Introductory Applied Machine Learning: Assignment 3

School of Informatics, University of Edinburgh

Instructors: Victor Lavrenko and Nigel Goddard
Assignment prepared by Sean J. Moran, revised by Boris Mitrovic

Handed out 30 Oct 2014, due by **1600** on **Nov 10 2014**.
Hard copy **and** electronic submission required. The time of the deadline will be strictly enforced.

Remember that plagiarism is a university offence. Please read the policy at
http://www.inf.ed.ac.uk/teaching/plagiarism.html .

Marking Breakdown

70-100% results/answer correct plus extra achievement at understanding or analysis of results. Clear explanations, evidence of creative or deeper thought will contribute to a higher grade.

60-69% results/answer correct or nearly correct and well explained.

50-59% results/answer in right direction but significant errors.

40-49% some evidence that the student has gained some understanding, but not answered the questions properly.

0-39% serious error or slack work.

Mechanics

You should produce a word processed report in answer to this assignment (e.g. with LaTeX).

- **pdf** format is acceptable for the report, other formats are not.
- you need to submit this report as a hard copy to the ITO **and** electronically as described below.

In question 2 you are also required to produce a file `iaml_assignment.res`. For the electronic submission place your report in a directory called `iamlans` and submit this using the `submit` command on a DICE machine. The format is

```
submit iaml 3 iamlans
```

You can check the status of your submissions with the `show_submissions` command.

NOTE: Your electronic submission will **not** count if you do not submit a hard copy of your report to the ITO.

Late submissions: The policy stated in the School of Informatics MSc Degree Guide is that normally you will not be allowed to submit coursework late.

Collaboration: You may discuss the assignment with your colleagues, provided that the writing that you submit is entirely your own. That is, you should NOT borrow actual text from other students. We ask that you list on the assignment sheet a list of the people who you've had discussions with (if any).

Important Instructions

(a) In the following questions you are asked to run experiments using WEKA. The WEKA version installed on DICE is **Version 3.6.2**. If you are working on a machine other than DICE (e.g. your laptop), please make sure that you download and install the **same** version. This is important as your results need to be reproducible on DICE.

(b) In many cases, the WEKA *Explorer* allows you to modify the random seed that will be used. Just using the default seed is fine. If you do change the seed you need to report the seed you have chosen.

(c) You may find that WEKA crashes with an out-of-memory exception. If this should occur, refer to the instructions in IAML lab 1 in order to run WEKA with a larger memory allocation.

(d) The .arff files that you will be using are available from the IAML website.

(e) **IMPORTANT:** Keep your answers brief and concise. NOTE: you may **lose points** for a report longer than **1000 words**. In Question 2 of the assignment you should not write more than 600 words in total.

(f) Please use the course NB site `http://nb.mit.edu/` for asking questions. You should aim to check this site regularly as assignment related clarifications will occasionally be posted.

(g) Please adhere to the following presentation guidelines when producing your report: (i) Please include your student number on the first page of the report. Please DO NOT include your exam number anywhere. (ii) Make sure to number your answers and keep them in the same order as they are presented in the assignment sheet. (iii) If you are asked to report a number, please do not report it as part of a sentence, but put it in a new line/table etc. (iv) Keep sentences/paragraphs short and to the point. (v) Print double-sided.

Description of the dataset

In this assignment our goal is to recognize objects from 19 different visual classes (e.g. person, dog, cat, car, ...) in realistic scenes. The dataset consists of several thousands photographs harvested from the web. Each object of a relevant class has been manually annotated with a bounding box. Images can contain none, one or multiple objects of each class. We have prepared a website where you can view the images [`http://www.inf.ed.ac.uk/teaching/courses/iaml/2014/assts/asst3/images.html`]. This website will also be linked from the course homepage.

Here we will focus on a single classification task: you will be required to classify images as to whether or not they contain a person. To save you time and to make the problem manageable with limited computational resources, we have preprocessed the dataset. We will use the "bag of visual words" representations. That is, each image is represented by a 500 dimensional vector that contains the normalized count for each of 500 different visual words present in the respective image (a similar representation is used for the spambase dataset, just for real words). Visual words are based on SIFT features. SIFT features are essentially local orientation histograms and capture the properties of small image regions. They possess attractive invariance properties which make them well suited for our task (you can read more about SIFT features in D.Lowe, IJCV 60(2):91- 110, 2004, but the details dont matter for the purpose of this assignment). Each SIFT feature is a 128 dimensional vector. From each image many SIFT features are extracted, typically > 2500 per image (features are extracted at regular intervals using a 15 pixel grid and at 4 different scales). To obtain visual words a representative subset of all extracted SIFT features from all images is chosen and clustered with k-means using 500 centres (such use of the k-means algorithm will be discussed in detail during the lecture). These 500 cluster centres form our visual words. The representation of a single image is obtained by first assigning each SIFT feature extracted from the image to the appropriate cluster (i.e. we determine the visual word corresponding to each feature by picking the closest cluster centre). We then count the number of features from that image assigned to each cluster (i.e. we determine how often each visual word is present in the image). This results in a 500 dimensional count vector for each image (one dimension for each visual word). The normalized version of this count vector gives the final representation

