Assignment One: Image classification, dimensionality reduction

CS909: 2018-2019

Submission: **12 pm (midday) Wednesday 13th February 2019**

Notes:

(a) Submit a zip file of the completed iPython notebooks to Tabula. Make sure to put comments in your code explaining what you have done and what you have observed from the results.

(b) This assignment will contribute to 25% of your overall mark.

This assignment consists of 7 exercises which are split into 4 parts. The first 3 parts can be found in the lab of **Classification** while the last part can be found in the lab of **Dimensionality Reduction**. Please download the corresponding ipynb files from the module web page https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs909.

Part 1: Binary Classification of MNIST Dataset

In the first set of tasks, you will evaluate a number of popular classifiers for the task of recognising handwritten digits from MNIST dataset. Specifically, we will focus on distinguishing between 7 and 9 which are known to be a hard pair. You will use the scikit-learn libraries.

Binary Classification in scikit-learn

All classifiers in scikit-learn follow a common pattern that makes life much easier. Follow these steps for all the tasks below.

1. Instantiate the classifier with appropriate parameters
2. Train/fit the classifier with training data and correct labels
3. Test the classifier with unseen data
4. Evaluate the performance of classifier

Exercise 1: Decision Trees (10 marks)

In the first task, you will use Decision trees (see http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier) for classification. Implement a CART decision tree with splitting criterion as entropy. Also print the learned decision tree using the supplied function "plot_tree".

Exercise 2: Naive Bayes (10 marks)

In this task, you will create a multinomial Naive Bayes classifiers and evaluate it. You might want to follow http://scikit-learn.org/stable/modules/naive_bayes.html

Exercise 3: Logistic Regression (10 marks)

Logistic regression is a simple classifier that converts a regression model into a classification one. Implement a logistic regression model by following the details at http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Exercise 4: Random Forests (10 marks)

Random Forests is a very popular ensemble method. See url http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html for details. Implement a Random Forest.

Part 2: Exploration of Different Evaluation Metrics for Binary Classification

In the second set of tasks, we will learn how (and when) to use various evaluation metrics for performance evaluation of the binary classification models that you created so far. You may want to check http://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics for additional details.

Exercise 5: Print the classification results (total: 20 marks)

For each of the models above:

- Task 5a: Print the classification report and confusion matrix (5 marks)
- Task 5b: Print the ROC curve (5 marks)
- Task 5c: Print the AUC curve (5 marks)
- Task 5d: Print the precision/recall curve (5 marks)

Part 3: Parameter Tuning through Grid Search/Cross Validation and Parallelisation

So far in this assignment, you manually tweaked the model till it became better. For complex models, this is often cumbersome. A common trick people use is called Grid Search where you exhaustively test various parameter combinations and pick the best set of parameter values. This is a VERY computationally intensive process and hence it will require some parallelisation.

In this part, you will learn how to tune Random Forest for MNIST dataset and then parallelise it so as to get results faster. You might want to take a look at http://scikit-learn.org/stable/modules/grid_search.html for additional details.

Exercise 6: Tuning parameters for Random Forest using grid search (15 marks)

Part 4. Dimensionality Reduction

Dimensionality reduction is a key data pre-processing technique. In this part, you will perform PCA, a popular dimensionality reduction technique, on MNIST data to see how it performs.

Exercise 7: Dimensionality Reduction (25 marks)

- Task 7a: Train a multi-class classifier (OneVsRest) with LinearSVC class and make predictions and print the training time and the classification accuracy on the test set. (5 marks)
- Task 7b: Perform PCA with 100 components on the training data, map both training and test data into 100-dimensional space by PCA, train a multi-class classifier (OneVsRest) with LinearSVC class using the training data, make predictions, print the training time and classification accuracy on the test set. (10 marks)
- Task 7c: One way to determine how much components needs for PCA is to find the smallest value such that it explained 95% of the variance. Using the PCA results obtained above, print the cumulative variance that is explained by 100 components. (10 marks)