УДК 681.518

Л.А. Демидова, Ю.С. Соколова

АСПЕКТЫ ПРИМЕНЕНИЯ АЛГОРИТМА РОЯ ЧАСТИЦ В ЗАДАЧЕ РАЗРАБОТКИ SVM-КЛАССИФИКАТОРА

Рассматриваются аспекты применения алгоритма роя частиц в задаче разработки SVM-классификатора с целью выбора типа функции ядра, значений параметров функции ядра и значения параметра регуляризации, обеспечивающих высокое качество классификации данных. Приведены результаты экспериментальных исследований, подтверждающие целесообразность использования алгоритма роя частиц в задаче разработки SVM-классификатора.

***Ключевые слова**: SVM-алгоритм, классификация, оптимизация, параметры функции ядра, параметр регуляризации, алгоритм роя частиц, кросс-проверка.*

Введение. В настоящее время для решения широкого спектра классификационных задач в различных прикладных областях успешно применяется SVM-алгоритм (Support Vector Machines, SVM), предложенный В.Н. Вапником [1]. Данный алгоритм осуществляет обучение по прецедентам («обучение с учителем») и входит в группу граничных алгоритмов и методов классификации.

SVM-алгоритм реализует построение бинарного SVM-классификатора, осуществляя перевод векторов характеристик классифицируемых объектов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве [1 – 5]. При этом по обеим сторонам разделяющей гиперплоскости строятся две параллельных гиперплоскости, задающие границы классов и находящиеся на максимально возможном расстоянии друг от друга. Предполагается, что чем больше расстояние между этими параллельными гиперплоскостями, тем меньше средняя ошибка SVM-классификатора. Векторы характеристик классифицируемых объектов, ближайшие к параллельным гиперплоскостям, называются опорными векторами. Пример построения разделяющей гиперплоскости в пространстве D-2 приведен на рисунке 1.

SVM-алгоритм предполагает выполнение обучения, тестирования и классификации.

Использование специальных типов функций, называемых ядрами, в процессе обучения SVM-классификатора позволяет в ряде случаев осуществить перевод векторов характеристик классифицируемых объектов в пространство более высокой размерности и реализовать уже в нём линейное разделение классов [1-5].

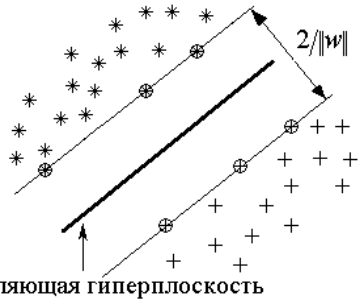

Рисунок 1 – Бинарное разделение классов

Существенный интерес при разработке SVM-классификатора представляет решение проблемы, связанной с выбором типа функции ядра, значений параметров функции ядра и значения параметра регуляризации SVM-алгоритма, обеспечивающих высокое качество классификации данных.

В простейшем случае решение данной проблемы осуществляется посредством перебора возможных типов функции ядра, значений параметров функции ядра и значения параметра регуляризации, что требует значительных вычислительных затрат. При этом для оценки качества классификации могут быть использованы показатели точности и полноты классификации и др. [6].

Используемые для оценки качества классификации показатели зачастую являются сложными, многоэкстремальными функциями, которые, к тому же, могут в явном виде не зависеть от оптимизируемых параметров, в связи с чем к ним не применимы классические методы оптимизации

(например, градиентные методы). В этом случае целесообразно применение поисковых алгоритмов стохастической оптимизации, таких как генетический алгоритм (ГА) [7, 8], иммунный алгоритм клональной селекции [9], алгоритм роя пчел [10], алгоритм колонии муравьев [10], алгоритм роя частиц (Particle Swarm Optimization, PSO-алгоритм) [10 – 12], в которых подбор оптимального решения производится сразу по всему пространству поиска. Эти алгоритмы основаны на моделировании законов функционирования и социального поведения живых организмов и реализуют: случайную генерацию начальной популяции решений; вычисление значения некоторой целевой функции (в качестве которой может использоваться, например, показатель качества классификации) для каждого решения, определяющего его близость к оптимуму; репродукцию популяции решений на основе значений целевой функции. При этом все вычисления завершаются при выполнении некоторых условий останова поискового алгоритма.

Долгое время при решении трудоемких задач оптимизации наиболее востребованными являлись генетические алгоритмы [7, 8]. Однако в 1995 году Джеймсом Кеннеди и Расселом Эберхартом был предложен алгоритм роя частиц (PSO-алгоритм), предназначенный для оптимизации непрерывных нелинейных функций, который по эффективности может конкурировать со многими алгоритмами глобальной оптимизации. При этом простота реализации PSO-алгоритма определяет его низкую алгоритмическую сложность.

PSO-алгоритм имеет много общего с ГА. Оба алгоритма, являясь алгоритмами случайно-направленного поиска, начинают работать со случайно сгенерированной популяцией решений и выполняют расчет значений своих целевых функций, выполняя в процессе эволюции поиск лучшего решения. Однако в PSO-алгоритме нет генетических операторов, подобных кроссинговеру и мутации в ГА. В PSO-алгоритме потенциальные решения-частицы имеют память, взаимодействуют и изменяют свои скорости. Механизмы передачи информации в ГА и PSO-алгоритме совершенно различны. В ГА хромосомы обмениваются информацией (генами) друг с другом, поэтому вся популяция движется как единая группа в область оптимума. В PSO-алгоритме только информация о глобально лучшей позиции среди всех частиц передается другим частицам, поэтому в большинстве случаев все частицы стремятся к лучшему решению быстрее, чем в ГА. Также преимущество PSO-алгоритма перед ГА заключается в том, что для реализации PSO-алгоритма необходимо уметь определять только значение целевой функции, и он имеет меньше параметров управления, чем ГА. Таким образом, можно сделать вывод о целесообразности выполнения анализа перспективности применения PSO-алгоритма в задаче разработки SVM-классификатора.

