Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математических методов прогнозирования

Байтеков Никита Вячеславович

Регрессионные логические корректоры

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:

д.ф.-м.н., доцент

Е. В. Дюкова

Москва, 2018

Содержание

1 Введение

Основными задачами машинного обучения по прецедентам являются задача классификации и задача восстановления регрессии. Далее рассматривается общая постановка для обеих задач.

Исследуется множество объектов M, которые описываются набором целочисленных признаков $\{x_1, ..., x_n\}$. Каждому объекту из M соответствует некоторое значение целевой переменной («ответа») из числового множества Y, которое, вообще говоря, неизвестно. Дано обучающее множество объектов $T = \{S_1, ..., S_m\}$ из M, такое, что для каждого обучающего объекта (прецедента) известно значение «ответа». Требуется по предъявленному набору значений признаков, описывающему некоторый объект из M определить значение его значение целевой переменной. В случае, когда $Y = \{1, \ldots, l\}, l \geq 2$, решается задача классификации, а при $Y = \mathbb{R}$ — задача восстановления регрессии.

Задачи машинного обучения являются актуальными, поскольку они возникают в целом ряде прикладных областей, таких как биология, медицина, телекоммуникации, информационная безопасность, банковская деятельность и т.д.

В настоящей работе рассматривается задача восстановления регрессии, для решения которых существует множество различных методов. Широко известны такие алгоритмы, как случайный лес, метод ближайших соседей, байесовская ридж-регрессия или нейронные сети. Один из способов решения рассматриваемой задачи заключается в её сведении к задаче классификации с той целью, чтобы потом применить методы классификации.

Среди современных моделей классификации показывают хорошие результаты модели логических корректоров [1] [2] [3], в основе которого лежат идеи логического [4] [5] и алгебраического подходов [6]. Классический логический подход заключается в использовании корректных алгоритмов, базирующихся на построении корректных элементарных классификаторов. Элементарный корректор (далее просто эл.кл) — это элементарная конъюнкция, определённая на признаковых описаниях объектов. Для логического подхода являются сложными задачи с признаками высокой значности, поскольку в этом случае не удаётся найти достаточное количество информативных корректных эл.кл.

Алгебраический подход позволяет корректировать работу нескольких распознающих алгоритмов, каждый из которых безошибочно классифицирует лишь часть объектов из обучающей выборки. Цель коррекции заключается в компенсации ошибок одних алгоритмов правильными ответами других, так что результирующий алгоритм оказывается лучше каждого из базовых алгоритмов в отдельности. В логических корректорах в качестве базовых алгоритмов используются произвольные эл.кл., в качестве корректирующей функции берётся булева функция, в частности, монотонная. Основным понятием является понятие монотонного корректного набора эл.кл. (далее просто м.кор. набора) — это такой набор эл.кл. для класса K, для которого монотонная корректирующая функция принимает разные значения для любой пары объектов $(S', S''), S' \in K, S'' \notin K$. Поиск таких наборов является сложной в вычислительном плане задачей.

Одним из первых успешных логических корректоров является алгоритм MON [7], который использует наиболее информативные м.кор. наборы эл.кл. ранга 1. В последующих работах ограничение на ранг эл.кл. было снято. Были предожены различные способы снятия вычислительной сложности, например, с помощью применения генетических алгоритмов и стохастического поиска корректных наборов эл.кл. в [8].

Целью данной работы является адаптация стохастического логического корректора MONS к задаче восстановления регрессии.

В настоящей работе разработаны стохастические регрессионные логические корректоры MONS-Rg1, MONS-Rg2 и MONS-Rg3 на основе сведения задачи восстановления регрессии к задаче классификации. Для кластеризации данных MONS-Rg1 использует динамический DBSCAN, MONS-Rg2 использует кластеризацию с помощью парзеновского окна постоянной ширины, а MONS-Rg3 — кластеризацию с помощью парзеновского окна переменной ширины. Проведено тестирование моделей, полученные результаты сопоставлены с результатами работы других известных классических методов регрессии, таких, как случайный лес(random forest), градиентный бустинг на решающих деревьях, байесовская ридж-регрессия, метод опорных векторов и метод ближайших соседей.

