2 слайд

Основные понятия и принципы машинного анализа данных (Machine Learning) произошли от более общего понятия Искусственного интеллекта (Artificial Intelligence), направленного на изучение искусственного поведения, обучения и адаптации.

Как правило, на вход системе подаются обучающие примеры, результаты экспериментов, частные случаи более общего явления, и требуется обобщить накопленный опыт в виде некоего знания, которое бы согласовывалось с имеющимися результатами, и способно было описывать новые факты.

В Machine Learning такое знание представимо нейронной сетью с фиксированным набором весов, решающим деревом или индуктивно-логическими правилами.

3 слайд

Определение. Будем говорить, что программа обучается на эксперименте **E** в классе задач **T** при данной мере выполнения **P**, если её выполнение задач **T** улучшается (в смысле **P**) с использованием опыта **E.**

Примером такой задачи **T** может являться задача “играть в шашки”.

4 слайд

Обучение с учителем (supervised learning). При таком типе обучения известны значения аппроксимируемой функции на некотором ограниченном наборе данных. Ставится задача построения гипотезы, которая была бы согласована с обучающими данными, и обладала предсказательной силой. Предположим, нам известны значения двумерной функции f в четырёх точках. По этим точкам восстановим параболическую поверхность h такую, что $h \approx f$ для этих точек. Точного совпадения значений f и h не требуется. В таком случае, поверхность h является нашей гипотезой относительно f.
Эта парадигма обучения работает в случае задач распознавания образов, классификации, предсказания.

Обучение без учителя (unsupervised learning). При данной парадигме обучения значения целевой функции априори не известны. Обучающие данные представлены векторами в многомерном пространстве, требуется выделить подмножества векторов, в соответствии с некоторым принципом. Эта парадигма применяется для задач кластеризации (иерархическая, агломеративная, самоорганизующиеся карты) при разном уровне детализации знаний, и, как частный случай кластеризации, для задач уменьшения разнообразия данных за счёт выделения прототипов и снижения размерности признакового пространства. Так как окончательный результат априори не известен, остро стоит проблема оценки качества полученной кластеризации.

5 слайд

Допустим, требуется описать концепцию “дни, благоприятные для игры в теннис”

Обучающие данные представлены в таблице.

Каждый день описан четырьмя признаками, которые принимают значения из фиксированного множества значений. Для некоторых комбинаций значений признаков известно, является ли день с такими характеристиками благоприятным для игры в теннис. Это есть множество обучающих примеров концепции.

Требуется при любой возможной комбинации значений уметь предсказывать значение концепции PlayTennis.

6 слайд

Формально, пусть имеется $X = \langle X_1, ..., X_n \rangle$ - набор m-мерных векторов (n – количество обучающих примеров, m - количество признаков).

Целевая концепция $\text{PlayTennis}: X \to \langle Yes, No \rangle$ представлена конечным набором реализаций $D = \langle X_i, PlayTennis(X_i) \rangle, i = 1...n$ - множество обучающих данных.

Требуется найти такую гипотезу $h: X \to \langle Yes, No \rangle$, $h \in H$, что для $\forall X_i \;\; h(X_i) = PlayTennis(X_i), i = 1...n$.

H – пространство решающих деревьев.

7 слайд

Определение. Деревом решений называется дизьюнкция коньюнкций значений признаков.

Дерево следует строить и интерпретировать снизу вверх, от корня к листьям. Промежуточные узлы дерева (не листовые) являются тестом для признаков. Каждое значение признака задаёт новую ветвь. Лист дерева содержит классификационную метку, значение целевой функции для фиксированного набора значений признаков вдоль соответствующей ветки.

Ветка дерева представляет собой коньюнкцию значений признаков и вместе со значением функции преобразуется в правило ЕСЛИ-ТО. Например,

ЕСЛИ Outlook = Sunny **И** Humidity = Normal, **ТО** PlayTennis = Yes.

Дерево представляет собой совокупность правил, дизьюнкцию коньюнкций значений признаков.

(Outlook = Sunny **AND** Humidity = Normal) **OR**
(Outlook = Overcast) **OR**
(Outlook = Rain **AND** Wind = Weak)

Булева функция, которую описывает дерево, однозначно восстанавливается по положительным реализациям.

8 слайд. Алгоритм ID3.

Осуществляет направленный поиск (greedy search) в пространстве возможных деревьев.