Цель работы – анализ аспектов применения алгоритма роя частиц при решении задачи разработки SVM-классификатора.

Теоретическая часть. Пусть имеется учебный набор вида: $\{(z_1,y_1),...,(z_s,y_s)\}$, в котором каждому объекту z_i поставлено в соответствие число y_i, принимающее значение 1 или −1, в зависимости от того, какому классу принадлежит объект z_i. При этом предполагается, что каждому объекту z_i поставлен в соответствие q-мерный вектор числовых значений характеристик $z_i=(z_i^1,z_i^2,\ldots,z_i^q)$ (обычно нормализованный значениями из отрезка [0, 1]), где z_i^j – числовое значение j-й характеристики для i-го объекта ($i=\overline{1,s}$, $j=\overline{1,q}$) [5]. Данный учебный набор может быть использован для разработки бинарного SVM-классификатора с целью его дальнейшего применения для классификации новых объектов.

При разработке SVM-классификатора необходимо реализовать многократное обучение и тестирование на разных случайным образом сформированных (на основе учебного набора) обучающем и тестовом наборах с последующим определением лучшего SVM-классификатора в смысле обеспечения максимально возможного качества классификации, для оценки которого обычно используют точность и полноту классификации [6].

SVM-классификатор может быть применен для классификации новых объектов, если качество обучения и тестирования является приемлемым [1].

Задача разработки бинарного SVM-классифи-катора с учетом теоремы Куна-Таккера эквивалентна двойственной задаче поиска седловой точки функции Лагранжа, которая сводится к задаче квадратичного программирования, содержащей только двойственные переменные [1 – 3]:

$$\begin{cases} -L(\lambda)=-\sum\limits_{i=1}^{S}\lambda_i+\frac{1}{2}\cdot\sum\limits_{j=1}^{S}\sum\limits_{i=1}^{S}\lambda_i\cdot\lambda_j\cdot y_i\cdot y_j\cdot k(z_i,z_j)\to\min\limits_{\lambda}, \\ \sum\limits_{i=1}^{S}\lambda_i\cdot y_i=0, \\ 0\le\lambda_i\le C,\ i=\overline{1,S}, \end{cases} \quad (1)$$

где λ_i – двойственная переменная; z_i – объект из обучающего набора; y_i – число (–1 или 1), характеризующее классовую принадлежность объекта z_i из обучающего набора; $k(z_i, z_j)$ – функция ядра; C – параметр регуляризации ($C > 0$); S – количество объектов в обучающем наборе; $i = \overline{1,S}$.

В случае линейной разделимости классов в исходном q-мерном пространстве характеристик в ходе обучения SVM-классификатора реализуется линейная классификация объектов. При этом в (1) в качестве ядра $k(z_i, z_j)$ используется скалярное произведение $< z_i, z_j >$.

В случае линейной неразделимости классов в исходном q-мерном пространстве характеристик и необходимости повышения размерности пространства реализуется нелинейная классификация объектов. При этом в (1) в качестве ядра $k(z_i, z_j)$, позволяющего разделить объекты разных классов, обычно используется одна из функций [3]:

- полиномиальная:
$k(z_i, z_j) = (< z_i, z_j > +1)^d$;
- радиальная базисная:
$k(z_i, z_j) = exp(- < z_i - z_j, z_i - z_j > /(2 \cdot \sigma^2))$;
- сигмоидная:
$k(z_i, z_j) = th(k_1 + k_2 \cdot < z_i, z_j >)$,

где d [$d \in N$ (по умолчанию $d = 3$)], σ [$\sigma > 0$ (по умолчанию $\sigma^2 = 1$)], k_1 [$k_1 < 0$ (по умолчанию $k_1 = -1$)] и k_2 [$k_2 > 0$ (по умолчанию $k_2 = 1$)] – некоторые параметры; th – гиперболический тангенс.

В результате обучения SVM-классификатора определяются опорные векторы, являющиеся векторами характеристик тех объектов z_i из обучающей выборки, для которых значения соответствующих им двойственных переменных λ_i отличны от нуля ($\lambda_i \neq 0$) [3].

Именно опорные векторы находятся ближе всего к гиперплоскости, разделяющей классы, и несут всю информацию о разделении классов.

Если задача квадратичного программирования (1) решена, то классификация произвольного объекта z может быть выполнена с использованием правила вида:

$$\alpha(z) = sign\ (\sum_{i=1}^{S} \lambda_i \cdot y_i \cdot k(z_i, z)\ -\ b), \qquad (2)$$

где $b = < w, z_i > -\ y_i$; $w = \sum_{i=1}^{S} \lambda_i \cdot y_i \cdot z_i$.

При этом суммирование в правиле (2) выполняется только по опорным векторам.

