Piecewise-linear fitting for multi-slope regression data

Nalinda Kulathunga[1], Yunjiao Wang[1], Daniel Vrinceanu[1], Nishanth Rajiv Ranasinghe[2], Lei Huang[2] and Yonggao Yang[2]

[1]Department of Mathematics, Texas Southern University, Houston, TX 77004.

[2]Department of Computer Science, Prairie View A & M University, Prairie View, TX 77446.

Abstract

In data science, regression analysis plays a vital role in predicting continuous target variables. The significance of a model depends on how well it extracts the hidden complexity of the data. We investigate an approximation method where continuous regression data can be described as a combination of multiple linear functions based on multiple distinct partitions of the data. To demonstrate the fitting process, the function space is solved using piecewise-linear fitting algorithm that could best describe the regression data. The algorithm was tested using simulated datasets with a positive convexity and known slopes. Partition optimization was done using a heuristic procedure, which works based on the maximum value of the predictions given by the spline fitting functions obtained from each partition, as explained in [1]. We used the exact equation as well as the gradient descent approximation to fit the data in each partition. Unstable gradient behavior was avoided using gradient clipping technique. The method was successful in identifying partitions with different slope parameters and the resulting function space agrees well with the known slopes of the data. Visualization is presented to explain the propagation of partition boundaries during the optimization process for 1-D and 2-D cases. Our results indicate that the partition boundaries were able to capture the boundaries of near- linear portions in data, which would help to approximate the convex regression data using max-affine splines. We tested the method on real data using the temperature vs. ocean depth profile from CalCOFI [8]. As future directions, we will discuss the applicability of this method to predict trends in seismic data and explain the deep learning [2].

Method

The k-term max-affine fitting problem:

$$\text{minimize} \quad J = \sum_{i=1}^{m} \left(\max_{j=1,\ldots,k} (a_j^T u_i + b_j) - y_i \right)^2 \qquad \ldots.(1)$$

Where, m is the number of data points, k is the number of partitions, a and b are slopes and bias respectively, y is the label.

First step is the initialization of the partitions. We have used both 1). random subsets of fixed size and 2). Initial clusters based on random cluster centers. Both methods show similar results. However the later method shows faster convergence. In the later method we randomly chose cluster centers $p_1, \ldots, p_K$, and the initial partitions were defined as,

$$P_j^{(0)} = \{ i \mid \|u_i - p_j\| < \|u_i - p_s\| \text{ for } s \neq j \}, \quad j = 1, \ldots, K \qquad \ldots.(2)$$

Where, P_j^0 partition contains the set of data points closest to p_j.

Then for each partition j = 1, ….., k, the least-square fit is done. The list of slopes and biases are updated and used for the updating of partitions before the next round of fitting. For the simplest case, there is a unique pair of a, b that minimizes.

$$\sum_{i \in P_j^{(l)}} (a^T u_i + b - y_i)^2 \qquad \ldots.(3)$$

Using there new values of a and b, the updating of new partitions is done by assigning a datapoint with index l to a partition with index s if,

$$f^{(l)}(u_i) = \max_{s=1,\ldots,k} (a_s^{(l)T} u_i + b_s^{(l)}) = a_j^{(l)T} u_i + b_j^{(l)} \qquad \ldots.(4)$$

This procedure is executed until the partition at an iteration is the same as the partition at the previous iteration or until a defined maximum number of iterations is reached.

The compete description of the method can be found in section 3 of reference : [1]

Loss minimization

Gradient Descent:

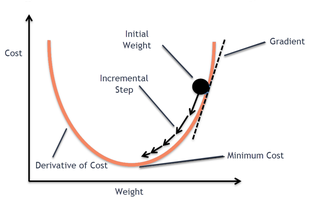
In gradient descent algorithms, models can be trained by minimizing the Mean Square Error (MSE). Consider the matrix representation of the problem: Y = θX, where Y represents labels, X represents features and θ represents the weights. The MSE(θ) and the gradients are defined as;

$$MSE(X, \theta) = \frac{1}{m} \sum_{i=0}^{m} (y_{pred}^{(i)} - y^{(i)})^2 = \frac{1}{m} \sum_{i=0}^{m} (\theta^T . x^{(i)} - y^{(i)})^2 \qquad \ldots.(5)$$

$$\frac{\partial}{\partial \theta_j} MSE(X, \theta) = \frac{2}{m} \sum_{i=0}^{m} (\theta^T . x^{(i)} - y^{(i)}) x_j^{(i)} \qquad \ldots.(6)$$

Here, m stands for number of data instances, *i* stands for the index related to length of the data set and *j* stands for the number of features (considering *j* = 0 for the bias term). Weights will be updated in each iteration so that the MSE(θ) reaches a global minimum with a rate η, which is also known as the learning rate.

