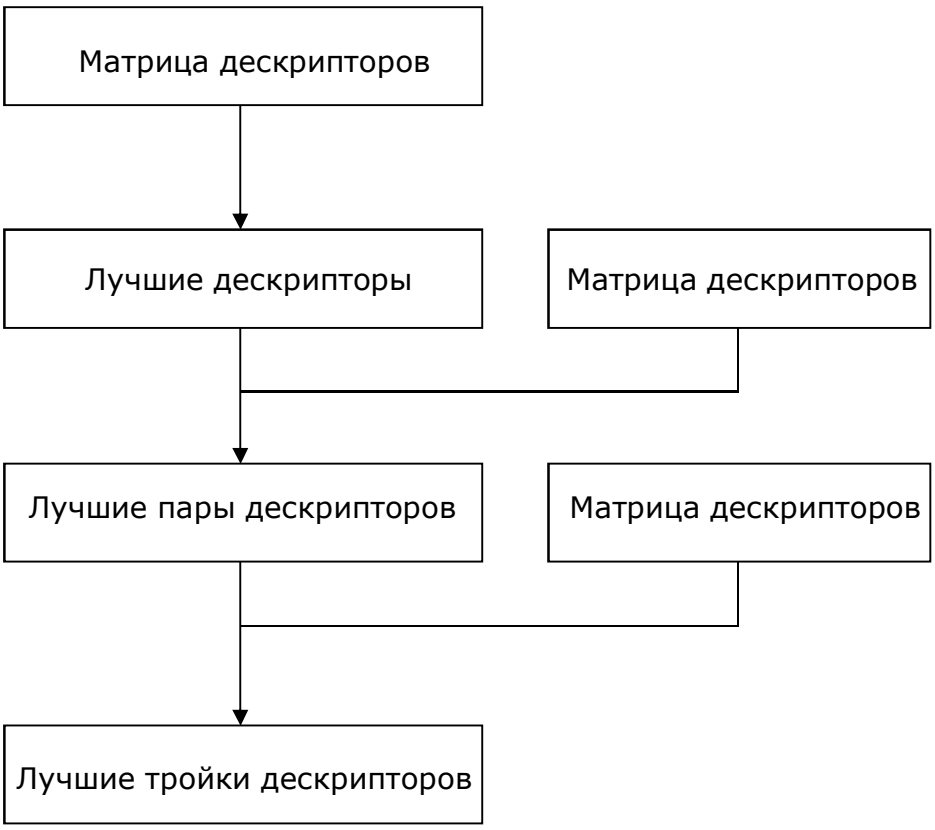
4) Проверяем, достигнуто ли условие остановки алгоритма; если оно не достигнуто, переходим к п. 2).

Приведем преимущества использования МГУА:

- Вычислительная сложность этого метода значительно меньше, чем у полного перебора - это имеет большое значение из-за большого числа дескрипторов (> 1000).
- Гарантируется нахождение наиболее точной модели - метод не пропускает наилучшего решения во время перебора всех вариантов (в заданном классе функций)
- Переборные алгоритмы МГУА довольно просто запрограммировать
- Метод использует информацию непосредственно из выборки данных и минимизирует влияние априорных предположений автора о результатах моделирования
- Подход МГУА используется для повышения точности других алгоритмов моделирования.

2.6 **Эволюционный алгоритм МГУА на k-NN.**

Приведем схему алгоритма.



Данную схему можно продолжать и дальше.

Перейдем словесному описанию алгоритма.

1. С помощью метода k-NN с ограничением радиуса по каждому дескриптору из матрицы *X,* прогнозируется активность и по значению функции качества *φ* отбирается *n* лучших представителей. Отбор лучших представителей будем называть селекцией.
2. К каждому набору дескрипторов, полученному на предыдущем шаге, по очереди добавляется еще один матрицы *X,* прогнозируется активность и по значению функции качества *φ* отбирается *n* лучших представителей. Мы получили n^2 лучших наборов дескрипторов.
3. Значения функции *φ,* полученные на втором шаге, сравнивается со значениями на первом. На тех наборах, где нет улучшения качества прогноза, считаем, что *φ=0.* Отбираем *n* лучших наборов.
4. Если *φ=0* для всех наборов или количество дескрипторов в наборе соответствует максимальному *m*, то завершаем расчет, иначе переходим ко второму шагу.

Главный член в оценке сложности алгоритма $\frac{3}{2}nm\mathrm{NM}^2$.

Вывод.

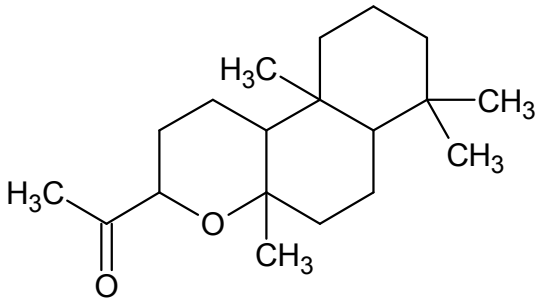
На этапе построения функциональной зависимости, для достижения наилучшего качества прогноза используются следующие параметры:

1. Количество кластеров, на которое мы разбиваем выборку
2. Значение ограничивающего радиуса в k-NN
3. Количество соседей использующихся для прогноза
4. Количество дескрипторов в каждой селекции
5. Количество селекций

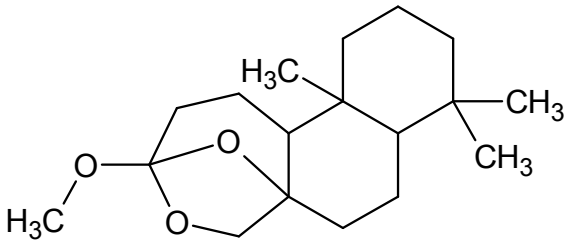
Глава 3. Результаты расчетов и их анализ.