Основная проблема, возникающая при обучении SVM-классификатора, связана с отсутствием рекомендаций по выбору таких значений параметра регуляризации C, функции, описывающей ядро $k(z_i, z_j)$, а также значений параметров самой функции ядра, при которых будет обеспечена высокая точность классификации объектов. Данная проблема может быть решена с применением тех или иных оптимизационных алгоритмов, в частности с использованием PSO-алгоритма, хорошо зарекомендовавшего себя при решении широкого спектра задач оптимизации.

В простейшем случае предлагается сначала выбрать несколько типов функции ядра $k(z_i, z_j)$, реализовать поочередно PSO-алгоритм для каждого такого типа функции ядра $k(z_i, z_j)$ с целью поиска оптимальной комбинации значений параметра регуляризации C и параметров функции ядра, обеспечивающей максимально возможную точность классификации, а затем определить в качестве искомых тот тип функции ядра $k^*(z_i, z_j)$ и те значения коэффициента регуляризации C^* и параметров функции ядра $k^*(z_i, z_j)$, при которых точность классификации максимальна.

В настоящее время известны различные модификации PSO-алгоритма, в основе которых лежит его классическая версия [12]. Пусть n – количество параметров, подлежащих оптимизации, а $x^1, x^2, ..., x^n$ – параметры оптимизации. При этом в аналитической записи целевой функции $f(x) = f(x^1, x^2, ..., x^n)$ алгоритма оптимизации (оптимум, например, минимум которой необходимо найти) эти параметры могут как присутствовать в явном виде, так и отсутствовать.

При реализации классического PSO-алгоритма n-мерное пространство поиска населяется роем из m агентов-частиц (элементарных решений). Положение (позиция) i-й частицы задается вектором $x_i = (x_i^1, x_i^2, \ldots, x_i^n)$, который определяет некоторый набор значений параметров оптимизации. В процессе инициализации роя частицы случайным образом располагаются по всей области поиска. При этом каждая i-я частица ($i = \overline{1,m}$) имеет свой собственный вектор скорости $v_i \in R^n$, которая в каждый конкретный момент времени, соответствующий некоторой итерации PSO-алгоритма, определенным образом влияет на значения координат позиции i-й частицы ($i = \overline{1,m}$). Координаты позиции i-й частицы

($i=\overline{1,m}$) в n-мерном пространстве поиска однозначно определяют значение целевой функции $f(x_i)=f(x_i^1,x_i^2,\ldots,x_i^n)$, которое является некоторым решением задачи оптимизации [10 – 12].

Для каждой позиции n-мерного пространстве поиска, в которой побывала i-я частица ($i=\overline{1,m}$), выполняется вычисление значения целевой функции $f(x_i)$. При этом каждая i-я частица запоминает, какое лучшее значение целевой функции она лично нашла, а также координаты позиции в n-мерном пространстве, соответствующие этому значению целевой функции. Кроме того, каждая i-я частица ($i=\overline{1,m}$) «знает», где расположена позиция, являющаяся лучшей (с точки зрения достижения оптимума целевой функции) среди всех позиций, которые «разведали» частицы (благодаря этому имитируется мгновенный обмен информацией между всеми частицами). На каждой итерации частицы корректируют свою скорость, чтобы, с одной стороны, быть поближе к лучшей позиции, которую частица нашла сама, и, с другой стороны, приблизиться к позиции, которая в данный момент является глобально лучшей. Через некоторое количество итераций частицы должны собраться вблизи наиболее хорошей позиции (глобально лучшей по результатам всех итераций). Однако возможно, что часть частиц останется где-то в относительно неплохом локальном оптимуме.

Сходимость PSO-алгоритма зависит от того, каким образом выполняется коррекция вектора скорости частиц. Известны различные подходы к выполнению коррекции вектора скорости v_i для i-й частицы ($i=\overline{1,m}$) [12].

В классической версии PSO-алгоритма коррекция каждой j-й координаты вектора скорости ($j=\overline{1,n}$) i-й частицы ($i=\overline{1,m}$) производится в соответствии с формулой [12]:

$$v_i^j=v_i^j+\varphi_p\cdot r_p\cdot(p_i^j-x_i^j)+\varphi_g\cdot r_g\cdot(g^j-x_i^j), \quad (3)$$

где v_i^j – j-я координата вектора скорости i-й частицы; x_i^j – j-я координата вектора x_i, задающего позицию i-й частицы; p_i^j – j-я координата вектора лучшей позиции, найденного i-й частицей за все время ее существования; g^j – j-я координата глобально лучшей позиции всего роя частиц, в которой целевая функция имеет оптимальное значение; r_p и r_g – случайные числа в интервале (0, 1), которые вносят элемент случайности в процесс поиска; φ_p и φ_g – личный и глобальный коэффициенты ускорения частиц, являющиеся константами и определяющие поведение и эффективность PSO-алгоритма в целом.

С помощью коэффициентов ускорения φ_p и φ_g в (3) масштабируются случайные числа r_p и r_g. При этом глобальный коэффициент ускорения φ_g управляет воздействием глобальной лучшей позиции на скорости всех частиц, а личный коэффициент ускорения φ_p – воздействием личной лучшей позиции на скорость некоторой частицы.

Довольно часто при выполнении коррекции вектора скорости v_i для i-й частицы ($i=\overline{1,m}$) используется модификация формулы (3):

$$v_i^j=\omega\cdot v_i^j+\varphi_p\cdot r_p\cdot(p_i^j-x_i^j)+\varphi_g\cdot r_g\cdot(g^j-x_i^j), \quad (4)$$

в которой перед j-й координатой v_i^j вектора скорости ($j=\overline{1,n}$) i-й частицы добавлен множитель – весовой коэффициент ω (коэффициент инерции), благодаря чему скорость изменяется более плавно.