$$\theta^{(next)} = \theta^{(now)} - \eta \nabla_\theta MSE(\theta) \qquad \ldots.(7)$$



Exact Equation:

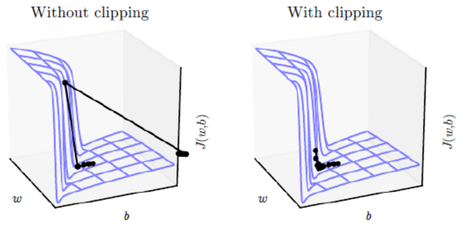
$$\begin{bmatrix} a_j^{(l+1)} \\ b_j^{(l+1)} \end{bmatrix} = \begin{bmatrix} \sum u_i u_i^T & \sum u_i \\ \sum u_i^T & |P_j^{(l)}| \end{bmatrix}^{-1} \begin{bmatrix} \sum y_i u_i \\ \sum y_i \end{bmatrix} \qquad \ldots.(8)$$

Exploding Gradient

In certain situations, due to the shape of the cost function, we may not be able to arrive at the global minimum efficiently. Exploding gradient is one of these situations. A sudden fall in the error surface as shown in the two figures to the right, results in very large gradients. To prevent the gradients from blowing up, the gradient-clipping can be implemented. The gradient (g) is controlled with the help of a threshold value (η) and prevent from taking huge steps along the error surface.

We have implemented clipping as shown in equation (9) . At the second stage of this analysis, when we work with real data, clipping is very useful to prevent slopes becoming blown-up.

Figure reference : [7]



$$g \leftarrow \frac{\eta g}{\|g\|} \qquad \ldots.(9)$$

Where g is the gradient, η is the threshold and $\|g\|$ is the norm of g.

Algorithm

Least-Square Partition Algorithm [1]

Given portion $P_l^{(0)}, \ldots\ldots\ldots, P_K^{(0)}$ of {1, ……. , m}, iteration limit max

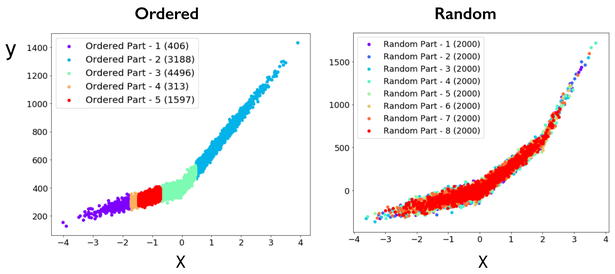
For l = 0, …. , l max

1. Compute a and b as in equation (8) or approximate using gradient descent.
2. Form the new partitions according to equation (4)
3. Quit if the current and previous portions contain the same data.

- **Tools:**
- Python 3.7
- numpy
- Pandas data frames
- Scikit-learn [3]
- TensroFlow [4]
- TensorFlow keras [5]
- Matplotlib [6]

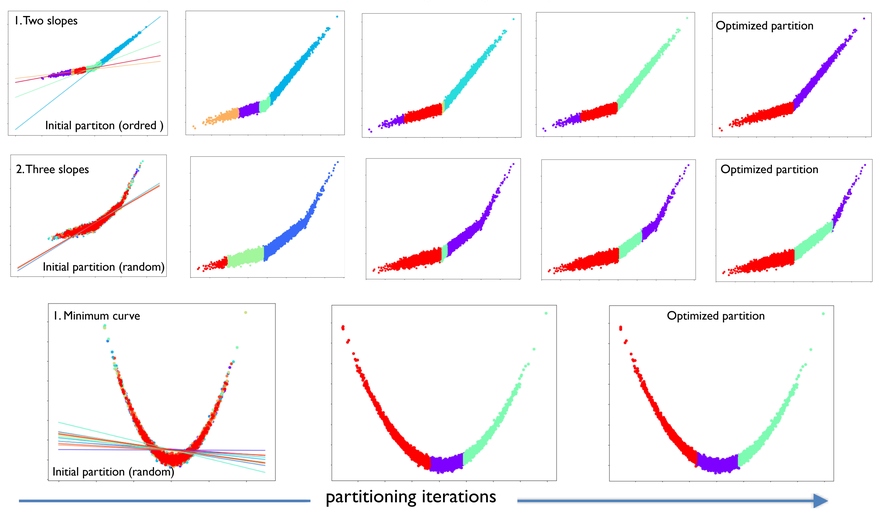
Initiation

We used both approaches 1). Using random subsets and 2). Clusters found using the equation (2) for the initiation of the partitions. Left figure shows the ordered partitions obtained using equation (2) and right figure shows partitions obtained using fixed random subsets.

