Final Project

Justin Fellers

3/26/2021

Introduction

This project explored how well the Decision Tree Algorithm can predict the best heuristic for a Capacitated Vehicle Routing Problem (CVRP) instance and supported my ongoing research in the Mechanical, Systems, and Industrial Engineering (MCISE) Department. The CVRP is defined as a problem where a fleet of homogenous vehicles is directed to service a set of cities from a single depot. The depot has zero demand and unlimited capacity, and each city has a positive and non-zero demand. The cities must all be serviced, and vehicle capacity cannot be violated. All problems exist in the 2-D Euclidean plane, and the objective is to minimize the total fleet distance travelled. In application, this problem is used to formulate routes for delivery vehicles across many industries.

Despite its simple definition, the CVRP is NP-Hard, meaning no polynomial time algorithm exists for exactly solving all problems to optimality. Global research has produced heuristic algorithms that provide good solutions within a reasonable run time, albetit a new algorithm's improvements can often be isolated to certain instance types (Wolpert and Macready 1997; Kothoff 2014; Steinhaus 2015).

The Decision Tree (DT) can be used for CVRP algorithm prediction by formulating the problem as a machine learning classification task. CVRP instances must be vectorized into relevant features and labeled by the heuristic which produced the best solution (i.e. minimum fleet distance travelled). This study used a set of 23 CVRP features found in the literature, which largely characterize instances by their size, spatial attributes, and vehicle requirements. Examples include the number of cities, minimum fleet size, standard deviation of demands, and average route length. For the purposes of this report, the features are listed by number with a logical short title. This study considered four solution heuristics to compete for the instance label:

1. The Clarke & Wright Savings Algorithm (CW),
2. The Sweep Algorithm (SP),
3. An accelerated Genetic Algorithm (GA),
4. The Self-Organizing Map (SOM).

I generated a novel library of 4,897 CVRP instances through earlier work in MCISE. I created the dataset used in this project by first extracting the 23 instance features and then mapping labels by solving all instances with all algorithms. The dataset is in csv format and is of dimension 4,897 rows by 32 columns. Each row constitutes a CVRP instance represented by its features and labeled by its best algorithm, along with other administrative information such as which city, depot, and demand functions were used in the instance generation algorithm.

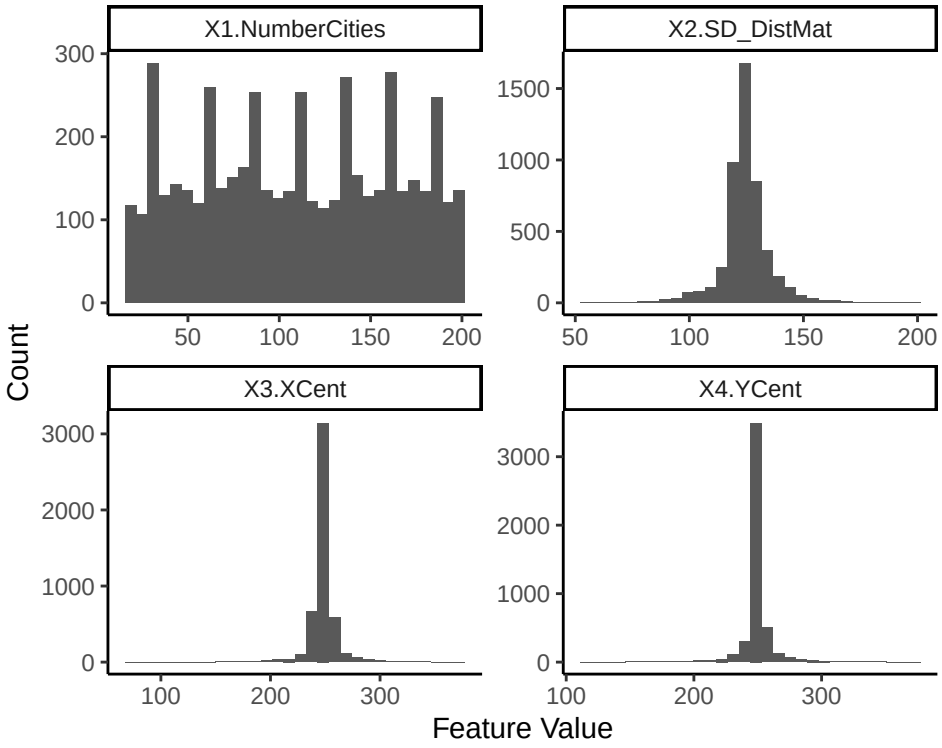
This project focused on exploring the data set and performing the classification task with the DT algorithm. The goals of this project are as follows:

1. Examine the feature distributions to consider the diversity of their values.
2. Present the algorithm label counts to gain summary insight to the algorithm performances.

