Introductory Applied Machine Learning: Assignment 2

School of Informatics, University of Edinburgh

Instructor: Nigel Goddard
Assignment prepared by Sean J. Moran, revised by Boris Mitrovic, Nigel Goddard

For due date and time, see the course web page.
Hard copy **and** electronic submission required. The time of the deadline will be strictly enforced.
Ensure that your name and exam number DO NOT appear on the document and that your matriculation number DOES appear on it.

Plagiarism and 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.

Please remember that you should acknowledge all forms of collaboration when you submit your coursework.

We ask that you list on the assignment sheet a list of the people who you've had discussions with (if any). You WILL NOT get penalised for mentioning collaboration. However, if you collaborate and don't mention it, this may be considered plagiarism.

Plagiarism carries very harsh penalties - a lot worse than getting a 0 on the coursework. For more information, please see the Informatics Policy on plagiarism: `http://www.inf.ed.ac.uk/admin/ITO/DivisionalGuidelinesPlagiarism.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** formats are 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 4 you are also required to produce a file `iaml_assignment.res`. For the electronic submission place your report and the file in a directory called `iamlans` and submit this using the `submit` command on a DICE machine. The format is

```
submit iaml 2 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. See `http://www.inf.ed.ac.uk/teaching/years/msc/courseguide10.html#exam` for exceptions to this, e.g. in case of serious medical illness or serious personal problems.

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 **1500 words**. In Question 3 of the assignment you should not write more than 300 words in total. In Question 4 of the assignment you should not write more than 600 words in total.

(f) A discussion forum is available `https://nb.mit.edu`. Please post questions regarding the assignment on this forum. You should aim to check this forum 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 Datasets

This assignment is based on **three** different datasets. In Question 1 you will conduct further data mining tasks on the 20 Newsgroups dataset which you first became accustomed to during Assignment 1. In Questions 2 and 3 you will explore the MNIST digits dataset. Both of these datasets are frequently used in the literature to evaluate cutting edge machine learning algorithms. In Question 4 there is an object recognition mini-challenge, in which you are asked to find the best classifier you can for the determining if a person is in the images in the dataset.

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 Clustering [30%]

In Question 1 we will explore the 20 Newsgroups dataset. This is the same dataset that you used in Assignment 1, but note that we use tf-idf weights rather than frequency counts (see below for details).

20 Newsgroups Dataset

The 20 Newsgroups dataset is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups, each corresponding to a different topic. Some of the newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc.hardware, comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale, soc.religion.christian). There are three versions of the 20 Newsgroups Dataset. In this assignment we will use the bydate matlab version in which documents are sorted by date into training (60%) and test (40%) sets, newsgroup-identifying headers are dropped and duplicates are removed. This collection comprises roughly 61,000 different words, which results in a bag-of-words representation with frequency counts. More specifically, each document is represented by a 61,000 dimensional vector that contains the counts for each of the 61,000 different words present in the respective document.

To save you time and to make the problem manageable with limited computational resources, we preprocessed the original dataset. We will use documents from only 5 out of the 20 newsgroups, which results in a 5-class problem. More specifically the 5 classes correspond to the following newsgroups 1:alt.atheism, 2:comp.sys.ibm.pc.hardware, 3:comp.sys.mac.hardware, 4:rec.sport.baseball and 5:rec.sport.hockey. Additionally, we computed the mutual information of each word with the class attribute and selected the 520 words out of 61,000 that had highest mutual information. Therefore, our dataset is a N$\times$500 dimensional matrix, where N is the number of documents. The resulting representation is much more compact and can be used directly to perform our experiments in WEKA.

In contrast to Assignment 1, we have opted for tf-idf weights (term frequency - inverse document frequency) for each word instead of the frequency counts. These weights represent the importance of a word to a document with respect to a collection of documents. The importance increases proportionally to the number of times a word appears in the document and decreases proportionally to the number of times the word appears in the whole corpus. Since it is difficult to compute the tf-idf weights in WEKA, we have computed the weightings for you in the provided dataset. Therefore, for the provided dataset, each attribute represents the tf-idf weight of a word for each document and the data are normalized to unit length vectors. The tf-idf weights will be referred to as *importance weights.*

Task

