Jeremy Kang
Jeremy.kang@gatech.edu
CS 7641: Machine Learning – Assignment 1

Datasets

1. The first dataset was obtained from the UCI Machine Learning Repository. This dataset was initially created to demonstrate a decision-making system to evaluate car acceptability (unacc, acc, good, v-good) according to six features: overall price (v-high, high, med, low), price of maintenance (v-high, high, med, low), number of doors (2, 3, 4, 5-more), person capacity (2, 4, more), size of luggage boot (small, med, big), and safety (low, med, high). There were 1728 instances with a total of 7 attributes (i.e., including the class attribute.) All of the variables were previously categorically encoded, as well as all of the instances did not have any missing values. Thus, the attributes did not have to be preprocessed – aside from formatting the dataset to be WEKA compatible (e.g., explicitly assigning a class attribute).

2. The second dataset was also retrieved from the UCI Machine Learning Repository. This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey where samples of married women that were not (or did not have confirmation of being) pregnant at the time of data collection. The objective of the study was to predict contraceptive method choice (no use, long-term methods, short-term methods) based on 9 other attributes:

- wife's age ([numerical]),
- wife's education (1 = low, 2, = medium-low, 3 = medium-high, 4 = high),
- husband's education (1 = low, 2, = medium-low, 3 = medium-high, 4 = high),
- number of children ([numerical]),
- wife's religion (0 = Non-Islam, 1 = Islam),
- wife's employment status (0 = yes, 1 = no),
- husband's occupation (1, 2, 3, 4),
- standard-of-living index (1 = low, 2, = medium-low, 3 = medium-high, 4 = high),
- media exposure (0 = good, 1 = not-good)

There were 1473 instances across a total of 10 attributes. This dataset required more pre-processing than the first as some of the attributes were not properly labeled and, minimally, the class attribute needed to be discretized (see conclusion for why the rest of the numerical values were not discretized) for WEKA to properly use all of the algorithms to analyze the data.

These datasets were specifically chosen because of their relatively clean formatting, categorical-orientation (note, however, the second dataset has two continuous attributes), as well as the fact that both are not missing any values. However, it is worthwhile pointing out that, compared to some of the other available datasets, there were a low number of instances, which may lead to some bias, which subsequently may lead to overfitting. In fact, it's important to acknowledge that the second data set was collected from a larger sample.

Software and Approach

WEKA's GUI Explorer and Experimenter were used to run all five algorithms on both datasets. The datasets were split into 80% training and 20% testing. In order to produce the learning curves, 10 iterations, 10-fold cross-validations, and a remove-percentage filter to measure performance when certain percentage of the instances are removed from both the testing and training set. Performance was measured in terms of error rate as a function of percent of data removed.

