Evaluation of Bernstein Polynomial as a Machine Learning technique

Deekshant Wadhwa¹  Nitesh Kumar Singh²  Nishtha Jatana³

¹ Student, Maharaja Surajmal Institute of Technology
deeckshantwadhwa2000@gmail.com

² Student, Maharaja Surajmal Institute of Technology
niteshks247@gmail.com

³ Assistant Professor, Maharaja Surajmal Institute of Technology
nishthajatana@msit.in

Corresponding Author Email: nishthajatana@msit.in

Abstract. Regression algorithms are used to device a function to predict the output based on that function for new unseen inputs. We have used Bernstein polynomial as a type of approximation curve to fit a polynomial between all features of a dataset so as to perform regression on it. As approximation curves do not generally meet most of the input points, which are used to create it, hence the chances of it over-fitting to a model are low. In this paper, we have used Bernstein polynomial separately for each feature versus output graph, assuming that each feature in the dataset is independent of each other and combines the outcome of each curve to predict the final result.

Keywords. Machine learning, Regression, Supervised learning, Algorithm, Approximation Curves, Bernstein Polynomial, Curve Fitting

1. Introduction

Machine learning is used to teach a machine, patterns among data to make them capable of giving outputs for the inputs for which they have not seen any output based on the patterns they have learned [1]. Humans are capable of looking at data and predicting its outcome till a certain limit of amount and complexity. However, through machine learning, computers can predict the outputs for data no matter its size or complexity given sufficient time, storage, and processing capabilities. The purpose of machine learning is to predict accurate outputs based on similar but mostly unseen data. Machine learning is broadly divided into two parts: regression and classification [2]. In this paper, we are discussing a regression machine learning algorithm using Bernstein Polynomial.

1.1. Formula Used
Bernstein Polynomial [3] is an approximation polynomial named after Sergei Natanovich Bernstein. The popular Bézier curve is also a special case of Bernstein Curve for ‘a = 0’ and ‘b = 1’. Bernstein polynomial is given by Eq.(1).
1.2. Purpose of Study
The purpose of this paper is to study the aspects of a rather rarely used methodology of curve fitting. There are several other algorithms for regression available, but this paper presents a rather simple approach to regression in machine learning. Also, this paper presents the effects of varying 'a' and 'b' parameters in Bernstein Polynomial over the datasets used. Furthermore, in this paper, we analyze the effect of using Bernstein Polynomial in creating the approximation equation for regression and analyze how each feature of data is used for the prediction of the outcome.

In this paper, we address the following research questions.
RQ1. How effective is Bernstein Polynomial in performing regression?
RQ2. What are the prospects of using Bernstein polynomial in the real world?
RQ3. What is the scalability of the approach?
RQ4. What is the significance/advantages of the approach described?
RQ5. What are the further enhancements of this approach?

2. Related Works
Bernstein's polynomials can be used to create a fuzzy regression model and perform better than existing conventional models on fuzzy real-life data [4]. They applied the Bernstein regression model to a real-life dataset and further improved this idea by combining its features with other models. This method can be used for much better performance by using Bernstein’s polynomials approach to expand the data into 'n' dimensions and using that expansion in different models. Lian Fang et al. [5] described the concept of parametric curve fitting and implemented it using the Hermite Basis function. Using unorganized data, the author tries to depict the model performance on messy real-life data collected in an unorganized manner. This paper addresses this concept of using Parametric Curve in regression models using Bernstein's Polynomials to show the performance of thus created model through the usage of a real-life dataset in a way where the organization does not matter. Forrest [6] depicts the use of Bezier Curves, a parametric curve using the basic function of Bernstein Polynomial, for the approximation of the curve. Bezier Polynomial, being globally affected for every change in any control point, performed quite well for this purpose. This paper takes the route of Bernstein Polynomials, which are locally affected for every change in any control point for the same. This optimization on local values gives stable and easier control to the user on the curve for optimizing purpose. Also, new data can be added to the model without disturbing the entire curve to optimize a specific local range of values. Linghui Zhao et al. [7] have used the Bezier Curve to fit the training data in a 2D space and optimized the parameters of their equation using genetic algorithm. This paper has implemented a mutation approach to achieve proper parameters to get their fitness function to a practical value. Chia-wei liao et al. [8] have applied Bernstein-bezier curves to smooth their data for handwriting recognition of handwritten Chinese language. Using the improved data after processing the Chinese characters under Bernstein-bezier curves, the authors have implemented stroke segmentation to achieve their task of recognition of handwritten Chinese characters.