Весовой коэффициент ω отвечает за баланс между размерами поискового пространства и вниманием к найденным субоптимальным решениям. В случае, когда $\omega>1$, скорости частиц увеличиваются, они разлетаются в стороны и исследуют пространство более тщательно. В противном случае скорости частиц со временем уменьшаются и скорость сходимости зависит от выбора значений коэффициентов ускорения частиц φ_p и φ_g. Таким образом, большие значения коэффициента ω способствуют исследованию пространства поиска, а малые – локализации решения.

В одной из самых распространенных модификаций PSO-алгоритма – канонической – предлагается выполнять нормировку коэффициентов ускорения φ_p и φ_g, чтобы сходимость алгоритма не так сильно зависела от выбора их значений [12]. При этом коррекция каждой j-й координаты вектора скорости ($j=\overline{1,n}$) i-й частицы ($i=\overline{1,m}$) производится в соответствии с формулой:

$$v_i^j=\chi\cdot[v_i^j+\varphi_p\cdot r_p\cdot(p_i^j-x_i^j)+\varphi_g\cdot r_g\cdot(g^j-x_i^j)], \quad (5)$$

где χ – коэффициент сжатия;

$\chi=2\cdot k/\left|2-\varphi-\sqrt{\varphi^2-4\cdot\varphi}\right|$; $\varphi=\varphi_p+\varphi_g$ ($\varphi>4$);

k – некоторый масштабирующий коэффициент, принимающий значения из интервала (0, 1).

При использовании формулы (5) для коррекции вектора скорости гарантируется сходимость PSO-алгоритма и нет необходимости в явном контролировании скоростей частиц [12].

Пусть коррекция вектора скорости i-й частицы ($i=\overline{1,m}$) выполнена в соответствии с одной из формул (3) – (5). Тогда коррекция j-й

координаты позиции i-й частицы ($i=\overline{1,m}$) выполняется в соответствии с формулой:

$$x_i^j = x_i^j + v_i^j. \qquad (6)$$

Далее для каждой i-й частицы ($i=\overline{1,m}$) рассчитывается новое значение целевой функции $f(x_i)$ и выполняется проверка: не стала ли новая позиция с вектором координат x_i лучшей среди всех позиций, в которых i-я частица ранее побывала. Если новая позиция i-й частицы признается лучшей на текущий момент времени, то информация о ней сохраняется в векторе p_i ($i=\overline{1,m}$) с «запоминанием» значения целевой функции $f(x_i)$ в этой позиции.

Затем среди всех новых позиций частиц роя осуществляется проверка на наличие глобально лучшей позиции. Если некоторая новая позиция признается глобально лучшей на текущий момент времени, то информация о ней сохраняется в векторе g с «запоминанием» значения целевой функции в этой позиции.

На рисунке 2 представлена схема канонического PSO-алгоритма. При этом $f: R^n \to R$ – целевая функция от n переменных, которую необходимо минимизировать.

При этом частицам роя могут быть сопоставлены векторы, описывающие их позиции в пространстве поиска и закодированные параметром регуляризации C и параметрами функции ядра:

- (C,d) – при использовании полиномиальной функции ядра;
- (C,σ) – при использовании радиальной базисной функции ядра;

(C,k_1,k_2) – при использовании сигмоидной функции ядра.