2 Логические корректоры в задаче классификации

2.1 Алгоритм MON

Рассмотрим задачу классификации и введём основные определения. Пусть обучающее множество объектов T разбито на l непересекающихся множеств $\{K_1, ..., K_l\}$, называемых *классами*, и для каждого прецедента известно, к какому из классов он принадлежит. Обозначим множество прецедентов из T, принадлежащих некоторому классу $K, K \in \{K_1, ..., K_l\}$, как $T \cap K$, а $T \cap \overline{K}$ — как множество прецедентов из T, не принадлежащих классу K. Пусть $H = \{x_{j_1}, ..., x_{j_r}\}$ — набор различных признаков, $\sigma = \{\sigma_1, ..., \sigma_r\}$ — набор допустимых значений этих признаков, где σ_q - допустимое значение признака $x_{j_q}, q = 1, \ldots, r$.

Пара (H, σ) называется *элементарным классификатором(эл.кл.)* Число r называется *рангом* эл.кл.(H, σ). Обозначим через $x_j(S)$ значение признака x_j на объекте S и через $H(S)$ вектор значений признаков $(x_{j_1}(S), \ldots, x_{j_r}(S))$. Говорят, что эл.кл. (H, σ) *выделяет* объект S, если $H(S) = \sigma$. Эл.кл. (H, σ) считается *корректным для класса* K, если не существует двух выделяемых эл.кл. (H, σ), таких, что $S' \in K, S'' \notin K$.

Пусть $P_{(H,\sigma)}(S)$ — *отклик* эл.кл. (H, σ) на объекте S, которой полагается равным 1 в случае, если эл.кл выделяет объект, и равным 0 в противном случае. Пусть $L_T = (l_{ij})$ — булева матрица, элемент l_{ij} равен отклику j-ого эл.кл. на i-ом объекте выборки T. Матрица $L_T \in \mathbb{R}^{m \times N}$ называется матрицей сравнения для выборки T, где N — мощность множества всевозможных эл.кл.

Пусть имеется упорядоченный *набор эл.кл.* $U = ((H_1, \sigma_1), \ldots, (H_d, \sigma_d))$. Набор U ставит в соответствие объекту S из M бинарный вектор $U(S) = ([H_1(S) = \sigma_1], \ldots, [H_d(S) = \sigma_d])$, который называется *откликом* набора эл.кл. U на объекте S (здесь через $[p]$ обозначается предикат, принимающий 1, если выражение p истинно, и 0 – в противном случае) Набор эл.кл. U называется *монотонным корректным для класса* K(м.кор. набор для K), где $K \in \{K_1, ..., K_l\}$, если существует монотонная функция алгебры логики $F_{U,K}$: $F_{U,K}(U(S')) \neq F_{U,K}(U(S'')), \forall (S', S''), S' \in K, S'' \notin K$. В таком случае, функция $F_{U,K}$ называется *монотонной корректирующей* для U. М.кор. набор U для K является *неприводимым*, если любой набор эл.кл., являющий-

ся подмножеством U, не является м.кор. для K. Неприводимый набор для класса K является *минимальным*, если не существует м.кор. набора для K меньшей мощности.

Процесс работы логического 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 м.кор. 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:

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

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

2.2 Алгоритм MONS

Алгоритм MONS является является модификацией алгоритма MON, которая использует не всю матрицу сравнения, но случайным образом формирует небольшое подмножество эл.кл.,которое также является подматрицей сравнения. Анализ такого подмножества позволяет получать наборы эл.кл., являющиеся приближённым решением исходной задачи, а также оценивать возможную при этом ошибку. Так как объём выборки существенно меньше множества всех возможных эл.кл., то появляется возможность обрабатывать таблицы куда больших размеров.

Пусть T — обучающая выборка, θ_s, θ_d — параметры разнообразия и различимости для подмножества Θ, а ε — порог для критерия останова нашего алгоритма.

В начале работы алгоритма выборка T разбивается на две: усечённую обучающую выборку T' и валидационную выборку V. Благодаря выборке V можно в точности определить момент, когда алгоритм сошёлся.

Обучающий цикл состоит из двух циклов: внешнего из max_i итераций и внутреннего по всем классам $K \in \{K_1, \cdots, K_l\}$, для каждого из которых будет отдельно строится семейство м.кор. наборов. В начале итерации с номером i для класса

K случайно формируется подмножество эл.кл. Θ по следующему принципу: сначала из признакового описания каждого объекта, принадлежащего рассматриваемому классу K, генерируется θ_s эл.кл., после чего по получившемуся подмножеству Θ' каждый объект сверяется на различимость с каждым объектом не из класса K, и, в противном случае, в Θ' добавляется θ_d различающих эл.кл. Итоговое подмножество позиционируется как $\Theta = \{P_{(H_1,\sigma_1)}, \ldots, P_{(H_{N_K},\sigma_{N_K}),}\}$.