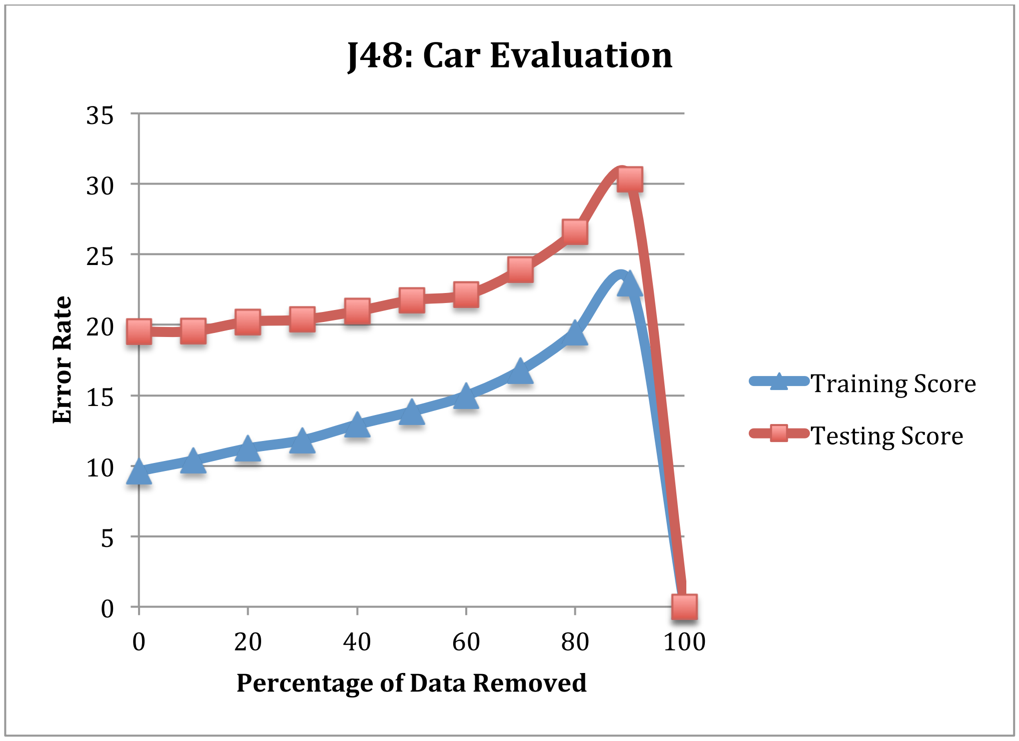
Classification Problems

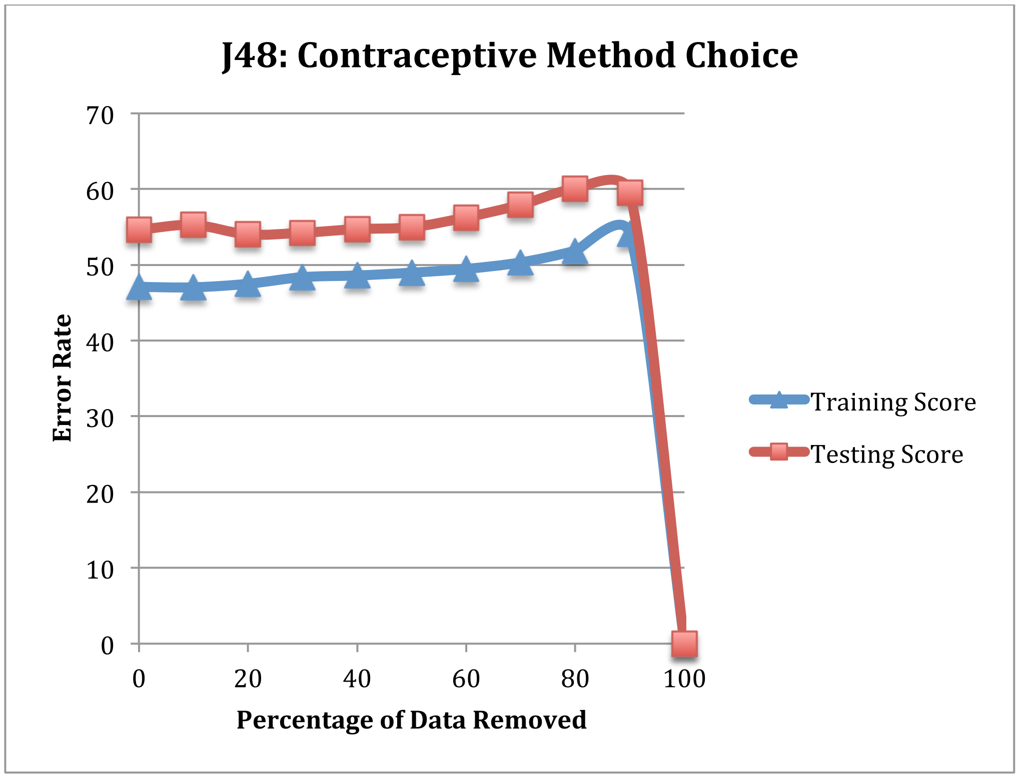
1. Classifying car acceptability (i.e., unacceptable, acceptable, good, or very good) based on the car's overall price, price of maintenance, number of doors, person capacity, size of luggage boot, and safety.

2. Classify contraceptive method choice (no use, long-term method, short-term method) based on wife's age, wife's education, husband's education, number of children, wife's religion, wife's employment status, husband's occupation, standard-of-living index, and media exposure.

Decision Tree (J48)

The J48 algorithm was used for the decision tree. There were many advantages of using J48, such as pruning the data as default, accounting for continuous attribute value ranges, and handling data with missing attribute values (not necessary with these two datasets.