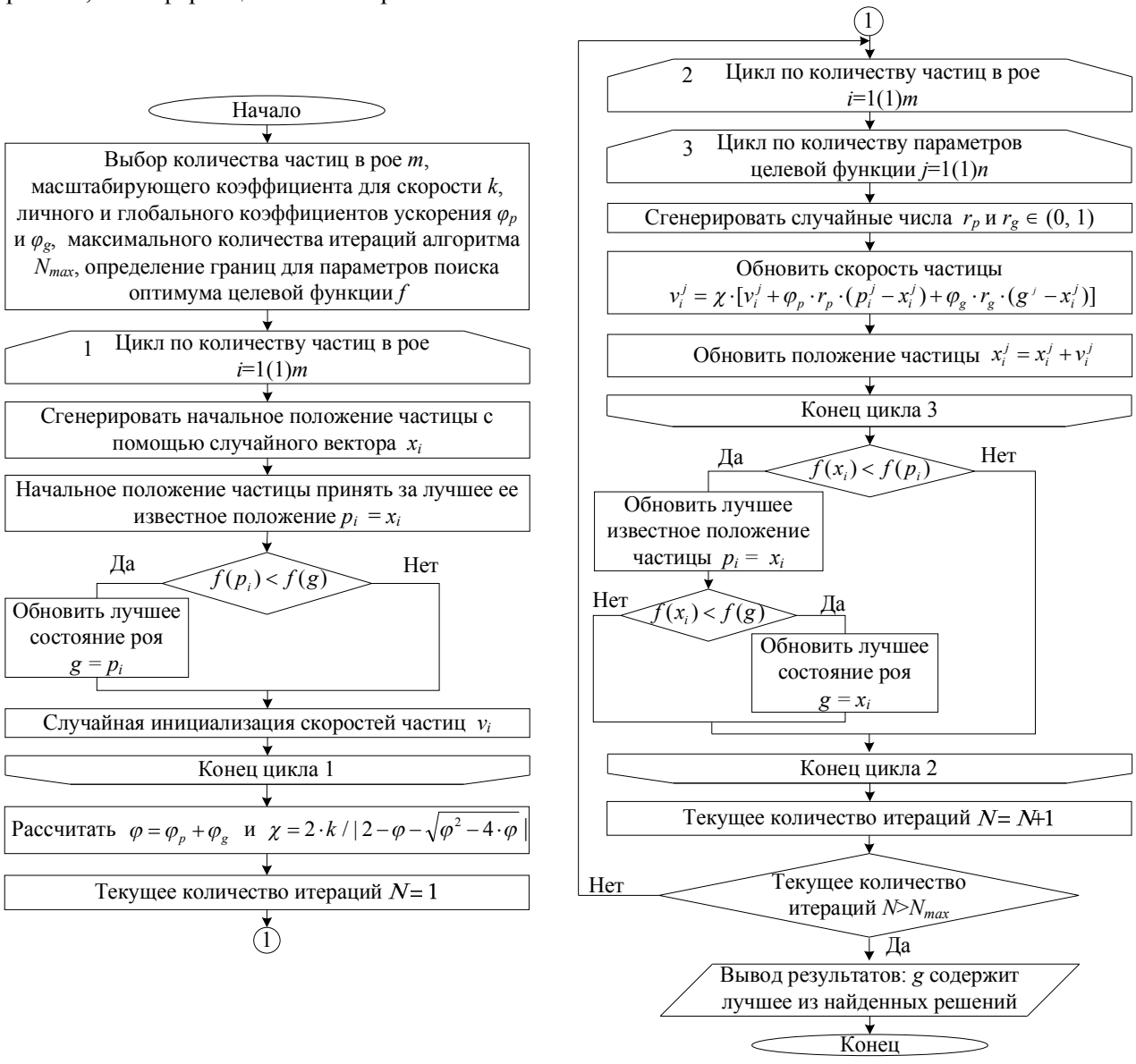

Рисунок 2 – Схема канонического PSO-алгоритма

Традиционный подход к применению PSO-алгоритма при разработке SVM-классификатора заключается в последовательном многократном применении PSO-алгоритма при фиксированном типе функции ядра с целью выбора оптимальных значений параметров функции ядра и значения параметра регуляризации C и последующем выборе лучшего типа функции ядра и соответствующих ему значений параметров функции ядра и значения параметра регуляризации.

Наряду с традиционным подходом к применению PSO-алгоритма при разработке SVM-классификатора предлагается применять новый подход, реализующий одновременный поиск лучшего типа функции ядра, значений параметров функции ядра и значения параметра регуляризации C. При этом инициализация роя частиц должна выполняться таким образом, чтобы каждому используемому в процессе поиска типу функции ядра соответствовало одинаковое количество частиц. Каждой частице роя соответствует вектор, описывающий ее позицию в пространстве поиска:

$$(C, T, x^1, x^2),$$

где C – параметр регуляризации; T – номер типа функции ядра (например, 1, 2, 3 – для полиномиальной, радиальной базисной и сигмоидной функций соответственно); x^1, x^2 – параметры функции ядра [при этом параметр x^1 полагаетсяравным параметрам функций ядра d, σ и k_1 (в зависимости от того, какому типу функции ядра соответствует частица роя); параметр x^2 полагается равным параметру функций ядра k_2, если частица роя соответствует сигмоидному типу функции ядра, в противном случае значение этого параметра считается равным нулю.

Сформированный таким образом рой частиц развивается так, что на каждой итерации PSO-алгоритма все частицы роя взаимодействуют по координате, отвечающей за тип функции ядра. При этом возможно «перерождение» частиц, если оказывается, что преобладающим становится некоторый тип функции ядра (в процессе движения роя к глобально лучшему решению, определяемому значением целевой функции).

В случае «перерождения» некоторой частицы происходит изменение ее типа функции ядра и соответствующих этому типу ядра значений параметров (с учетом диапазонов изменения их значений). Частицы, которые не подверглись «перерождению», осуществляют движение в своем собственном пространстве поиска (той или иной размерности).

Использование такого подхода к применению PSO-алгоритма в задаче разработки SVM-классификатора позволяет снизить временные затраты на построение искомого SVM-классификатора.

Оценка качества SVM-классификатора может быть выполнена с применением различных показателей качества классификации [6]. Это могут быть показатель, учитывающий данные кросс-проверки, показатели точности и полноты классификации, показатель, основанный на анализе ROC-кривой, и т.п.

Суть метода кросс-проверки (cross-validation), используемого для расчета показателя качества классификации, заключается в следующем [6]. Набор объектов Z, используемых для построения классификатора, разбивается на t непересекающихся поднаборов Z^l $(l = \overline{1,t})$: $Z = \bigcup_{l=1}^{t} Z^l$, $Z^l \cap Z^r = 0$ $(l \neq r)$, каждый из которых поочередно используется для тестирования SVM-классификатора, а остальные $t-1$ для обучения. В результате процедура обучения SVM-классификатора повторяется t раз и каждый из поднаборов Z^l $(l = \overline{1,t})$ используется для тестирования (рисунок 3). В качестве искомой точности SVM-классификатора полагается средняя точность по всем t циклам обучения и тестирования.