После этого составляется булева матрица L_K из всевозможных строк вида $B(S', S''), S' \in T \cap K$, а $S'' \in T \cap \overline{K}$, где бинарный вектор $B(S', S'') = (b_1, \ldots, b_{N_K})$ определяется как

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

. Заметим, что L_K является подматрицей сравнения для L_T, а каждый её столбец соответствует определённому эл.кл. из Θ, так что набор таких столбцов задаёт набор эл.кл. Очевидно, что любой набор эл.кл. U_K является монотонным 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 (известен также как задача дуализации монотонной КНФ),который относится к числу трудоёмких перечислительных задач. Для решения используются приближённые алгоритмы поиска, в нашем случае это генетический алгоритм. После работы алгоритма полученные м.кор. наборы добавляются в соответствующее семейство W_K м.кор. наборов для K.

После завершения внутреннего цикла происходит проверка критерия останова. Для этого вычисляется средний «отступ» предсказаний модели от истинных значений меток классов по следующей формуле:

$$M(S) := \frac{1}{i}\sum_{j=1}^{i} \Gamma_{y(S)}(W^j_{y(S)}, S) - \max_{K \in \{K_1,\ldots,K_l\} \setminus y(S)} \Gamma_K(W^j_K, S),$$

где $y(S)$ — истинная метка класса для объекта S, W^i_K — семейство м.кор. наборов для K на i-той итерации алгоритма, а $\Gamma_K(S)$ — ранее упомянутая оценка принадлежности

объекта S к классу K. Если $|M_i(S) - M_{i-1}(S)| < \varepsilon$ для любого объекта $S \in V$, то алгоритм завершается.

Ниже приведена схема алгоритма, описанная на псевдокоде:

Algorithm 1 Процедура обучения алгоритма MONS

Вход: T — обучающая выборка, max_i — максимальное число итераций, ε — порог, θ_s — разнообразие Θ, θ_d — различимость Θ;

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

$V := V(T)$; // *случайно выделить выборку $V \subset T$ в качестве валидационной*

$T' := T \backslash V$; // *усечь обучающую выборку*

для $i = 1, \ldots, max\_i$

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

    $\Theta := \Theta(\theta_s, \theta_d)$; // *случайно сформировать подмножество эл.кл. Θ*

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

    $W_K := W_K \cup W_K^i$; // *добавить 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*

    $M_i^{avg}(S) := \frac{1}{i} \sum_{j=1}^{i} M_j(S)$; // *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 поиска м.кор. наборов эл.кл

На каждом шаге генетического алгоритма обновляется множество приближённых решений задачи, называемое *популяцией* в ходе итераций, называемых *поколениями*. Элементы популяции 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 особи для скрещивания* осуществляется с помощью метода рулетки, когда вероятность p_i выбора i-ой особи зависит от значения функции приспособленности f_i этой особи:

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

На каждой итерации для скрещивания выбирается ровно одна пара родителей, так как в этом случае лучшие особи с высокой вероятностью 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. Затем восстанавливается допустимость решения, так как, вообще говоря, полученный набор столбцов не является неприводимым покрытием. На последнем этапе одна из наименее приспособленных особей замещается потомком.

В качестве критерия останова используется сходимость популяции с задержкой. Пусть $P_i = \{U_1, \cdots, U_{N_{P_i}}\}$ — популяция на i-ой итерации, а $f(P_i) = \frac{1}{N_{P_i}} \sum\limits_{j=1}^{N_{P_i}} f_j$ — её средняя приспособленность. Тогда генетический алгоритм завершается, если неравенство $|f(P_j) - f(P_{j-1})| < \varepsilon$ выполняется для $\forall j \in \{i - d, \cdots, i\}$, где i — некоторый существующий номер итерации, а параметры ε и d заранее заданы. Такой критерий останова позволяет усиливающемуся коэффициенту мутации дольше воздействовать на нашу популяцию, чтобы увеличить шанс выбраться из гипотетического локального максимума.

