Table 7: Comparison of the least square error E^C and A.E. in Example 8 corresponding for different values of λ and n.

λ		0.50	0.75	1.00	1.25
$n=2$	E^c	5.96e-1	3.19e-1	1.29e-1	4.31e-2
	A.E.	2.22e-0	2.08e-1	3.28e-2	2.67e-4
	CPU Time (in seconds)	2.69	3.08	3.28	3.16
$n=4$	E^c	7.93e-3	2.33e-4	5.10e-7	1.85e-9
	A.E.	2.09e-0	2.19e-2	4.27e-4	3.45e-5
	CPU Time (in seconds)	5.89	9.33	10.67	13.90
$n=6$	E^c	4.59e-6	2.57e-11	9.79e-10	3.05e-10
	A.E.	1.72e-1	8.81e-6	3.27e-5	1.18e-5
	CPU Time (in seconds)	16.55	31.06	34.29	37.73
$n=8$	E^c	3.08e-45	6.61e-14	1.53e-11	2.16e-11
	A.E.	4.40e-16	1.34e-6	5.79e-6	3.98e-6
	CPU Time (in seconds)	39.55	75.15	77.97	81.87
$n=10$	E^c	2.69e-47	2.27e-15	6.45e-13	2.35e-12
	A.E.	4.40e-16	3.12e-7	1.53e-6	1.59e-6
	CPU Time (in seconds)	80.23	149.29	183.23	241.30

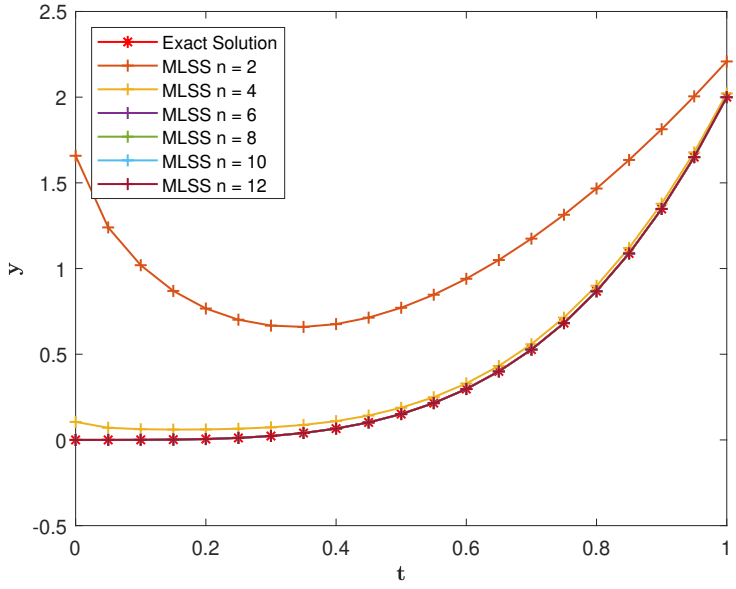

Figure 1: Exact solution and approximate solution for Example 8 with $\lambda = 0.75$ and different values of n.

6.2 Application in machine learning

Machine learning is a field of artificial intelligence which gives computer systems the ability to learn using available data. Recently machine learning algorithms become very popular for analyzing of data and make predictions. The least-squares method is widely used in machine learning to analyze data for regression analysis and classification. In particular, in the regression analysis, the goal of people is to plot a best fit curve or line between data. If someone is interested in discovering the best fit line for one variable using the least squares method for the data, then, in this case, the form of polynomials

(hypothesis/model) is

$$P_1(x;1) = a_0 + a_1 x.$$

We have already shown the advantage of the modified least squares method in Section 5 while searching the best fit curve between data. Also, some data need to have features vector like x^λ, for example $x^{0.5}$. Therefore, we must choose the polynomials (hypothesis/model) of the form

$$P_2(x;0.5) = a_0 + a_1 x^{0.5}.$$

Example 9. Consider the data in Table 8, which is related to pharmaceutical sales of some company.

Table 8: Pharmaceutical sales of some company [1]

Years	2014	2015	2016	2017	2018
Sales	10000	21000	50000	70000	71000

For Example 9, we fit the data with the $P_1(x;\lambda) = a_0 + a_1 x^\lambda \in M_1^\lambda$. To simplify the calculation, the years in Table 8 are replaced by the coded values. For example, 2014 is 1, 2015 is 2, and so forth on. We use the data from 2014 to 2017 to fit the curve $a_0 + a_1 x^\lambda$ and predict the sales in 2018. After implementation of the proposed method discussed in Section 4, the results are described in the Table 9. From the Table 9, one can observe that for $\lambda = 0.5$, the predicted sales in 2018 is close to the real value in 2018.

