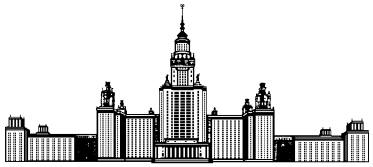
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

КУРСОВАЯ РАБОТА СТУДЕНТА 317 ГРУППЫ

« Логические корректоры в задаче восстановления регрессии »

Выполнил:

студент 3 курса 317 группы

Байтеков Никита Вячеславович

Научный руководитель:

д.ф-м.н., доцент

Дюкова Елена Всеволодовна

Москва, 2017

Содержание

1 Введение

Рассмотрим множество объектов M, которые описываются набором целочисленных признаков $\{x_1, ..., x_n\}$. Каждому объекту соответствует значение из множества ответов Y. Дано обучающее множество объектов $T = \{S_1, ..., S_m\}$ из M, такое, что для каждого обучающего объекта(прецедента) известно значение ответа. Требуется построить алгоритм распознавания $A_T : M \to Y$, ставящий в соответствие произвольному объекту из M ответ из Y. В случае, когда $Y = \mathbb{R}$, решается задача восстановления регрессии, а при $Y = \{0, 1, \ldots, l\}$ — задача классификации.

В работе рассматривается задача восстановления регрессии, для решения которой существует много методов. Чаще всего используется многомерная линейная регрессия, когда зависимость ответа от признаков предполагается линейной. Хорошо показывает себя метод ближайших соседей, который исходит из предположения, что у похожих объектов будут похожие ответы. Для решения задачи также используются адаптации методов классификации. В качестве примера можно рассмотреть вариации решающих деревьев, которые стабильно показывают высокое качество полученных решений.

Одной из современных моделей классификации является логический корректор [1], в основе которого лежат идеи как логического(использование семейств простых логических моделей), так и алгебраического подходов(взаимное корректирование ошибок различных алгоритмов)[2]. Метод хорошо зарекомендовал себя при решении прикладных задач. В [5] разработан стохастический вариант логического корректора MONS. Целью данной работы является адаптация этого алгоритма к задаче восстановления регрессии.

Введём основные определения. Задано обучающее множество T, и для каждого прецедента из T известно, к какому из l непересекающихся множеств $\{K_1, ..., K_l\}$, называемых *классами*, он принадлежит. Пусть $T \cap K$ — множество прецедентов из T, принадлежащих классу K, а $T \cap \overline{K}$ — множество прецедентов из T, не принадлежащих классу K. Конъюнкция вида $P_{(H,\sigma)} = x_{j_1}^{\sigma_1} ... x_{j_r}^{\sigma_r}$ называется *элементарным классификатором(эл.кл.)*, где $H = \{x_{j_1}, ..., x_{j_r}\}$ — набор целочисленных признаков, $\sigma = \{\sigma_1, ..., \sigma_r\}$ — набор допустимых значений этих признаков, а r — *ранг эл.кл.*. Эл.кл. считается *корректным для класса* K, если $\nexists (S', S'')$ — пары обучающих объ-

ектов, $S' \in K, S'' \notin K$, такой, что $P_{(H,\sigma)}(S') = P_{(H,\sigma)}(S'') = 1$. Объект S *содержит эл.кл.* $P_{(H,\sigma)}$, если $P_{(H,\sigma)}(S) = 1$. Пусть $U = \{P_{x_{j_1},\sigma_{j_1}}, ..., P_{x_{j_q},\sigma_{j_q}}\}$ - *набор эл.кл.*. Набор эл.кл. U называется *монотонно корректным для класса* K, где $K \in \{K_1, ..., K_l\}$, если $\forall(S', S'')$ — пары обучающих объектов, $S' \in K, S'' \notin K$ существует монотонная функция алгебры логики $F_{U,K}$: $F_{U,K}(U(S')) \neq F_{U,K}(U(S''))$. В таком случае, функция $F_{U,K}$ называется *монотонно корректирующей* для U. Монотонно корректный набор U для класса K является *неприводимым*, если любой набор эл.кл., являющийся подмножеством U, не является монотонно корректным. Неприводимый набор для класса K является *минимальным*, если не существует монотонно корректных наборов для класса K меньшей мощности.