of the image (normalized means that we divide the count vector by the total number of visual words in the image, i.e. the normalized counts sum to 1 for each image). Our dataset with all images is thus a N $\times$ 500 dimensional matrix where N is the number of images. The resulting representation is much more compact and can be used directly to perform classification in WEKA.

The full dataset has 520 attributes (dimensions). The first attribute (`imgID`) contains the image ID which allows you to associate a data point with an actual image. The next 500 attributes (`dim1...dim500`) correspond to the normalized count vector. The last 19 attributes (`is_ <class>`) indicate the presence of at least one object of a particular class in the image. In most of the experiments (unless explicitly noted otherwise) you will be asked to train classifiers for classifying person vs. non-person images and only the `is_person` attribute and the 500 dimensional feature vector will be used. Do not use the additional class indicator attributes as features unless explicitly told to do so.

In Part A we provide you with a training (`train_images_partA.arff`) and a validation (`valid_images_partA.arff`) dataset. In Part B we provide three data sets: a training set (`train_images_partB.arff`), a validation set (`valid_images_partB.arff`), and a test set (`test_images_partB.arff`). The training and validation set contain valid labels. In the test set the labels are missing. The files are available from the IAML website.

Important: *Throughout the assignment you will be given various versions of the dataset that are relevant to a particular question. Please be careful to use the correct version of the dataset when instructed to do so. If you use the wrong version of the dataset by mistake no marks will be awarded.*

1 Exploration of the dataset [60%]

Important: *In questions (a),(b),(c) below all experiments should be performed on the training data using an 80% `Percentage split` (the default setting is a 10-fold cross-validation). All class indicator attributes except the `is_person` attribute should be removed. The `img_id` attribute should also be removed.*

(a) Train a `Logistic` classifier and a `SimpleLogistic` classifier (using the default parameters for both classifiers) using 80% `Percentage split` on the training set. Now explore the "usefulness" of the 500 features using the attribute evaluator `InfoGainAttributeEval` (in the *Select Attributes tab*). Answer the following question:

- What is the difference between the `SimpleLogistic` and the `Logistic classifier`? Look at the classification performance and the models learned. Explain your answer in light of the results you have obtained using the attribute evaluator `InfoGainAttributeEval`.

(b) Train a `Logistic` classifier using different values of the `ridge` parameter. You should use 80% `Percentage split` on the training set. Try the values 0.0001, 1, 100, 200, 800, 4000, and plot the percent correct as a function of the `ridge` value. Include a copy of your graph in your report (ensure the x-axis is scaled appropriately so that any trend is fully visible) and answer the following question:

- Explain the role of the `ridge` parameter and compare regularization to feature selection. Interpret your results in the light of the observations you made in question (a).

(c) Although linear classifiers often do well, in particular for noisy, high-dimensional data we might be able to do better with more complex classifiers. To investigate this possibility for our data set try the SVM classifier (`SMO`) with an RBF kernel. As in the previous experiments use 80% `Percentage split` on the training set (all class indicator attributes except `is_person` removed) to evaluate the classifier. Two parameters primarily affect the performance for this kernel: the kernel width, which is controlled by `gamma` and the complexity parameter `c`. Explore the effect of these parameters by answering the following three questions:

- First, explore the effect of `gamma` using the following values: 0.001, 0.01, 0.1, 0.3, 0.5. Plot the percent correct as a function of `gamma`. Include a copy of your graph in your report (ensure that the x-axis is appropriately scaled so that any trend is fully visible).
- For the kernel width that gives the best results you should then explore different values of the `c` parameter (using the values 0.001, 0.1, 1, 100). Plot the percent correct as a function of `c`. Include a copy of your graph in your report (use log-scale for the x-axis so that any trend is fully visible).
- Is this procedure guaranteed to find the values of `gamma` and `c` that lead to the highest percentage correct (PC) performance for our model? Briefly explain your answer.