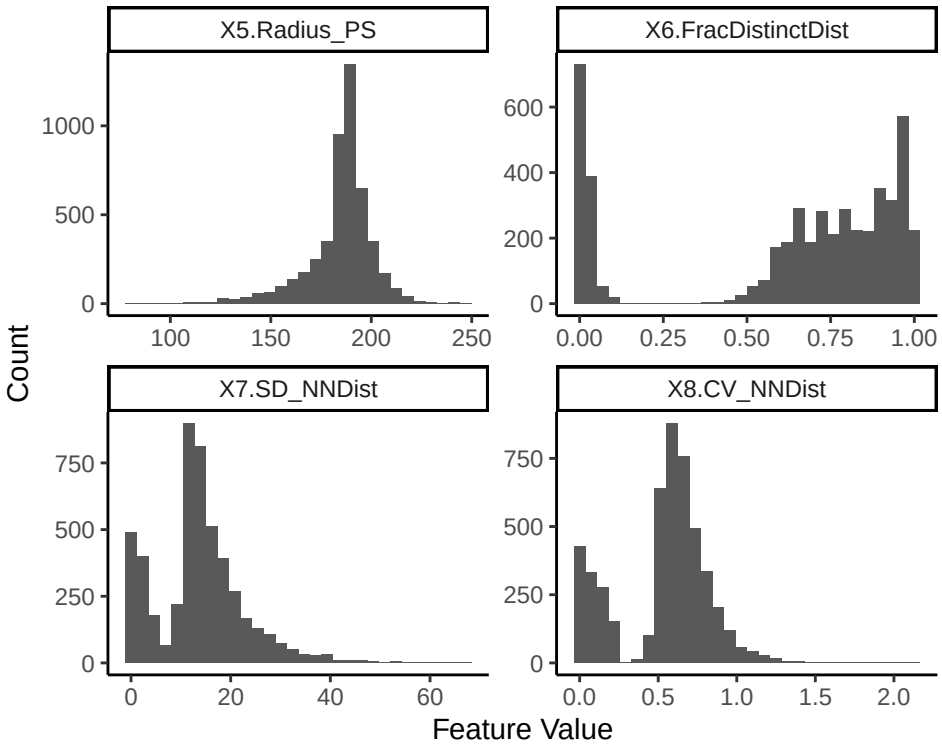
3. Explore feature-to-label relationships for insight to the range of feature values where algorithms perform best.
4. Report the prediction accuracy of the optimal DT model and compare it to a metric found in the literature.

Feature Distributions

This section presents the distributions of all 23 features, which are wrapped into figures each containing four features. Examining the diversity of feature values is a subjective, yet intuitive, way to estimate how valuable a feature can be to the DT algorithm. By inspection, a feature's diversity may be evaluated by considering the range of its values along the X variable. This stems from the concept that features with the most variation contain the most information about the dataset. The features are short titled and employ free axes to avoid distortion.