3.1. **Выборка амбровых одорантов.**

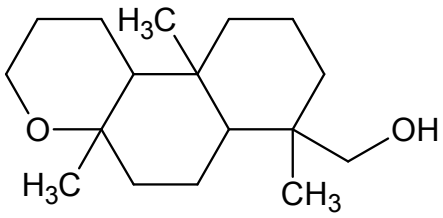
Была предоставлена выборка амбровых одорантов (низкомолекулярных соединений, обладающих амбровым запахом), состоящая из 129 молекул. Для каждого соединения выборки было дано 3D-описание соответствующего молекулярного графа в отдельном файле с расширением *.mol.* В данном файле перечислены вершины графа (атомы) с дополнительными атрибутами: символом химического элемента, трехмерные координаты в ангстремах и электрический заряд. В файле с расширением *.sdf* кроме уже описанной информации были представлены сведения о биологической активности соединений. Ниже даны примеры структурных формул некоторых из соединений выборки:

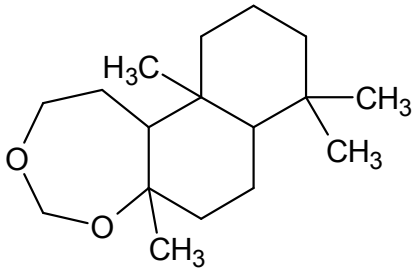


$C_{19}H_{32}O_2$



$C_{18}H_{30}O_3$





$C_{17}H_{30}O_2$

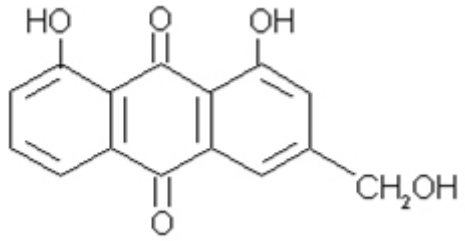
На данной выборке проводилось сравнение агоритма эволюционного алгоритма на k-NN (k-NN МГУА) с k-NN на всех признаках и на всех факторах. Приведем результаты вычислений.

	k-NN МГУА		k-NN на всех признаках		k-NN на факторах			
	прогноз	отказ	прогноз	отказ	прогноз	отказ		
1	75.9689	0	59.6899	0	59.6899	0		
2	74.4186	0	59.6899	0	60.4651	0		
3	75.1937	0	59.6899	0	63.5658	0		
4	76.7441	0	59.6899	0	59.6899	0		
5	75.9689	0	60.1562	1	61.2403	0		
6	77.5193	0	59.6899	0	60.4651	0		
7	76.7441	0	59.6899	0	61.2403	0		
8	76.7441	0	59.6899	0	61.2403	0		

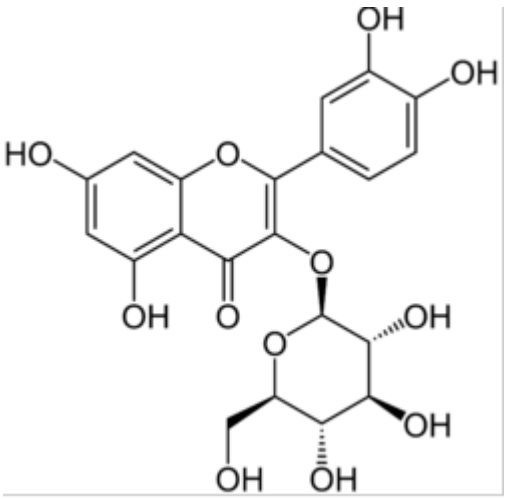
Из таблицы видно, что эволюционного алгоритма на k-NN существенно превосходит обычный k-NN на всех признаках и факторах. Прогноз полученный на факторах лучше чем на всех признаках. Отсюда можно сделать вывод, что в матрицах большинство дескрипторов отражают активность.

3.2 **Выборка гликозидов.**

Гликозиды - органические соединения, молекулы которых состоят из двух частей: углеводного (пиранозидного или фуранозидного) остатка и неуглеводного фрагмента. В качестве гликозидов в более общем смысле могут рассматриваться и углеводы, состоящие из двух или более моносахаридных остатков. Преимущественно кристаллические, реже аморфные вещества, хорошо растворимые в воде и спирте. Гликозиды представляют собой обширную группу органических веществ, встречающихся в растительном (реже в животном) мире и/или получаемых синтетическим путём. Соединения нужно было разбить на два класса активные и не активные. Приведем химические формулы некоторых гликозидов.



Алоэ – эмодин



Гликозид кверцетина

Выборка из 76 веществ извлечена из Базы данных РОНЦ, в которой содержатся структурные формулы, номенклатурные характеристики, физико-химические свойства и результаты изучения цитотоксической активности *in vitro* и противоопухолевой активности *in vivo* около 12000 оригинальных отечественных синтетических веществ и природных экстрактов, которые изучались в РОНЦ или других учреждениях России и стран СНГ [10, 10,11].

Расчеты проводились на 24 различных матрицах дескрипторов. Целью являлось сравнение прогнозов на кластерах и на всей выборке. Приведем таблицу с результатами прогноза.

№	без кластеризации			с использованием кластеризации				
				кластер 1		кластер 2		отказ
	размер	прогноз	отказ	размер	прогноз	размер	прогноз	
1	76	90.7894	0	34	100	22	100	20
2	76	94.7368	0	33	100	23	100	20
3	76	96.0526	0	31	96.7741	20	100	25
4	76	92.1052	0	32	100	22	100	22
5	76	89.4736	0	33	100	17	100	26
6	76	94.7368	0	34	100	17	100	25
7	**76**	**96.0526**	**0**	**34**	**100**	**24**	**100**	**18**
8	76	94.7368	0	24	100	31	100	21
9	76	92.1052	0	33	96.9696	19	100	24
10	76	88.1578	0	33	96.9696	20	100	23
11	76	93.4210	0	33	100	23	100	20
12	76	96.0526	0	24	100	31	100	21
13	76	90.7894	0	32	96.8750	0	100	44
14	76	90.7894	0	47	91.4893	0	100	29
15	76	92.1052	0	45	91.1111	0	100	31
16	76	90.7894	0	47	95.7446	0	100	29
17	76	89.4736	0	44	97.7272	0	100	32
18	76	86.8421	0	45	95.5555	0	100	31
19	76	93.4210	0	45	95.5555	0	100	31
20	76	90.7894	0	53	94.3396	0	100	23
21	76	92.1052	0	45	97.7777	0	100	31
22	76	89.4736	0	44	95.4545	0	100	32
23	76	89.4736	0	40	95.0000	0	100	36
24	76	92.1052	0	53	92.4528	0	100	23

Из таблицы видно, что использование разбиения на кластеры и удаление выбросов существенно улучшают качество прогноза.

