Machine Learning 2016/2017, Assignment 3

Yevgeny Seldin, Christian Igel
Department of Computer Science, University of Copenhagen

The deadline for this assignment is **12:00 pm (noon, not midnight) 12/12/2016**. You must submit your individual solution electronically via the Absalon home page.

A solution consists of:

- A PDF file with detailed answers to the questions, which may include graphs and tables if needed. Do *not* include your source code in this PDF file.
- Your solution source code (Matlab / R / Python scripts or C / C++ / Java code) with comments about the major steps involved in each question (see below). Source code must be submitted in the original file format, not as PDF.
- Your code should be structured such that there is one main file that we can run to reproduce all the results presented in your report. This main file can, if you like, call other files with functions, classes, etc.
- Your code should also include a README text file describing how to compile and run your program, as well as list of all relevant libraries needed for compiling or using your code.

1 Summarization by the mean

Let's consider $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ with $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$. Prove that the solution of

$$\text{argmin}_{\boldsymbol{b}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{b}\|^2$$

is given by the empirical mean

$$\boldsymbol{b} = \overline{\boldsymbol{x}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \ .$$

Figure 1: Examples from the traffic sign data set.

That is, the mean is the best vector summarizing/representing a sample set in the least-squares sense.

Deliverables: formal proof

2 PCA for high dimensional data and small samples

The time complexity of computing the eigenvalue decomposition of an d times d matrix is cubic in d. The number d of dimensions easily gets very large, for example, if we want to analyze text documents. Let us write the data covariance matrix as

$$\boldsymbol{S} = \boldsymbol{X}_0^{\mathrm{T}} \boldsymbol{X}_0 \ , \tag{1}$$

where the $N \times d$ data matrix $\boldsymbol{X}_0$ is composed of the training data points after subtracting the mean such that the ith row corresponds to $(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^{\mathrm{T}}$.

Now let N be smaller than d. In the following, we consider how to compute the N principal components of the $d \times d$ matrix $\boldsymbol{S}$ by working with an $N \times N$ matrix.

The restriction to the first N principal components is no limitation. Basic linear algebra teaches us that the rank of $\boldsymbol{S}$ and the number of non-zero eigenvalues is not larger than N anyway.

Let us consider the $N \times N$ matrix $\boldsymbol{X}_0 \boldsymbol{X}_0^{\mathrm{T}}$. Prove that if $\boldsymbol{v}$ is an eigenvector of the $N \times N$ matrix $\boldsymbol{X}_0 \boldsymbol{X}_0^{\mathrm{T}}$ with eigenvalue λ, then $\boldsymbol{X}_0^{\mathrm{T}} \boldsymbol{v}$ is a (not normalized) eigenvector of $\boldsymbol{S} = \boldsymbol{X}_0^{\mathrm{T}} \boldsymbol{X}_0$ with the same eigenvalue.

Thus, if $N < d$ it is possible to consider the smaller eigenvalue problem of decomposing $\boldsymbol{X}_0 \boldsymbol{X}_0^{\mathrm{T}}$. The equation $\boldsymbol{u}_i = \boldsymbol{X}_0^{\mathrm{T}} \boldsymbol{v}_i$ relates the ith largest eigenvalue of the smaller matrix to the ith principal component of $\boldsymbol{S}$.

Deliverables: formal proof

3 The Traffic Sign Recognition Data

We consider a small subset of a traffic sign recognition benchmark data set [1]. Recognition of traffic signs is a challenging real-world problem of high industrial relevance. Traffic sign recognition can be viewed as a multi-class classification problem with unbalanced class frequencies, in which one has to cope with large variations in visual appearances due to illumination changes, partial occlusions, rotations, weather conditions, etc. However, humans are capable of recognizing the large variety of existing road signs with close to 100 % correctness – not only in real-world driving situations, which provides both context and multiple views of a single traffic sign, but also when looking at single images. Now the question is how good a computer can become at solving this problem.

The data set consists of traffic sign images of different sizes, see Figure 1 for examples. However, we consider already preprocessed images. The data $\mathcal{S}$ are contained in `ML2016TrafficSignsTrain.csv`.

These data have been produced by transforming the images into a reduced size, fixed length, real-valued feature vector, which is nice for machine learning. The features are histograms of oriented gradients (HOG). There are 43 classes, each class corresponds to one type of traffic sign. Reliable ground-truth data were obtained by semi-automatic annotation. We refer to a point in the data set also as an *image*. The images were generated in the following way: Videos (*tracks*) were shot from a driving car and the traffic signs were extracted from these videos. This implies that the data set contains several images of the same *physical traffic sign* taken from subsequent frames of the video.

Each line in `ML2016TrafficSignsTrain.csv` corresponds to the feature of one image plus the corresponding label. The last column in a line indicates the label.

Note that the data requires a considerable amount of disk space and that some of the algorithms, in particular the PCA, require a considerable amount of time when applied to the full data set. It should be possible to process the data on a standard laptop, but if you for some reason run into computation time problems you can subsample the data. If you do so, please, state it clearly in your report and provide a script that you have executed (that subsamples the data) and a separate script that can be run on the full dataset.

3.1 Data understanding and preprocessing

Plot a histogram showing the distribution of class frequencies (i.e., for each of the 43 traffic signs plot the number of observations in the training data set divided by the total number of observations in the training data set).

Deliverables: description of software used; single histogram plot

3.2 Principal component analysis

Perform a principal component analysis of $\mathcal{S}$. Plot the eigenspectrum. How many components are necessary to "explain 90 % of the variance"? Visualize the data by a scatter plot of the first two principal components. Use different colors for the different shapes in the plot. Table 1 provides the mapping from class index to shape.

Deliverables: description of software used; plot of the eigenspectrum; number of components necessary to explain 90 % of variance; scatter plot for first two principal components with different colors indicating the five different shapes

3.3 Clustering

Perform 4-means clustering of $\mathcal{S}$. For the submission, please, initialize the cluster centers with the first four data points in $\mathcal{S}$ (when you are playing around with the data you are welcome to try other initializations as well; the "Learning from data" book describes some fancy heuristics). Plot the cluster centers projected to the first two principal components. That is, add the cluster centers to the plot from the previous Q. Briefly discuss the results: Did you get meaningful clusters?

Deliverables: description of software used; one plot with cluster centers; explain how you projected the high dimensional cluster centers down to two dimensions; short discussion of results

References

[1] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012.

Table 1: Mapping from class number to shape of the encoded traffic sign.

class index	shape
00000	Round
00001	Round
00002	Round
00003	Round
00004	Round
00005	Round
00006	Round
00007	Round
00008	Round
00009	Round
00010	Round
00011	Upwards pointing triangle
00012	Diamond
00013	Downwards pointing triangle
00014	Octagon
00015	Round
00016	Round
00017	Round
00018	Upwards pointing triangle
00019	Upwards pointing triangle
00020	Upwards pointing triangle
00021	Upwards pointing triangle
00022	Upwards pointing triangle
00023	Upwards pointing triangle
00024	Upwards pointing triangle
00025	Upwards pointing triangle
00026	Upwards pointing triangle
00027	Upwards pointing triangle
00028	Upwards pointing triangle
00029	Upwards pointing triangle
00030	Upwards pointing triangle
00031	Upwards pointing triangle
00032	Round
00033	Round
00034	Round
00035	Round
00036	Round
00037	Round
00038	Round
00039	Round
00040	Round
00041	Round
00042	Round