Partitioning Nearest Neighbors approach to Regression Variation Improvement in tree-based Approaches

Abhinav Mathur[1], Sunny Verma[2]

1 – Clinton Health Access Initiative, 2 – Absolut Data Research & Analytics Private Ltd

Abstract

Good generalized machine learning models should have high variability post learning [1]. Tree-based approaches[2] are very popular due to their inherent ability in being visually representable for decision consumption as well as robustness and reduced training times. However, tree-based approaches lack the ability to generate variations in regression problems. The maximum variation generated by any single tree-based model is limited to the maximum number of training observations considering each observation to be a terminal node itself. Such a condition is an overfit model. This paper discusses the use of a hybrid approach of using two intuitive and explainable algorithms, CART[2] and KNN[3] regression to improve the generalizations and sometimes the runtime for regression-based problems. The paper proposes first, the use of using a shallow CART algorithm (Tree depth lesser than optimal depth post pruning). Following the initial CART, a KNN Regression is performed at the terminal node to which the observation for prediction generation belongs to. This leads to a better variation as well as more accurate prediction than by just the use of a CART or a KNN regressor as well as another level of depth over an OLS regression[1]. The paper shall present the algorithm referred to as PNN (Partitioned Nearest Neighbors) and its benefits when dealing with higher dimension data.

Introduction to CART & KNN algorithm

CART (Classification & Regression Trees)

CART stands for **C**lassification **A**nd **R**egression **T**rees and among one of the most popular and widely used types of tree based algorithms. CART uses the concept of MSE (Mean Squared Errors) for regression as a loss function and Gini impurity/entropy for classification with respect to the bins of the target variable to partition data into smaller buckets. The numeric value given is the mean of the target variable in the data points in the terminal bucket, also called as the leaf node. Another useful feature of CART is taking a bivariate approach in portioning, i.e. – For each split or partition performed by the algorithm, it will only add a condition to one variable. This allows us to also reconstruct variable importance. The variable with the first split or the primary split has the maximum predictive capability with respect to the target variable.

$$I(\text{node}) = \underbrace{\text{MSE}(\text{node})}_{mean-squared-error} = \frac{1}{N_{node}} \sum_{i \in node} \left(y^{(i)} - \hat{y}_{node}\right)^2$$

$$\underbrace{\hat{y}_{node}}_{mean-target-value} = \frac{1}{N_{node}} \sum_{i \in node} y^{(i)}$$

Figure 1.

Since partitioning of data can be reconstructed as a top down approach with the variable and level explaining the highest amount of MSE/gini/entropy, a hierarchy of the splits also provides an insight into variable importance.

A sequence of data partitions can be visualized as a set of decision rules in the form of a tree of if-else statements leading to the popular term of ‘decision trees’.

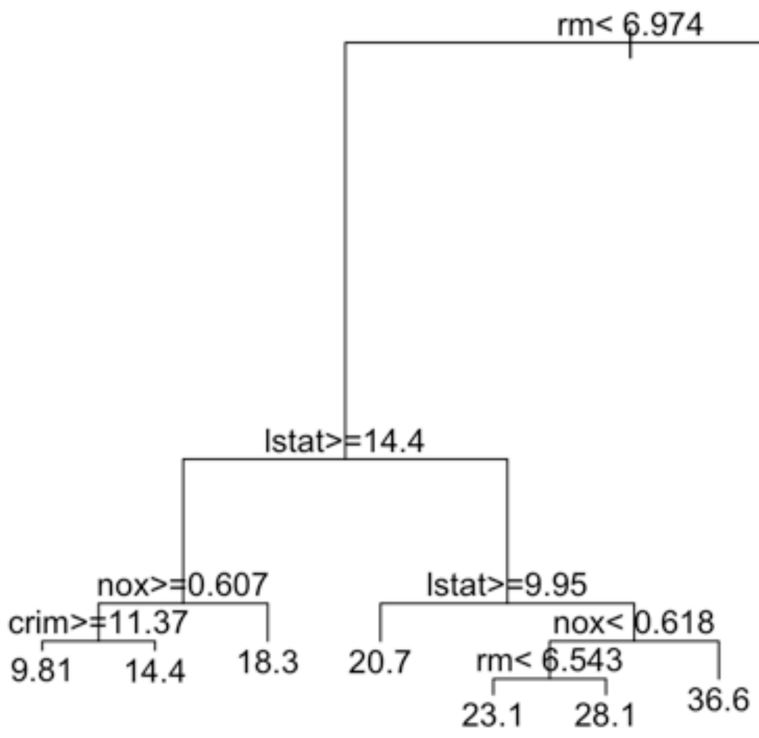


Figure 2

KNN (K- Nearest Neighbors regression)

KNN is a lazy evaluation algorithm that finds ‘K’ most similar observations in terms of mathematical distances in an N-dimensional space where N is the number of the independent variables used to predict the dependent variable. Two of the popular distance measures used are Euclidean, Manhattan distances explained as below.

