# Using Unsupervised Learning to Inform a Binary Classifier Graph for Multiclassification

Gabriel Spil
SunetID – gspil

Stanford CS 229 Spring 2021

*Abstract*— **Can a Supervised Learning model be boosted by knowledge gained from Unsupervised Learning? Can predefined labels hide latent patterns in data that may be helpful in designing a classifier? In this project, Unsupervised Learning is used to create an algorithmic template for a Supervised Learning Ensemble Model. The model is tested on the MNIST database of hand-written digits and can achieve a greater than 20% misclassification reduction over the SciKit Learn Logistic Regression Classifier in OVR model.**

## I. Introduction

Binary classifiers can be used in multiclass problems. One approach to multiclass problems using a binary classifier is to use the One-vs-Rest division of labor. In this case, one would build N (one per classification) One-vs-Rest models and use these models to build an ensemble model where the highest probability of a class denotes classification.

For the context of the project, we will use the MNIST database (Modified National Institute of Standards and Technology) of handwritten digits dataset. This is a set of 70,000 samples that are represented as 28 by 28-pixel matrixes. Each pixel has a value from 0-255.

The original impetus for the project was based on the intuition that digits seem to have strong similarities which each other and group into some sub-classes. At least this is true from the viewpoint of human perception: some are rounded, some are linear and some a combination of both. Additional we may discover mathematical groupings outside of what is humanly perceptible.

The goal is to see what innate structure might be inherent in a dataset that is already labeled. Classification in Supervised Learning is to a degree a modeling assumption about the nature and similarities of the data entities. It is common to address data mining with the assumption that the labeled data best represents the strongest cohesion of the data set. By putting this assumption aside, we can pose two questions about the data set and the problem in general. The first question is "are the current classifications correct for the goal(s) of data mining?" And the second question "is there any latent cohesion in the data outside of the labeling that might be leveraged during analysis?" The former is a question that should be asked in any engineering process to ensure understanding of both the problem and solution domain. The latter might lead to insight on approaches or ways to reduce noise in the data set.

## II. Exploration of Unsupervised Learnings

### A. Overview

The first phase of the project was to understand what type of latent natural cohesion might exist in the digit dataset. The goal is to use Unsupervised Learning to discover cohesion in a data set, where the cohesion may be orthogonal to the labels. Then to see if following the natural cohesion of the data set gives increases in classification accuracy over solely using the labels in Supervised Learning.

For the purposes of initial prototyping and investigation, a subset (~1700 images) of the MNIST handwriting digits dataset with feature reduction is used.

### B. Algorithm

Here is the general algorithm used to explore the natural cohesion of the dataset subset described above.

```
Start with the digit set 0-9.
    For each set of digits For cluster size in (2 to digit classes) run K-Means
    Determine the highest clustering based on cluster size. Reduce the digit set based on best fit and repeat.
    If the digit set is 2 then finish.
```

The above algorithm is a divide-and-conquer approach that divides the problem at each step into a subset of digits to classify into a binary set. This is represented in a graph where the leaf nodes are the final evaluations of a digit.

## C. *Results*

The Unsupervised Learning results suggest that this approach yields a higher level of classification then standard K-Means using 10 clusters (one for each digit). It was also found that precision on the initial set of all digits reduces as the number of clusters is increased. This leads to the conclusion that there is stronger cohesion between sub-sets of digits that can be leveraged to boost Supervised Learning. Below is a graph showing the decline of precision for each digit and the number of K-Means clusters are increased.

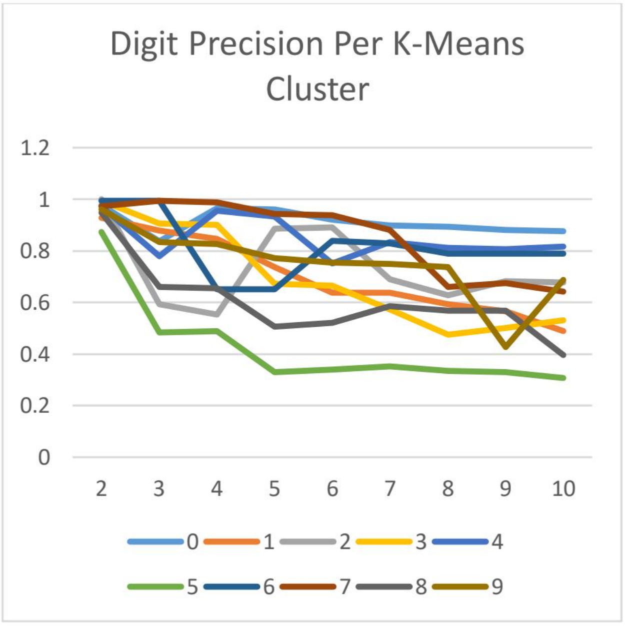

*Figure 1 Per Cluster Precision*

Figure 2 shows the relationship between the graph-based classification and a 10-cluster based K-Means classification. The average precision for K-Means is 63% vs 70% for the graph-based classification.