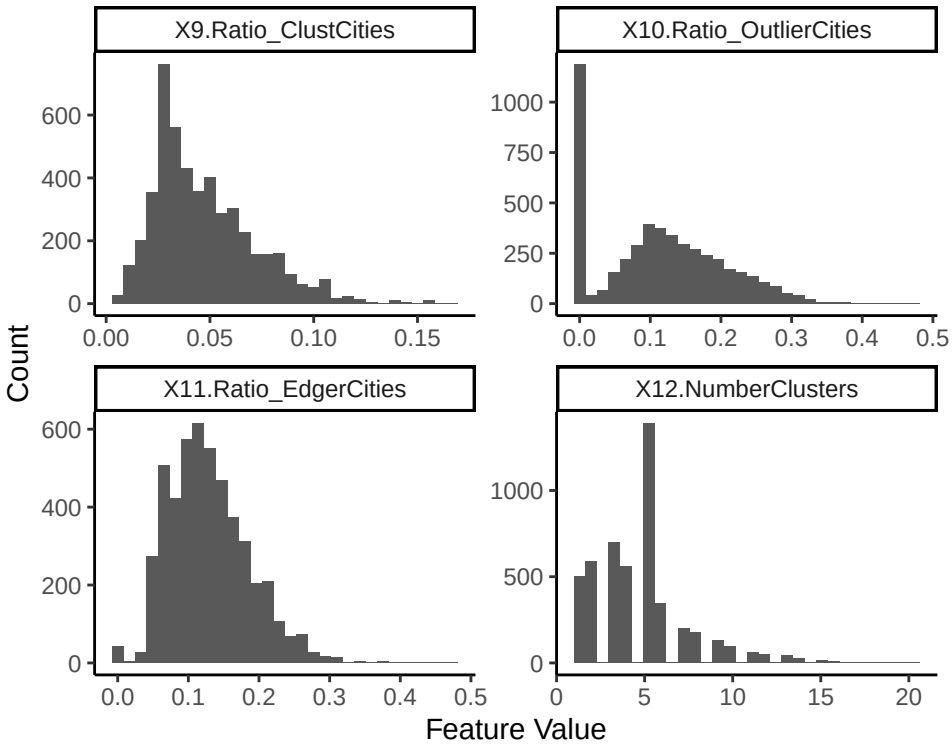
The feature 1.NumberCities shows the most diversity in its values out of this set. The other three features, especially 3.XCent and 4.YCent are much more limited.



The second set of plots exhibit greater diversity across the features. Of note is feature 6.FracDistinctDist, which has an apparently abnormal amount of lower end values. The values of this feature are explained by the methods used to position cities in the instance generation algorithm. One of these methods is designed to position cities at equidistant, or nearly equidistant, locations from one another (the module is referred to as 'Equidistant'). This minimizes the fraction of distinct values in the instance's distance matrix, which is represented by feature 6. The lower end values of this feature likely come from the 'Equidistant' positioning module, since the feature contains 1189 lower end values and the 'Equidistant' method is also used exactly 1189 times.

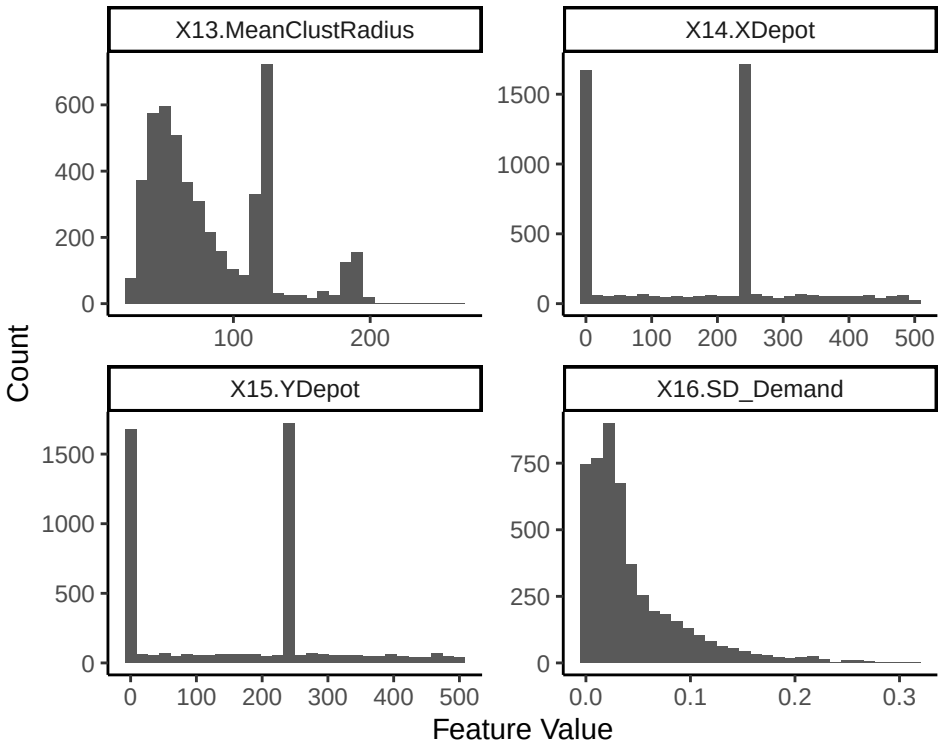
The features 7.SD_NNDist and 8.CV_NNDist also have an unusual amount of lower end values. These features, which represent the standard deviation of the instance's nearest neighbor distance and the coefficient of variation of the instance's nearest neighbor distance are going to be related to the 6.FracDistinctDist. This can be shown through a Pearson correlation matrix of these features:

```
##                     X6.FracDistinctDist X7.SD_NNDist X8.CV_NNDist
## X6.FracDistinctDist           1.0000000    0.8226188    0.7850338
## X7.SD_NNDist                  0.8226188    1.0000000    0.7394171
## X8.CV_NNDist                  0.7850338    0.7394171    1.0000000
```