We are interested in clustering the newsgroups documents using the k-means algorithm (known as `SimpleKMeans` in WEKA which can be found under the `Cluster` tab). The most common measure to evaluate the resulting clusters is the aggregate intra-cluster distance. We will additionally use the *Classes to Clusters* evaluation which is straightforward to perform in WEKA and look at the percentage of correctly clustered instances. Use the `train_20news_partA.arff` dataset and the **default seed (10)** to train the clusterers.

(a) First, train and evaluate a `SimpleKMeans` clusterer with 5 clusters (you need to change the *numClusters* option, but keep all other `SimpleKMeans` settings on default) using the *Classes to clusters evaluation* option. Look at the *Classes to Clusters* confusion matrix.

- Do the individual clusters correspond to classes? Which classes are more confused with each other? Can you explain why?

(b) Under the `Cluster` tab select `Use training set`. Now, train different `SimpleKMeans` clusterers using 2,3,4,5,6,7,8 clusters (use the **default seed (10)** and keep all other `SimpleKMeans` settings on default). Plot the within-cluster sum of squared errors (y-axis) against cluster number (x-axis) and answer the following questions:

- How many clusters would you select using this graph? Does this agree with your expectations? Explain why or why not. Include a copy of your graph with your report (ensure that the graph is appropriately labelled and the axes are scaled so that any trend is clearly visible).
- Is it safe to make this decision based on experiments with only one random seed? Why?

Hint: *The within cluster sum of squared errors value can be found close to the top of the clusterer output buffer.*

(c) Now consider the model with 5 clusters learned using k-means. After k-means converges, each cluster is described in terms of the mean for each of the 500 attributes computed from the documents assigned to the respective cluster. Since the attributes are the normalized importance weights for each word in a document, the mean vectors learned by k-means correspond to the importance weights for each word in each cluster.

For each of the 5 clusters, we selected the 20 attributes with the highest mean values. Open the file `cluster_means.txt` (this can be found on the IAML course website). The 20 attributes for each cluster are displayed column-wise together with their corresponding mean value. By looking at the words with the highest importance weights per cluster, try to answer the following questions:

- Which column (cluster) would you assign to each class (newsgroup topic) and why?
- Which two clusters are closest to each other?

2 PCA [25%]

In Questions 2 and 3 we will explore the MNIST digits dataset.

MNIST Dataset

This MNIST Dataset is a collection of 60,000 train and 10,000 test samples of handwritten digits. The samples are partitioned (nearly) evenly across the 10 different digit classes $\{0, 1, \ldots, 9\}$. Each sample is a 28×28 pixel image containing one digit. The digits are scaled to fit a 20×20 pixel box, which is then positioned in the 28×28 image by placing its centre of mass to the centre of the image. For further details on how the digits are normalized and centered to create the final images refer to the MNIST webpage[1]. The images are grayscale, with each pixel taking values in $\{0, 1, \ldots, 255\}$, where 0 corresponds to black (weakest intensity) and 255 corresponds to white (strongest intensity). Therefore, the dataset is a $N \times 784$ dimensional matrix where each dimension corresponds to a pixel from the image and N is the number of images. Again, to save you time and to make the problem manageable with limited computational resources, we have created a smaller dataset by selecting a random subset of images from the original dataset. For Question 2 of the assignment we have also reduced the number of pixels from 784 to 670, giving an $N \times 670$ dimensional matrix. This smaller dataset has been converted to the sparse arff format[2], and can be thus used directly to perform our experiments in WEKA.

Task

We expect the digits to lie in a lower-dimensional manifold and want to examine the representation we get by applying Principal Components Analysis (PCA). PCA maps the data into a new space by effectively

[1] *http://yann.lecun.com/exdb/mnist/*

[2] See `http://weka.wikispaces.com/ARFF+(stable+version)#Sparse ARFF files` for a description of the sparse arff format.

rotating the base vectors of the input space to the directions with the highest variance. We will assess the impact of this mapping to the classification task and the separability of the data in the PCA space.

Important: *All experiments should be performed on the training data using 5-fold CV and the default options unless otherwise indicated in the questions that follow. For the SVM classifier ensure that the filterType is set to No normalization/standardization (this can be found in the individual Classifier options under the Classify tab).*

(a) Load the training dataset `train_mnist_dd01_partB.arff`. Train a Naive Bayes (`NaiveBayes`) and a linear Support Vector Machine (`SMO`) using 5-fold CV. For the SVM classifier ensure that the `filterType` is set to `No normalization/standardization` (this can be found in the individual `Classifier` options under the `Classify` tab). For now, use the default settings for the classifiers (except for the `SMO filterType` option noted previously). We are not interested in optimizing their parameters, we just want to get a first idea of the dataset.

- Write down the accuracy (percent correct, PC) of the different classifiers.

Now look at the histograms of individual attributes in the bottom, right-hand corner of the `Preprocess` tab.

- Can you identify any attributes that contribute no information to the classification task? If so, explain why you think these attributes are not useful.