2 Логические корректоры в задаче классификации

2.1 Алгоритм MON

Одним из первых логических корректоров, показывающих хорошие результаты, является алгоритм MON [4], который использовал монотонно корректные наборы, состоящие из эл.кл. ранга 1.

Процесс работы корректора MON разбивается на два этапа: обучение модели и распознавание. На этапе обучения для каждого класса K, $K \in \{K_1, ..., K_l\}$, распознающий алгоритм A строит семейство монотонно корректных наборов эл.кл. $W_A(K)$. При распознавании для каждого объекта S вычисляется оценка принадлежности этого объекта к каждому из классов K по формуле:

$$\Gamma_K(S) = \frac{1}{|W_A(K)|} \sum_{U \in W_A(K)} \frac{1}{|T \cap K|} \sum_{S' \in T \cap K} \delta_U(S, S').$$

где $\delta_U(S, S')$ называется *функцией голосования* и определяется как:

$$\delta_U(S, S') = \begin{cases} 1, & \text{если } \omega_U(S) \succeq \omega_U(S'); \\ 0, & \text{иначе.} \end{cases}$$

Алгоритм относит неизвестный объект S к тому классу, для которого оценка принадлежности будет максимальной.

2.2 Алгоритм MONS

Алгоритм MONS является улучшенной моделью алгоритма MON, который использует т.н. *локальный базис* — множество эл.кл., на основе которого базовые алгоритмы строят наборы эл.кл. Использование локального базиса является компромиссом между желанием использовать в логическом корректоре эл.кл. ранга больше 1 и трудоёмкостью вычислений при поиске монотонно корректных наборов эл.кл. Формирование локального базиса само по себе является непростой задачей, и каждый алгоритм решает её по-разному. В нашем случае MONS использует стохастические методы, каждый раз формируя случайно небольшой локальный базис и на его основании составляя новые наборы эл.кл., которые добавляются к растущим семействам.

Для поиска минимальных наборов эл.кл. используется следующий алгоритм. Пусть $LB = \{P_{(H_1,\sigma_1)}, \ldots, P_{(H_{N_K},\sigma_{N_K}),}\}$ – локальный базис из N_K эл.кл., составленный для некоторого фиксированного класса $K \in \{K_1, ..., K_l\}$. Рассмотрим всевозможные пары объектов вида (S', S'') и сопоставим каждой паре бинарный вектор вида $B(S', S'') = (b_1, \ldots, b_{N_K})$, где:

$$b_i(S', S'') = \begin{cases} 1, & \text{если } P_{(H_i,\sigma_i)}(S') = 1 \text{ и } P_{(H_i,\sigma_i)}(S'') = 0; \\ 0, & \text{иначе.} \end{cases}$$

Составим булеву матрицу L_K из всевозможных строк вида $B(S', S'')$, где $S' \in T \cap K$, а $S'' \in T \cap \overline{K}$. Заметим, что каждый столбец L_K соответствует определённому эл.кл. из LB, так что набор таких столбцов задаёт набор эл.кл. Очевидно, что любой набор эл.кл. U_K является монотонно корректным тогда и только тогда, когда набор признаков $H = \{H_1, \ldots, H_{N_K}\}$ является *покрытием матрицы* L_K – таким набором столбцов, что любая строка на пересечениях с этими столбцами имеет хотя бы одну единицу. Более того, любому неприводимому(минимальному) набору эл.кл однозначно соответствует неприводимое(минимальное) покрытие. Таким образом мы свели задачу к поиску неприводимого покрытия булевой матрицы, которой известен как приложение задачи дуализации [**?**]. Для решения используются приближённые алгоритмы поиска, такие как градиентный спуск или генетический алгоритм, так как задача дуализации относится к трудно решаемым перечислительным задачам, и полиномиальных алгоритмов для её решения не было открыто до сих пор.

