Multi Classifier Cross Validation

COGS 118A

Ahsan Bari, A13865703

Abstract

In this paper I will be determining the efficiency of 4 different classifiers on 3 different datasets. The datasets are all relatively large with a mix a quantitative and qualitative data. As the datasets became larger and larger; however, the efficiency of some classifiers degraded while others did better. In order to determine the effectiveness of the classifiers, each classifier would sort each dataset in one of 3 splits, 20/80 train/test, 50/50 train/test, and 80/20 train/test. The classifiers I used for this project were K-nearest Neighbors, Decision Tree, Support Vector Machine — both RBF and Linear Kernels— and Random Forest.

Introduction

For this experiment I used 3 different datasets from the UCI Machine Learning Data Repository. The datasets were all classification tasks: the first one, a car evaluation task to determine the value of a car based off different features of the car. The car dataset was relatively small compared to the other two datasets at (1728,6) . Being a smaller dataset I decided that Linear SVM, KNN, and Decision Tree would be ideal classifiers to use for this task. The second dataset examined different species of mushrooms, the task was to determine which mushrooms would be toxic based off different features. The Mushroom dataset was slightly larger than the car dataset at (8124,22) so I determined that it could use the same classifiers as the car dataset. Lastly, the third dataset was a large bank evaluation dataset. The task for this set was to determine which individuals would apply and successfully receive a loan based on a number of factors. This dataset provided a mix of quantitative data as well as qualitative data which required the use of one-hot encoding to effectively utilize the classifiers. For This dataset I decided this one was too large for Linear SVM so I used KNN, Decision Tree, and Random Forests

Methods

In order to make the tests as consistent as possible, I created a script to shuffle and split the datasets into training and test data. Once the training and test datasets were established the classifiers would loop through the tree splits of training and test data and would return optimal features, training accuracy, and classifier accuracy. The classifiers I used generally worked well for each dataset and only very large datasets would need to be reexamined.

1. K-Nearest Neighbors (KNN)

   For KNN I used a standard, uniform weighted KNN. I varied the size of n depending on the size of the dataset and used ‘n_neighbors’ as the ideal parameter to train the classifier.

2. Decision Tree

   For decision tree I used ‘entropy’ as the standard criterion because there was a gain of information when the classifier would run. I also used ‘max_depth’ as the ideal parameter to train the classifier.

3. Support Vector Machine (SVM)

   For SVM, I used both ‘linear’ and ‘rbf’ kernels and made the c_list and gamma_list to be the range of the size of the dataset. For the ideal parameter, I used C.

4. Random Forest

   For random forests, I used ‘entropy’ as the main criterion because a random forest is a collection of decision trees; therefore, I wanted the random forest to be consistent with decision tree. Additionally, I sed ‘max_depth’ as the ideal parameter to train the classifier.

Tests

Because the size of the datasets became quite large, establishing consistent hyperparameters for all tests was quite challenging. To resolve this, to a certain extent, I measured the datasets by optimal feature for classification. Classifiers that consistently measured the optimal feature would be considered successful.

Results for Car Evaluation are as follows:

	Classifier Accuracy	Best Training Accuracy	Optimal Feature
KNN 20/80	0.266088	0.459408	6
KNN 50/50	0.218750	0.473658	6
KNN 80/20	0.164740	0.416964	7
Decision Tree 20/80	0.292842	0.391279	4
Decision Tree 50/50	0.315972	0.370651	4
Decision Tree 80/20	0.300578	0.356727	4
SVM 20/80	0.272596	0.340642	4
SVM 50/50	0.266204	0.290213	7
SVM 80/20	0.254335	0.283656	6

Results for Mushrooms are as follows:

	Classifier Accuracy	Best Training Accuracy	Optimal Feature
KNN 20/80	1.000000	1.0	1
KNN 50/50	1.000000	1.0	1
KNN 80/20	0.999385	1.0	1
Decision Tree 20/80	1.000000	1.0	1
Decision Tree 50/50	1.000000	1.0	1
Decision Tree 80/20	1.000000	1.0	1
SVM 20/80	1.000000	1.0	1
SVM 50/50	1.000000	1.0	1
SVM 80/20	1.000000	1.0	1

Results for Bank Evaluations are as follows:

	Classifier Accuracy	Best Training Accuracy	Optimal Feature
KNN 20/80	0.033150	0.035941	10
KNN 50/50	0.034593	0.036232	2
KNN 80/20	0.033175	0.034227	4
Decision Tree 20/80	0.058503	0.075785	5
Decision Tree 50/50	0.054145	0.082968	7
Decision Tree 80/20	0.057061	0.062230	4
Random Forest 20/80	0.058033	0.053422	5
Random Forest 50/50	0.057374	0.049117	6
Random Forest 80/20	0.063143	0.049828	5

Conclusion

By comparing the consistency between optimal features I was able to conclude that Decision Tree was the best classifier. In order to make this experiment better, I would expand the number of classifiers I used as well as explore different hyperparameters that could better classify the datasets.

Link to code: https://github.com/ahsanbari/118aFinalProject

References

https://github.com/ahsanbari/118aFinalProject

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

https://scikit-learn.org/stable/modules/tree.html

https://archive.ics.uci.edu/ml/index.php