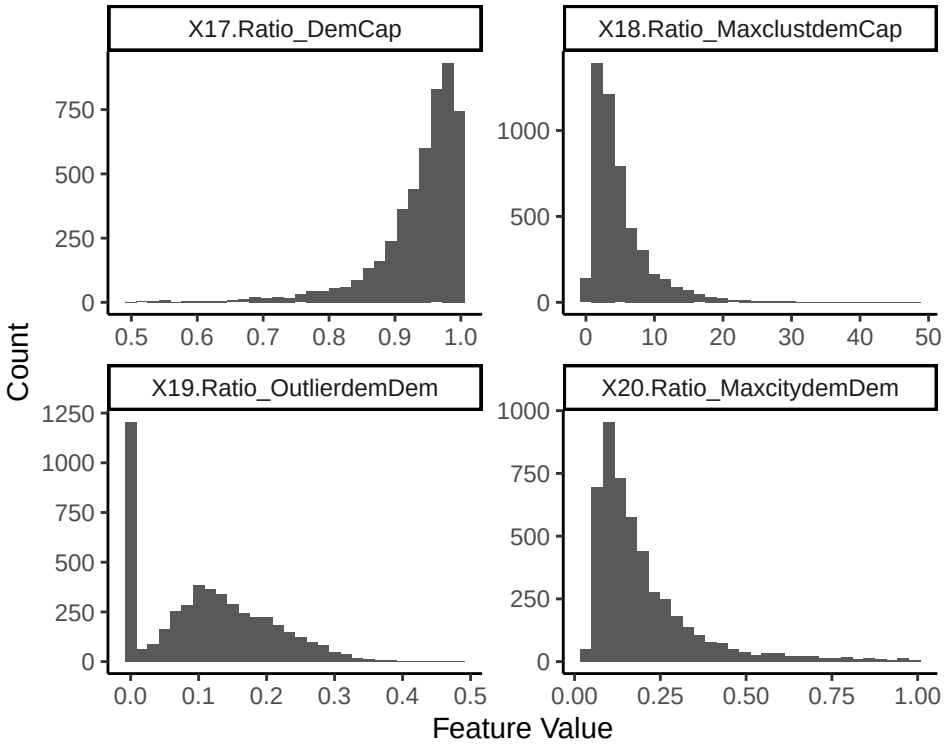


The next batch of plots further support diversity of feature values. These features all derive from an automated city clustering technique using the DBSCAN algorithm. Of interest is the apparently high value of 5 clusters in 12.NumberClusters. Since the number of clusters in an instance is principally related to the spatial location of the cities, examining the city positioning methods that result in exactly 5 clusters can be insightful.

Indeed, further analysis revealed that the 'Equidistant' city positioning module made up for 68.4 % of all instances containing 5 clusters. The generation algorithm contained four different methods to position cities, so this is a notable find worthy of futher investigation into the clustering methods of DBSCAN.