Table 9: Prediction of pharmaceutical sales of some company in the year 2018 with different values of λ

λ	0.5	0.75	1.00	1.25	1.5
Predicted sales in 2018	69692	80546	90000	98307	105870

Example 10. In this Example, we will fit the data generated by the solution of the fractional differential equation, which describes the dynamics of the world population.

$$\begin{aligned} {}_{\mathrm{C}}D^{\alpha}_{0,x} y(x) &= Py(x), \quad 0 < x \leq 1, \\ y(0) &= y_0 \end{aligned} \tag{6.7}$$

The solution of the fractional differential equations is given by $y(x) = y_0 E_\alpha(Px^\alpha)$, where y_0 is the population of the world at initial time, $E_\alpha(\cdot)$ is the Mittage-Leffler function, P is the production rate, and α is the fractional order of the model.

In [5], authors are finding the values of P and α using the world population data from 1910 to 2010, which are $1.3502e-2$ and 1.39, respectively. For this Example, we generating the data of $y(x)$ with the help of the exact solution of the fractional differential equation (6.7) at 11 equispaced points on the interval $[0,1]$ for the values of P and α. Here, we consider the two cases: first, we approximate $y(x)$ with the $\sum_{i=0}^{n} a_i x^{i\lambda}$, while in the second case, we approximate $y(x)$ with the $\sum_{i=0}^{n} b_i P_i(x;\lambda)$, where $P_i(x;\lambda),\ i = 0,1,\ldots,n$, are the orthogonal fractional polynomials with respect to data $\{x_j,\ j = 0,1,\ldots,10\}$, and we generated $P_i(x;\lambda),\ i = 0,1,\ldots,n$, with help of the Theorem 4.2. In the first case for finding the value of parameter $a_i,\ i = 0,1,\ldots,n$, we have to solve the system of equations because $\{1, x^\lambda, \ldots, x^{n\lambda}\}$ are non-orthogonal fractional polynomials with respect to data $\{x_j,\ j = 0,1,\ldots,10\}$, and for a large value of n, we may end with the ill-conditioned coefficient matrix. In the second case, we are directly finding the parameter values $b_i,\ i = 0,1,\ldots,n$ using Equation (4.13). Table 10 demonstrates the absolute error (A.E.) at $x = 0.55$ with different values of λ and n for both cases. From Table 10, one can observe that in the case of $\lambda = 1.39$, we get the better results compared to other values of λ for each n. This happens because the exact data is generated for $\alpha = 1.39$.

Table 10: A.E. at $x = 0.55$ for Example 10 with different values of λ and n.

For non-orthogonal fractional polynomials				
n	$\lambda = 0.50$	$\lambda = 1.00$	$\lambda = 1.50$	$\lambda = 1.39$
2	1.96e-4	4.16e-5	4.78e-7	6.68e-10
3	8.79e-7	1.61e-5	1.50e-5	6.86e-13
4	4.19e-7	7.01e-6	1.62e-6	5.10e-15
5	2.64e-8	2.00e-6	4.79e-6	5.11e-15
6	3.61e-9	1.35e-6	8.31e-7	2.21e-13
For orthogonal fractional polynomials				
n	$\lambda = 0.50$	$\lambda = 1.00$	$\lambda = 1.50$	$\lambda = 1.39$
2	4.61e-4	2.01e-3	1.08e-4	8.36e-9
3	1.05e-5	9.55e-5	8.68e-5	4.76e-12
4	1.98e-6	6.29e-5	1.00e-4	3.18e-15
5	2.63e-7	4.93e-5	1.54e-4	4.83e-15
6	5.57e-8	4.24e-5	3.12e-4	2.09e-16

7 Conclusions and future work

The introductory Section shows the demands of the least squares method in various fields. So the modification in the least squares method is the demand of time due to its application. The main idea of this work has been to modify the least squares method using the space M_n^λ. The numerical results for test Examples have been reported to show the efficiency of the modified least squares method over the classical least squares method. We can use the current work in the support vector machines in the future. Support vector machines are part of machine learning to analyze data for classification and regression analysis. In most cases, data are non-linear. So, we find some non-linear transformation ϕ that can be mapped the data onto high-dimensional feature space. The transformation is chosen in such a way that their dot product leads the kernel style function

$$K(x, x_i) = \phi(x).\phi(x_i).$$

If we choose the polynomial classifiers [11] of degree 2 and we have n data set $x_i,\ i = 1, \ldots, n$, in this case

$$\phi(x) = (1\ x_1\ x_2 \ldots x_n\ x_1^2\ x_1x_2 \ldots x_2^2 \ldots x_n^2)^T.$$

However, one can use fractional polynomial classifiers instead of classical polynomial classifiers for more accurate results. Further, one can also use fractional orthogonal polynomials as an activation function in the neural networks to avoid the vanishing gradient problem.

Acknowledgements

The First author acknowledges the support provided by University Grants Commission (UGC), India, under the grant number 20/12/2015(ii)EU-V. The second author acknowledges the support provided by the SERB India, under the grant number SERB/F/3060/2021 − 2022. The third author acknowledges the financial support from the North-Caucasus Center for Mathematical Research under agreement number 075 − 02 − 2021 − 1749 with the Ministry of Science and Higher Education of the Russian Federation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.