ПОДХОД К АВТОМАТЗИРОВАННОМУ ОТБОРУ ЗНАЧИМЫХ ПРИЗНАКОВ И ОЦЕНКЕ РЕЗУЛЬТАТОВ В ЗАДАЧЕ ПОИСКА ГЕНОМНЫХ МАРКЕРОВ ЛЕКАРСТВЕННОЙ УСТОЙЧИВОСТИ МИКРООРГАНИЗМОВ

В.В. Сатаневский
Студент БГУ, инженер компании InData Labs

Белорусский государственный университет, Республика Беларусь
E-mil: satanevsky.vlad@yandex.ru

Abstract. This paper is devoted to the automated selection of significant features and result estimation, paying special attention on the possibility of overfitting. The estimation was made on the task of revealing of genome-wide markers of microorganism drug-resistance mutations.

Введение. Развитие высокопроизводительных методов секвенирования дает значительный толчок биологическим исследованиям и становлению персонализированной медицины. Получение полного генетического кода живых организмов стало гораздо более доступным.

Однако, несмотря на это, использование генетического кода довольно затруднительно. Эти трудности связаны с тем, что влияние отдельных аллелей и нуклеотидов на различные процессы, происходящие в живых организмах, изучены лишь частично. В связи с этим, большое распространение получили алгоритмы машинного обучения, позволяющие автоматически находить закономерности в геноме, влияющие на исследуемые процессы.

При этом изучаемая задача имеет некоторые особенности, затрудняющие объективное тестирование различных подходов к её решению.

В докладе будет рассмотрена исходная задача, некоторые подходы к её решению, возникшие в процессе работой над задачей проблемы и предложенный способ их решения. Также будут рассмотрены некоторые способы и конкретная библиотека, реализующая один из рассмотренных подходов, позволяющая реализовать автоматизированный выбор наилучшей модели и её параметров.

Постановка задачи. Рассматривается задача полногеномного поиска ассоциаций, где анализируются мутации (однонуклеотидные полиморфизмы) в последовательностях ДНК микробактерий туберкулеза. Цель заключается в нахождении таких участков генома, мутации в которых влияют на наличие либо отсутствие лекарственной устойчивости к определенному препарату. Для

определения мутаций использовалось сравнение геномной последовательности с некоторой референсной последовательностью. Любое отличие символа в исходной последовательности и символа референсной последовательности считалось мутацией.

На вход поступает описание мутаций в последовательностях ДНК микробактерий туберкулеза. Для более наглядного представления введем матрицу X размера $n \times m$, где n – число наблюдаемых объектов, m – число различных позиций, в которых наблюдались мутации. Элемент матрицы x_{ij} равен 1, если у объекта под номером i наблюдалась мутация в j-й по счету позиции среди тех, в которых наблюдалась мутация хотя бы одном среди имеющихся n объектов, $x_{ij} = 0$, если в этой же позиции не наблюдается мутация и $x_{ij} = -1$, если не удалось установить, была ли мутация в соответствующей позиции. Информация о чувствительности к определенному препарату закодирована в векторе y, где $y_i = 1$, если установлена лекарственная устойчивость i-го объекта к выбранному препарату, $y_i = 0$, если у соответствующего объекта установлена лекарственная чувствительность к препарату и $y_i = -1$, если сведения отсутствуют.

Выходные данные представляют собой следующее:

5 Список мутаций, влияющих на чувствительность или устойчивость к препарату.

6 Модель, использующая описанный выше список мутаций, позволяющая для новых геномов микробактерий предсказывать, являются ли они устойчивыми или чувствительными к лекарственному препарату

Существующие подходы и возникшие проблемы. Для решения подобной задачи часто используют методы машинного обучения. Существуют методы, позволяющие как отбирать значимые признаки (в нашем случае, значимые мутации), так и строить модели для предсказания лекарственной устойчивости для новых объектов.

Данная нам задача анализа лекарственной устойчивости обладает рядом своих особенностей. Среди них малое число объектов и большое число признаков. К нашему сожалению, именно в таких ситуациях велик риск переобучения. В связи с этим, чтобы полученным результатам можно было доверять, важно построение правильного окружения, позволяющее проводить различные эксперименты. Также, для одних и тех же объектов даны результаты проверки устойчивости к нескольким различным препаратам.

Также хотелось бы, чтобы различные подходы к решению задачи проверялись автоматически (ведь различных препаратов у нас несколько), объективно, а результаты проверки не зависели от конкретных гиперпараметров рассматриваемых подходов.

Опишем основные пожелания к тестированию наших подходов в виде требований к тестирующему окружению, которых будем придерживаться при его проектировании и реализации:

1 Независимость от гиперпараметров. То есть, для наших экспериментов мы задаём лишь структуру нашей модели, но не указываем конкретные значения параметров. Это важно для того, чтобы тестировать структуру моделей, а не умение подбора параметров.