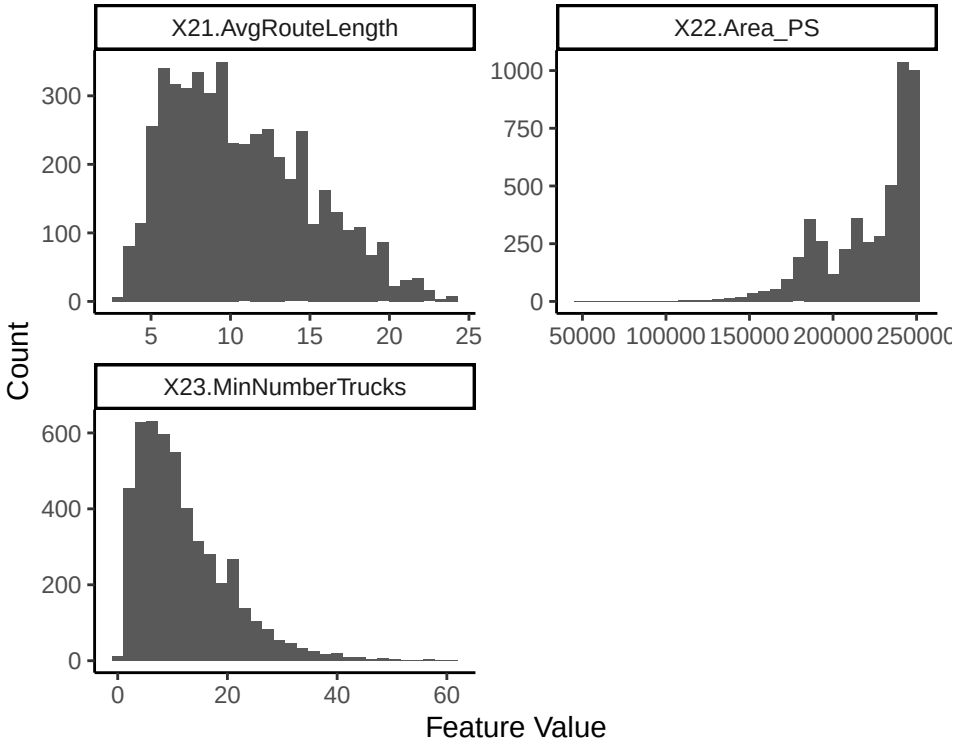
The subsequent set of plots displayed in figure 16 continue to reflect diversity in the problem space. The distributions of 14.XDepot and 15.YDepot, which correspond to the coordinate (X,Y) of the depot assignment, reveal the positioning methods for the depot. It is readily apparent the depot is either positioned at the origin (0,0), the center (250,250), or some random point.



Of interest in this set of plots is 19.Ratio OutlierdemDem, which displays a high count where the ratio of outlier city demand to total city demand is equal to 0. This is only feasible if the outlier demand is 0; therefore, it can only be possible if the instance has no outlier cities since all cities have a positive non zero demand.

For an instance to have no outliers all of its cities must be classified within a cluster. As discussed earlier, the creation method that ultimately leads to clustering results is the city positioning module. Therefore, by examining the city positioning methods which result in a value of 0 in 19.Ratio_OutlierdemDem we may see if there is a pattern between how cities are positioned and if they allow for outliers in the clustering methodology.

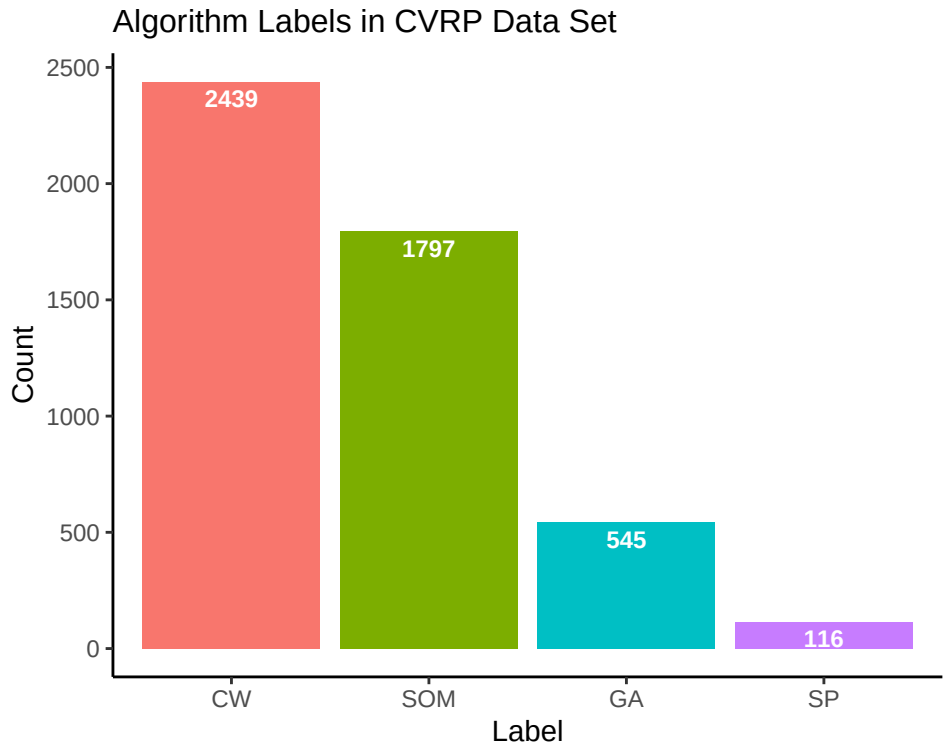
This analysis revealed an interesting pattern as the 'Equidistant' city module accounted for 97.8 % of all instances containing with no outlier demand. Upon reviewing the literature further, this behavior is consistent with the DBSCAN documentation (Pedregosa et al. 2011). DBSCAN is a hierarchical clustering algorithm that produces no outliers on data spread uniformly throughout a problem area.