3.3 **Выборка из токсичных соединений.**

Сначала скажем несколько слов о самой выборке. Химическая токсичность может вызывать множество биологически опасных эффектов, таких как повреждение генов, или даже привести смерти человека или животных. Около 120000 химических соединений будут протестированы в ближайшие 10 лет. Для этого потребуется до 45 миллионов лабораторных животных. Альтернативой может послужить прогноз токсичности на основе моделей построенных на уже протестированных соединениях. Исследуемая нами выборка поделена на две части. Первая из 644 молекул предназначена для построения моделей, а вторая (449 соединений) для их проверки. Всего имеется 5 различных признаковых пространств. Модели будем строить методом группового учета аргументов на основе k ближайших соседей с ограничением

по радиусу. Активность в отличие от предыдущих выборок не является дискретной. На данной выборке проверялось качество моделей, то так по ним будут прогнозироваться новые (не учавствовавшие в построении модели) молекулы. Ниже приведена таблица с результатами.

№	обучающая выборка		тестовая выборка			
			с выбросами		без выбросов	
	качество	отказ	качество	отказ	качество	отказ
1	0,6114	6	0,6153	7	0,618	10
2	0,7441	7	0,6877	5	0,6873	7
3	0,7297	8	0,6833	4	0,6886	4
4	0,7809	8	0,8099	8	0,809	8
5	0,7852	4	0,7605	3	0,7582	4

Из таблицы видно, что почти во всех признаковых пространствах результаты на обучающей выборке лучше, чем на тестовой. Это связано с тем, что признаки, на которых построена модель не являются универсальными. Но, тем не менее, результат ухудшился не сильно, а значит, модель может быть использована для прогноза. Удаление выбросов почти не как не сказалось на качестве прогноза. Это связано с тем, что выбросы составляют незначительную часть выборки и сходны по активности с остальными молекулами. Тем не менее, отказ от прогноза удаленных молекул может существенно улучшить конечный результат, так как большое настояние между соединениями можем означать, что они принадлежат разным классам, а нам нужно прогнозировать только сходные молекулы.

Заключение.

В дипломной работе был разработан и программно реализован эволюционный алгоритм на основе k-NN с ограничением радиуса. Так же было проведено тестирование алгоритма, проверенна эффективность эволюционного отбора, замечено улучшение качества прогноза при переходе к кластерам, отмечена пригодность полученных моделей для прогнозирования активности новых соединений.

Реализация эволюционный алгоритм на основе k-NN позволяет осуществить обратную связь между этапом поиска функциональной зависимости и построением дескрипторов. В дальнейшем, возможно, проводить детализацию дескрипторов, на которых получается наилучшее качество прогноза, что поможет сократить количество не информативных дескрипторов и ускорить работу алгоритма.

Алгоритм позволяет осуществлять прогноз любой активности: двоичной, дискретной, не дискретной.

Приведем также результаты прогноза на исследованных выборках:

1. Выборка амбровых одорантов.
   Лучший прогноз 77% правильных предсказаний, что на 16% лучше, чем прогноз, полученный с помощью обычного k-NN на всей матрице и на факторах.
2. Выборка гликозидов.
   Лучший прогноз 96% правильных предсказаний на всей выборке. В результате удаления выбросов и разбиения на кластеры удалось увеличить этот процент до 100.
3. Выборка из токсичных соединений.
   Значение функции качества для наилучшего прогноза 0.78, при этом при переходе к тестовой выборке качество прогноза увеличивается до 0.8.

Список литературы

1. Wiener H. Structural determination of paraffin boiling points. J Am Chem Soc 1947, vol.69, pp.17–20
2. Kumskov M.I., Zyryanov I.L., Svitan'ko I.V. A New Method for Representing Spatial Electronic Structures of Molecules in the Problem of Structure-Biological Activity Relationship. Pattern Recognition and Image Analysis, 1995, n.3, pp.477-484.
3. Кумсков М.И., Пономарева Л.А., Смоленский Е.А., Митюшев Д.Ф., Зефиров Н.С. Метод автоматического формирования структурных дескрипторов органических соединений для количественных корреляций «структура-активность». Изв. РАН, серия Химическая, 1994, с.1391-1393.
4. Randic M. On Characterization of Molecular Branching. Journal of the American Chemical Society, 1975, vo.97, pp.6609-6615.
5. Carhart R et al. Atom Pairs as Molecular Features in Structure-Activity Studies: Definition and Applications. J. Chem. Inf. Comput. Sci.; 1985; 25(2) pp 64–73
6. Makeev G.M., Kumskov M.I., Svitan'ko I.V., Zyryanov I.L. Recognition of Spatial Molecular Shapes of Biologically Active Substances for Classification of Their Properties. Pattern Recognition and Image Analysis, 1996, v.6, n.4. p.795-808.
7. Кумсков М.И., Смоленский Е.А., Пономарева Л.А., Митюшев Д.Ф., Зефиров Н.С. Системы структурных дескрипторов для решения задач «структура-активность». Доклады Академии Наук, 1994, 336, п.1., с.64-66.
8. Кохов В. А. Метод количественного определения сходства графов на основе структурных спектров // Известия РАН, Техническая Кибернетика. - 1994. - №5. - С.,143-159.
9. Апрышко Г.Н. Информационная система РОНЦ им. Н.Н. Блохина РАМН по противоопухолевым агентам. Общий обзор // НТИ. Сер. 2. – 2007. – № 1. –С. 18-22.
10. Апрышко Г.Н. Биологическая информация в электронной базе данных по противоопухолевым веществам НИИ ЭДИТО РОНЦ РАМН // Вестник РОНЦ. – 2007. – № 2. – С. 25-31.
11. Апрышко Г.Н., Решетникова В.В. Регистрационно-номенклатурный и химический модули электронной базы данных Информационной системы

по противоопухолевым агентам // НТИ. Сер. 2. – 2007. – №6. – С. 24-31.)

12. Moriguchi I., Kanada Y. Use of van der Waals volume in structure-activity studies. Chem. Pharm. Bull. (Tokyo), vol.25, pp.925-936, 1977.
13. Labute P. A widely applicable set of molecular descriptors. J. Mol. Graph. Mod., vol.18, pp. 464-477, 2000.
14. Rouvray D.H. (Ed.) Computational Chemical Graph Theory. / Nova Publ., New York, 1989.
15. Hoffman B., Cho S.J., Zheng W., Wyrick S., Nicols D.E., Mailman R.B., Tropsha A. Quantitative structure-activity relationship modelling of dopamine D[1] antagonists using comparative molecular field analysis, generic algorithms-partial least-squares, and K nearest neighbour methods. J.Med.Chem., vol.42, pp.3217-3226, 1999.
16. Zheng W., Tropsha A. Novel variable selection quantitative structure-property relationship approach based on the k-nearest-neighbour principle J.Chem.Inf.Comput.Sci., vol.40, pp.185-194, 2000.