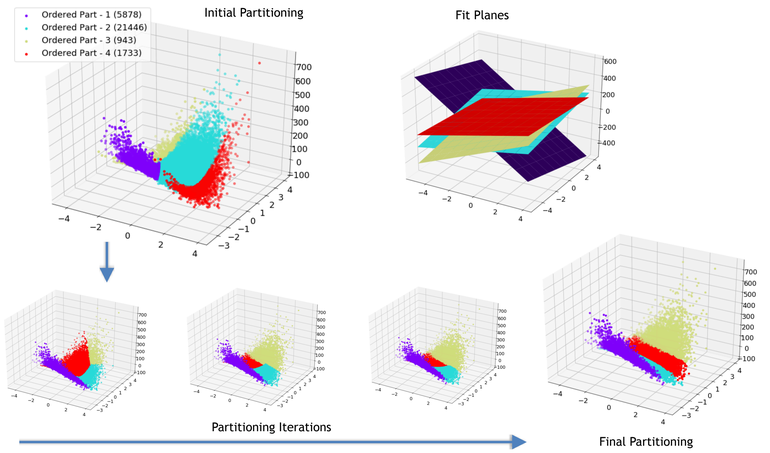


Results - I (1-D)

The partitioning propagation is visualized below for three types of 1-D convex data. 25 partitioning iterations were used and only the iterations where the partitioning was significantly different are shown below.

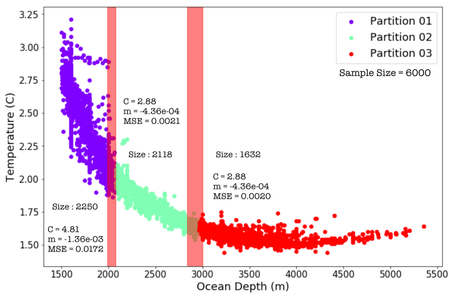


Results - II (2-D)



Above figures show the partition propagation in 2-D and in the bottom right figure the final (optimized) partitions are shown. It is clear that the final partitions are formed in parallel to each other and along the direction of the convexity of the data.

Results - III (Real Data)



Test on oceanic data:

This fitting algorithm was used to identify the slopes of temperature vs ocean depth profile. We used the CalCOFI [8] data set (deep sea : > 1500 m) for the testing. Outliers were removed and initial partition was done randomly with fixed partition size (10 partitions, each with size 600). Slopes (m) and Intercepts (C) of the each linear fitting is shown in the figure with MSE of the fit in each partition. Red bars show the standard error in the partition boundaries. We will be investigating why the rate of temperature drop changes at 2033.25 +/- 41.73 (m) and 2890.50 +/- 84.27 (m).

Discussion

We have presented our results from the piece-wise linear fitting method tested on regression data with visualizations up to 2-D. In this method, the piece-wise fitting is possible only if the data are convex. The partitioning procedure was followed according to the steps explained in [1]. Both random and ordered initial partitioning methods were tested and showed similar results. However, the ordered initial partitioning helped to achieve the convergence in partitioning faster. We visualize the partitioning procedure using simulated 1-D (2-slopes, 3-slopes and minimum curve) and 2-D regression data. Results show that the partitioning algorithm was able to capture the boundaries of the clusters with distinct slope parameters efficiently. We have encountered non-convergence in very few occasions but the change of initial partitioning helped to fix the problem. Another problem with fitting of higher dimensional data was the exploding gradient. We have implemented the clipping technique to avoid the exploding gradient form taking huge steps in the slope space. We successfully used the fitting method to cluster temperature vs. ocean depth (CalCOFI [8]) data into linear partitions. We are working on understanding and evaluating the convexity of the data which will eventually help us to develop a pre-processing method to find out data sets where this method can be deployed. The max affine spline approximation has been discussed in the context of deep learning [2] and we also plan to explore the partitioning and affine spline fitting in the hidden layers of deep nets as a future direction.

References :

1. Magnani, A. & Boyd, S.P. (2009) Convex piecewise-linear fitting. Optim Eng, 10(1):1-17. https://doi.org/10.1007/s11081-008-9045-3
2. Balestriero, R. & Baraniuk, R. (2018) Mad-max: Affine Spline Insight into Deep Learning, arXiv:1805.06576v5
3. Scikit-learn packages, https://scikit-learn.org/stable/
4. TensorFlow packages, https://www.tensorflow.org/
5. Keras packages, https://keras.io/
6. Matplotlib packages, https://matplotlib.org/
7. Ian Goodfellow et. al, "Deep Learning", MIT press, 2016
8. California Cooperative Oceanic Fisheries Investigations (CalCOFI) data set, https://www.kaggle.com/sohier/calcofi

Acknowledgment :

This work has been supported by National Science Foundation through an Excellence in Research award (CNS-1831980).

Contact : nalinda05kl@gmail.com