3 Логический корректор восстановления регрессии MONS-Rg

В данной работе на базе стохастического логического корректора MONS были разработаны алгоритмы *MONS-Rg1*,*MONS-Rg2* и *MONS-Rg3*, которые решают задачу восстановления регрессии. На первом этапе обучающие объекты разбиваются на несколько кластеров с помощью выбранного алгоритма кластеризации(для MONS-Rg1 используется динамический DBSCAN, для MONS-Rg2 используется кластеризация парзеновским окном постоянной ширины, а для MONS-Rg3 — переменной ширины. На втором этапе применяется алгоритм MONS для решения возникшей задачи классификации. На третьем этапе для распознаваемого объекта восстанавливается значение целевой переменной по полученной на втором этапе метке класса.

3.1 Алгоритм кластеризации 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 [9] — 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.

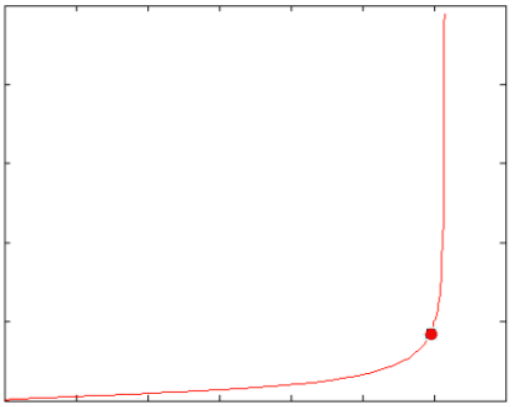

Рис. 1: Кривая k-расстояний для одного уровня плотности

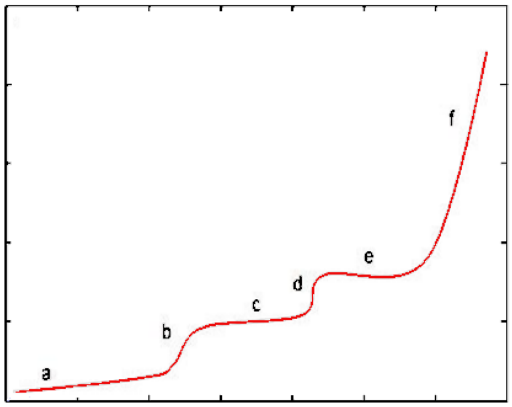

Рис. 2: Кривая k-расстояний для нескольких уровней плотности

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

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 $i = 1, \ldots, |BD|$

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

3.2 Алгоритмы кластеризации Parzen и vParzen

В данном методе для кластеризации будут использоваться только значения целевой переменной объектов обучающей выборки. Такой подход не использует предположение о данных, что близость признаковых описаний объектов соответствует близости их ответов. Для выделения кластеров в данных требуется оценить плотность вероятностного распределения значений целевой переменной нашей выборки. Тогда локальные максимумы плотности распределения будут соответствовать центральным элементам кластеров, а локальные минимумы – их границам.

Для того, чтобы оценить плотность распределения, воспользуемся непараметрическим методом парзеновского окна, который использует локальную оценку Парзена-Розенблатта. Пусть h — неотрицательная ширина парзеновского окна, m — количество объектов обучающей выборки T, S_i — вектор признакового описания i-го объекта, а $K(z)$ — функция, называемая *ядром*, которая обязана быть чётной и удовлетворять нормировке $\int K(z)dz = 1$. Тогда локальная оценка Парзена-Розенблатта имеет вид:

$$\hat{p}_h(S) = \frac{1}{mh}\sum_{i=1}^{m} K\left(\frac{\|S - S_i\|}{h}\right).$$

Функция $\hat{p}_h(S)$ обладает той же степенью гладкости, что и ядро, и, благодаря нормировке, может действительно интерпретироваться как некоторый аналог плотности. На практике используются несколько типов ядер, начиная от обычного прямоугольного и заканчивая нормальным, но в нашем случае используется *ядро Епанечникова*, так как оно специально приспособлено для оптимизации критерия MSE, с помощью которого в данной работе оценивается качество методов восстановления регрессии. Формула ядра Епанечникова:

$$K_E(z) = \frac{3}{4}\left(1 - z^2\right)\left[|z| \leqslant 1\right], z \in \mathbb{R}.$$

Обоснованием корректности оценки Парзена-Розенблатта служит следующая:

Теорема 1 (доказана в [11]). *Пусть выполнены следующие условия:*

1. *выборка T получена из распределения $p(S)$;*
2. *ядро $K(z)$ непрерывно и удовлетворяет условию $\int K^2(z)dz \leqslant \infty$;*

3. последовательность h_m такова, что $\lim\limits_{m\to\infty} h_m = 0$, а $\lim\limits_{m\to\infty} mh_m = \infty$.