Приложение.

1. **Описание модулей основых модулей.**

test8nv4.

Строит наилучшую модель для предсказания активности в рамках эволюционного алгоритма на основе k-NN метода с ограничением по радиусу.

параметры:

M- метрика (значение параметра пока не влияет на результат)

MAX - кол-во дескрипторов в выборке

n_molek- количество молекул в выборке

kol- количество дескрипторов, которое будет отобрано при каждом проходе

koef- параметр функции качества(кол-во правильно спрогнозированных молекул+koef*(кол-во отказов),если koef=2 то функция качества считается по формуле (кол-во правильно спрогнозированных молекул)/(количество молекул в выборке - кол-во отказов)*100

l_granutcua - ошибка в классификации считается допустимой, если она больше значения l_granutcua

r_granutcua - ошибка в классификации считается допустимой, если она больше значения r_granutcua

glybuna - количество селекций МГУА

kol_coced - количество соседей

Rad- правило поиска ограничения по радиусу (допустимые значения натуральные числа и 0)

0- нет ограничений по радиусу

n- радиус равен n-тому по длине ребру минимального покрывающего дерева

Вход

y.txt -фаил с активностью

result.txt- матрица дискрипторов (по строкам дескрипторы по столбцам молекулы в начале каждой строки название дескриптора)

Выход

out.txt- параметры запуска и результат (count - значение функции качества, max - кол-во правильно спрогнозированных молекул, neopr - кол-во отказов, k - количество соседей, num1 135 .... numn 227 , verno - кол-во процентов правильно предсказанных молекул в выборке)

result1.txt- матрица дескрипторов без повторений

Пример запуска

test8nv4('euk',547,56,6,0,-0.5,0.5,4,10,0)

claster.

Задача - разбиение молекул на кластеры методом минимального покрывающего дерева (МПД).

Парамеры:

num_dick- допустимые значения натуральные числа и 0
n- номер дескриптора по которому строится МПД
0- МПД строится по всем дескрипторам
kol_kl- количество кластеров, которое хотим получить
n_molek- количество молекул в выборке
radiys- максимальное расстояние между молекулами в кластере
0- любое

Вход

y.txt -фаил с активностью
result.txt- матрица дискрипторов (по строкам дескрипторы по столбцам молекулы в начале кждой строки название дескриптора)

Выход

yn.txt- фаил с активностью для n-того кластера
clastern.txt-матрица дискрипторов для n-того кластера (!!!ПО СТРОКАМ МОЛЕКУЛЫ ПО СТОЛБЦАМ ДИСКРИПТОРЫ!!!)

claster_inf.txt - распределение молекул по кластерам и условие принадлежности к кластеру.

Пример запуска

claster(0,10,644,0)

2. Листинг модулей.