Important: *In question (d) below we will evaluate our models on the validation set* `valid_images_partA.arff` *(instead of a percentage split on the training set). In the* `Classify` *tab choose* `Supplied test set` *in the* `Test options` *box and set the dataset set as appropriate. The validation dataset contains all 520 attributes. So you will have to remove attributes as required (remove the* `imgId` *attribute in all cases and the class indicator attributes as indicated in the relevant question below).*

(d) Reload the full training data that contains *all indicator variables for all object categories*. Remove the `imgId` attribute but keep all of the class indicator variables in the dataset this time. Compute the feature information gain ranking with respect to the `is_person` class indicator (i.e. all other class indicator variables are now also considered as features; make sure that in the "Select attributes" tab the `is_person` variable is selected as relevant class variable). Answer the following question:

- Look at the list of the best 50 features. What do you notice?

We will evaluate our classifier on the validation set. Again you will need to remove the `imgId` attribute from the validation dataset but do not remove the class indicator variables. Now train a simple `SimpleLogistic` classifier on this extended data set using default parameters (again, make sure that `is_person` is selected as the class attribute for classification in the `Classify` tab). Having done this, try to answer the following two questions:

- How does the performance differ with respect to the case when the additional class indicator variables are not present? Relate your observations to the observed feature ranking.
- Would it be easy to make use of the results in practice? Briefly explain your reasoning.

2 Mini Challenge [40%]

Important: *You are allowed to write up to a maximum of 600 words in this Part of the assignment. The thoroughness of the exploration and the quality of the resulting discussion is just as important as the final percentage correct result of your chosen method(s) and credit will be divided accordingly.*

In this final part of the assignment we will have a mini object-recognition challenge. Using the data provided you are asked to find the best classifier for the *person/no person* classification task. You can apply any preprocessing steps to the data that you think fit and employ any classifier you like (with the proviso that you can explain what the classifier/preprocessing steps are doing). You can also employ any lessons learnt during the course, either from previous Assignments, the Labs or the lecture material to try and squeeze out as much performance (in terms of percentage correct) as you possibly can. The only restriction is that all steps must be performed in WEKA.

We provide you with three new data sets: a training set (`train_images_partB.arff`), a validation set (`valid_images_partB.arff`), and a test set (`test_images_partB.arff`). You must use the former two for training and evaluating your models (as you see fit). Once you have chosen your favourite model (and

pre-processing steps) you should apply it to the test set (for which no labels are provided). Classify the data points in the test set and submit the classification results as part of your answer.

Your results will be evaluated in terms of the percentage correct. You also need to submit a brief description of your approach, and a short explanation of why you chose it. **The thoroughness of the exploration and the quality of the resulting discussion is just as important as the final percentage correct result of your chosen method(s) and credit will be divided accordingly.**

How to submit the results:

- Load the data set that you would like to train the your final classifier on. Choose `Supplied test set` in the `Test options` box and select `test_images_partB.arff` (or an appropriately preprocessed version thereof).
- Click on `More options` and make sure that the `Output predictions` box is checked. Then start the training and evaluation.
- When the training/evaluation is finished right click on the entry in the result list and choose `Save result buffer`. Save the result buffer as `iaml_assignment.res`. ***Important:*** *Make sure that the files is named exactly as specified: i.e.* `iaml_assignment.res` *otherwise it will not be evaluated. Do not zip or otherwise compress the file.*
- ***Important:*** *Make sure that the "Output predictions" box in the "More options" dialog is checked - otherwise the output buffer will not contain predictions for the individual data points and your results cannot be evaluated!! The result summary will not be meaningful for the test set because no labels are provided!*

Hint: *Feature engineering, feature combination, model combination and model parameter optimization can significantly improve performance.*

Hint: `Re-evaluate model on current test set`*-option which you get when you right-click on an entry in the result list might be useful for checking that your classifier is doing what it is supposed to do (it allows you to check multiple test sets with the same classifier).*

Hint: *If you train a model on one pre-processed variant of the datasets you must also validate and test it on the same pre-processed variant. For example a model trained on a training dataset that has some attributes removed will also need to be tested on a test set that has those same attributes removed. Otherwise WEKA will complain that the train and test set are not compatible. If you wish to apply nested data transformations and still have the validation and testing sets be compatible with your training dataset you can use the the meta classifier* `FilteredClassifier`*. The* `FilteredClassifier` *can be used to encapsulate an attribute selection process with a classifier. The attribute selection and the classifier are learned on the training data; any validation/test instances have the identical transformation applied before being passed to the classifier for prediction. This ensures both datasets are compatible. Multiple levels of transformation can be achieved by using the* `MultiFilter`*.*