Конструирует дерево последовательно, от корня к листьям, каждый раз отбирая признак, который наилучшим образом (в смысле информационной меры) разделяет обучающие примеры.

Без обратного хода, т.е. не возвращается, не меняет локального решения.

9 слайд. Сведения из теории информации. Энтропия.

Математическая формализация понятия об информации, которую ввели в первой половине XX века Л. Хартли и К. Шеннон относится к числу величайших достижений науки.

Информация есть устранённая неопределённость для достижения цели.

Энтропия (информационная) — мера хаотичности информации, неопределённость появления какого-либо символа первичного алфавита. При отсутствии информационных потерь численно равна количеству информации на символ передаваемого сообщения.

Так, возьмём, например последовательность символов, составляющих какое-либо предложение на русском языке. Каждый символ появляется с разной частотой, следовательно, неопределённость появления для некоторых символов больше, чем для других.

Информационная энтропия для независимых случайных событий x с n возможными состояниями (от 1 до n) рассчитывается по формуле:

$$H(x) = -\sum_{i=1}^{n} p(i) \log_2 p(i)$$

Эта величина также называется *средней энтропией сообщения*. Величина $\log_2 \frac{1}{p(i)}$ называется *частной энтропией*, характеризующей только i-е состояние. Таким образом, энтропия события *x* является суммой с противоположным знаком всех произведений относительных частот появления события *i*, умноженных на их же двоичные логарифмы (основание 2 выбрано только для удобства работы с информацией, представленной в двоичной форме). Это определение для дискретных случайных событий можно расширить для функции распределения вероятностей.

Шеннон вывел это определение энтропии из следующих предположений:

- мера должна быть непрерывной; т. е. изменение значения величины вероятности на малую величину должно вызывать малое результирующее изменение энтропии;
- в случае, когда все варианты (буквы в приведенном примере) равновероятны, увеличение количества вариантов (букв) должно всегда увеличивать полную энтропию;
- должна быть возможность сделать выбор (в нашем примере букв) в два шага, в которых энтропия конечного результата должна будет являтся суммой энтропий промежуточных результатов.

Шеннон показал, что любое определение энтропии, удовлетворяющее этим предположениям, должно быть в форме:

$$-K \sum_{i=1}^{n} p(i) \log_2 p(i)$$

где *K* — константа (и в действительности нужна только для выбора единиц измерения).

Шеннон определил, что измерение энтропии ($H = -p_1 \log_2 p_1 - \ldots - p_n \log_2 p_n$), применяемое к источнику информации, может определить требования к минимальной пропускной способности канала, требуемой для надежной передачи информации в виде закодированных двоичных чисел. Для вывода формулы Шеннона необходимо вычислить математическое ожидание «количества информации», содержащегося в цифре из источника информации. Мера энтропии Шеннона выражает неуверенность реализации случайной переменной. Таким образом, энтропия является разницей между информацией, содержащейся в сообщении, и той частью информации, которая точно известна (или хорошо предсказуема) в сообщении. Примером этого является избыточность языка — имеются явные статистические закономерности в появлении букв, пар последовательных букв, троек и т.д.

10 слайд. Энтропия.

Пусть:

S - набор обучающих примеров, $S \subseteq D$

p_+ - часть позитивных примеров в S

p_- - часть негативных примеров в S

Тогда энтропия множества S вычисляется как:

$$Entropy(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-).$$

Число бит информации, требуемое для того, чтобы закодировать произвольный элемент *S*. Т.е. тот минимальный объём информации, которым надо располагать, чтобы закодировать принадлежность к классу произвольного элемента из *S*.

Заметим, что энтропия равна нулю, если все элементы *S* – положительные/отрицательные.

Энтропия равна единице, максимальна, если положительных и отрицательных примеров поровну.

11 слайд. Обучающие данные.

12 слайд. Информационная мера признака.

Для реализации алгоритма ID3 необходимо на каждом шаге уметь определять признак, который наилучшим образом классифицирует примеры.

Для этого вводится понятие информационной меры признака *X* (мера эффективности классификации на основании признака *X*).

$$\text{Gain}(S,X)=\text{Entropy}(S) - \sum_{v\in \text{values}(X)} |S_v|/|S| \text{ Entropy}(S_v),$$

values(*X*) - набор возможных значений признака *X*.

$$S_v = \left\langle s \in S \middle| X(s) = v \right\rangle,$$

такие примеры из *S* , для которых значение признака *X* равно *v* .

|*S*|- мощность множества *S*.

