PR2: Image Classification

Sunny Udhani
Student ID: 012457289
Rank: 4
F1-score: 0.8579

Approach

1. Preprocessing - Feature Selection/Dimensionality Reduction

- Fetch the training data, training classes and test data from train.dat, train.labels and test.dat files respectively to pandas dataframes with float_precision parameter as 'high' so as to make sure we don't lose information.
- Fix the issue of an imbalanced dataset using imbalanced librarys' oversampling technique SMOTE.
- Randomize the training data using train_test_split and then feed the pipeline for fitting the model.
- Dimensionality reduction gives us a subset of features to be trained in the model so as to make the model simple throwing out redundant or irrelevant features. After playing with a bunch of techniques and cross-validating the output scores on my training data, I ended up using SelectKBest with chi2 scoring which is a feature selecting technique where sklearn's webpage comparing different techniques was very useful.

2. Classification

- Used sklearn's pipeline feature for simplifying tasks of transforming datasets and estimating results. The pipeline is used to assemble several steps together so that the results can be cross-validated using different parameters.
- Tested multiple classifiers against the training data using a dictionary of classifiers. Dictionary in method 'give_me_a_classifier' contains classifiers of different types. After cross-validating scores of different classifiers, I ended up using an ensemble classification technique called ExtraTreesClassifier.
- Used split training data to test the classifiers locally before submitting to CLP.
- Stacked the training data split as explained earlier to perform prediction against test data and write the result to an out.dat file.

Methodology

1. ### Preprocessing
    - The very first step in processing the data was to read the data in a way to preserve the significant digits in the features. The objects in the dataset were having 17 significant digits so I used pandas float_precision parameter to preserve complete values.
    - The dataset was having invariant features that were irrelevant to the classification problem, so I created a plot of a number of uniques values in each feature and sent to the professor for review, which resulted into Professor acknowledging the problem and allowing students to only work with 48 features from 832-880.
    - Following the suite from program 1 as we have imbalanced data I used an oversampling technique SMOTE to oversample under-represented classes resulting in better classification accuracy as the model was being tested with a balanced dataset and fitted with an imbalanced one.
    - Tested multiple Dimensionality reduction techniques with my dictionary of DR techniques in function ‘give_me_a_dr’. There was not much variance when used with balanced training data and reducing dimensions using different techniques but surely SelectKBest outperformed when used against test data and submitted in CLP. Also reducing features to 35, 30, 25… etc. yielded fewer scores as compared to 40 features. Explored methods PCA, TruncatedSVD, LLE-standard and spent significant time at T-SNE, fasttsne and MultiCoreTSNE.
2. ### Classification
    - I used multiple classification techniques as found in my dictionary in method ‘give_me_a_classifier’. Also used sklearn’s Pipeline feature to test all my models in conjunction with the dimensionality reduction technique.
    - Spent a significant amount of time researching Neural Networks and their compatibility with the problem in hand but in vain. The ensemble techniques were proving to be a better fit for the problem. Also spent a considerable amount of time exploiting AdaBoost on the dataset and tuning input parameters.
    - Shown from my personal submissions graph I was never afraid of testing new classification techniques.
    - From the very start made an overall structure of code using utility functions and sklearn’s pipeline to make testing models simple with less code. Also explored techniques like OpenCV and GridsearchCV which while didn’t end up in my program but are very useful.
    - Sklearn’s webpage showing a comparison between different ensemble models was very helpful in understanding the ExtraTreesClassifier. ExtraTreesClassifier was simply a better choice because it uses averaging of scores on random sub-samples and controls over-fitting internally.
    - Achieved F1-score of 0.96 locally with training data and 0.8579 on CLP using 45 features and 500 trees.