3. Bernstein Polynomial
Named after Sergei Natanovich Bernstein, Bernstein Polynomial [3] are approximate curves used for curve fitting. Bernstein Polynomial is used to create this formula, which is given by Eq. (2).

$$B^n_i(t) = \frac{n!}{i!(n-i)!} (t-a)^i (b-t)^{n-i} \frac{1}{(b-a)^n}$$

For i = 0, 1, ..., n

$$G(t) = \sum_{i=0}^{n} P_i B^n_i$$ (2)
Where, 

- $G(t)$ is the generator function used to create the parametric equation for plotting in each dimension.
- $P_i$ is the $i^{th}$ input to the generator function.
- $B^i_n$ is the Bernstein coefficient

\[
B^i_n(t) = \binom{n}{i} (t-a)^i (b-t)^{n-i} \quad (3)
\]

Hence,

\[
G(t) = P_0 B_0^n + P_1 B_1^n + P_2 B_2^n + \cdots + P_n B_n^n \quad (4)
\]

This generator function is used to plot each feature of the dataset independently against the output. Hence for each feature of the data, a generator function equation is created, and using this equation along with the generator equation of the output as parametric equations, graphs are plotted for each feature independently.

\[
G_X(t) = \sum_{i=0}^{n} P_i B_i^n \quad (6)
\]

\[
G_Y(t) = \sum_{i=0}^{n} P_i B_i^n \quad (7)
\]

As 't' in these parametric equations is a variable on which all the equations depend hence the value of 't' needs to be fixed before plotting any equation. In Bernstein Polynomial, the value of 't' lies between [a, b]. Although fixing this value between [a, b] makes sure that the regression takes place between the lowest and highest values of the training data, but this leads to two downsides of Machine Learning through Bernstein Polynomial, that is, the values for prediction through this model should lie between the minimum and maximum training value and that prediction of points near the minimum and maximum value of training data at the time of testing is not very accurate.

All the equations are dependent on the value of t, which lies between [a, b]; hence the value of change between 'a' and 'b' or the sampling rate should be chosen wisely. The sampling rate is the number of samples the algorithm takes between 'a' and 'b' to plot the parametric curve for prediction. It is the number of parts in which the entire plotting range is divided into for the curve to be plotted. Low sampling rates can lead to fast results but would decrease the accuracy of the algorithm. Also, this would lead to broken points in the graph, and the algorithm would need to interpolate for the desired points would further decrease the accuracy of the algorithm.

![Sampling 10](image)

**Figure 1. Sampling is 10**

In Fig.1, it is quite evident that the Bernstein curve was not appropriately plotted due to the low sampling rate as it a parametric curve and has led to an approximation between two sampling points.
Setting the sampling rate to a higher value will lead to better and more accurate results, and the graphs plotted with it will also be smooth. Altogether this would make the predictions of the algorithm more accurate but would take more time and computation to compute the answer.

![Sampling](image-url)

Figure 2. Sampling is $10^4$

Hence, a balance is to be maintained between the low and high values of the sampling rate. It should neither be very high nor very low. It should be kept moderate, like in Fig.2, according to the situation it is being used in, computation power available, computation time available, number of features available, number of samples available, and most importantly, the maximum amount of accuracy required. (Negative sampling rate is a wrong value and would lead to no answer from this algorithm).

4. Approach
The basis of our approach is the Bernstein Polynomial, which approximates the input values to create an approximate curve that can be used in machine learning for regression. Fig.3 shows the basic structure of any training dataset, which can be used for regression.

| F1 | F2 | ... | Fm |
|----|----|-----|----|
| X1 | A11| A12 | ... | A1m |
| X2 | A21| A22 | ... | A2m |
| ...| ...|     | ... |     |
| Xn | An1| An2 | ... | Anm |

Figure 3. Input training Data for regression

| Y: | Y1 | Y2 | ... | Yn |
|----|----|----|-----|----|

Figure 4. depicts the general form of the actual output of the training dataset.