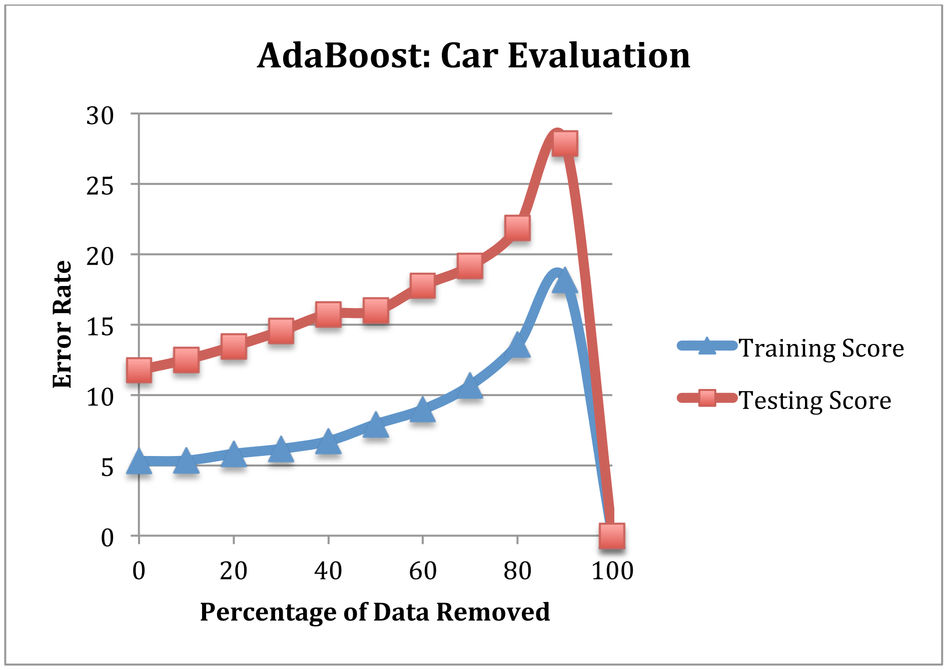


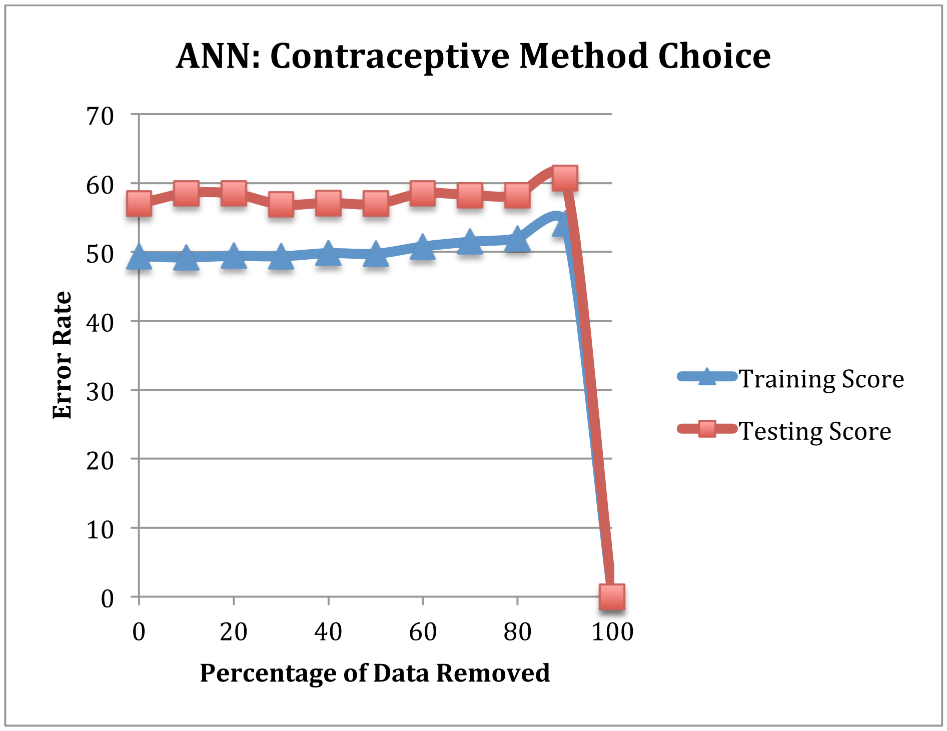
The figures above shows the error rate plotted as a function of percentage of data removed from both the training and testing sets (i.e., the x-axis represents what percent of the data is removed and given to the sets). From first hand, it appears that there is a large amount of difference between the training and testing set for the Car Evaluation data. However, there is only about a 7.39% error rate difference between the two sets (i.e., the training set performs about 7.39% better than the testing). For the Contraceptive data set, the difference is about the same at 6.21% better performance in the training set. However, the average performance is at 44.85% and 51.06% for the training and testing set, respectively. Compared to all of the other algorithms, the Contraceptive J48 performed the best with an average training score of 46.61%. Note that the Car Evaluation J48 vastly outperforms the Contraceptive J48 with an average error rate difference of 31.75% on the training set and 30.57% on the testing set. What's even more peculiar is that the Contraceptive dataset stays around 50% error, regardless of how the data is split. Meanwhile, the J48 performs most optimally with only about 40% of the data from both the testing and training sets.