Заметим, что второе слагаемое есть ожидаемое значение энтропии после разбиения *S* признаком *X*, с весом.

Gain(*S*,*X*) – это количество бит информация о значении целевой функции при известных значениях признака *X*, или, количество бит информации, которое экономится при кодировании значения целевой функции произвольного члена *S* при условии, что известно значение признака *X*.

Для выбора корневого признака решающего дерева тестируем все имеющиеся признаки:

Gain(*S*, Outlook) = 0.246

Gain(*S*, Temperature) = 0.029

Gain(*S*, Humidity) = 0.151

Gain(*S*, Wind) = 0.048

Признак Outlook обладает большей информационной мерой, следовательно, алгоритм ID3 объявляет его корнем и выпускает ветки в соответствии с количеством значений признака.

13 слайд. Алгоритм ID3.

Далее поиск осуществляется локально, с учетом заданной ветки.

Вычисляем информационную меру признаков при фиксированном значении корневого признака:

$\text{Gain}(S_{\text{sunny}}, \text{Humidity})=0.970-(3/5)0.0 – 2/5(0.0) = 0.970,$

S_{sunny} − подмножество обучающих примеров из *D*, для которых значение признака Outlook равно Sunny.

$\text{Gain}(S_{\text{sunny}}, \text{Temperature})=0.970-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570,$

$\text{Gain}(S_{\text{sunny}}, \text{Wind})=0.970= -(2/5)1.0 – 3/5(0.918) = 0.019.$

Терминальными условиями алгоритма ID3 являются:

- вдоль ветки исчерпались все признаки

-энтропия равна нулю (примеры однозначно классифицируются)

14 слайд. Априорные предположения алгоритма ID3.

Поиск решения осуществляется в пространстве деревьев в условиях индуктивного предположения: “предпочесть более короткие деревья, листья которых ближе к корню обладают большей информационной мерой”. Данное предположение является адаптацией известного методологического принципа “бритвы Оккама”: “Сущности не должны быть умножены без необходимости”, что означает “предпочесть наиболее простую гипотезу, которая удовлетворяет данным”.

Комментарии к алгоритму ID3:

- ID3 осуществляет поиск в пространстве деревьев, которые формализуют класс конечных, булево-значных функций.

- Не отвечает на вопрос, сколько всего деревьев описывает обучающую выборку. Находит одно из…

- Алгоритм “жадный”, т.е. есть опасность прийти к локально-оптимальному решению. Однажды выбрав признак, алгоритм не возвращается назад.

- Использование информационной меры признака в основе алгоритма обеспечивает устойчивость алгоритма к шумам в данных.

Преимуществом решающих деревьев является проста и ясность их применения, естественная интерпретация результата классификации. По сравнению с другими методами, деревья устойчивы к шумам. Достоинством так же является простота языка метода и богатый класс функций, представимых деревьями.

Однако, деревья трудно оптимизировать (метод построения дерева исключает обратный ход), и как следствие, есть опасность прийти к локально-оптимальному решению. Остро стоит проблема переобучения, зависящая от длины дерева. В книге Митчелл (Mitchell, 1997) обсуждается проблема поиска оптимальной длины предсказывающего дерева (основанная на скользящем контроле). К ограничениям так же относится то, что вид решающих правил задается априори.

Пример функций, плохо представимых деревьями (language bias):

- - *Parity*: output is true if an even number of attributes are true
- *Majority*: output is true if more than half of the attributes are true

15 слайд.

Анализ регуляторных последовательностей генов представляет собой актуальную проблему биологии и вызов теории анализа данных и машинного обучения (Data Mining and Machine Learning). Действительно, сложность экспериментальных исследований регуляции экспрессии, наличие большого количества зашумленных и неполных данных, включая предсказанные с некоторой степенью достоверности признаки, требуют адекватных математических подходов для решения этой проблемы.

В статье (Wang, D. *et al*, 2001) при помощи метода деревьев решается задача классификации белков в соответствии с их функциями на основании комбинации мотивов в последовательностях белков.