Тогда $\hat{p}_{h_m}(S)$ сходится к $p(S)$ при $m \to \infty$ для почти всех $S \in M$, причём скорость сходимости имеет порядок $O\left(m^{-\frac{2}{5}}\right)$.

В случае, когда значения выборки расположены сильно неравномерно, то возникает так называемая *проблема локальных сгущений*. В таком случае фиксированная ширина окна приводит к чрезмерному сглаживанию плотности в одних областях пространства M и к недостаточному сглаживанию в других. Решением данной проблемы является метод *переменной ширины парзеновского окна*, который в точке признакового пространства S выбирает ширину окна равной расстоянию данной точки до ближайшего k-го соседа из обучающей выборки. Параметр k, как и параметр h, как правило, настраивается с помощью контроля по отдельным объектам, более известным как leave-one-out.

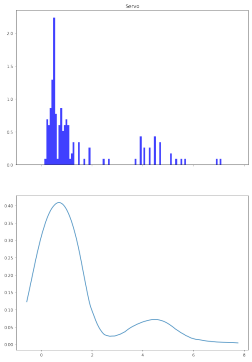

Рис. 3: Датасет Servo

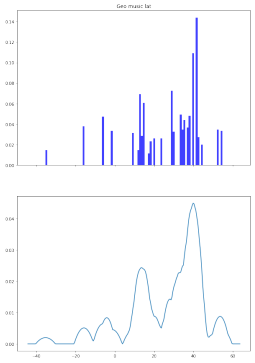

Рис. 4: Датасет Geomusic(lat)

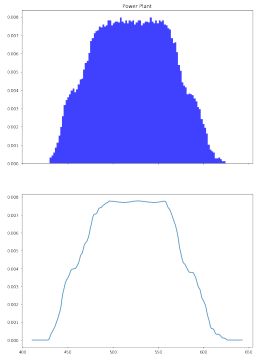

Рис. 5: Датасет Cycle Combine Power Plant

3.3 Восстановление целевой переменной в алгоритме MONS-Rg

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. В данной работе использовался метод взвешенных ближайших соседей, который выдаёт в качестве ответа сумму значений целевых переменных соседей, причём с весами, обратно пропорциональными их расстоянию до нашего исходного объекта. Выбор метода обоснован тем, что в одном классе обучающие объекты при схожих значениях целевой переменной могут быть далеко расположены друг от друга в признаковом пространстве.

4 Вычислительные эксперименты

Алгоритмы *MONS-Rg1, MONS-Rg2 и MONS-Rg3* сравнивались с 5 основными моделями, решающими задачу восстановления регрессии: байесовской ридж-регрессией(далее обозначаемой как BR), градиентным бустингом (GB), методом ближайших соседей(kNN), случайным лесом(RF) и адаптацией метода опорных векторов (SVR). Реализация данных моделей была взята из библиотеки машинного обучения scikit-learn. Для настройки моделей использовался перебор по сетке параметров. Для тестирования было выбрано 20 прикладных задач из ресурса UCI. Для оценки качества работы применялся скользящий контроль на 10 разбиениях по функционалу качества MSE по следующей формуле:

$$MSE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y_i})^2,$$

где y - вектор истинных значений ответов для объектов проверочной выборки, $\hat{y}$ - вектор предсказанных моделью ответов, n - размер выборки.