Базовыми характеристиками качества классификации являются уровни ошибок первого и второго рода. В контексте задачи классификации объектов ошибка первого рода («ложный пропуск», false negative) возникает, когда искомая принадлежность объекта к классу ошибочно не обнаруживается. Ошибка второго рода («ложное обнаружение», false positive) возникает, когда при отсутствии искомой принадлежности объекта к классу ошибочно принимается решение о её наличии. Виды ошибок, которые могут возникнуть при выполнении бинарной классификации объектов, приведены в таблице.

Пусть количество объектов равно, N_0 из них N_p – количество «положительных» (с меткой +1) объектов, а N_n – количество «отрицательных» (с меткой – 1) объектов, т.е. $N_0 = N_p + N_n$. Пусть количество «ложных пропусков» равно FN, а количество «ложных обнаружений» – FP. Тогда количество верных пропусков определяется как $TN = N_n - FP$, а количество верных обнаружений как $TP = N_p - FN$.

В этом случае можно рассчитать нормированные уровни ошибок первого и второго рода (nFN и nFP соответственно), а также долю верно распознаваемых пропусков nTN и обнаружений nTP [6]:

$$nFN = \frac{FN}{N_p} \cdot 100\%; \; nFP = \frac{FP}{N_n} \cdot 100\%;$$

$$nTN = \frac{TN}{N_n} \cdot 100\%;\ nTP = \frac{TP}{N_p} \cdot 100\%.$$

Эти показатели наглядно отражают качество классификации, поскольку не зависят от количества классифицируемых объектов.

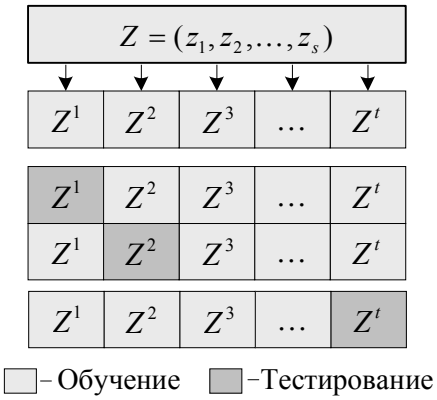

Рисунок 3 – Иллюстрация перекрестной проверки

Виды ошибок

Предсказанный класс	Результат классификации	Реальный класс	
		+1 (класс 1)	**–1 (класс 2)**
	+1	true positive *TP* Нет ошибки	false positive *FP* *Ошибка II рода*
	-1	false negative *FN* *Ошибка I рода*	true negative *TN* Нет ошибки

Показатель полноты (recall) классификации оценивает долю найденных классификатором объектов из класса среди всех объектов этого класса:

$$Recall = \frac{TP}{TP + FN}. \quad (7)$$

Показатель точности (precision) оценивает долю найденных объектов из класса среди всех объектов, которые отнесены в этот класс классификатором (то есть сколько объектов, отнесенных классификатором к классу, на самом деле ему принадлежит):

$$Precision = \frac{TP}{TP + FP}. \quad (8)$$

Также качество классификатора может быть оценено с помощью анализа ROC-кривой (Receiver Operation Characteristic), которая отображает зависимость доли верно классифицированных объектов (с меткой +1) в общем количестве объектов (с меткой +1) nTP от доли неверно классифицированных объектов (с меткой –1) в общем количестве объектов (с меткой –1) nFP.

В дальнейшем предлагается считать качество SVM-классификатора высоким, если количество ошибок на обучающем и тестовом наборах минимально, причем количество ошибок SVM-классификатора на объектах тестового набора не сильно отличается от средней ошибки на обучающем наборе (во избежание переобучения SVM-классификатора).

Экспериментальные исследования. Целесообразность использования PSO-алгоритма при разработке SVM-классификатора была подтверждена на тестовых и реальных данных.

Модельные данные для проведения экспериментальных исследований были взяты из репозитория задач машинного обучения UCI Machine Learning Repository (University of California, School of Information and Computer Sciences, Irvine, California – Калифорнийский университет, школа информационных и компьютерных наук, Ирвин, Калифорния). В частности, была использована выборка данных хирургического отделения университета штата Висконсин о диагностике рака молочной железы, содержащая информацию о 569 пациентах.

При этом у 212 пациентов (класс 1) была диагностирована злокачественная опухоль, а у 357 пациентов (класс 2) диагноз не был подтвержден (ftp://ftp.cs.wisc.edu/math-prog/cpo-dataset/machine learn/cancer/WDBC/). Информация о каждом пациенте содержала 30 атрибутов, описывающих характеристики клеточных ядер на основе оцифрованных изображений.

При построении SVM-классификатора был использован PSO-алгоритм с целью выбора оптимальных типа функции ядра $k(z_i, z_j)$ и значений его параметров, а также оптимального значения параметра регуляризации C. При этом в поиск были включены ядра с полиномиальной, радиальной базисной и сигмоидной функциями, для которых были установлены следующие диапазоны изменения значений параметров: $3 \le d \le 10$, $d \in N$ (для полиномиальной функции); $0{,}1 \le \sigma \le 5$ (для радиальной базисной функции); $-8 \le k_0 \le -0{,}1$ и $0{,}1 \le k_1 \le 5$ (для сигмоидной функции). Для параметра регуляризации C диапазон изменения был определен как $0{,}1 \le C \le 10$. Кроме того, были заданы следующие параметры PSO-алгоритма: количество частиц m в рое, равное 600 (по 200 на каждый тип функции ядра); количество итераций $N_{max} = 20$; коэффициенты личного и глобального ускорения, равные соответственно $\varphi_p = 2$ и $\varphi_g = 5$; масштабирующий коэффициент $k = 0{,}3$.