Градиентный алгоритм хорошо показывает себя на случайных булевых матрицах, но сильно ошибается на «разряженных» матрицах, которые содержат относительно малое количество единиц. Этого недостатка лишён генетический алгоритм, который и используется в данной работе.

При обучении алгоритма MONS исходное обучающее множество T (или просто обучающая выборка) разбивается на две части - усечённую обучающую выборку T' и валидационную выборку V. После выше описанного алгоритма для получения минимальных наборов эл.кл. для каждого класса требуется проверить критерий останова, ведь найдено лишь приближённое решение, а первой итерации может не хватить. Зададим отступ объекта S как разность между оценками истинного и предсказанного классов. Тогда алгоритм остановится, когда прирост качества для всех объектов станет меньше фиксированного ε:

Algorithm 1 Процедура обучения алгоритма MONS

Вход: T — обучающая выборка, max_i — максимальное число итераций, ε — порог;

Выход: $W_K, K \in \{K_1, \ldots, K_l\}$ — семейства мон. корректных наборов;

$V := V(T)$; // *случайное выделение выборки $V \subset T$ в качестве валидационной*

$T' := T \backslash V$; // *усечение обучающей выборки*

для $i = 1, \ldots, max\_i$

  для всех $K \in \{K_1, \ldots, K_l\}$

    $LB := LB()$; // *случайным образом сформировать локальный базис*

    $W_K^i := GA(T', LB, K)$; // *ГА возвращает семейство мон. наборов из LB*

    $W_K := W_K \cap W_K^i$; // *добавляем полученные наборы в семейство W_K*

  для всех $S \in V$

    $M_i(S) := \Gamma_{y(S)}(W^i_{y(S)}, S) - \max\limits_{K \in \{K_1, \ldots, K_l\} \backslash y(S)} \Gamma_K(W_K^i, S)$ // *отступ*

    $M_i^{avg}(S) := \frac{1}{i} \sum_{j=i}^{i} M_j(S)$; // *средний отступ по предыдущим итерациям*

  если $\forall S \in V M_i^{avg}(S) - M_{i-1}^{avg}(S) < \varepsilon$ **то**

    выход

2.3 Генетический алгоритм

На каждом шаге генетического алгоритма обновляется множество приближённых решений задачи, называемое *популяцией* в ходе итераций, называемых *поколениями*. Элементы популяции также называются особями. Качество найденного приближенного решения характеризуется *функцией приспособленности*. При развитии популяции можно придерживаться различных подходов, к примеру, стратегия частичной замены подразумевает, что часть популяции переходит в следующее поколение без изменений. При таком подходе популяция постепенно избавляется от «наименее приспособленных» особей, но для этого необходимо поддерживать разнообразие особей, иначе генетический алгоритм преждевременно сойдется, попав в локальный минимум. Получение новых особей происходит с помощью операторов скрещивания и мутации, которые имеют явные параллели со своими биологическими аналогами.

В данном случае особями популяции являются покрытия булевой матрицы, кодирующие соответственные наборы эл.кл. Функция приспособленности для набора эл.кл. U задаётся следующей формулой:

$$\begin{cases} f(U_i) = \tau(U_i) - \min\limits_{j \in \{1,\dots,N\}} \tau(U_j) + 1, \\ \tau(U_i) = \frac{1}{|T' \cap K|} \sum_{S \in T' \cap K} \frac{1}{|V \cap K|} \sum_{S' \in V \cap K} \delta_{U_i}(S, S'), \end{cases}$$