It is well known that a separate cross validation set is helpful for decision trees. Unfortunately, I ran out of time to go through this procedure, but it would have been interesting to see what differences may have been shown if a separate set was held for the J48 algorithms. Regardless, a 10-fold cross validation within the test and training set may have been enough to show an accurate representation of error rate of both data sets.

Boosting (AdaBoost:J48)

AdaBoost was used with the J48 decision tree algorithm as the base learner. Boosting is known for helping learning algorithms by reconsidering more difficult examples and accounting for them in later nodes of the tree.





The figures above represents AdaBoost applied to the J48 decision tree. The Contraceptive actually performed slightly worse than using just the J48 learner. In addition, the learner stayed around 50-60% error rate for both sets. Taking a look at the classifier statistics sheds some light on why the performance is low.

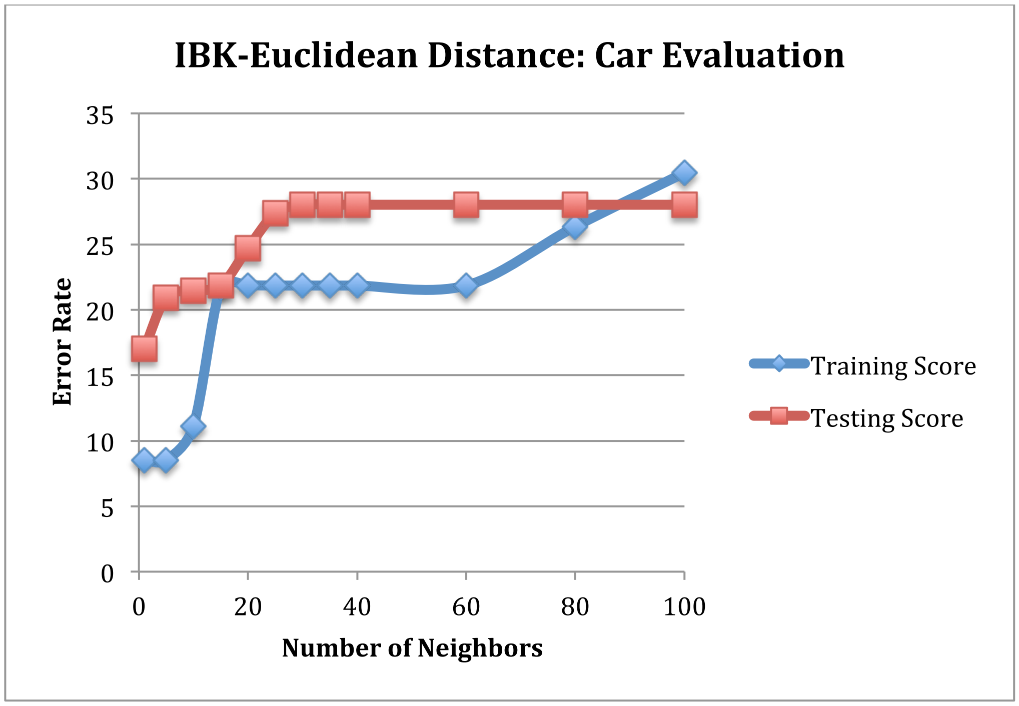
```
Number of Leaves  : 201
Size of the tree : 337
Weight: 0.38
Total Number of Instances         1178
Number of performed Iterations: 10
Time taken to build model: 0.5 seconds

=== Evaluation on test split ===
Correctly Classified Instances        502              42.6146 %
Incorrectly Classified Instances      676              57.3854 %
Mean absolute error                 0.3775

=== Detailed Accuracy By Class ===
TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
0.52      0.347     0.525     0.52      0.523      0.645    No-use
0.347     0.234     0.298     0.347     0.321     0.64     Long-term
0.363     0.298     0.399     0.363     0.38      0.587    Short-term
weightav  0.426     0.304     0.43      0.426     0.428     0.623

=== Confusion Matrix ===
a      b    c   <-- classified as
260  96  144 |   a = No-use
88    91   83  |   b = Long-term
147 118 151 |   c = Short-term
```