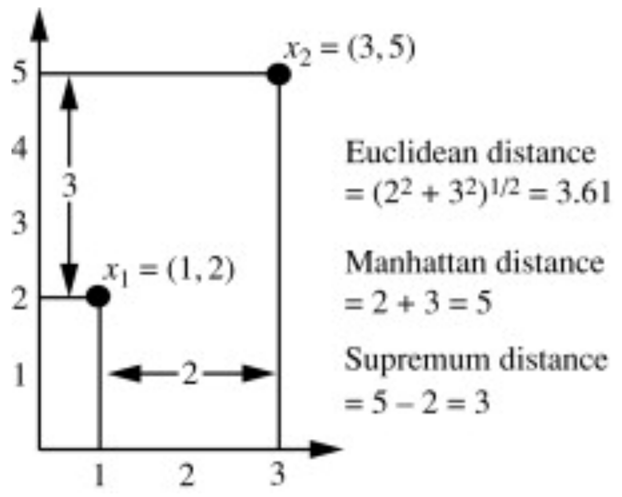


Figure 3

A KNN calculates distances for a cartesian product between all train and test data observations and returns the mathematical average of the dependent variable of the ‘K’ similar observations. Since KNN computes the distance between each possible pair of observations in the training and test data sets, it is again an intuitive algorithm which can be understood in terms of the most possible value arising between similar observations.

Drawbacks of CART & KNN algorithm

While CART and KNN are easily intuitive algorithms, they suffer from some critical computational and predictive limitations on their own. Some of the most common limitations in these algorithms can be summarized in the following points

Drawbacks of CART

(1) CART does not give good generalizations for regressions as the number of possible regression values generated from CART are in the superset of the distinct regression values present in the training dataset
(2) CART is susceptible to overfitting if not pruned carefully to get good accuracy on predictions, while this would improve generalizations and variability in the train predictions, it would lead to overfitting in test predictions.

Drawbacks of KNN

(1) KNN algorithms are computationally very expensive as they calculate an N dimensional distance over a cartesian product between the training and test datasets
(2) Even with space portioning techniques (kd-tree)[4], bisecting clusters etc., the space reduction happens only by the scale of distance, this could result in the loss of important values with respect to target variable. This again becomes more prominent in sparse datasets.
(3) KNN algorithms run time increases at an exponential rate based on the increase in training data observations

Proposed Algorithm

The algorithm proposed in this paper, hence referred to as Partitioned Nearest Neighbors (PNN) throughout the course of the paper is an ensemble of a weak learner (i.e. a shallow decision tree) along with a KNN run on the terminal nodes for the test observations which leads to a reduced search space for each observation (rather than the whole training dataset). Further to improve the computational efficiency we can run KNN in parallel at all the leaf nodes using multi-threading.

A CART algorithm is first run on the training dataset with a reduced depth (less depth than a pruned model or arbitrarily chosen by the Data Scientist) which would generate partition rules based on Information gain with respect to the target variable. A shallow CART will allow to find the most important variables with the highest predictive power and hence

The rules can be applied to get the partition index or the terminal node for the test data observations. For each test data point in a partition index, the algorithm will seek to run to find its 'K' nearest Neighbors algorithmically with Euclidean distance. For the purposes of this paper, the value of K is hard coded to be 3 when samples in terminal node <10, else 5. However, as continuation of work, a suitable K selection algorithm will be developed as well.

The KNN regression returns the values of the arithmetic mean of the values of the nearest Neighbors as the prediction for the observation.

The algorithm is expected to perform better in the following ways on large datasets with high dimensionality of independent variables