где T' - усеченная обучающая, а V - валидационная выборки. Функцию $f(U_i)$ требуется максимизировать — это важно, так как при разных функциях приспособленности могут возникнуть задача как максимизации, так и минимизации. *Начальная популяция* формируется из случайных наборов столбцов, которые, если понадобится, дополняются до покрытий. Если случайно сгенерированная особь уже присутствует в начальной популяции, то ничего не добавляется. *Выбор подпопуляции для скрещивания* осуществляется с помощью метода рулетки, когда вероятность выбора зависит от значения функции приспособленности у особи:

$$p_i = \frac{1/f_i}{\sum_{j=1}^{N_P} 1/f_j}$$

Такая формула позволяет вычислять относительное значение приспособленности вместо абсолютного, предотвращая проблему равных вероятностей выбора особей поздних популяций. На каждой итерации для дальнейшего скрещивания выбирает-

ся ровно одна пара родителей, так как в этом случае лучшие особи остаются в популяции и сразу же могут участвовать в следующих скрещиваниях (метод *последовательной замены*). Новая особь получается с помощью взвешенного однородного кроссовера, когда значение каждой хромосомы копируется в набор потомка из набора одного из родителей с вероятностью, пропорциональной их значениям функции приспособленности. В алгоритме также используется особый оператор мутации, который повышает вероятность мутации с течением времени по следующей формуле:

$$k(t) = k_0 \left(1 - \frac{1}{C \cdot t + 1}\right),$$

где $k(t)$ - количество мутируемых хромосом на шаге t, k_0 - количество мутируемых хромосом на последнем шаге алгоритма, коэффициент C отражает скорость стремления $k(t)$ к k_0. После чего замещаем потомком одну из наименее приспособленных особей, попутно восстанавливая допустимость решения, так как после операторов скрещивания и мутации полученный набор столбцов может не являться неприводимым покрытием или покрытием в принципе. В качестве критерия останова используется *сходимость популяции с задержкой*: если прирост качества популяции меньше заранее заданного порога ε определенное число t_{tc} шагов подряд, то выполнение алгоритма прерывается. Это понижает вероятность преждевременной сходимости, так как у алгоритма больше шансов, мутировав, «выбраться» из локального минимума.

3 Сведение задачи восстановления регрессии к задаче классификации

Для применения алгоритма MONS в задаче восстановления регрессии предлагается адаптировать нашу задачу к задаче классификации. Мы можем либо исходить из предположения, что у похожих объектов близкие значения целевой переменной, либо не допускать подобного. В первом случае логично разбить объекты(точки в пространстве признаковых описание) по их признаковым описаниям на несколько кластеров, каждому из которых будет присвоена своя метка класса, и сведём задачу к предыдущей. Во втором случае мы можем разбить на классы исходя только из значения самой целевой переменной.

3.1 Алгоритм DM-DBSCAN

Для разбиения данных на кластеры можно использовать разные алгоритмы, но важно, чтобы алгоритм умел самостоятельно определять количество кластеров, присутствующих в выборке, отфильтровывал выбросы, умел работать с нелинейной геометрией данных а также был быстрым и простым в реализации. Таким параметрам удовлетворяет алгоритм DM-DBSCAN[6] — динамический плотностной алгоритм кластеризации пространственных данных с присутствием шума. Он, как и его предшественник DBSCAN, основывается на предположении, что внутри каждого кластера наблюдается типичная плотность объектов, которая заметно выше плотности снаружи, в то время как шумовые объекты разрежены сильнее, чем информативные.