test8nv4.

```
function [] = test8nv4
(M,MAX,n_molek,kol,koef,l_granutcua,r_granutcua,glybuna,kol_coced,Rad)
n_molecules=n_molek;
myfid =  fopen ('result.txt', 'r');
myformat = '%*s';
myformat2 = '%s';
for i = 1 : n_molecules
    myformat = strcat (myformat, '%f');
end
for i = 1 : n_molecules
    myformat2 = strcat (myformat2, '%*f');
end
ar = textscan (myfid, myformat2, -1);
name = ar{1, 1};
name = [name ar{1, 1}];
fclose (myfid);
myfid =  fopen ('result.txt', 'r');
cell_array = textscan (myfid, myformat, -1);
data = cell_array{1, 1};
for i = 2 : n_molecules
    data = [data cell_array{1, i}];
end
fclose (myfid);
```

```
n_molecules = n_molek;
size(data)
i1=1;
for i=1:MAX
    if(i1>MAX)
        break;
    end
    j1=i1+1;
    for j=i+1:MAX
        if(j1>MAX)
            break;
        end
        cor=coreluacua (data(i1,:),data(j1,:));
        if cor==0
            data=[data(1:j1-1,:);data(j1+1:MAX,:)];
            name=[name(1:j1-1,:);name(j1+1:MAX,:)];
            j1=j1-1;
            MAX=MAX-1;
        end
        j1=j1+1;
    end
    i1=i1+1;
end
size(data)
mkdir (fullfile (matlabroot, 'work','МГУАk-NN','out'));
fid = fopen (fullfile (matlabroot, 'work', 'МГУАk-NN','out','result1.txt'),
'w') ;
for j=1:MAX
   name1= char(name(j));
   fprintf (fid,'%s               ',name1);
   for i=1:n_molecules
        fprintf (fid,'%f ',data(j,i));
   end
   fprintf (fid,'\r\n');
end
fclose(fid);
y = textread(fullfile (matlabroot, 'work', 'МГУАk-NN','y.txt'));
y1=y(:,1);
y=y1;
size(y)
sampl = data;
for i=1:n_molecules
    for j=1:kol_coced
        Class(i,j)=0;
    end
end
for i=1:5
    for j=1:kol
        result(i,j)=0;
    end
end
num=0;
max=0;
J=0;
k1=0;
t=1;
j1=0;
filename='result.txt';
fid = fopen (fullfile (matlabroot, 'work', 'МГУАk-NN','out',filename), 'a') ;
for j=1:MAX

    sampl1=sampl(j,:);
    if Rad~=0
        raduys=opr_raduys(sampl1,Rad);
```

```
    else
        raduys=10000000000;
    end
    B=sampl1(2:n_molecules);
    y2=y(2:n_molecules,:);
    training = B';
    group = y2;
    Molekula=sampl1(1,1);
    Class(1,:) = Knnv4_1(Molekula, training, group,kol_coced,M,raduys);
    for i=2:n_molecules-1
        A=sampl1(1:i-1);
        y1=y(1:i-1,:);
        B=sampl1(i+1:n_molecules);
        y2=y(i+1:n_molecules,:);
        training = [A,B]';
        group = [y1;y2];
        Molekula=sampl1(1,i);
        if i==33
        end
        Class(i,:) = Knnv4_1(Molekula, training, group,kol_coced,M,raduys);
    end
    A=sampl1(1:n_molecules-1);
    y1=y(1:n_molecules-1,:);
    training = A';
    group = y1;
    Molekula=sampl1(1,n_molecules);
    Class(n_molecules,:) = Knnv4_1(Molekula, training,
group,kol_coced,M,raduys);
    Class1=Class;
    for i=1:kol_coced
        Class(:,i) =Class(:,i)-y;
    end
   max=-1;
    for i1=1:kol_coced
        neopr=0;
        Da=0;
        for i=1:n_molecules
            if
Class(i,i1)>l_granutcua&&Class(i,i1)<r_granutcua&&(Class(i,i1)+y(i)>0||Class(
i,i1)+y(i)<0)
                Da=Da+1;
            end
            if Class(i,i1)+y(i)==0
                neopr=neopr+1;
            end
        end
        if koef==2
            count=Da/(n_molecules-neopr)*100;
        else
        count=Da+koef*neopr;
        end
        if count>max
            max=count;
            DA=Da;
            Neopr=neopr;
            k1=i1;
            J=j;
        end
    end
    if num < kol
        num=num+1;
        result(1,num)=max;
        result(2,num)=DA;
        result(3,num)=Neopr;
```

```
        result(4,num)=k1;
        result(5,num)=J;
    else
        I=1;
        lok_min=result(1,1);
        for i=2:kol
            if lok_min>result(1,i)
                I=i;
                lok_min=result(1,i);
            end
        end
        if result(1,I) < max
            result(1,I)=max;
            result(2,I)=DA;
            result(3,I)=Neopr;
            result(4,I)=k1;
            result(5,I)=J;
        end
    end
    max=-1;

end
fclose(fid);
for i=1:kol
    for i1=1:kol-1
        if result(1,i1)< result(1,i1+1)
            s=result(:,i1);
            result(:,i1)= result(:,i1+1);
            result(:,i1+1)=s;
        end
    end
end
filename='out.txt';
fid = fopen (fullfile (matlabroot, 'work', 'МГУАк-NN','out',filename), 'a') ;
fprintf (fid,'Parametru test8v4:\r\n metruk %s kol_disk %d kol_mol %d
kol_otdor %d koef %f l_granutcua %f r_granutcua %f glybuna %d kol_coced %d
raduys %d\r\n\r\n',M,MAX,n_molek,kol,koef,l_granutcua,r_granutcua,glybuna,kol
_coced,Rad);
for i=1:num
    klast=n_molecules-result(3,i);
    if klast==0
        klast=-1;
    else
        klast=result(2,i)/(n_molecules-result(3,i))*100;
    end
    fprintf (fid,'count %f  max %d  neopr %d    k %d    num1 %d
verno %f\r\n',result(1,i),result(2,i),result(3,i),result(4,i),result(5,i),kla
st);
end
fprintf (fid,'\r\n\r\n\r\n');
fclose(fid);

max=0;
for j1=1:glybuna

[result1]=estumate_v4(M,MAX,n_molecules,num,koef,l_granutcua,r_granutcua,glyb
una,kol_coced,data,result(:,1),y,Rad);
    for j=2:num
        j

[res]=estumate_v4(M,MAX,n_molecules,num,koef,l_granutcua,r_granutcua,glybuna,
kol_coced,data,result(:,j),y,Rad);
        result1=[result1,res];
```

```
    end
    kou=0;
    i1=1;
    for i=1:kol*kol
        if result1(1,i)<= result(1,i1);
            result1(1,i)=0;
        end
        kou=kou+1;
        if(kou/kol>0.9999999999)
            i1=i1+1;
            kou=0;
        end
    end
    for i=1:kol*kol
        for i1=1:kol*kol-1
            if result1(1,i1)< result1(1,i1+1)
                s=result1(:,i1);
                result1(:,i1)= result1(:,i1+1);
                result1(:,i1+1)=s;
            end
        end
    end
    fid = fopen (fullfile (matlabroot, 'work', 'МГУАк-NN','out',filename),
'a') ;
    A=size(result1);
    N=A(1)-5;
    for i=1:num*num
        fprintf (fid,'count %f  max %d  neopr %d    k %d    num1 %d
',result1(1,i),result1(2,i),result1(3,i),result1(4,i),result1(5,i));
        for i1=1:N
            fprintf (fid,'num%d %d   ',1+i1,result1(5+i1,i));
        end
        klast=n_molecules-result1(3,i);
        if klast==0
            klast=-1;
        else
            klast=result1(2,i)/(n_molecules-result1(3,i))*100;
        end
        fprintf (fid,'verno %f\r\n',klast);
    end
    fprintf (fid,'\r\n\r\n\r\n');
    fclose(fid);
    result=result1(:,1:num);
end
```

estumate_v4

```
function [res] =
estumate_v4(M,MAX,n_molecules,kol,koef,l_granutcua,r_granutcua,glybuna,kol_co
ced,sampl,vector,y,Rad)
num=0;
t=1;
max=0;
J=0;
k1=0;
A=size(vector)-4;
N=A(1);
for i=1:n_molecules
    for j=1:kol_coced
        Class(i,j)=0;
    end
end
for i=1:N+5
    for j=1:kol
        result(i,j)=0;
```

```
    end
end

for j=1:MAX
    Da=0;
    Neopr=0;
    sampl1=sampl(j,:);
    for j1=1:N
        sampl1=[sampl1;sampl(vector(j1+4),:)];
    end
    if Rad~=0
        raduys=opr_raduys(sampl1,Rad);
    else
        raduys=10000000000000;
    end
    training =sampl1(:,2:n_molecules)';
    group = [y(2:n_molecules)];
    Molekula=sampl1(:,1)';
    Class(1,:) = Knnv4_1(Molekula, training, group,kol_coced,M,raduys);
    for i=2:n_molecules-1
        training = [sampl1(:,1:i-1),sampl1(:,i+1:n_molecules)]';
        group = [y(1:i-1);y(i+1:n_molecules)];
        Molekula=sampl1(:,i)';
        Class(i,:) = Knnv4_1(Molekula, training, group,kol_coced,M,raduys);
    end
    training = sampl1(:,1:n_molecules-1)';
    group = y(1:n_molecules-1);
    Molekula=sampl1(:,n_molecules)';
    Class(n_molecules,:) = Knnv4_1(Molekula, training,
group,kol_coced,M,raduys);
    for i=1:kol_coced
        Class(:,i) =Class(:,i)-y;
    end
    for i1=1:kol_coced
        neopr=0;
        Da=0;
        for i=1:n_molecules
            if
Class(i,i1)>l_granutcua&&Class(i,i1)<r_granutcua&&(Class(i,i1)+y(i)>0||Class(
i,i1)+y(i)<0)
                Da=Da+1;
            end
            if Class(i,i1)+y(i)==0
                neopr=neopr+1;
            end
        end
        if koef==2
            if(n_molecules-neopr==0)
                count=0;
            else
            count=Da/(n_molecules-neopr)*100;
            end
        else
        count=Da+koef*neopr;
        end
        if count>=max
            max=count;
            DA=Da;
            Neopr=neopr;
            k1=i1;
            J=j;
        end
    end
    if num < kol
```

```
        num=num+1;
        result(1,num)=max;
        result(2,num)=DA;
        result(3,num)=Neopr;
        result(4,num)=k1;
        for j1=1:N
            result(4+j1,num)=vector(4+j1);
        end
        result(5+N,num)=J;
    else
        I=1;
        lok_min=result(1,1);
        for i=2:kol
            if lok_min>result(1,i)
                I=i;
                lok_min=result(1,i);
            end
        end
        if result(1,I) < max
            result(1,I)=max;
            result(2,I)=DA;
            result(3,I)=Neopr;
            result(4,I)=k1;
            result(5+N,I)=J;
        end
    end
    max=0;
end
res=result;
```

