Searching for Similarity

Using kNN and Decision Trees for Classification and Regression

kNN is a conceptually simple algorithm which is surprisingly useful and effective for classification. Fundamentally, kNN relies on the intuition that objects of a similar class should share similar features [1]. To prevent overfitting and to adjust the specificity with which similarity is determined, the 'k' hyperparameter of kNN (which stands for k-Nearest Neighbors) may be increased, in order to consider more neighbors and decrease variance/overfitting, or decreased, in order to consider less neighbors and increase bias.

Although kNN is often used for classification, the algorithm can still used for regression analysis where the target variable is continuous in nature. The idea of utilizing kNN for linear regression is to approximate target values with a best predicted value for any given input [1]. A straightforward approach would be to compute the average of ‘k’ number of data points for all possible input values that will yield a set of predictions, which can then be all connected to reveal some regression line.

The decision tree algorithm classifies objects by building a tree of questions about the objects' features that can be answered to determine which class the object belongs to [2]. Usually these questions are numerical comparisons (for example, "petal width <= 0.5" when classifying flowers by type). Decision trees present several hyperparameters which may be tuned to optimize performance; some common examples are maximum depth (how many levels the tree may contain), the minimum number of examples of a feature required to cause a new node/split in the tree, and a threshold by which a split must increase performance [3].

When applying the decision tree algorithm for regression, an impurity metric that is suitable for continuous variables is necessary, which can be defined using the weighted mean square of the decision tree’s children nodes [2]. Similarly to how kNN can be used for regression, the decision tree algorithm can be applied for all possible input values to reveal a some regression trend line, which may be smoothed out using dimensionality reduction techniques to reduce the issue of overfitting the regression trend line.

kMeans Clustering, Hierarchical Clustering, and Model-Based Clustering

The k-Means algorithm is similar to the kNN algorithm but differs in that it performs unsupervised clustering rather than supervised classification. Furthermore, the 'k' hyperparameter no longer refers to the number of neighbors; rather it refers to the number of classes which the model is to consider. The k-Means are the centroids which each data point is repeatedly assigned to and then averaged out to compute the next centroid. The algorithm stops learning when the number of centroids no longer changes.

Hierarchical clustering is similar to the decision tree algorithm in that it forms a tree of the data based on features which can then be walked in order to find out what cluster an object belongs to. The diagrams generated by a hierarchical clustering model are called

dendograms [5], and when the algorithm functions successfully, the features used to identify each branch of the dendogram should make logical sense to the model developer based on their domain knowledge of the dataset.

Model-based clustering is also known as a "mixture model" since it attempts to fit the data with multiple models and chooses the optimal one "according to BIC for EM initialized by hierarchical clustering for parameterized Gaussian mixture models [6]." This makes model-based clustering a more automatic process in terms of optimization, as it performs a significant amount of experimentation on behalf of the user and selects an optimal model as opposed to manually testing each different type and analyzing the results yourself. Of course, model-based clustering does not necessarily outperform manually optimized models in any situation.

Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA)

These two algorithms are the main algorithms of dimensionality reduction, which is the process of reducing the number of random variables to consider for possible predictors by obtaining a set of principle variables [7]. PCA is helpful in situations such as if someone is ensure that their variables are independent of each other or are unsure what variables to completely remove from consideration.

LDA is a type of linear combination or a mathematical process of using various data items and applying a function to them to separately analyze multiple classes of objects or items [7]. This algorithm is commonly used for supervised classification problems, mainly to project features in a higher dimension space into a lower dimension space [8]. Using only a single feature to classify classes would likely result in some overlap, so LDA works by continuing to increase the number of features to consider for a proper and efficient classification.

References

[1] https://towardsdatascience.com/the-basics-knn-for-classification-and-regression-c1e8a6c955
[2] https://towardsdatascience.com/https-medium-com-lorrli-classification-and-regression-analysis-with-decision-trees-c43cdbc58054
[3] https://towardsdatascience.com/decision-tree-hyperparameter-tuning-in-r-using-mlr-3248bfd2d88c
[4] https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
[5] https://towardsdatascience.com/hierarchical-clustering-explained-e59b13846da8
[6] https://www.statmethods.net/advstats/cluster.html
[7] https://medium.com/machine-learning-researcher/dimensionality-reduction-pca-and-lda-6be91734f567
[8] https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/