CS412 IML HW5 (Option 1) - Written Part – Chinmay Gangal

Goal: Predicting "Empathy" of a person based on learning from a huge collection of survey response for Young People

Data Preprocessing Steps:

- Filling nulls: Several columns have null values. Although these are very less in count compared to the total dataset size, still these need to be either eliminated or filled with some value. In our case, we fill null spaces with the "mode" of the respective feature, i.e. the value that appears maximum times in the column
- Encoding categorical features: There are 11 categorical features i.e. text features in our data. Simply converting these to numbers using mapping wouldn't be sufficient, since these integers would reflect intensity, which isn't the case. Hence we apply OneHot encoding, which splits each category value into a new binary feature
- Feature scaling, standardization & Normalization: When we use models for learning, then imbalanced feature ranges (for those that depend on vector space), biases & abnormal distribution hurt the time spent on training as well as accuracy to a small extent. Thus, we perform the respective operations on our data to solve this issue.

Models experimented with:

Following are the models tried with tuning and their respective accuracies on development data.

We have used a 60:20:20 split for train:dev:test. Since the data isn't very huge, we need considerable data for validation to correctly tune hyperparameters, hence have not chosen the popular 80:10:10 split.

- K nearest neighbors — Accuracy: 0.27
- Decision Tree — Accuracy: 0.37
- Perceptron — Accuracy: 0.38
- **Random Forest — Accuracy: 0.49 — Test accuracy : 0.396**
- Gaussian Naïve Bayes — Accuracy: 0.37
- Polynomial SVM — Accuracy: 0.37
- SVM with RBF — Accuracy: 0.38
- Linear SVM — Accuracy: 0.39

(a) The data was *Young People Survey* dataset, which consists of over a 1000 survey responses and 150 different features. The task is to predict Empathy based on the person's various preferences in Music, personality traits, etc

(b) The final Machine Learning solution chosen for this task was the Random Forest Classifier, which is an ensemble method since it trains on multiple decision trees. It is to be noted that although other ensemble methods were tried, they did not yield an increment in the results on dev and instead demanded larger time for training, hence were not chosen. The Random Forest classifier clearly proved to give better results than other models on the development data, and was notably faster than the polynomial SVM, which usually takes more time depending on the degree. Here we have tuned the hyperparameters for improving results.

(c) Accuracy was used to evaluate success of every model, i.e. on the development data

(d) Jupyter notebook was used for experimenting since it allows breaking the entire code into sections (cells) and run each section individually. This saves time spent on executing certain parts of the script of which we already know the results

(e) The results, as shown above, help us infer that the ensemble classifier Random Forest gives best accuracy, although there is a huge drop in accuracy for the test data

(f) We can observe from the notebook some examples where our approach incorrectly and correctly classified. A possible improvement could be applying boosting through a variety of tuned models.