Machine Learning Final Exam (Fall 2019)

Vahid Khalkhali
Department of Electrical and Computer Engineering, Temple University
vahid.khalkhali@temple.edu

Introduction: There are two datasets of two-dimensional and five-dimensional data. Each dataset contains one large training file, one development file and one evaluation file. It is supposed to train a classifier on train data file, develop it on development data file and finally evaluate it on eval file. These three files are available completely for two-dimensional data, but for five-dimensional data, the eval data file does not contains labels and it is unknown. The competition is guessing the labels for five-dimensional data.

Two algorithms have been used for classification. Before training and running each algorithm, all data have been normalized with Scikit-Learn normalization library. The normalization has been done with maximum value normalizer. Therefore, it is expected that all the data would be in the range of zero and one.

Besides the following two algorithms, SVM was also implemented on 2D dataset. As it was expected, the difference was negligible.

Algorithm No. 1 Description (GMM): After plotting the 2D data, the visualization gave the idea that they should be generated by Gaussian distribution. Therefore, Gaussian Mixture Model used as the first algorithm. The number of clusters in GMM is very important. In the used algorithm, each class clustered separately with variable number of clusters, then the distribution of each class estimated. Based on the estimation of each class, ML estimation of PDFs used for classification. By looping over multiple cluster numbers, the optimum number found and used for estimation of PDFs. This algorithm has been used for 2D and 5D data. GMM implementation of Scikit-Learn library in Python has been used. This implementation, based on the documentation on their website, initialize the first centers of clusters with K-means algorithm. So, there is probability that it cannot properly find the best probability distributions for fitting to data. In general, GMM is like many machine learning algorithms, sub-optimal and not perfectly optimum.

Algorithm No. 2 Description (MLP): A simple Multi-Layer Perceptron implemented in TensorFlow has been used for classification. Several parameters of MLP have been changed. Firstly the optimizer parameter determined. Then almost all the activation functions in TensorFlow have been used. Then the number of layers increased. When the number of layers has been increased, the error rates on train data significantly decreased but the error rates on development data increased. This observation clearly shows that increasing the number of layers will cause over fitting. It means that the classifier just adjusted to train data and has lost generalization.

Finally, based on several trial and error, two hidden layers with 60 and 40 neurons respectively have been implemented. Since optimization of MLP is sensitive to initialization, the seed values have been changed in a range and the best overall classifier has been chosen as the winner MLP classifier. The same architecture has been used in 2D and 5D.

Results: The results show that even for data with Gaussian behavior, the neural networks results are good.

The running time for each algorithms and each dataset was different. The machine which the algorithms has been run on was Core I7-Forth generation with 8GB Ram and Python version 3. GMM on 2D data finished too quickly. GMM on 5D data took about a minute to run. MLP for 2D data with 20 different

seeds took about ten minutes. But the same MLP architecture with 20 seeds took more than 3 hours for 5D dataset.

In 2D dataset, GMM proposed 5 clusters for class one and 4 clusters for class 2. In 5D dataset, GMM proposed 3 clusters for class 1 and 4 clusters for class 2. The best seed for initialization of MLP for 2D dataset was 5 and for 5D dataset was 15. The error rates are shown in following table.

Algorithm	**2 D**			**5 D**		
	Train	**Dev**	**Eval**	**Train**	**Dev**	**Eval**
GMM	7.97%	7.65%	8.05%	37.23%	36.80%	37.13
MLP	8.07%	8.15%	8.30%	36.31%	36.60%	36.96

Based on the observation of evaluation on unknown sets, the classifier works fine and followed the underline pattern on data. Both GMM and MLP produced almost similar results on the same datasets. The difference between two classifiers are not statistically significant.

Conclusions: The numerical simulations show that almost all classifiers with some tuning will have similar results. The tuning usually is usually done to avoid overfitting or underfitting. In both cases, the evaluation dataset was not considered important and just the train and development has been used. In several cases, especially for 2D dataset which the eval file was known, it was observed that the high error rates of train and dev datasets prevents a classifier with lower error rates on eval set, be chosen. Considering the correlation between an evaluation dataset and training dataset with bring a good classifier but when classifier used to predict data which the characters have not seen before is usually will be done by chance. Since the results' difference are not statistically significant, it can be implicated that almost all classifiers have the same quality and choosing one against another is based on the situation and application.