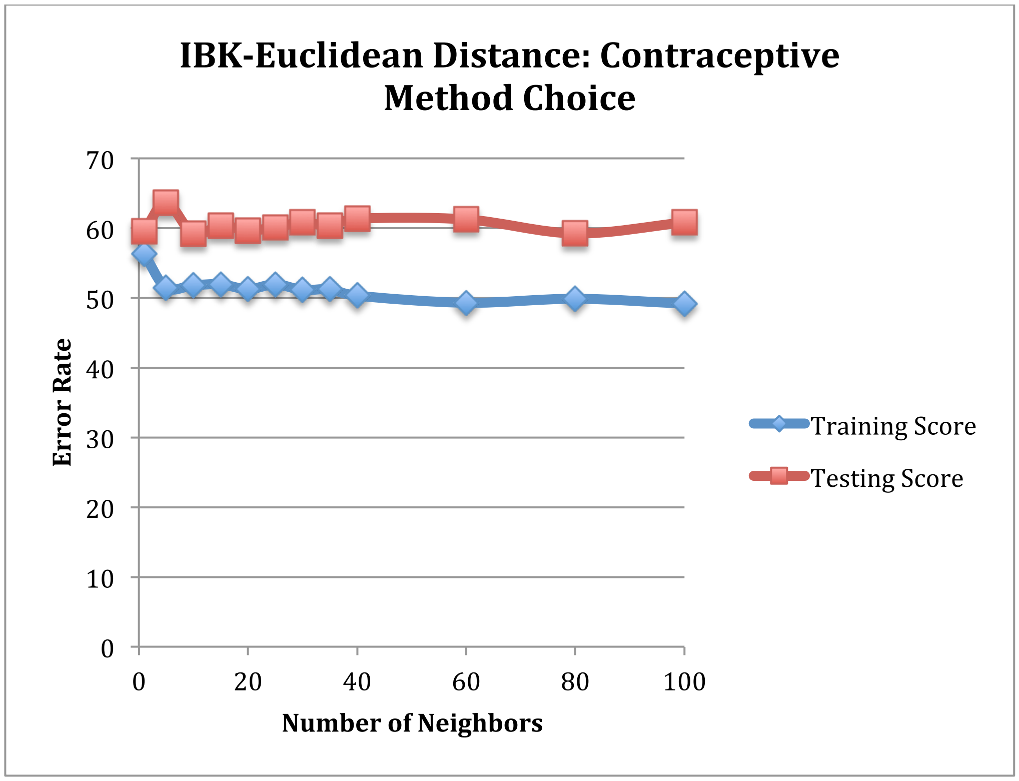
The confusion matrix suggests that the AdaBoosted J48 is significantly misclassifying the class attribute. There are a number of factors that could possible explain this and it is discussed further in the conclusion.

In contrast, the Car Evaluation AdaBoosted J48 performed with an average 8.06% error rate for the training set and 15.51% for the testing set. This boosted decision tree performed, on average, 5.03% better on the training set and 4.97% better on the test set when compared to just the J48 classifier.