Предсказание функции неизвестных белков является важнейшей задачей функциональной геномики. Ряд работ [Hudak and McClure, 1999] указывает на то, что функция белка коррелирует с высоко-консервативными участками аминокислот, мотивами. Созданные базы экспериментальных данных (Prosite [Hoffman et al., 1999], Pfam [Bateman et al., 2000]) содержат мотивы (кластеры мотивов) и при запросе определённого мотива способны возвращать функцию, ассоциированную с этим мотивом. Однако такая модель является слишком упрощенной, т.к. многие белки содержат несколько мотивов, и, в общем случае, функция белка определяется паттерном мотивов, присутствием/отсутствием определенных мотивов.

16 слайд. Модель испытаний.

Последовательности белков с известными функциями разделяются на обучающие и контрольные данные. На основании алгоритма построения решающих деревьев и используя набор обучающих данных, получаем дерево классификации белков, точность которого оценивается на контрольных данных. Итоговое дерево представляет собой совокупность правил, позволяющих относить последовательности новых белков к известным функциональным семействам в соответствии с расположением консервативных мотивов.

17 слайд. Представление данных.

Каждая аминокислотная последовательность белка должна быть преобразована в соответствующую запись значений признаков, т.е. представлять строку в таблице "объект-признак". Выбор признаков является критическим шагом в построении модели. В данном случае каждая последовательность должна быть представлена в терминах присутствия/отсутствия определенных мотивов из подходящего словаря мотивов.
Предположим, что словарь содержит N мотивов. Тогда каждую последовательность кодируем двоичной записью длины N, причём на i-том месте стоит единица, если соответствующий мотив представлен в последовательности, в противном случае ставится ноль. Каждая такая строка ассоциирована со своим функциональным классом.

18 слайд. Построение решающих деревьев.

Обучающие данные содержали 585 белков, принадлежащих к 10 известным функциональным классам, и одному негативному классу (false positive). Решающие деревья строились для разных объёмов обучающих данных (11, 20…585 последовательностей). Каждый эксперимент, для фиксированного объёма данных, проводился три раза, весь объём данных случайным образом делился на обучение и контроль. Усреднённая точность (для 3-х независимых запусков) решающих деревьев, в зависимости от объёма обучающих данных, представлена в таблице. Из таблицы видно, что, обучившись всего на 10% данных, решающее дерево способно предсказывать с высокой точностью.

В соответствии с базой данных Prosite каждый функциональный класс представлен одним, или более, мотивами. С другой стороны, каждому мотиву соответствует уникальная запись в базе, т.е. один и тот же мотив не может характеризовать два разных класса. Анализ полученного дерева показывает, что роль каждого мотива критична для классификации. В связи с этим встает вопрос, не является ли полученное оптимальное решающее дерево простым переписыванием информации из базы, или всё же содержит дополнительное знание? Проведённый анализ ошибок передпредсказания (второго рода) показал, что ошибки второго рода для деревьев, начиная с определённого объёма обучающих данных (40%), значительно меньше ошибки, получаемой при запросе через базу данных. Таким образом, полученное дерево решений открывает закономерности расположения мотивов в белковых последовательностях, которые явно не содержаться в базе данных.

20 слайд. Построение решающих деревьев для белковых семейств, содержащих общие мотивы.

Обучающие данные содержали 73 последовательности белков, принадлежащих к 5 известным функциональным классам. Выбранные классы имели пересекающиеся множества функциональных мотивов, т.е. один мотив мог характеризовать два разных класса. В этом случае использование базы данных Prosite приводит к большим ошибкам второго рода. В то время как решающее дерево при использовании всего 30% данных для обучения, имеет высокую точность (95%) распознавания классов.

Пример дерева, построенного для 58 обучающих белков, приведён на рисунке. Это решающее дерево различает белки, принадлежащие классам PDOC00360 и PDOC00605,

на основании присутствия мотива PS50064 первом классе, но не во втором, хотя оба класса содержат мотив PS50010.

Решающе дерево представляют собой более адекватную модель функциональных классов, чем модель, основанную на базе данных консервативных мотивов. Решающее дерево обнаруживает взаимосвязи между присутствием/отсутствием определённых мотивов, в общем случае, достаточно удалённых друг от друга. Такие взаимосвязи играют критическую роль в определении 3-х мерной структуры и функции белков.

21 слайд. Генетические алгоритмы.