В ходе реализации PSO-алгоритма происходило не только изменение координат частиц, отвечающих за параметры функции ядра $k(z_i, z_j)$ и параметр регуляризации C, но изменялся и тип функции ядра.

На рисунках 4 – 6 показаны примеры распо-

ложения роя частиц в пространстве поиска D-2 в момент инициализации, а также на 10-й и 20-й итерациях соответственно для частного случая реализации PSO-алгоритма при использовании только радиальной базисной функции ядра в собственном пространстве поиска. При этом сами частицы помечены маркерами-звездочками, а лучшая позиция в пространстве поиска (лучший набор значений параметров C и σ) – белым маркером квадратной формы. Как видно из рисунков, в ходе реализации PSO-алгоритма частицы роя движутся к некоторой лучшей (оптимальной) для текущей итерации позиции, демонстрируя коллективный поиск лучшей позиции в ореоле обитания. При этом корректируются скорость и направление движения каждой частицы.

На рисунке 7 показано положение роя частиц в пространстве поиска D-3 на 20-й итерации для частного случая реализации PSO-алгоритма при использовании только сигмоидной функции ядра в собственном пространстве поиска.

При реализации PSO-алгоритма было получено, что в качестве оптимальной функции ядра следует выбрать радиальную базисную функцию с параметром $\sigma = 4{,}105$, установив, кроме того, значение параметра регуляризации C равным 3,145. При этом SVM-классификатор, обученный на данных о 338 пациентах, неправильно классифицировал 4 пациентов из класса 1, а все пациенты из класса 2 были верно классифицированы. Тестирование, проведенное на данных о 231 пациенте, показало, что ошибочно классифицированы по 2 пациента из каждого класса. Точность классификации составила 98,59 %, а точность кросс-проверки – 98,27 %.

Таким образом, SVM-классификатор абсолютно верно подтвердил диагноз для 206 пациентов и верно не диагностировал заболевание у 355 пациентов. Количество ошибок I рода, допущенных SVM-классификатором, оказалось равным 6, а количество ошибок II рода – 2.

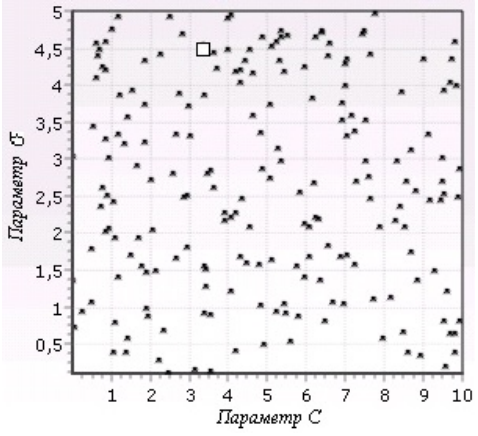

Рисунок 4 – Расположение частиц в рое в момент инициализации (с радиальной базисной функцией ядра)

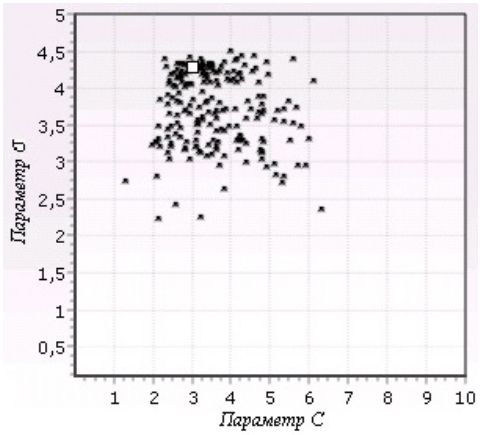

Рисунок 5 – Расположение частиц в рое на 10-й итерации (с радиальной базисной функцией ядра)

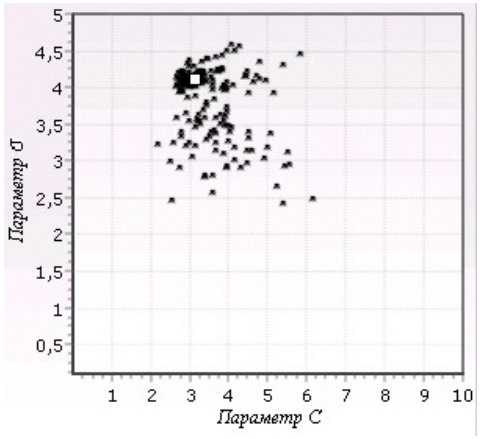

Рисунок 6 – Расположение частиц в рое на 20-й итерации (с радиальной базисной функцией ядра)

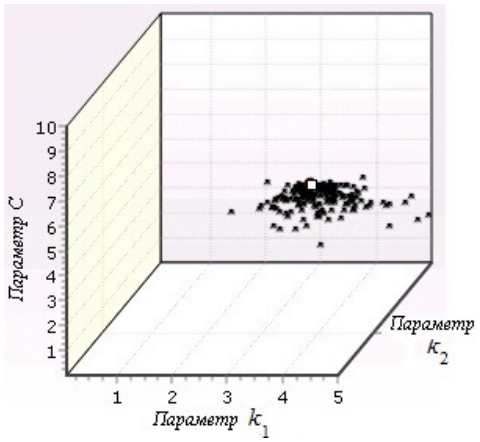

Рисунок 7 – Расположение частиц в рое на 20-й итерации (с сигмоидной функцией ядра)

Аналогичные результаты качества классификации были получены и при классификации 690 заявок на потребительские кредиты в Австралии (http://www.ics.uci.edu/~mlearn/databases/statlog/australian/), где каждая заявка описывалась набором из 14 характеристик [2]. Использование PSO-алгоритма при разработке SVM-классификатора позволило выполнить классификацию

данных с точностью 92,17 % при выбранных PSO-алгоритмом радиальной базисной функции ядра с $\sigma = 3{,}435$ и значении параметра регуляризации $C = 48{,}762$.