This last batch of plots also showcase diversity in these features in the data set, particularly 21.AvgRoute-Length and 23.MinNumberTrucks. Though not exact, the 21.AvgRouteLength feature resembles a T[3,6,25] distribution, which was the sampling distribution used in the instance generation algorithm.

Label Distribution

This section explores the distribution of labels in the dataset and considers how it may influence the selection model.



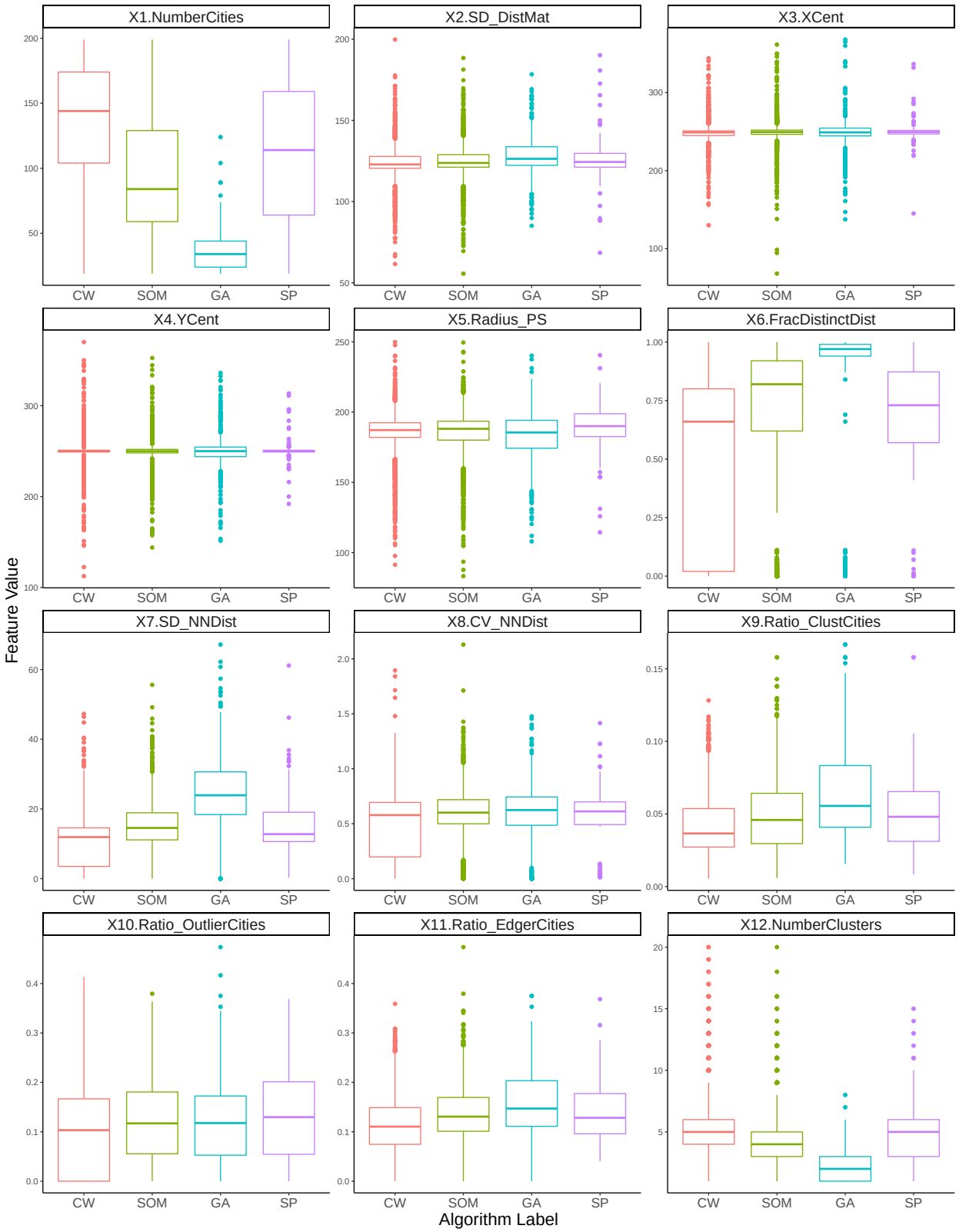
Since the CW and SOM earned significantly more labels than the GA and SP, the DT algorithm will be provided a greater number of training examples for these two heuristics. This bias may allow for simpler prediction of their labels in the test data, though is not guaranteed (Domingos 2012).