Knnv4_1

```
function [Res] = Knnv4_1(Molekula, training, group,kol,M,raduys);
e=0.0000000001;
A=size(Molekula);
siz1=A(2);
A=size(training);
siz2=A(1);
sosed=0;
for k=1 : kol
    dist(k)=0;
    pres(k)= 0;
    Res(k)= 0;
end
for j = 1 : siz2
    s=0;
    for i = 1 : siz1
        s=s+(Molekula(i)-training(j,i))*(Molekula(i)-training(j,i));
    end
        if sosed<kol
           f=0;
           for l=1 :sosed
               if s - dist(l)<e && s - dist(l)>-e
                 pres(l)= pres(l)+group(j);
                 f=1;
               end
           end
           if f==0
                sosed=sosed+1;
                dist(sosed)=s;
                pres(sosed)=group(j);
           end
        else
           f=0;
           for l=1 :sosed
```

```
                if s - dist(l)<e && s - dist(l)>-e
                  pres(l)= pres(l)+group(j);
                  f=1;
                end
            end
            if f==0
                 num=1;
                 max=dist(1);
                 for i=1:sosed-1
                     if max<dist(i+1)
                         num=i+1;
                         max=dist(i+1);
                     end
                 end
                 if s - dist(num)<e && s - dist(num)>-e
                       pres(num)= pres(num)+group(j);
                 end
                 if s - dist(num)<-e
                     dist(num)=s;
                     pres(num)=group(j);
                 end
            end
        end
end
    for j=1 : sosed
        for i=1 : sosed-j
            if dist(i)>dist(i+1)
                s=dist(i);
                dist(i)=dist(i+1);
                dist(i+1)=s;
                s=pres(i);
                pres(i)=pres(i+1);
                pres(i+1)=s;
             end
        end
    end
for k=1 : kol
    for j=1 : k
        if(dist(j)<raduys)
            Res(k)=Res(k)+pres(j);
        end
    end
end
for i=1 : kol
    if Res(i) > e
        Res(i)=1;
    end
    if Res(i) < -e
        Res(i)=-1;
    end

end
```

Knnv4_2

```
function [Res1] = Knnv4_2(Molekula, training, group,kol,M,raduys);
e=0.0000000001;
A=size(Molekula);
siz1=A(2);
A=size(training);
siz2=A(1);
sosed=0;
group_size=4;
```

```
for k=1 : kol
    dist(k)=0;
    for j=1:group_size
        pres(k,j)= 0;
        Res(k,j)= 0;
    end
    Res1(k)=0;
end
for j = 1 : siz2
    s=0;
    for i = 1 : siz1
        s=s+(Molekula(i)-training(j,i))*(Molekula(i)-training(j,i));
    end
        if sosed<kol
           f=0;
           for l=1 :sosed
               if s - dist(l)<e && s - dist(l)>-e
                 pres(l,group(j))= pres(l,group(j))+1;
                 f=1;
               end
           end
           if f==0
                sosed=sosed+1;
                dist(sosed)=s;
                pres(sosed,group(j))=1;
           end
        else
           f=0;
           for l=1 :sosed
               if s - dist(l)<e && s - dist(l)>-e
                 pres(l,group(j))= pres(l,group(j))+1;
                 f=1;
               end
           end
           if f==0
                num=1;
                max=dist(1);
                for i=1:sosed-1
                    if max<dist(i+1)
                        num=i+1;
                        max=dist(i+1);
                    end
                end
                if s - dist(num)<-e
                    dist(num)=s;
                    for i1=1:group_size
                        pres(num,i1)= 0;
                    end
                    pres(num,group(j))=1;
                end
           end
        end
end
    for j=1 : sosed
        for i=1 : sosed-j
            if dist(i)>dist(i+1)
                s=dist(i);
                dist(i)=dist(i+1);
                dist(i+1)=s;
                s=pres(i,:);
                pres(i,:)=pres(i+1,:);
                pres(i+1,:)=s;
            end
        end
```

```
    end
for k=1 : kol
    for j=1 : k
        if(dist(j)<raduys)
            Res(k,:)=Res(k,:)+pres(j,:);
        end
    end
end
for k=1 : kol
    max=0;
    num=0;
    for j=1 : group_size
        if(max<Res(k,j))
            max=Res(k,j);
            num=j;
        end
    end
    if(num>0.1)
        Res(k,num)=0;
    end
    for j=1 : group_size
        if(max==Res(k,j))
            num=0;
        end
    end
    Res1(k)=num;
End
```