For the training of the model, that is, for the creation of the Bernstein polynomial equation $G(t)$, the training data is first split into 'm' individual datasets containing the input from a single feature and the corresponding output as shown in Fig.5.

| F1 | Y |     | Fm | Y |
|----|---|-----|----|---|
| A11| Y1|     | A1m| Y1|
| A21| Y2|     | A2m| Y2|

4
Each of these individual tables is then sorted in ascending order individually, pivoted on 'T' keeping the (Anm, Yn) features together. This is done to keep future calculations simple and fast. Once sorted, each table is then used to create the generator function of their own – \((X1(t), Y1(t)), \ldots, (Xm(t), Ym(t))\) where \(Xm(t)\) and \(Ym(t)\) represent \(G_X(t)\) and \(G_Y(t)\) respectively. They are then used to plot 'm' individual graphs for prediction of the output during the testing.

These equations are then used against sample training data (as shown in Fig.6)

Each feature of this data is then split into its corresponding feature, similar to that during training, but as this prediction is made from the previously formed equations, hence no sorting is required. For each value of \(Aa,m\), its corresponding generator equation in \(X\) (\(G_X(t)\)) is solved for 't' and using that value of 't' in the corresponding \(G_Y(t)\) equation, the prediction for that particular feature is made. The prediction done for every input depends on the number of features describing it. Using these equations and the sample testing data, 'm' predictions are made for an input which has 'm' features. All these predictions for an input are combined to create one single output. The predicted outputs can be combined in several ways based on the situation in which it is being used –

1. Mean – average of all predictions for an input is used as the final output of the algorithm
2. RMS – root mean square can be used in cases where a direct average of predictions does not produce a useful outcome
3. Median – it represents the central value of all feature predictions so as to eliminate peak values.
4. Other Methods – Other ways of combining these predictions can be mode, geometric mean, or any other custom method.

5. Experiments
In this work, two datasets have been used for training and prediction, with their individual predictions combined using mean, RMS, and median methods. The datasets are randomly split into training and testing data. The predictions for each dataset and its combination method (three per dataset) were analyzed using the r-squared score and mean absolute percentage error. The sampling rate was chosen high enough to get highly accurate predictions. The variation in 'a' and 'b' are ranging from 0 to 5 each. The datasets used are:

1. Housing Rates dataset [9] - A Simple Dataset which deals with the prediction of House rates based on features - lot size, number of bathrooms, number of bedrooms, number of floors (4 features), and the output is the price of the house. Three hundred twenty-seven inputs have been used for training, and two hundred nineteen inputs are used for testing purpose.
2. Used Cars Price Prediction dataset [10] – It is a dataset containing details of used cars with features used for prediction such as the total kilometers driven in the car by the previous owner(s) in KM, the standard mileage offered by the car company in kilometer per liter (kmpl),
the displacement volume of the engine in cubic centimeters (cc), the maximum power of the engine in brake horsepower (bhp), and the number of seats in the car. The current selling price of the used car is the output of the model. Three hundred inputs are used for training, and two hundred inputs are used for the testing.

6. Result
This section presents the answers to RQs stated in section 1.2

Answer to RQ1. Bernstein Polynomial has not yielded significant results for performing regression. The R2 score of this method on various datasets and with various combination methods is very low but it being a relatively crude method of regression compensates this downside by being fast. In this study, only three basic combinational methods (Mean, Median, and RMS) have been used on the results of Bernstein Polynomial. As Bernstein Polynomial tends not to overfit the training set, it is possible that on applying some other sophisticated combinational methods (like KVM, neural nets) on the results of Bernstein Polynomial, its R2 score on the testing dataset may thereby improve. The results of applying Bernstein Polynomial on the Housing Rates [9] dataset is as follows –

Figure 6. Bernstein polynomial prediction graph for Lot size v/s House Price (Output) with a = 2, b = 4

Figure 7. Bernstein polynomial prediction graph for Number of Bedrooms v/s House Price (Output) with a = 2, b = 3
Figure 8. Heatmap for R2 Score with Combination Method = Average

Figure 9. Heatmap for R2 Score with Combination Method = Median

Figure 10. Heatmap for R2 Score with Combination Method = RMS
Mean absolute percentage error –
Combination Method: Mean = 33.91%
Combination Method: Median = 25.59%
Combination Method: RMS = 39.36%

The results of applying the Bernstein Polynomial method on Used Cars Price [10] Prediction dataset are as follows –

Figure 11. Bernstein polynomial prediction graph for Kilometers driven by car v/s Price of the car (Output) with a = 1, b = 5

Figure 12. Bernstein polynomial prediction graph for Displacement Volume of engine v/s Price of the car (Output) with a = 0, b = 1

Figure 13. Heatmap for R2 Score with Combination Method = Average
Answer to RQ2. In this study, the effectiveness of this method over any dataset has not given results high enough to be used on real-world applications where the correctness of such models needs to be almost perfect. Although this technique does not prove to be of direct use in real-world applications, yet, as it considers each feature of a dataset as an independent, unrelated entity so the uncombined results of Bernstein polynomial can be used to increase the number of features of a dataset for other machine learning or regression models to increase their efficiency on real-world applications.