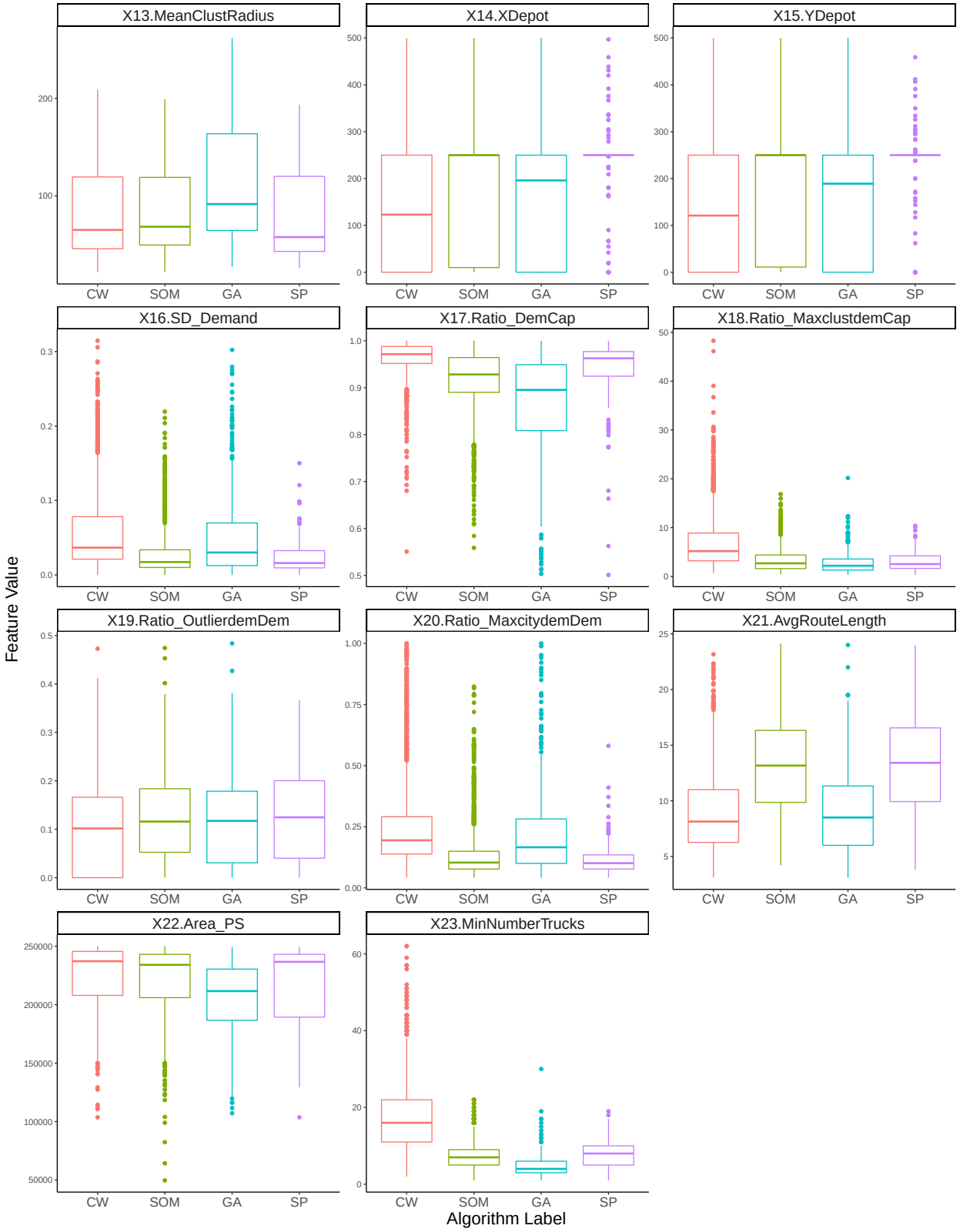
For evaluating the effectiveness of an algorithm selection model, one may compare it to a simulated model which universally predicts the most frequently occurring label in the dataset to all test instances. This is called the Single-Best-Solver model, and, in this study, would yield an accuracy of 49.8% by using the CW algorithm. This comparison metric is discussed later to evaluate the model created by the DT algorithm.