Knnv4_1

```
function [Res] = Knnv4_1(Molekula, training, group,kol,M,raduys);
e=0.0000000001;
A=size(Molekula);
siz1=A(2);
A=size(training);
siz2=A(1);
sosed=0;
for k=1 : kol
    dist(k)=0;
    pres(k)= 0;
    Res(k)= 0;
    count(k)=0;
end
for j = 1 : siz2
    s=0;
    for i = 1 : siz1
        s=s+(Molekula(i)-training(j,i))*(Molekula(i)-training(j,i));
    end
        if sosed<kol
           f=0;
           for l=1 :sosed
               if s - dist(l)<e && s - dist(l)>-e
                 pres(l)= pres(l)+group(j);
                 count(l)= count(l)+1;
                 f=1;
               end
           end
           if f==0
                sosed=sosed+1;
                dist(sosed)=s;
                pres(sosed)=group(j);
                count(sosed)=1;
           end
        else
           f=0;
           for l=1 :sosed
```

```
                if s - dist(l)<e && s - dist(l)>-e
                  count(l)= count(l)+1;
                  pres(l)= pres(l)+group(j);
                  f=1;
                end
            end
            if f==0
                 num=1;
                 max=dist(1);
                 for i=1:sosed-1
                     if max<dist(i+1)
                         num=i+1;
                         max=dist(i+1);
                     end
                 end
                 if s - dist(num)<e && s - dist(num)>-e
                       pres(num)= pres(num)+group(j);
                       count(num)= count(num)+1;
                 end
                 if s - dist(num)<-e
                     dist(num)=s;
                     pres(num)=group(j);
                     count(num)=1;
                 end
            end
        end
end
    for j=1 : sosed
        for i=1 : sosed-j
            if dist(i)>dist(i+1)
                s=dist(i);
                dist(i)=dist(i+1);
                dist(i+1)=s;
                s=pres(i);
                pres(i)=pres(i+1);
                pres(i+1)=s;
                s=count(i);
                count(i)=count(i+1);
                count(i+1)=s;
             end
        end
    end
for k=1 : kol
    if k==500
        fyfh=1;
    end
    kol_s=0;
    for j=1 : k
        if(dist(j)<raduys)
            kol_s=kol_s+count(j);
        end
    end
    for j=1 : k
        if(dist(j)<raduys)
            Res(k)=Res(k)+pres(j);
        end
    end
    if(kol_s==0)
        Res(k)=1000000;
    else
        Res(k)=Res(k)/kol_s;
    end
end
```

opr_raduys

```
function [raduys] = opr_raduys(data,Rad)
s=size(data);
for i=1:s(2)
    min(i)=10000000;
    J(i)=0;
end

for i=1:s(2)
    for j=1:s(2)
        x=0;
        if(j~=i)
            for k=1:s(1)
                x=x+(data(k,i)-data(k,j))*(data(k,i)-data(k,j));
            end
            if(min(i)>x)
                min(i)=x;
                J(i)=j;
            end
        end
    end
end
I=1;
MIN=min(1);
for i=2:s(2)
    if(min(i)<MIN)
        MIN=min(i);
        I=i;
    end
end
derevo(1,1)=I;
derevo(2,1)=0;
derevo(1,2)=J(I);
derevo(2,2)=MIN;
for i=1:s(1)
    derevo(2+i,1)=data(i,I);
    derevo(2+i,2)=data(i,J(I));
end
if(derevo(1,1)<derevo(1,2))
    data=[data(:,1:derevo(1,1)-1),data(:,derevo(1,1)+1:derevo(1,2)-
1),data(:,derevo(1,2)+1:s(2))];
else
    data=[data(:,1:derevo(1,2)-1),data(:,derevo(1,2)+1:derevo(1,1)-
1),data(:,derevo(1,1)+1:s(2))];
end
n=s(2)-2;
for i1=1:n
    s=size(data);
    d=size(derevo);
    for i=1:s(2)
        min(i)=10000000000;
    end
    for i=1:s(2)
        for j=1:d(2)
            X(j)=0;
                for k=1:s(1)
                    X(j)=X(j)+(data(k,i)-derevo(2+k,j))*(data(k,i)-
derevo(2+k,j));
                end
                if(min(i)>X(j))
                    min(i)=X(j);
                end
        end
```

```
    end
    I=1;
    MIN=min(1);
    for i=2:s(2)
        if(min(i)<MIN)
            MIN=min(i);
            I=i;
        end
    end
    derevo(1,i1+2)=I;
    derevo(2,i1+2)=MIN;
    for i=1:s(1)
        derevo(2+i,i1+2)=data(i,I);
    end
    data=[data(:,1:derevo(1,i1+2)-1),data(:,derevo(1,i1+2)+1:s(2))];
end
for i=1:Rad
    for j=1:n+1
        if derevo(2,j)>derevo(2,j+1)
            w=derevo(2,j+1);
            derevo(2,j+1)=derevo(2,j);
            derevo(2,j)=w;
        end
    end,
end
raduys=derevo(2,n+3-Rad);
```

