Summary on the Assignment for the position of augmented reality software engineer at DataVerse, by Loizos Shianios

Dependencies and packages used: tensorflow, scipy, sklearn, pandas, numpy, prince, xgboost, matplotlib. Note that to use basinhopping in tensorflow a slight modification to scipy minimize module is needed. The mod is included in the repository.

The problem at hand is a binary classification problem with mixture of data. Two methods were examined as to how to convert the data into numeric form. The first was via the use of dummy variables, and each category of our categorical variables is introduced as a variable in the data set. The second was via the use of FAMD (Factor Analysis for Mixed Data), that combines PCA and MCA as well as CA. To use FAMD in Python we need prince and it is only compatible with Python 3 or above.

The idea was to examine three models and compare them based on their performance on the validation and testing sets. The first model which is used as a base model is a logistic regression, where a decision boundary is drown between the data to form the two classes. The second model is a model built via XGBoost. It combines several small decision trees, and uses gradient boosting to optimize the ensemble. It aims to predict the class of the input vector directly. The final one is a two hidden layer neural network, built in a fashion as to output the probability of the input being a member of each class, and the predicted class for the input is the one with the higher probability.

Due to lack of time we curry minimal parameter tuning and the results were very low; specifically, for Logistic regression the F1 score was about 0.68, for XGBoost 0.1 and for the NN about 0.2. Also the NN was tested on the full adult.test set found in the repo, with some very low scores but comparable with the ones achieved for the validation set.

As a further evaluation measure we use the confusion matrix for each method. We found that the data set was imbalanced, there is about 3 times more data for the less than 50 K class, and all methods tent to output a lot more this class, which suggest that a slight modification to the sampling method is needed, to help the algorithm achieve a better training.

It was also found that the scaling of the data was greatly influencing the results and further investigation was need as to find the optimal scaling method. Also when using FAMD the number of output dimensions affect dramatically the results as well. Some plots can be found in the 'Results' folder. Also if you have tensorboard installed the graph of the neural network is stored in the logs folder

A couple of final points. First there are a couple of spotted 'mistakes' in the code, as well as a few comments on how to further investigate a particular part of the problem. This is to let you see how I tend to work and keep my work while under development. Also I try to include as many different techniques as time would allow, rather than focusing on just some standard methods. On this note a couple of further suggestion are, first we could also use t-SNE for dimensionality reduction or even t-SNE could follow FAMD. Also we could use SOM neural network, a class of unsupervised classifiers; a generic form of SOM can be found in my other Git repository, supplied in my CV. Also the ART networks (Adaptive Resonance Theory) and their advancements like fusion-ART, are well suited for the job.

Thank you, and have a good day.