- Remove these attributes from the dataset. Include a short description of how you removed the attributes (including how many attributes you removed).

Now retrain the Naive Bayes and the SVM classifiers using the reduced dataset. For the SVM classifier ensure that the `filterType` is set to `No normalization/standardization` (this can be found in the individual `Classifier` options under the `Classify` tab). Report their performance and compare it to the performance before the attributes were removed.

- Do the results confirm your hypothesis about the usefulness of the specific attributes?

Important: *We now provide you with a <u>new</u> dataset (train_mnist_dd02_partB.arff). You <u>must</u> use this dataset for the remainder of the questions in this part of the assignment.*

(b) Load the dataset `train_mnist_dd02_partB.arff`. Perform PCA using the attribute evaluator `Principal Components` (in the *Select attributes* tab). Change the `varianceCovered` option to 1.0, in order to get the eigenvalues for all the eigenvectors of the input space.

- Plot the eigenvalues in descending order. What do you notice? What does this suggest? Include a copy of your graph in your report (ensure that the graph is appropriately labelled and the axes are scaled so that any trend is clearly visible).

(c) Reload the `train_mnist_dd02_partB.arff` dataset. Now go to the *Preprocess* tab and select the `AttributeSelection` filter. Click on it and change its options to `PrincipalComponents` for the evaluator and `Ranker` for search. Ensure that `varianceCovered` is set to 1.0. This filter will map our dataset into the principal components and will use as many eigenvectors as to retain 100% of the variance in the data.

- Retrain the Naive Bayes (`NaiveBayes`) and linear SVM (`SMO`) classifiers using 5-fold CV and report their performance. ***Important:*** *For the SVM classifier ensure that the filterType is set to No normalization/standardization (this can be found in the individual Classifier options under the Classify tab).*

Comparing the performance of the classifiers to that obtained in question 2a, answer the following two questions:

- Why did the performance of one classifier improve?
- Why did the performance of the other classifier remain the same?

Hint: *The correct answer to this question has absolutely nothing to do with the fact that we switched the SMO filterType to No normalization/standardization.*

3 Feature Engineering [15%]

In this Question we will again examine the MNIST digits dataset. You may write up to 300 words in this section. We introduce a new dataset (`train_mnist_binary_partC.arff`) for this section. This is a boolean/binary version of the MNIST dataset. This dataset is not comparable to that used in Question 2 of the assignment. You should therefore not relate any lessons learnt in Question 2 to your use of the dataset in this part of the assignment.

An interesting set of features that we can engineer for this dataset is the set of adjacent pixel conjunctions. Pixel conjunctions describe the concurrence of two pixels being non-zero. Each conjunction is a new attribute created by multiplying the values of two pixels. In this question we examine a dataset with conjunctions. This dataset was created from the original dataset (`train_mnist_binary_partC.arff`) as follows:

- Extract pairwise conjunctions by considering the following combinations for each pixel x in the image: $\{x, x_{right}\}, \{x, x_{down}\}, \{x, x_{right+down}\}, \{x, x_{right+up}\} \rightarrow 2,970$ new binary attributes +784 original binary attributes

Having extracted these features we provide you with the following additional dataset:

- `train_mnist_binary_pairConj_partC.arff`: contains the binary pixels and the pairwise conjunctions

Given both of these datasets, try to answer the following questions:

Important: *All experiments should be performed on the training data using 5-fold CV. For the SVM classifier ensure that the filterType is set to No normalization/standardization (this can be found in the individual Classifier options under the Classify tab).*

- Train a linear SVM (choose `PolyKernel` and set `exponent` to 1.0 in the `SMO` classifier options) using 5-fold CV on the `train_mnist_binary_pairConj_partC.arff` dataset. Keep all other settings on default except the `filterType` as noted above. Report the percentage correct performance.
- Now train an SVM with a second order polynomial kernel (choose `PolyKernel` and set `exponent` to 2.0 in the `SMO` classifier options) on the `train_mnist_binary_partC.arff` dataset using 5-fold CV. Keep all other settings on default except the `filterType` as noted above. Report the percentage correct performance.
- Can you explain the difference in performance between the linear SVM on the binarized dataset with the pairwise conjunction features and the polynomial SVM on the binarized dataset without the pairwise conjunction features? Relate your answer to the models built and the nature of the feature sets used.

4 Mini Challenge [30%]

Important: *You are allowed to write up to a maximum of* <u>*600 words*</u> *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.*

Image Dataset

In this question your 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/2015/assts/asst2/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.

We provide three data sets: a training set (`train_images_partD.arff`), a validation set (`valid_images_partD.arff`), and a test set (`test_images_partD.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.

Task

In this Question 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_partD.arff`), a validation set (`valid_images_partD.arff`), and a test set (`test_images_partD.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`*.*