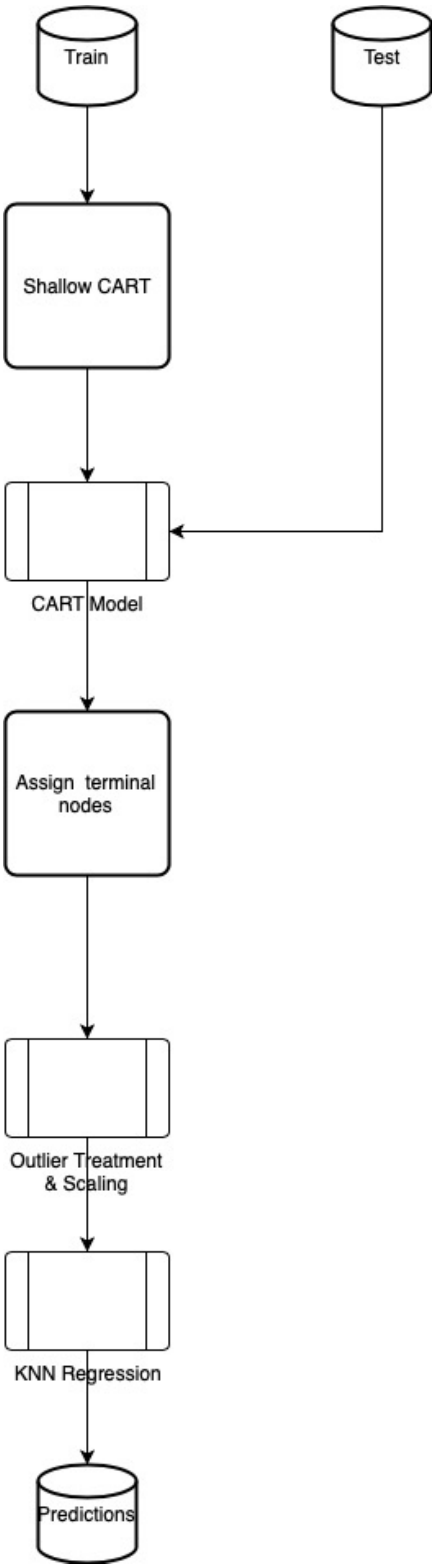
(1) Add more variation & generalization to predictions
(2) Improve performance over CART & less computationally intensive than KNN
(3) Make the algorithm more intuitive

As further development, The K shall be automated to the following scenarios

(1) When samples in terminal node are less than 50
(2) When samples in terminal node are less than 100
(3) When samples in terminal node exceed 100

The algorithm is intended to be explainable as well and the intuitive explanation of the algorithm for a tree with 2 splits is as

"Observation N in the test data set has been predicted to have a value 'Y' based on the following logics. If condition 1 is satisfied and condition 2 is satisfied, then the Y is the average of the 3 most similar observations in the training set that satisfy condition 1 and condition 2"


Train
Test
Shallow CART
CART Model
Assign terminal nodes
Outlier Treatment & Scaling
KNN Regression
Predictions

Performance

PNN was evaluated for comparison against a CART and Random Forest (sklearn) on the house pricing dataset and it shows an improvement in the error metrics of MAPE and RMSE in the following table

Algorithm	RMSE	MAPE
CART	234357.19	32.08%
RF	239401.49	33.15 %
PNN	220270.50	26.61 %

PNN was able to outperform CART & a bagging algorithm based on CART by ~ 5.5 – 6.5% in MAPE and > 6% in RMSE without the tuning of its hyperparameter of K.

This leads to the belief that PNN can perform even better on large datasets with high dimensions even more when the K hyperparameter is automated.

PNN can be further tuned for multi-threading by running all KNN computations for terminal nodes in parallel on multiple cores, this will ensure a faster run time on larger datasets. This shall be delivered once PNN is prepared as a package and available on python.

Code Repository

The jupyter notebook used for the paper can be found in the following repository along with the evaluation dataset

https://github.com/sv9469/decision_tree_with_knn_regression/blob/master/Decision%20trees%20with%20KNN%20regression%20with%20House%20pricing%20dataset.ipynb

References

1 – Bengio, Y., 2009. Learning deep architectures for AI. Foundations and trends® in Machine Learning, 2(1), pp.1-127.

2 – Li, B., Friedman, J., Olshen, R.A. and Stone, C.J., 1984. Classification and Regression Trees (CART). Encyclopedia of Ecology, 40(3), pp.582-588.

3 – Varmuza, K., 1980. K—Nearest Neighbour Classification (KNN-Method). In Pattern Recognition in Chemistry (pp. 62-71). Springer, Berlin, Heidelberg.

4 – Ramasubramanian, V. and Paliwal, K.K., 1989, November. A generalized optimization of the kd tree for fast nearest-neighbour search. In *Fourth IEEE Region 10 International Conference TENCON* (pp. 565-568). IEEE.

Figure 1 – Datacamp credited on https://medium.com/@mathanrajsharma/fundamentals-of-classification-and-regression-trees-cart-e9af0b152503

Figure 2 – http://www.sthda.com/english/articles/35-statistical-machine-learning-essentials/141-cart-model-decision-tree-essentials/

Figure 3 – https://www.sciencedirect.com/topics/computer-science/manhattan-distance