How kNN and decision trees work for classification and regression

For classification:

In conclusion, DT is better than kNN for classification. The accurancy of DT is .80 and the accurancy of kNN is 0.785.

The KNN algorithm has a relatively high degree of adaptation to numerical data, and has less preprocessing. Generally, a single type of data can be normalized, and the formula for selecting distance can also be carried out based on the actual situation. One advantage of the KNN algorithm is that it is not sensitive to outliers, but during preprocessing, if the extreme data can be removed and then normalized, the classification effect will be better.

The decision tree algorithm has higher requirements for data preprocessing and requires pre-classification. The classification process will be better if it is oriented to the problem itself. If it is used to make the actual algorithm and put it into use, it will be better for a specific user to perform a scaling effect by the user. However, when there is too much data, more consideration should be given to the pruning of the decision tree, but it will inevitably reduce the accuracy.

For regression:

Decision trees are supervised and non-parametric, and have a structure similar to a flowchart. These trees are easily interpretable because you can see each “decision” the machine made and why it got what it got. kNN, or K-Nearest Neighbor, is also non-parametric and supervised. In kNN, the k is a constant defined by the user.

Both methods being non-parametric mean the distribution of data cannot be defined in a few parameters. (No assumptions) Decision trees are better are finding non-linear relationships, making them worse than say linear regression. In addition, bigger trees mean more overfitting, which means we must prune it. kNN on the other hand becomes slower with larger datasets, and becomes sensible to outliers.

kNN works with regression by approximating the relationship between independent variables and the continuous outcome via the average of the observations. Decision trees will regress the data by asserting true or false to specific questions, causing it to branch off – like a tree- into a true and false.

How the 3 clustering methods of step 3 work?

Clustering is a category of unsupervised machine learning that aims to group data together through their similarities with no ground truth, though the data set used for this assignment did have labels for some surface-level comparisons. The first method used was the k-means algorithm, which chooses k random centroids and assigns each data point to a random group. K in this context is the number of desired clusters, which can be determined by iterating through the algorithm several times to minimize the sum of squared distance from each centroid, or it can be assigned arbitrarily based on domain knowledge as was done here. The model then iteratively assigns each data point to the closest centroid and recalculates the location of each centroid until clusters have formed. This is an instance of an expectation-maximization algorithm, where parameters are updated to match a hypothesis and a hypothesis is updated to match parameters until convergence.

Hierarchical clustering works a bit differently. Initially, each observation is placed in its own cluster, and the algorithm iteratively finds the distance between every cluster and combines the two nearest clusters. For clusters with multiple observations, distance can be single-linkage, the minimum distance between data points, complete-linkage, the maximum distance between data points, or average-linkage, which is fittingly the average distance between all data points in each cluster. The last option is the least susceptible to outliers, so it was the one chosen for this assignment. Large data sets bog down the algorithm's ability to produce reasonable links and a meaningful dendrogram, so the output here was all over the place.

Model-based cluster analysis is a method used to find the optimum hyperparameters for clustering. Here, various potential models were scored by their Bayes Information Criterion, or BIC, to select a good hierarchical clustering for parameterized Gaussian mixture models. Instead of strictly defining an observation to a certain cluster, Gaussian mixture models define probability regions for each cluster with potential for overlap. The specifics of the optimal model match the model with the highest BIC, and the analysis for this data set recommended 9 regions with ellipsoids of varying volume, shape, and orientation. Gaussian mixture models are great if the underlying data follows a Gaussian distribution, otherwise it will overfit due to its high bias nature. None of the models provided a particularly good visualization of the clusters or regions since the input parameters were multidimensional, and accuracy could not be tested because the models are all unsupervised.

How PCA and LDA work, and why they might be useful techiques for machine learning

PCA (Principal Component Analysis) is a type of unsupervised learning, which removes correlation between independent variable and looks for hidden latent variable, and reduces noise.

LDA (Linear Discriminant Analysis) is a supervised algorithm that performs classification and dimensionality reduction at the same time. The key to LDA is that when classification is performed, the variance within the class is minimized, but the variance between classes is maximized.

The difference in these two comes from PCA finding a new axis that allows the data to be widely distributed, while the LDA finds a new axis that best classifies data.

PCA performs better when the difference in class comes from the difference in variance, and LDA performs better when the difference in class of data comes from the difference in mean. Also, When data is represented in a 3-dimensional plot, LDA shows a better performance than PCA as well in classification.