K-Nearest Neighbors (IBK-Euclidean Distance)

K-Nearest Neighbors seek to find the most optimal number of neighbors that has the lowest error rate. Instead of the percentage of data removed, K-NN uses the Euclidean distance metric to pair every new test sample to previous observed samples. Weka's Instance-Based Learner (IBK) was used to run the algorithms.





The figures above show the most interesting results compared to the rest of the algorithms. Although performance of all testing sets are expected to be lower than the training set because of how the data was split – taking a look at the Contraceptive data – at 5 kfolds, there appears to be a spike in error, whereas for the training set, the error rate decreases at the same fold. Still, however, error rate hovers around 50% for the training set and 60% for the testing set, performing the worst out of all the algorithms.

What is also interesting is that for the Car Evaluation data, both the training and testing data suggest about 15 neighbors will provide the optimal accuracy at about 21%. Although having the neighbors between 1 and 5 has an error rate at 8.53% and at 10 neighbors training error is 11.1%, the highest accuracy is probably more along the lines of 19% (i.e. the average error rate) because of how the points average around at 20 neighbors for *both* the training and test sets. This is explained by its locality preference bias in which near points are more similar than further points.

Artificial Neural Network (Multilayer Perceptron)

Multilayer Perceptron was chosen for the ANN algorithm to be applied to the datasets.