В работе [2] для этой же выборки данных о заявках на потребительские кредиты с применением стандартного подхода к обучению и тестированию SVM-классификатора удалось построить SVM-классификатор, обеспечивший при значениях параметра регуляризации $C = 50$ и параметре радиальной базисной функции ядра $\sigma = 15^{0,5}$, найденных методом проб и ошибок [2], соответственно 88,44 % и 74,59 % верно классифицированных одобренных и неодобренных заявок. Результирующая точность классификации составила 81,51 %.

Выводы. Результаты экспериментальных исследований, полученные на основе тестовых данных, традиционно используемых для оценки качества классификации, подтверждают перспективность использования PSO-алгоритма при разработке SVM-классификатора для решения задачи выбора оптимальных типа функции ядра, значений параметров функции ядра, а также значения параметра регуляризации, обеспечивающих высокое качество классификации объектов при приемлемых временных затратах.

Дальнейшие исследования предполагается выполнить при решении вопросов разработки SVM-классификаторов, предназначенных для классификации инновационных и инвестиционных проектов, конкурсных заявок и т.п. [8, 13 – 15].

Библиографический список

1. *Chapelle O., Vapnik V., Bousquet O., Mukherjee S.* Choosing Multiple Parameters for Support Vector Machines // Machine Learning. 2002. № 46 (1-3). P. 131-159.

2. *Lean Yu, Shouyang Wang, Kin Keung Lai, Ligang Zhou.* Bio-Inspired Credit Risk Analysis. Computational Intelligence with Support Vector Machines. Springer-Verlag Berlin Heidelberg. 2008. p. 244.

3. *Вьюгин В.В.* Элементы математической теории машинного обучения: учеб. пособие. М.: МФТИ, 2010. 252 с.

4. *Sokolova Yu.S.* Cluster ensembles development on the base of SVM-algorithm // В сборнике: Modern informatization problems in economics and safety. Proceedings of the XX-th International Open Science Conference (Yelm, WA, USA, January 2015). Editor in Chief Dr. Sci., Prof. O.Ja. Kravets. Yelm, WA, USA, 2015. P. 38-43.

5. *Демидова Л.А., Соколова Ю.С.* Использование SVM-алгоритма для уточнения решения задачи классификации объектов с применением алгоритмов кластеризации // Вестник Рязанского государственного радиотехнического университета. 2015. № 51. С. 103-113.

6. *Вежневец В.* Оценка качества работы классификаторов // Компьютерная графика и мультимедиа. Выпуск № 4 (1). 2006.

7. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы, нечеткие системы / пер. с польск. И.Д. Рудинского. М.: Горячая линия-Телеком. 2004. 452 с.

8. *Ковшов Е.Е., Горяева О.В.* Применение генетического алгоритма при оценке рисков инновационных проектов // Российское предпринимательство. 2010. № 11-3. С. 85-91.

9. *Демидова Л.А.* Модели прогнозирования временных рядов с короткой актуальной частью на основе модифицированного алгоритма клонального отбора // Вестник Рязанского государственного радиотехнического университета. 2012. № 39-2. С. 64-71.

10. *Зайцев А.А., Курейчик В.В., Полупанов А.А.* Обзор эволюционных методов оптимизации на основе роевого интеллекта // Известия Южного федерального университета. Технические науки. 2010. № 12 (113). С. 7-12.

11. *Курейчик В.М., Кажаров А.А.* Использование роевого интеллекта в решении NP-трудных задач // Известия Южного федерального университета. Технические науки. 2011. № 7 (120). С. 30-36.

12. *Jun Sun, Choi-Hong Lai, Xiao-Jun Wu.* Particle Swarm Optimisation: Classical and Quantum Perspectives. CRC Press, 2011. P. 419.

13. *Гусева М.В., Демидова Л.А.* Многокритериальная классификация инвестиционных проектов на основе систем нечеткого вывода и мультимножеств // Научно-техническая информация. Серия 2: Информационные процессы и системы. 2006. № 12. С. 16-20.

14. *Гусева М.В., Демидова Л.А.* Генерирование решающих правил классификации инвестиционных проектов на основе систем нечеткого вывода и мультимножеств // Системы управления и информационные технологии. 2006. Т. 26. № 4. С. 46-53.

15. *Демидова Л.А.* Классификация инвестиционных проектов на основе мультимножеств и нечеткой кластеризации // Известия Южного федерального университета. Технические науки. 2006. № 15 (70). С. 72-79.

16. *Демидова Л.А., Соколова Ю.С.* Лингвистический подход к задаче классификации конкурсных проектов с применением инструментария теории мультимножеств // Вестник Рязанского государственного радиотехнического университета. 2014. № 50-1. С. 109 – 17.