2 Подбор параметров эксперимента проводится независимо от тестирования. Это означает, что тестовые данные не участвуют при подборе гиперпараметров. То есть, тестирование должно проводиться ровно один раз, после того, как параметры перебраны. Это важно, потому что, если, перебирая параметры моделей (ручным или автоматическим способом), мы будем иметь информацию о качестве модели на тестовых данных, очень просто переобучиться. Обычно, этому уделяют не так много внимания, потому что в выборках достаточно много объектов и значительные изменения качества модели с большой вероятностью свидетельствуют именно о лучшем качестве, а не о переобучении.

3 Устойчивость. Это означает, что оценка качества должна слабо зависеть от разбиения выборки на обучающую и тестовую.

Построение тестирующего окружения. В связи с этими требованиями, построение тестирующей системы мы решили производить следующим образом.

1 На вход поступают данные по некоторому лекарству, модель, а также пространство неинициализированных параметров этой модели.

2 Используя эту модель и пространство параметров мы создаем так называемую мета-модель, у которой уже нет никаких параметров, кроме переданной модели и пространства возможных значений параметров (на самом деле, мета-модель как раз инкапсулирует в себе подбор этих самых гиперпараметров).

3 Мета-модель тестируется на имеющихся данных. На выход поступают результаты тестирования – метрики качества мета-модели и отобранные ею признаки.

Визуальное описание архитектуры тестирующего окружения можно видеть на рисунке 1.

Тестирование мета-модели проводилось посредством скользящего контроля по 5 блокам. В качестве метрик качества использовались правильность, F_1, точность, полнота, матрица неточностей.

Заметим, что такой способ тестирования обеспечивает нам выполнения сразу трех пунктов. Во-первых, так как тестируется мета-модель, выполняется требование 1, ведь мета-модель не зависит от гиперпараметров. Так как мета-модель сама является моделью, а тестирование методом скользящего контроля по 5 блокам удовлетворяет условиям 2 и 3, они также выполнены.

Мета-модель представляет собой модель, которая принимает на вход другую модель и пространство её гиперпараметров, подбирает наилучшие гиперпараметры и использует их для обучения и предсказания. Рассмотрим это более детально.

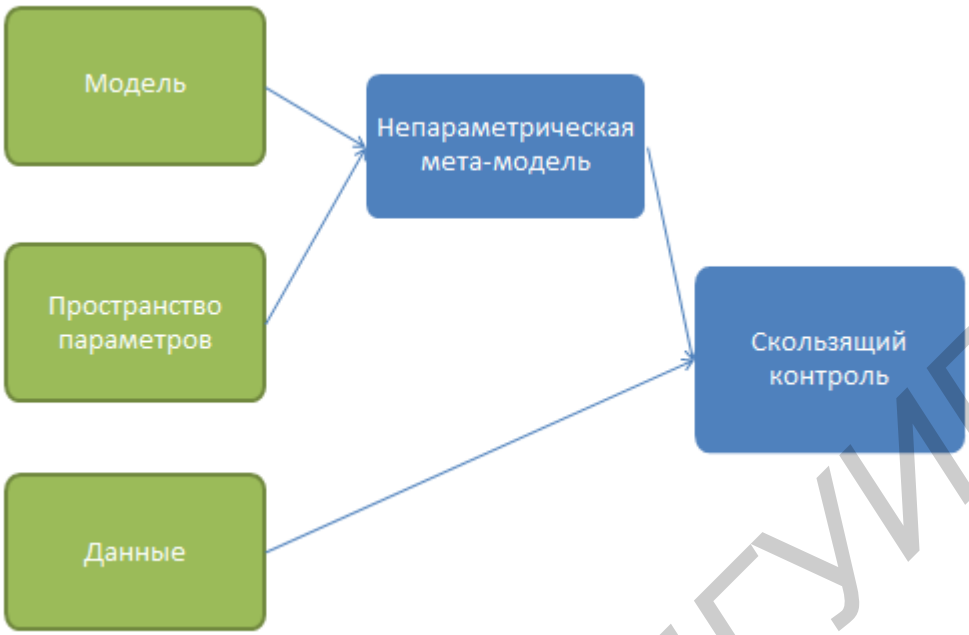

Рис. 1. Структура тестирующего окружения

При обучении мета-модели на вход поступают тренировочные данные – признаковое описание объектов и скрытые переменные. Обучение модели состоит из следующих последовательных стадий:

1 Автоматический подбор оптимальных гиперпараметров с помощью скользящего контроля по 5 блокам.

2 Обучение внутренней модели с использованием гиперпараметров, подобранных на шаге 1.

При предсказании мета-модели на вход поступают тестовые данные, представленные признаковым описанием тестовых объектов. Скрытые переменные при этом неизвестны. В этом случае производится предсказание скрытых переменных с использованием обученной на стадии обучения внутренней модели.