Feature-to-Label Analysis

This section explores the distribution of features by their algorithm label. Presented as Box-and-Whisker plots, these visualizations quickly indicate the range of feature values where algorithms performed best. These plots are relevant for intuition on the data set provided to the DT algorithm, which learns from previous feature-to-label relationships to predict the labels of new instances from its feature values.

The plots are first presented for all 23 features and then commented upon after observing any patterns or indications of algorithm performance within the data.


X1.NumberCities
X2.SD_DistMat
X3.XCent
X4.YCent
X5.Radius_PS
X6.FracDistinctDist
X7.SD_NNDist
X8.CV_NNDist
X9.Ratio_ClustCities
X10.Ratio_OutlierCities
X11.Ratio_EdgerCities
X12.NumberClusters
Feature Value
Algorithm Label
CW
SOM
GA
SP



The feature 1.NumberCities presents an interesting result. The median value for the GA does not intersect with the box of any other algorithm, indicating it may be grouped differently from the others. By observation, it appears the GA performed best on instances containing lower city counts.

In both 6.FracDistinctDist and 7.SD NNDist the GA appears in its own group. As presented earlier, these two features have a strong correlation. When an instance's fraction of distinct distances in its distance matrix are minimal, and so is the standard deviation of its nearest neighbor distance. We also know from earlier analyses that low values in these features are attributed to cities being positioned under the 'Equidistant' city module. Consequently, it is likely the GA performed poorly solving instances created under the 'Equidistant' module.

The features 21.AvgRouteLength and 23.MinNumberTrucks also reveal noteworthy algorithm behavior. Two groups possibly exist in 21.AvgRouteLength, one containing CW and GA and another with SOM and SP. This suggests one group performed better with more cities per route and another with less. Since the GA and SOM are both metaheuristics, perhaps increasing their search space in future studies would first lead to domination of its group. In 23.MinNumberTrucks notice the CW in a clear group of its own, outperforming all other algorithms when the instance requires a higher number of trucks. Indeed, with the exception of one GA outlier, the CW performed best on all instances requiring more than approximately 25 routes.

Decision Tree Model

Using the python3 script **DSP539_DT**, I was able to explore different parameter settings for the DT algorithm using 10-fold cross validation. Once the best parameters were found, I trained an optimal model using all training data. The classification accuracy for the DT model is 76.4%.

The confusion matrix for a classification model presents the probability of correct and incorrect predictions for each label in the test set, and is presented below.

Figure 1: Confusion Matrix for label predictions by DT model.

The SP algorithm is never predicted as the best performing algorithm, which could be expected due to its infrequency of occurrence. Note that 63% of the true SP labels were predicted as SOM.Researching their performance similarities may be insightful in the future. The CW algorithm is correctly predicted 87% of the time, though is misclassified as GA or SOM with 2% and 11% probability, respectively.

Overall, the DT accuracy of 76.4% well outperforms the simulated SBS model that would produce 49.8% with universal application of the CW algorithm. This positively suggests the model learned from the data before making a prediction, which also verifies the features chosen to represent the instances.

Bibliography

Domingos, Pedro. 2012. "A Few Useful Things to Know about Machine Learning." *Communications of the*

Association for Computing Machinery 55: 78–87.

Kothoff, Lars. 2014. "Algorithm Selection for Combinatorial Search Problems: A Survey." *Artificial Intelligence Magazine* 35: 48–60.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12: 2825–30.

Steinhaus, Meghan K. 2015. "The Application of the Self Organizing Map to the Vehicle Routing Problem." PhD thesis, University of Rhode Island.

Wolpert, David H., and William G. Macready. 1997. "No Free Lunch Theorems for Optimization." *IEEE Transactions on Evolutionary Computatation* 1: 67–82.