Идея алгоритма основана на биологической эволюционной теории, парадигме Дарвина, и заключается, во первых, в продукции гипотез в соответствии с мутационными и рекомбинационными биологическими событиями, и, во вторых, в отборе конкурентоспособных кандидатов, которые лучше адаптированы к окружающей среде.

Следует заметить, что генетические алгоритмы и входящие в них понятия есть чисто математические термины, которые стали использоваться в компьютерной биологии относительно недавно, гораздо позже, чем в физике и других науках.

Схема алгоритма показана на рисунке.

Инициализировать случайным образом популяцию гипотез;

Для каждой гипотезы популяции оценить её степень соответствия данным, предсказательную способность, так называемую функцию Fitness;

Если значение Fitness для гипотез меньше некоторого порога, то организовывать новую популяцию гипотез;

Иначе предложить гипотезу с максимальным значением функции Fitness.

22 слайд. Представление гипотез.

В генетических алгоритмах гипотезы представляются бинарными строками для удобной манипуляции операторами мутации, кроссинговера.

Признаки записываются по порядку, и для каждого признака резервируется количество мест в соответствии с его значениями.

Например, гипотеза:

(Outlook = Overcast v Rain) AND (Wind = Strong) THEN PlayTennis = yes

перепишется как:

011 10 10

Признак Outlook принимает три значения, следовательно, для него зарезервировано три места, в данной гипотезе Outlook = Overcast или Outlook = Rain, т.е. на соответствующих местах стоят единицы и т.д.

При воздействии операторами мутации и кросссовера на гипотезы, последняя воспринимается как строка символов, а не как осмысленная запись гипотезы, в которой каждое место зарезервировано под значение конкретного признака. Таким образом, могут возникнуть бессмысленные гипотезы (например, содержащие две единицы на конце), которые автоматически наделяются низким значением функции Fitness.

Полученные представления гипотез имеют одинаковую длину (даже если нет ограничений на значения некоторого признака, всё равно он учитывается в записи гипотезы), что важно при интерпретации окончательно выбранной гипотезы.

23 слайд. Генетические операторы.

Одноточечный кроссовер.

Гипотезы-родители разрезаются в одной точке, и обмениваются частями. У двух родителей возникает два потомка. Точка разрезания выбирается случайным образом.

Двухточечный кроссовер.

Случайным образом выбирается две точки разрезания.

24 слайд. Генетические операторы.

Оператор кроссовер в общем виде.

Маска кроссовера – бинарная строка, генерится случайным образом и обозначает, какая информация должна быть взята от каждого из родителей.

Оператор мутация.

Случайным образом выбирается одно из значений, и переключается.

25 слайд. Параметры алгоритма.

Основные параметры генетического алгоритма:

Функция **Fitness**, степень соответствия гипотезы данным, предсказательная способность гипотезы. Например,

$$\text{Fitness}(h)=(\text{correct}(h))^2,$$

correct(h)- точность классификации гипотезой данных, часть обучающих примеров, верно классифицированных гипотезой h.

Порог соответствия, **threshold**.

p: число гипотез в текущей популяции

r: часть гипотез, замещаемая оператором кроссинговер

m: уровень мутаций

26 слайд. Формирование популяции.

Выбрать (1-r)p членов популяции $P_{current}$ и добавить к популяции P_{new}. Вероятность выбора индивида:

$$P(h_i) = \text{Fitness}(h_i) / \Sigma_j \text{Fitness}(h_j)$$

Таким образом, вероятность того, что гипотеза будет выбрана пропорциональна её значению функции Fitness, и обратно пропорциональна значениям функции Fitness для других конкурирующих гипотез в популяции. Выбор, осуществляемый в соответствии с заданной таким образом вероятностью, называется выбором по правилу рулетки (roulette wheel selection), имитируя нанесение на рулетку областей, пропорциональных значениям функции Fitness.

Кроссовер- выбрать rp/2 пар гипотез из $P_{current}$ в соответствии с $P(h_i)$. Для каждой пары (h1,h2) произвести два потомка посредством применения оператора кроссовер и добавить к P_{new}.

Мутация. Выбрать m членов P_{new} с равномерным распределением. Изменить один бит в записи на случайно выбранном месте.

Заменить $P_{current}$ на P_{new}

Для каждой гипотезы h оценить значение функции Fitness.

27 слайд. Поиск в пространстве гипотез.