Автоматический подбор гиперпараметров. Зачастую, алгоритм машинного обучения имеет набор параметров (который называется гиперпараметрами), от правильности задания которого зависит качество результирующей модели. Поэтому, для того, чтобы оценить, какой метод машинного обучения лучше, гиперпараметры подбираются достаточно близко к оптимальным (в той мере, в которой это удалось), затем сравниваются алгоритмы, инициализированные наилучшими найденными параметрами. Поэтому, для того, чтобы сравнить алгоритмы правильно, необходимо уметь правильно выбирать гиперпараметры. Поэтому, рассмотрим основные методы, позволяющие выбрать гиперпараметры близко к оптимальным значениям. Далее, рассмотрим некоторые способы подбора гиперпараметров:

1 Поиск по сетке. Представляет собой исчерпывающий поиск среди параметров, заданных на некоторой сетке (вообще говоря, не равномерной)

2 Случайный поиск, как и поиск по сетке, представляет собой перебор некоторого множества вариантов гиперпараметров, только, в отличие от поиска по сетке, эти варианты генерируются из некоторого распределения, заданного на вход алгоритму.

3 Методы байесовской оптимизации. На данный момент это наиболее передовые методы, обеспечивающие сходимость за небольшое число итераций. Именно поэтому мы и использовали их при автоматизации подбора гиперпараметров. Рассмотрим их далее.

Методы, основанные на байесовской оптимизации являются адаптивными, отличие которых от всех рассмотренных ранее методов заключается в том, что выбор вариантов гиперпараметров для проверки осуществляется с использованием результатов предыдущих проверок.

В данном случае мы решаем задачу максимизации метрики качества. При байесовском подходе искомая метрика качества рассматривается как случайная функция, для которой задано некоторое априорное распределение. Далее, при вычислении значения этой функции в некоторой точке, вычисляется апостериорное распределение этой случайной функции, которое в дальнейшем используется в качестве априорного. При выборе следующей точки используют некоторый критерий и априорное распределение на значение рассматриваемой случайной функции (самым распространенной метрикой в этом случае является “матожидание улучшения качества”).

В отличие от случайного поиска, использование результатов предыдущих вычислений позволяет существенно ускорить сходимость. Правда, недостатком метода является вычислительно более сложный выбор следующей точки, что не подходит в случаях, когда метрика качества вычисляется очень быстро (в таком случае лучше воспользоваться случайным поиском).

Библиотека hyperopt. В работе мы использовали алгоритм дерева парзеновских оценок (Tree of Parzen Estimators), реализованный в библиотеке hyperopt языка Python. Основным отличием рассматриваемого алгоритма и библиотеки является следующее:

1 Возможность подбирать различные типы параметров (действительные, дискретные упорядоченные, дискретные неупорядоченные).

2 Параметры можно задавать древовидно, что позволяет перебирать лишь необходимые параметры. Например, мы можем выбирать модель машинного обучения среди, например, логистической регрессии и случайного леса, при этом, если на некоторой итерации мы выбрали логистическую регрессию, далее мы будет перебирать лишь ее параметры (например, величину и тип регуляризации), а выбрав случайный лес, перебирать его параметры (например, глубину деревьев, критерий разделения).

3 Возможность распределенного подбора параметров. При этом создается управляющий процесс (master), создающий задачи, вычислением которых занимаются вычислительные процессы (worker), которые могут находиться на различных физических узлах, принимая команды по сети. Координация работы, а

также сохранение окончательных и промежуточных результатов производится через MongoDB.

7. Проведенные эксперименты и полученные результаты.

Далее, опишем проведенные эксперименты и их результаты:

1 Базовая модель (логистическая регрессия, случайный лес или градиентный бустинг).

2 Отбор признаков (статистический либо на основе модели) и базовая модель.

3 Сеть релевантных признаков (более подробно о ней можно почитать в) и базовая модель.

Результаты проведенных экспериментов:

Таблица 1. Результаты

		AMIK: Amikacin	CAPR: Capreomycin	ETHI: Ethionamide/ Prothionamide	OFLO: Ofloxacin	PARA: Para-aminosalicyclic acid
Accuracy	a	0,89	0,83	0,75	0,83	0,88
	b	0,89	0,82	0,79	0,87	0,88
	c	0,88	0,82	0,76	0,85	0,88

Литература

[1]. Сергеев Р.С. и др. Алгоритмы поиска мутаций лекарственной устойчивости в геномах микобактерий туберкулеза // Информатика. - 2016. - № 1(49). - С. 75-91.

[2]. Liu, J. High-Dimensional Structured Feature Screening Using Binary Markov Random Fields / J. Liu [et al.]. // JMLR workshop and conference proceedings. – 2012. - №22. – P. 712–721.