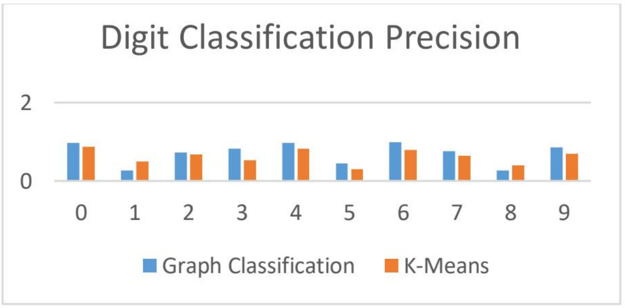

*Figure 2 Final Precision*

The results above lead to the observation that gains in accuracy may be attained by using the generated classification graph to build an ensemble of classifiers based on the graph.

## III. Design

### A. *Dataset*

The dataset selected is the full 70,000 sample MNIST database of handwritten digits. Each sample is a graphic with 28 by 28 pixels, which each pixel having a value from 0-255. The dataset has a dimensionality of 784 features, one per each pixel. To avoid issues around high dimensionality we reduce the dimensions to lower levels using the UMAP Python library. We test with various dimensions and find that 64 features gives stable results for the project.

### B. *Models*

We produce various models to explore the relationship between different approaches and effects on accuracy.

### C. *Graph Models*

The Graph Model is the primary model used to test the premise of using Unsupervised Learning to boost Supervised Learning. The model is composed of two primary components. The first component is a Mapping object that contains the decomposition of the classifications learned from Unsupervised Learning. Figure 3 contains the graph generated by a decomposition process run based on the MNIST dataset compressed from 784 to 64 features.

The Mapping object represents this decomposition in a Binary Tree, where each node contains a set of digits and the leaves of the Binary Tree represent a single digit.

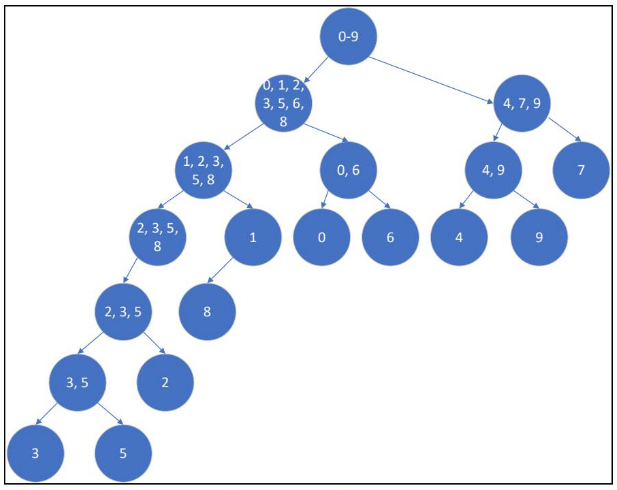

*Figure 3 Classification Decomposition*

The second component is a Binary Tree of Classifiers. The Classifier Binary Tree nodes mirror the Mapping Binary Tree nodes. Each Classifier is trained on the set of the training data scoped by the mirrored node in the Mapping. For example, the root Classifier is trained to classify (4,7,9) versus the other digits.

Predictions are based on cumulative probability of a classification along a chain of classifiers starting from the root to the leaf containing the digit. For each classifier *k* in the chain of classifiers that are trained to predict the digit we return:

$$\prod_{i=1}^{k} P_i(x)$$

### D. *OVR (One vs Rest) Model*

The OVR Model is a One vs Rest model that uses the Scikit Learn Logistic Regression Classifier as a primitive. The model produces a set of classifiers, one per each class. The classifier learns the single class versus the remaining classes. The purpose of implementing a model based on the primitive Scikit Learn Logistic Regression Classifier was for comparison against the Graph Model. And this avoided comparison against a model that was a black box. In the end the project OVR implementation and the Scikit Learn Logistic Regression Classifier in OVR mode performed identically.

The probability of a sample classified as a digit is the average probability of each classifier.

$$1/n \sum_{i=1}^{k} P_i(x)$$

The figure 4 shows the static class design for the Graph and OVR models described above.

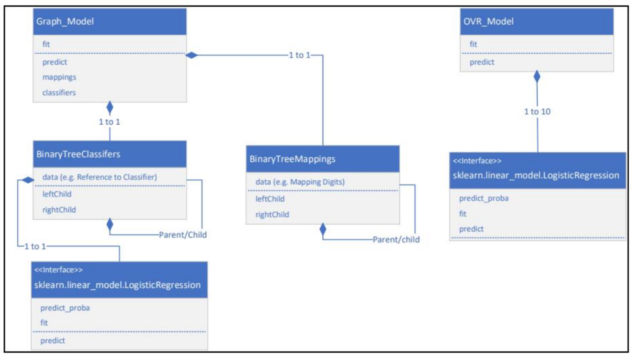

*Figure 4 Class Designs for Graph and OVR Models*

### E. *Hybrid Ensemble*

The Hybrid Ensemble uses the Graph and OVR model in unison. Each prediction that does not match in both models in decided by the highest probability prediction returned by the models for the sample.