Answer to RQ3. In this paper, before applying the Bernstein polynomial, an assumption is taken that each feature of the dataset is independent of others. For each feature, a new Bernstein equation has to be created. The number of inputs in the training set is also directly proportional to the complexity of the Bernstein equation hence formed, and so is the time taken to create that equation. Hence, the time taken for the algorithm to be trained on a training set increases as the number of inputs or the number of features in the training set increases. However, this time may be very less as compared to the time taken by other regression methods used on the same data. Henceforth, the scalability of the proposed approach is relatively high in terms of execution time needed.

Answer to RQ4. The results of the approach described are not quite significant, yet they open up an opportunity to explore other advantages of this algorithm. The algorithm, although being quite simple, it does not give many accurate results but is still very quick. This method is a straightforward approach towards regression; hence it is quite fast. Therefore, using it as a baseline model for actual real-world models is one of the advantages of the approach described. The results of this model can be considered as the minimum outcome of any model used on that data, achieved almost instantly.
Answer to RQ5. Bernstein Polynomial method for regression can be improved by not directly using its outcome as the final result but by utilizing its uncombined results as an input for other machine learning or regression techniques (like neural networks, KVMs, and so on) which require a large number of different features to give better results. In the approach described above, only one feature is transformed at a time in 2D space using Bernstein polynomial, but if more than one, say two features are considered at a time then \( \binom{n}{2} \) features can be created out of just \( \binom{n}{1} \) features. Hence, as the number of features thus created increases, so will the accuracy, and therefore the results of the model might also increase.

7. Conclusion
The application of Bernstein Polynomial in machine learning regression has been studied for various combination methods and over various datasets. The variation of 'a' and 'b' parameter and their effect over the output has also been studied, as shown previously in this paper. R2 score and mean absolute percentage error for both the datasets for various combinations techniques, with varying 'a' and 'b' parameters have been determined. Although the results of this algorithm over these datasets are not entirely promising, yet it can be considered as a baseline model for real-world models. Bernstein Polynomial algorithm can also be used to increase the number of features for some other regression or machine learning algorithm. This can be achieved by not combining the outputs of this algorithm but instead using them separately as new features along with old features of the dataset for some other machine learning or regression method.

References

[1] M. D, S. DJ and T. CC, "Machine learning," Neural and Statistical Classification, pp. 1 - 298, 1994.
[2] A. Dey, "Machine learning algorithms: a review.," International Journal of Computer Science and Information Technologies, vol. 7, no. 3, pp. 1174-1179, 2016.
[3] T. A. Pastva, Bezier Curve Fitting, CA: NAVAL POSTGRADUATE SCHOOL MONTEREY, 1998.
[4] d. Hierro, A. F. R. López, J. Martínez-Moreno, C. A. Peña and C. R. L. d. Hierro, "A fuzzy regression approach using Bernstein polynomials for the spreads: Computational aspects and applications to economic models.," Mathematics and Computers in Simulation, vol. 128, pp. 13-25, 2016.
[5] L. Fang and D. Gossard, "Multidimensional curve fitting to unorganized data points by nonlinear minimization.," Computer-Aided Design, vol. 27, no. 1, pp. 48-58, 1995.
[6] A. R. Forrest, "Interactive interpolation and approximation by Bézier polynomials.," The Computer Journal, vol. 15, no. 1, pp. 71-79, 1972.
[7] Zhao, Linghui, J. Jiang, C. Song, L. Bao and J. Gao, "Parameter optimization for Bézier curve fitting based on genetic algorithm.," in International Conference in Swarm Intelligence, Berlin, Heidelberg, 2013.
[8] Liao, Chia-Wei and J. S. Huang, "Stroke segmentation by Bernstein-Bezier curve fitting," Pattern Recognition, vol. 23, no. 5, pp. 475-484, 1990.
[9] [Online]. Available: https://www.kaggle.com/ananthreddy/housing.
[10] [Online]. Available: https://www.kaggle.com/avikasliwal/used-cars-price-prediction.