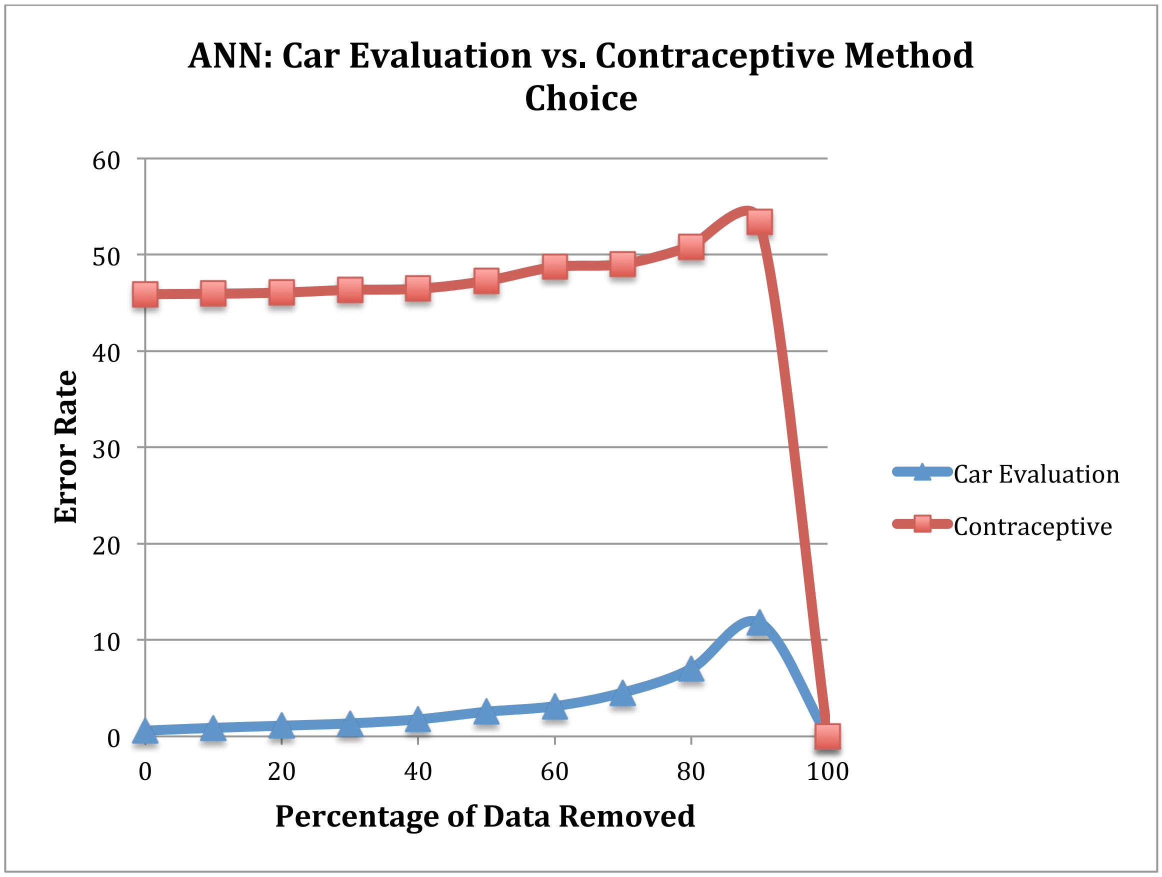
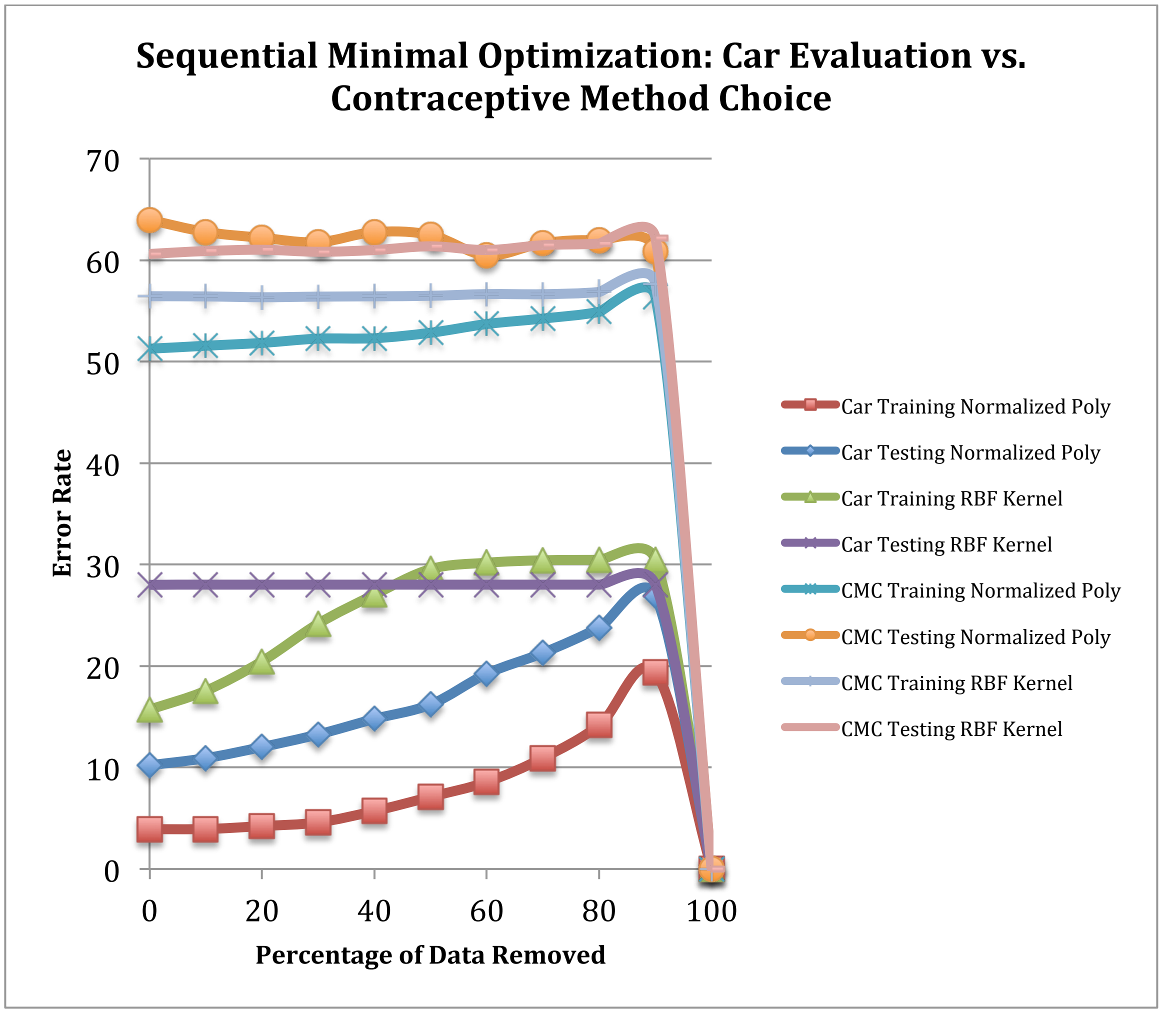


Figure 4: Artificial Neural Network (Multilayer Perceptron) Learning Curve of Both Datasets.