### F. *Discriminant Ensemble*

The Discriminant model uses classifiers trained on specific pairs of digits. The goal was to attempt to address specific misclassifications that were evident in the Confusion Matrix. Commonly misclassified pairs are [1,7] and [3,8] for example.

## IV. Model Evaluation

The primary goal was to evaluate the Graph Model against the baseline to determine if Unsupervised Learning could help design an ensemble model. The Graph Model performed better than any other model evaluated. The Graph Model has a 50% decrease in misclassification over the baseline. It also performed about 1% better in accuracy than the OVR model. This aligns to a reduction in misclassification of about 20%. Open-source classifiers are very efficient on this dataset and incremental improvements are somewhat unexpected.

The Hybrid and Discriminant Models did not add any meaningful accuracy above the OVR Model. They are mentioned here as potential theories for improvements which did not behave differently than the OVR model.

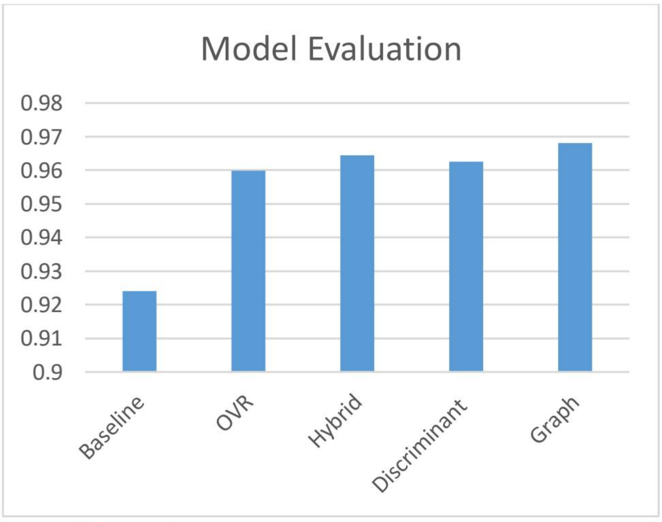

*Figure 5 Model Evaluation*

## V. Project Artifacts

The project artifacts are in the following Jupyter Notebooks written in Python. The project depends on the following Python libraries: Numpy, UMAP and Scikit Learn.

### A. *LoadProcessData.ipynb*

This notebook loads the original handwritten digit dataset from MNIST (Modified National Institute of Standards and Technology database).

### B. *CreateMappings.ipynb*

This notebook deals with creating Mapping objects and persisting them to disk.

### C. *BuildTestModel.ipynb*

This notebook tests models on the original handwritten digit dataset from MNIST (Modified National Institute of Standards and Technology database).

## VI. Related Works

There is some work on digit and alphabet topology which takes a topological view of character similarities. This is similar to the intuition that was the driving impetus for the project. Coincidentally, some of the topological classes defined were "discovered" by the Unsupervised Learning algorithm.[4]

K-means clustering has also been used to train per digit autoencoders on the handwritten-digits dataset. The approach is more detailed and uses EM and GMMs to classify digits.[6]

There have also been projects that use Unsupervised Learning to divide the dataset without labels and use 2 generative models to classify. This is an ensemble approach that uses the two models for consensus. [5]

The last two references show that using Unsupervised Learning in conjunction with Supervised Learning can boost classification. My approach is much simpler, but I could not find an existing project with the exact approach.

## VII. Conclusions

Latent structures discovered in Unsupervised Learning can be used to train binary classifiers on multi class problem. One approach of using learned patterns is to divide and conquer based on recursive division of the classification problem.

Future work would include more experimentation on different datasets to see if other datasets might also show similar patterns.

## References

[1] Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

[2] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020)..

[3] Leland McInnes and John Healy and James Melville. (2020). "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction".

[4] O.Knill, (2011) Lecture 9: Topology of Alphabet. Section 2: Warm-up with digits. (http://people.math.harvard.edu/~knill/teaching/mathe320_2011/handouts/08-worksheet.pdf).

[5] H. Lee, T. Kim, E. Song and S. Lee, "Collabonet: Collaboration of Generative Models by Unsupervised Classification," 2018 25th IEEE International Conference on Image Processing (ICIP), 2018, pp. 1068-1072, doi: 10.1109/I.

[6] G E Hinton, M Revow, P Dayan, "Recognizing Handwritten Digits Using Mixtures of Linear Models" Department of Computer Science, University of Toronto. Toronto, Ontario, Canada M5S 1A4.

[7] Sener, O., Hyun Oh Song, Ashutosh Saxena and S. Savarese. "Learning Transferrable Representations for Unsupervised Domain Adaptation." NIPS (2016).