Поиск гипотез в генетическом алгоритме характеризуется как “внезапный”, “хаотичный”, рандомизированный (траектория алгоритма не может быть предсказана, даже если известна отправная точка) в отличие от направленного поиска гипотез, от гипотез специального вида к более общему, например.

Имеет место феномен перенаселения (crowding), соответствующий быстрому воспроизводству небольшого числа наиболее приспособленных гипотез (с максимальным значением функции Fitness). В этом случаем, большую часть популяции составляют “копии” одной гипотезы, что значительно уменьшает разнообразие популяции.

Для преодоления этого феномена используют, другие критерии выбора гипотез, например, турнир (tournament selection), по правилам которого две гипотезы популяции выбираются случайным образом, из них с предустановленной вероятностью p выбирается одна из гипотез, и с (1-p) другая. Другой подход заключается в назначении штрафов гипотезам, имеющим копии в популяции.

Для увеличения разнообразия популяции проделывают так называемый акт “геноцида”, когда 80% популяции замещается случайными гипотезами, таким образом добавляются различные “сущности”.

Терминальными условиями алгоритма являются

- отсутствие событий
- значение максимальной приспособленности не меняется

28 слайд. Эволюция популяции.

Теоретическое обоснование генетических алгоритмов основано на понятии схемы.

Определение. Схемой является запись, состоящая из 0, 1, и *, и представляющая собой набор бинарных строк, в записи которых стоит 0 и 1 в соответствии со схемой, и произвольные значения на местах, обозначенных *.

Например, схема * * 1 * * 0 0 * * 1 представляет все строки длин 10, где на третьем и последнем месте стоят единицы, а на шестом и седьмом нули. Бинарные строки, удовлетворяющие условиям схемы, называются представителями схемы.

Теорема схем (Schema Theorem) исследует эволюцию популяции в генетических алгоритмах в терминах эволюции схем, т.е. изменения числа индивидов, представляющих схему.

Вероятность того, что схема изменится при мутациях зависит от числа фиксированных бит в схеме (порядок схемы). Чем более жесткая, определённая схема (высокий порядок), тем больше вероятность быть разрушенной в результате мутаций.

Вероятность того, что схема изменится при воздействии оператора кроссовер зависит от расстояния между первым и последним фиксированными битами. Чем больше это расстояние, тем больше вероятность того, что разрезание операции произойдёт прямо между битами, и, таким образом, потомки перестанут представлять ту же схему, что и родители.

Итак, схемы низкого порядка с небольшими расстояниями обладают высокой степенью выживаемости при смене поколений.

В процессе генетического алгоритма явным образом оценивается приспособленность каждой из гипотез, и, как следствие (неявным образом), оценивается выживаемость каждой схемы.

Прежде чем сформулировать саму теорему введём некоторые обозначения:

- H – схема, имеющая по крайней мере одного представителя на момент времени t.
- $m(H, t)$ – количество представителей H на момент t.
- $f(H, t)$ - среднее значение функции Fitness гипотез из H на момент t.
- $f(x)$ – значение функции Fitness для произвольной гипотезы x;
- $\overline{f}(t)$ - среднее значение функции Fitness для популяции на момент t.
- Мы хотим вычислить $E(m(H, t + 1))$, ожидаемое число представителей схемы H на момент времени $t + 1$.

29 слайд. Эволюция популяции, выбор гипотез.

Эволюция популяции в генетическом алгоритме определяется тремя этапами: выбором гипотез, рекомбинацией и мутацией.

При условии отсутствия двух последних шагов, т.е. ограничиваясь случаем, когда в следующую популяцию попадают гипотезы с вероятностью, пропорциональной их приспособленности, ожидаемое число представителей схемы H на момент времени $t + 1$ вычисляется следующим образом:

$$E\left(m\left(H,t+1\right)\right)=\sum_{x\in H}\frac{f\left(x\right)}{\bar{f}\left(t\right)}=m\left(H,t\right)\sum_{x\in H}\frac{f\left(x\right)}{m\left(H,t\right)\cdot\bar{f}\left(t\right)}=m\left(H,t\right)\cdot\frac{f\left(H,t\right)}{\bar{f}\left(t\right)},$$

где $x\in H$ означает, что гипотеза x является представителем схемы H .