Согласно алгоритму, объекты разбиваются на три типа: «ядровые», «граничные» и «шумовые». Для каждой точки в заранее заданном радиусе ε считается число соседних – если оно больше некоторого порога, то она находится в середине предполагаемого кластера и называется *ядровой*. В таком случае ей присваивается некоторая метка класса, и процесс рекурсивно повторяется для всех её соседей, которые ещё не были обработаны. Если же соседей меньше, чем нужно, но в их числе есть хоть один ядровой объект, то текущий объект является *граничным*. После первоначальной разметки(когда обрабатываются только ядровые точки) граничным объектам присваиваются метки ближайших ядровых. В противном случае, если в окрестности

слишком мало соседей, а также нет ядровых точек, то объект помечается как *шумовой*. Стоит отметить, что у алгоритма DBSCAN два входных параметра: радиус распознавания соседей ε и порог для определения ядровых точек n_{min}. Но так как плотность получается фиксированной, то DBSCAN не умеет обрабатывать кластеры с различной плотностью. DM-DBSCAN лишён этих недостатков, так как оценивает уровни плотности каждого из кластеров по графику кривой расстояний до k-го ближайшего соседа.

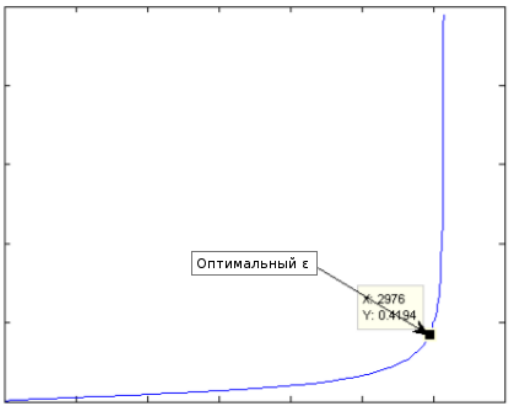


Рис. 1: Кривая k-расстояний для одного уровня плотности

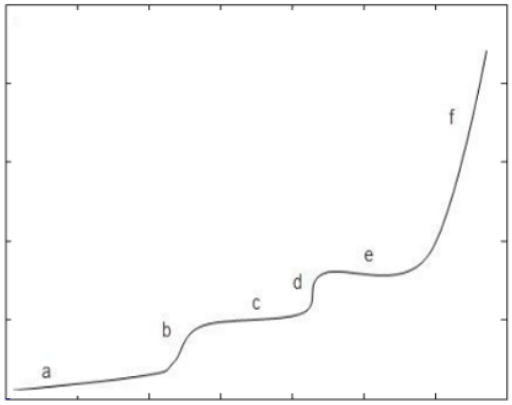


Рис. 2: Кривая k-расстояний для нескольких уровней плотности

На графиках можно наблюдать горизонтальные участки, соответствующие уровням плотности в данных, в то время как вертикальные участки соответствуют уровням шумовых точек. Для определения оптимальных значений используются точки перемены знака второй производной графика, которая вычисляется по стандартной разностной схеме.

После разбиения на кластеры используется алгоритм MONS для решения уже знакомой задачи классификации, настраивая модель. При распознавании нового объекта алгоритм MONS в качестве ответа выдаст один из кластеров, к которому объект, скорее всего, принадлежит. Найдём значение целевой функции объекта, основываясь на значениях объектов из данного кластера. Есть несколько вариантов поведения, например брать среднее значение или медиану всех значений целевой переменной в классе или у некоторых объектов в нашем классе, которые наиболее похожи на тестируемый — можно использовать разные метрики, в зависимости от типа данных.

Algorithm 2 Алгоритм DM-DBSCAN

Вход: T — множество данных из n точек, n_{min} — минимальное число соседей;

Выход: Y — вектор разметки данных по кластерам;

$D = dist\_mat(T)$; // *матрицы расстояний по выбранной метрике*

$D^d = k\_deriv(D)$; // *вторых производных графика k-расстояний*

$E = \emptyset, N = \emptyset$; // *инициализация множеств порогов плотностей и шумов*

для $i = 1, \ldots, n-1$

  если $D_i^d < 0$ **и** $D_{i+1}^d > 0$ и $D_i^d - D_{i+1}^d < \delta$ **то**

    $E = E \cup \{\frac{D_i^d + D_{i+1}^d}{2}\}$ // *добавляем в множество порогов плотностей*