Ниже в таблице 1 приведены результаты работы перечисленных моделей на соответствующих задачах:

Данные	Размер ($m \times n$)	MRg(best)	kNN	GB	BR	SVR	RF
Servo	167×4	0.2045	0.7291	**0.1334**	1.3195	0.4447	0.1784
Computers	209×5	**2442.8**	3250.4	5833.7	4693.2	9752.4	2939.4
Yachts	308×6	1.587	63.273	**0.87048**	82.315	36.737	1.1024
Concrete	103×7	11.966	12.475	11.092	7.2968	**1.7544**	11.443
Fertility	100×10	**0.0983**	0.10395	0.15011	0.11011	0.10205	0.11346
Cancer	198×31	**0.0867**	0.0992	0.15806	0.1520	0.1036	0.1245
Autos	205×25	1.0705	1.2975	**0.28264**	0.6838	0.3493	0.2982
Forest Fire	517×12	**3872.6**	4072.3	4190.1	4053.3	4178.8	4653.2
Cancer (wdbc)	205×25	0.04670	0.0460	0.0481	0.0594	0.0357	**0.0338**
Cancer (wpbc)	198×33	0.1681	0.1735	0.1786	0.1522	**0.1511**	0.1617
Autos MPG	398×7	14.576	14.463	7.2976	11.499	**7.1182**	7.4283
Geomusic(lat)	1059×68	255.30	244.36	247.76	285.42	**244.32**	248.18
Geomusic(lon)	1059×68	1811.65	1803.28	1744.08	1943.47	**1678.69**	1717.53
Geomusic chromo (lat)	1059×116	256.34	255.04	244.43	278.64	**240.68**	244.80
Geomusic chromo (lon)	1059×116	1695.84	1698.92	**1586.42**	1889.16	1611.59	1613.75
WineQ (red)	1599×11	0.4350	0.4066	**0.3271**	0.4255	0.3906	0.3315
WineQ (white)	4898×11	0.4905	0.4299	**0.3438**	0.5680	0.4767	0.3558
Cycle Plant	9568×4	8.1377	**2.539**	26.970	89.600	82.596	28.554

Таблица 1: Качество работы алгоритмов (MSE)

Из таблицы видно, что модели *MONS-Rg* превосходят другие алгоритмы восстановления регрессии на четырех задачах из списка и показывают схожее качество на остальных. Исследование этих задач показало, что в данных выделяются «подзадачи», для решения которых применяется уже другой алгоритм восстановления регрессии (таким образом восстанавливается значение целевой переменной объекта).

Таким образом повышается общая устойчивость алгоритма к выбросам и неоднородному распределению значений «ответов» в представленных данных. Из минусов можно отметить большее время работы по сравнению с остальными алгоритмами, так как модель является более сложной и таблицы видно, что модель способна соперничать с ведущими моделями, а порой и превосходить их на отдельных задачах.

5 Заключение

В данной работе рассматривалась задача восстановления регрессии, которая была решена с помощью сведения к задаче классификации. Разработана модель стохастического регрессионного логического корректора. В процессе работы реализованы и протестированы такие алгоритмы, как стохастическая модификация логического корректора MONS, динамический плотностной алгоритм кластеризации DM-DBSCAN и два варианта алгоритма кластеризации на основе парзеновского окна. Проведено исследование эффективности модели при решении прикладных задач в зависимости от выбранного алгоритма кластеризации. Эксперименты показывают стабильное качество работы алгоритмов MONS-Rg, сравнимое с работой классических алгоритмов, решающих задачу восстановления регрессии. Найден ряд задач, на которых построенная модель доминирует над остальными алгоритмами.

Список литературы

[1] *Дюкова Е.В., Журавлёв Ю.И., Прокофьев П.А.* Логические корректоры в задаче классификации по прецедентам // Ж. вычисл. матем. и матем. физики, 2017, 57:11 С.141-162

[2] *Djukova E.V., Zhuravlev Yu.I., Sotnezov R.M.* 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.

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

[4] *Баскакова Л.В., Журавлёв Ю.И.* Модель распознающих алгоритмов с представительными наборами и системами опорных множеств // Ж.вычисл.матем.и.матем.физ. 1981. Т.21 №5. С. 1264–1275

[5] *Дмитриев Е.А., Журавлёв Ю.И., Кренделев Ф.П.* О математических принципах классификации предметов или явлений // Дискретный анализ. Новосибирск: ИМ СО АН СССР, 1966. Вып. ?. С. 3–17

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

[7] *Sotnezov R.M.* 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

[8] *Любимцева М.М.* Логические корректоры в задачах распознавания // Тезисы лучших дипломных работ факультета ВМК МГУ, 2014. С.47-49.

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

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

[11] *Parzen E.* On the estimation of a probability density function and mode // Annals of Mathematical Statistics. — 1962. — Vol. 33. — Pp. 1065–1076.