claster

```
function [] = claster(num_dick,kol_kl,n_molek,radiys)
n_molecules=n_molek;
kol_kl=kol_kl-1;
myfid =  fopen ('result.txt', 'r');
myformat = '%*s';
myformat2 = '%s';
for i = 1 : n_molecules
    myformat = strcat (myformat, '%f');
end
for i = 1 : n_molecules
    myformat2 = strcat (myformat2, '%*f');
end
ar = textscan (myfid, myformat2, -1);
name = ar{1, 1};
name = [name ar{1, 1}];
fclose (myfid);
myfid =  fopen ('result.txt', 'r');
cell_array = textscan (myfid, myformat, -1);
data = cell_array{1, 1};
for i = 2 : n_molecules
    data = [data cell_array{1, i}];
end
fclose (myfid);
y = textread(fullfile (matlabroot, 'work', 'МГУАk-NN','y.txt'));
y1=y(:,1);
y=y1;
s=size(data)
if num_dick==0
    sz=s(1);
else
    sz=0;
end
for i=1:n_molek
```

```
    num_mol(i)=i;
end
e=0.0000000001;
for i=1:s(2)
    min(i)=10000000;
    J(i)=0;
end

for i=1:s(2)
    for j=1:s(2)
        x=0;
        if(j~=i)
            for k=1:sz
                x=x+(data(k,i)-data(k,j))*(data(k,i)-data(k,j));
            end
            if num_dick~=0
                x=x+(data(num_dick,i)-data(num_dick,j))*(data(num_dick,i)-
data(num_dick,j));
            end
            if(min(i)>x)
                min(i)=x;
                J(i)=j;
            end
        end
    end
end
I=1;
MIN=min(1);
for i=2:s(2)
    if(min(i)<MIN)
        MIN=min(i);
        I=i;
    end
end
derevo(1,1)=I;
derevo(2,1)=0;
derevo(1,2)=J(I);
derevo(2,2)=MIN;
for i=1:s(1)
    derevo(2+i,1)=data(i,I);
    derevo(2+i,2)=data(i,J(I));
end
derevo(3+s(1),1)=num_mol(I);
derevo(3+s(1),2)=num_mol(J(I));
derevo(4+s(1),1)=y(I);
derevo(4+s(1),2)=y(J(I));
if(derevo(1,1)<derevo(1,2))
    data=[data(:,1:derevo(1,1)-1),data(:,derevo(1,1)+1:derevo(1,2)-
1),data(:,derevo(1,2)+1:s(2))];
    num_mol=[num_mol(1:derevo(1,1)-1),num_mol(derevo(1,1)+1:derevo(1,2)-
1),num_mol(derevo(1,2)+1:s(2))];
    y=[y(1:derevo(1,1)-1);y(derevo(1,1)+1:derevo(1,2)-
1);y(derevo(1,2)+1:s(2))];
else
    data=[data(:,1:derevo(1,2)-1),data(:,derevo(1,2)+1:derevo(1,1)-
1),data(:,derevo(1,1)+1:s(2))];
    num_mol=[num_mol(1:derevo(1,2)-1),num_mol(derevo(1,2)+1:derevo(1,1)-
1),num_mol(derevo(1,1)+1:s(2))];
    y=[y(1:derevo(1,2)-1);y(derevo(1,2)+1:derevo(1,1)-
1);y(derevo(1,1)+1:s(2))];
end
n=s(2)-2;
for i1=1:n
    s=size(data);
```

```
    d=size(derevo);
    for i=1:s(2)
        min(i)=10000000000;
    end
    for i=1:s(2)
        for j=1:d(2)
            X(j)=0;
                for k=1:sz
                    X(j)=X(j)+(data(k,i)-derevo(2+k,j))*(data(k,i)-
derevo(2+k,j));
                end
                if num_dick~=0
                    X(j)=X(j)+(data(num_dick,i)-
derevo(2+num_dick,j))*(data(num_dick,i)-derevo(2+num_dick,j));
                end
                if(min(i)>X(j))
                    min(i)=X(j);
                end
        end
    end
    I=1;
    MIN=min(1);
    for i=2:s(2)
        if(min(i)<MIN)
            MIN=min(i);
            I=i;
        end
    end
    derevo(1,i1+2)=I;
    derevo(2,i1+2)=MIN;
    for i=1:s(1)
        derevo(2+i,i1+2)=data(i,I);
    end
    derevo(3+s(1),i1+2)=num_mol(I);
    derevo(4+s(1),i1+2)=y(I);
    data=[data(:,1:derevo(1,i1+2)-1),data(:,derevo(1,i1+2)+1:s(2))];
    num_mol=[num_mol(1:derevo(1,i1+2)-1),num_mol(derevo(1,i1+2)+1:s(2))];
    y=[y(1:derevo(1,i1+2)-1);y(derevo(1,i1+2)+1:s(2))];
end
derevo(2,:)
mkdir (fullfile (matlabroot, 'work','МГУАк-NN','out'));
cd('out');
fid=fopen('rastoyania.txt', 'w');
for j=1:n+2
    fprintf (fid,'%d\r\n',derevo(2,j));
end
fclose(fid);
MAX=0;
MIN=derevo(2,2);
if radiys==0
    for i=1:kol_kl
        max=0;
        for j=1:n+2
            if derevo(2,j)>max
                max=derevo(2,j);
                if i==1
                    MAX=max;
                end
            end
        end
        for j=1:n+2
            if (derevo(2,j)- max < e)&&(derevo(2,j)- max > -e)
                derevo(2,j)=-1;
            end
```

```
        end
    end
else
    max=radiys;
    for j=1:n+2
        if derevo(2,j)>radiys
            derevo(2,j)=-1;
        end
    end
end
max=0;
for j=1:n+2
    if derevo(2,j)>max
        max=derevo(2,j);
    end
end
num=1;
OutFile='claster';
OutFile=cat(2, OutFile, num2str(num));
OutFile=cat(2, OutFile, '.txt');
OutId=fopen(OutFile, 'w');
OutFile1='claster_inf';
OutFile1=cat(2, OutFile1, '.txt');
OutId1=fopen(OutFile1, 'w');
fprintf (OutId1,'параметры запуска  num_dick %d kol_kl %d n_molek %d
radiys %d\r\n',num_dick,kol_kl+1,n_molek,radiys);
OutFile2='y';
OutFile2=cat(2, OutFile2, num2str(num));
OutFile2=cat(2, OutFile2, '.txt');
OutId2=fopen(OutFile2, 'w');
for i=1:n+2
    if derevo(2,i)+1==0
        fprintf (OutId1,'\r\n');
        fclose(OutId);
        fclose(OutId2);
        num=num+1;
        OutFile='claster';
        OutFile=cat(2, OutFile, num2str(num));
        OutFile=cat(2, OutFile, '.txt');
        OutId=fopen(OutFile, 'w');
        OutFile2='y';
        OutFile2=cat(2, OutFile2, num2str(num));
        OutFile2=cat(2, OutFile2, '.txt');
        OutId2=fopen(OutFile2, 'w');
        for j=1:s(1)
            fprintf (OutId,'%d   ',derevo(j+2,i));
        end
        fprintf (OutId1,'%d   ',derevo(s(1)+3,i));
        fprintf (OutId2,'%d   ',derevo(s(1)+4,i));
        fprintf (OutId2,'\r\n');
        fprintf (OutId,'\r\n');
    else
        for j=1:s(1)
            fprintf (OutId,'%d   ',derevo(j+2,i));
        end
        fprintf (OutId1,'%d   ',derevo(s(1)+3,i));
        fprintf (OutId2,'%d   ',derevo(s(1)+4,i));
        fprintf (OutId2,'\r\n');
        fprintf (OutId,'\r\n');
    end
end
fprintf (OutId1,'\r\n');
MIN
```

```
fprintf (OutId1,'%d   условие кластеризации MAX %d
MIN %d',sqrt(max),sqrt(MAX),sqrt(MIN));
fprintf (OutId1,'   условие кластеризации проверка   %d',max);
max
MAX
fprintf (OutId1,'\r\n');
fclose(OutId1);
fclose(OutId);
fclose(OutId2);
cd('..');
```