  иначе если $D_i^d > 0$ **и** $D_{i+1}^d < 0$ и $D_i^d - D_{i+1}^d < \delta$ **то**

    $N = N \cup \{\frac{D_i^d + D_{i+1}^d}{2}\}$ // *или добавляем в множество порогов шумов*

$PQ = \emptyset, C = 0$; // *очередь обработки и переменная для меток класса*

$PD = \emptyset, BD = \emptyset$; // *множества обработанных и граничных точек*

для $i = 1, \ldots, n$

  если $type(T_i, E, N) =$ «ядровая точка» **то**

    $Y_{T_i} = C$; // *присвоим новую метку класса*

    $PD = PD \cup \{T_i\}$; // *помечаем точку как обработанную*

    $PQ = PQ \cup (neighboors(T_i, E) \backslash PD)$; // *добавляем в PQ всех соседей T_i*

    пока $PQ \neq \emptyset$

      если $PQ_1 \notin PD$ **и** $type(T_i, E, N) =$ «ядровая точка» **то**

        $Y_{PQ_1} = C$; // *присвоим новую метку класса*

        $PQ = PQ \cup (neighboors(PQ_1, E) \backslash PD)$; // *добавляем соседей PQ_1*

      иначе если $PQ_1 \notin PD$ **и** $type(T_i, E, N) =$ «граничная точка» **то**

        $BD = BD \cup \{T_i\}$;

      $PQ = PQ \backslash \{PQ_1\}$;

    $C = C + 1$;

  иначе если $type(T_i, E, N) =$ «граничная точка» **то**

    $BD = BD \cup \{T_i\}$;

  иначе если $type(T_i, E, N) =$ «шум» **то**

    $Y_i = -1$; // *шум помечается отдельной меткой*

для $i = 1, \ldots, |BD|$

  $Y_{BD_i} = label(closest\_core(BD_i, D))$;

4 Вычислительные эксперименты

4.1 Исходные данные и условия эксперимента

4.2 Результаты эксперимента

4.3 Выводы

5 Заключение

В данной работе была рассмотрена задача восстановления регрессии. Предложен новый подход к решению, основанный на использовании модели логического корректора. Реализованы стохастическая модификация логического корректора алгоритм MONS и динамический плотностной алгоритм кластеризации DM-DBSCAN. Поставленная задача решена на основе ансамбля алгоритмов. Приведены результаты тестирования алгоритма на прикладных задачах.

Список литературы

[1] *E.V. Djukova, Yu.I. Zhuravlev, R.M. Sotnezov.* Construction of an Ensemble of Logical Correctors on the Basis of Elementary Classifiers // Pattern Recognition and Image Analysis, 2011, Vol. 21, No4, pp. 599—605.

[2] *Ю. И. Журавлёв* Об алгебраическом подходе к решению задач распознавания или классификации // Проблемы кибернетики, 1978, вып. 33, 5–68

[3] *Е. В. Дюкова, Ю. И. Журавлёв, К. В. Рудаков* Об алгебро-логическом синтезе корректных процедур распознавания на базе элементарных алгоритмов // Журнал вычислительной математики и математической физики, 1996, 36:8 215–223

[4] *R. M. Sotnezov.* Genetic Algorithms for Problems of Logical Data Analysis in Discrete Optimization and Image Recognition // Pattern Recognition and Image Analysis, 2009, Vol. 19, No. 3, 469–477

[5] *Любимцева М.М., Дюкова Е.В.* Логические корректоры в задачах распознавания // 2014

[6] *M.T.H. Elbatta, W.M. Ashour* A Dynamic Method for Discovering Density Varied Clusters // International Journal of Signal Processing, Image Processing and Pattern Recognition Vol.6, No. 1, 2013

[7] *Дюкова Е.В.* Об асимптотически оптимальном алгоритме построения тупиковых тестов // ДАН СССР. 1977. Т. 233. № 4. С. 527-530.