Таким образом, ожидаемое число представителей схемы в следующий момент времени пропорционально среднему значению функции приспособленности в настоящий момент времени, и обратно пропорционально среднему значению функции приспособленности для всей популяции в настоящий момент времени. Согласно теореме, со временем начинают преобладать более приспособленные схемы.

Теперь вычислим значение $E(m(H, t + 1))$ с учетом эффекта операторов мутации и рекомбинации. Если учитывать только максимально негативный эффект, то мы получим оценку снизу для $E(m(H, t + 1))$. Получим оценку только для случая одноточечного кроссовера.

- p_{xc} - вероятность того, что одноточечный кроссовер будет применен к строке
- $Sc(H)$ – вероятность того, что схема H выживет после применения одноточечного кроссовера, т.е. по крайней мере один из потомков представителя H останется представителем H.
- $L(H)$ - расстояния между первым и последним фиксированными битами схемы H.
- l – длина гипотезы.

30 слайд. Эволюция популяции. Эффект одноточечного кроссовера.

Вероятность того, что схема будет разрушена не хуже, чем:

$$p_{xo}\left(\frac{L(H)}{l-1}\right),$$

что позволяет оценить вероятность того, что схема выживет при воздействии одноточечным кроссовером:

$$S_c(H) \geq 1 - p_{xo}\left(\frac{L(H)}{l-1}\right).$$

31 слайд. Эволюция популяции. Эффект мутаций.

Теперь необходимо оценить эффект мутаций, привносимый в разрушение схемы при формировании новой популяции.

- p_m - вероятность мутации одного бита.
- $\boldsymbol{o(H)}$ – порядок схемы $\boldsymbol{H}$.
- $\boldsymbol{S_m(H)}$ – вероятность того, что схема $\boldsymbol{H}$ выживет при мутации представителя $\boldsymbol{H}$.

Имеет место неравенство:

$$S_m(H) \geq (1 - p_m)o(H)$$

Итак, используя выше полученные неравенство, получаем оценку снизу для значения $E(m(H, t + 1))$, ожидаемое число представителей схемы H на момент времени t+1:

$$E\left(m(H,t+1)\right) \geq \frac{f(H,t)}{\bar{f}(t)} \cdot m(H,t) \cdot \left(1 - p_{xo} \cdot \frac{L(H)}{l-1}\right) \cdot (1-p_m)^{o(H)}.$$

Грубая интерпретация этого неравенства заключается, в том, что наибольшей степенью выживаемости и способностью разрастаться обладает наиболее приспособленная схема (средняя приспособленность схемы сопоставима со средней приспособленностью популяции в каждый момент времени), содержащая меньшее число определённых бит и кратчайшее расстояние между первым и последним битами.

32 слайд.

Такие важнейшие проблемы биоинформатики, как проблема выравнивания генетических последовательностей, проблема сборки фрагментов ДНК, картирования генов, сводятся к так называемой NP-трудной проблеме (нелинейная полиномиальная оценка числа итераций) построения гамильтонова цикла графа (цикл в графе, проходящий через каждую вершину в точности один раз).

Пусть *1*, *2*,..*n* – метки *n* городов и $C = [c_{i,j}]$ - матрица стоимостей $c_{i,j}$ между городами, *i* и *j*. Требуется найти наиболее дешевый маршрут, соединяющий все города, т.е. минимизировать функционал (совокупную стоимость перемещений):

$$A(n) = \sum_{i=1}^{n-1} c_{i,i+1} + c_{n,1}$$

33 слайд. Применение генетических алгоритмов к задаче сборки фрагментов ДНК.

Проблема сборки фрагментов ДНК, связанная с секвенированием ДНК, заключается в получение консенсусной последовательности по ДНК фрагментам, по своей природе, является NP-трудной задачей.

Цепь ДНК, длиннее 500 оснований не может быть секвенирована без большой погрешности. Таким образом, ДНК реплицируется несколько раз, и индивидуальные цепочки двойной спирали случайным образом разрезаются на небольшие фрагменты, которые секвенируется без сохранения их порядка на родительской цепи. Порядок определяется степенью близости между соседними фрагментами. Восстановление последовательности по 4-м фрагментам показано на рисунке.

Важно отметить, что в отличие от задачи с городами, где маршрут является циклическим, нам важно знать отправную и конечную точки маршрута, также важен порядок.