Out of all the other algorithms, ANN performed the best with the Car data with an average error rate of 4.0% for the training set which was 42.61% better than the training error for CMC data. Although the Contraceptive data remains within the 45-55% error rate range, it appears to have some benefit by using the ANN.

Support Vector Machines (SMO)

Radial Basis Function (RBF) Kernel and Normalized Polynomial Kernel were both used for the SVM algorithms.



Normalized Poly performed better than the RBF kernel for both data sets. The RBF training for the Car had a higher increase in error rate than the others across the rest of the kernels. The Car training RBF kernel had a slight linear increase in error rate compared to a more curved increased seen on the normalized polynomial kernel. The Car test RBF had a peculiar behavior in which all of the rates, regardless of how much you take away from the test set had a value of 28.02%

Times (Hr:Min:Sec)

Classifier	Training	Test
Decision Trees	*Car* 00:00:04 *CMC* 00:00:08	*Car* 00:00:02 *CMC* 00:00:03
Boosting	*Car* 00:00:24 *CMC* 00:01:09	*Car* 00:00:07 *CMC* 00:00:13
ANN	*Car* 00:56:03 *CMC* 00:34:13	*Car 00:19:08* *CMC* 00:07:31
SVM	*Car Norm Poly* 00:09:31 *CMC Norm Poly* 00:16:00 *Car RBF* 00:05:06 *CMC RBF* 00:05:10	*Car Norm Poly* 00:00:46 *CMC Norm Poly* 00:00:38 *Car RBF* 00:00:27 *CMC RBF* 00:00:40
KNN	*Car* 00:00:23 *CMC* 00:00:22	*Car* 00:00:05 *CMC* 00:00:04

Conclusion

It is worthwhile to reiterate the benefits and weaknesses of both datasets and how they were analyzed using all of the algorithms. The Contraceptive data remained at around 40-60% error rate consistently for all of the algorithms. This indicates a couple of notions. The attributes in the data may not be good predictors of the class attribute. Since the data was extracted from a subset of a different study, this steady error rate across the algorithms may also be because of a sampling issue. Subsequently, there were a low number of instances comparatively to other available datasets and the high error rate, as well as low performance across all algorithms may be attributed to this fact when looking at the Contraceptive data. With some of the algorithms, the larger differences in performance within the training and testing sets suggest that there is a considerable amount of variance that could have been reduced if a larger number of instances were given.

One method of addressing this issue is by normalizing/discretizing the other two numerical values in the Contraceptive dataset (e.g., -inf to 20, 21-50, 50-inf). However, categorizing these variables did not improve the performance and, in some instances, increased the error rate. This reiterates the notion that the data might not be particularly good and when looking at the classification, short-term appears to be equally distributed (i.e., confusion is high), even though the other class values are more distributed.

Another consideration is that all of the algorithms were generated using a 10-fold cross validation. In hindsight, this may have caused unreliable smaller tests because outliers may have skewed the average. Unfortunately, I did not have enough to split the data further for a validation set for any of the algorithms. The J48 and Multilayer Perceptron would have particularly benefited from a separate cross validation to test and improve accuracy performance.

However, taking into consideration the number of instances for both datasets and the fact that the car evaluation data set performed relatively well across all five algorithms, it is more likely that other factors played a role in such poor performance with the Contraceptive dataset. Regardless of various speculations, a different dataset may have provided a more interesting comparison of the different algorithmic performance against the Car Evaluation data. In future assignments, I may consider datasets with significantly more instances and noise to see what challenges and benefits appear while applying them to different algorithms.

Nevertheless, it is also important to acknowledge (and appreciate) clean and thorough data. Although this may be rare in the real world, as well as the fact that some algorithms such as J48 have methods in handling noisy data, having well-organized and full datasets allow more thorough and less biased predictions using these algorithms.