Применение генетического алгоритма:

- Пусть имеется *n* фрагментов цепи ДНК *1*, *2*,..*n*, и известна матрица близости всех попарно взятых фрагментов. Метод динамического программирования обеспечивает лучшее выравнивание двух фрагментов, лучшее, в смысле различных вариантов пересечения и ориентации.
- $f_1, f_2, ..., f_n$ - некое упорядочивание фрагментов, в котором нет повторов, и запись $f_i = j$ означает, что фрагмент i начинается в позиции j результирующей последовательности (которую определяет упорядоченный набор фрагментов). Функция Fitness результирующей последовательности определяется как:

$$Fitness = \sum_{i=1}^{n-1} W_{f_i, f_{i+1}},$$

где $W_{f_i, f_{i+1}}$ - значение матрицы близости фрагментов f_i и f_{i+1}.

Т.е. имеющиеся фрагменты случайным образом разбросаны вдоль отрезка заданной длины. Этот упорядоченный набор фрагментов является гипотезой относительно консенсусной последовательности. Далее к данному типу гипотез применяются операторы кроссовера и мутации с учётом того, что бы в результирующей последовательности не было повторяющихся фрагментов.

Окончательная консенсусная последовательность получается на основании порядка, обладающего максимальным значением функции Fitness.

34 слайд. Применение генетических алгоритмов к задачам биоинформатики.

Генетические алгоритмы использовались для:

- задач множественного выравнивания (Notredame *et al.*, 1996);
- распознавания структуры РНК (Shapiro *et al.*, 2001);
- извлечения регуляторных генных сетей из профилей экспрессии генов (Ando *et al.*, 2001);
- распознавания сайтов связывания транскрипционных факторов
- разработки медицинских препаратов (Douguet *et al*., 2000).

Вообще, генетические алгоритмы предназначены для решения сложных многопараметрических задач оптимизации, где прямой перебор параметров затруднен или невозможен.

Генетические алгоритмы используются автономно, например, применительно к вышеуказанным задачам и как, принцип, вовлекаются в другие методы для улучшения скорости их сходимости и поиска оптимального решения. В том числе существует направление генетического программирования, где членами популяции являются самостоятельные компьютерные программы (Poli *et al*., 2002). Генетический алгоритм это изящный метод, претендующий на решение задач высоких размерностей.

Несмотря на то, что предсказательная способность окончательного решения может быть достоверно оценена, сам способ получения решения, его динамика в процессе итераций алгоритма, не является осмысленной.

1. Mitchell, T. (1997). Machine Learning. New York: McGraw Hill.
2. Hudak, J. and McClure, M.A. (1999) A Comparative Analysis of Computational Motif Detection Methods. Pacific Symposium on Biocomputing 4:138-149 (1999).
3. Hofmann K., Bucher P., Falquet L., Bairoch A. (1999) The PROSITE database, its status in 1999 Nucleic Acids Res. 27:215-219.
4. Bateman, A. Birney, E.., Durbin, R., Eddy, S., Howe, K., and Sonnhammer, E. (2000) Nucleic Acids Research, 28:263-266.
5. Wang, D., Wang, X., Honavar, V., and Dobbs, D. (2001). Data-Driven Generation of Decision Trees for Motif-Based Assignment of Protein Sequences to Functional Families.
6. Pal S.K. Bandyopadhyay S. Ray S.S. (2006) Evolutionary Computation in Bioinformatics: A Review, IEEE Transactions on Systems, Man, and Cybernetics, Part-C, 36(5), 601-615.
7. Notredame C., Higgins D.G. (1996) SAGA: Sequence alignment by genetic algorithm, Nucleic Acids Res., 24(8): 1515-1524.
8. Shapiro B.A., Wu J.-C., Bengali D., Potts M.J. (2001) The massively parallel genetic algorithm for RNA folding: MIMD implementation and population variation. Bioinformatics 17(2): 137-148.
9. Ando S., Iba H. (2001) Inference of gene regulatory models by genetic algorithms. Proceedings of the 2001 IEEE Congress on Evolutionary Computation: 712-719.
10. Douguet D., Thoreau E., Grassy G. (2000) A genetic algorithm for the automated generation of small organic molecules: drug design using an evolutionary algorithm. J Comput Aided Mol Des., 14(5):449-66.
11. Poli R., Langdon W.B. (2002) Foundations of Genetic Programming. Springer.