Machine Learning Fall 2017

Final Graded Exam

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

You must submit your individual solution of the exam electronically via the **Digital Exam / Digital Eksamen** system. The deadline for submitting the exam is **16:00, 26 January 2018**. The exam must be solved **individually**. You are **not allowed** to work in groups or discuss the exam questions with other students. For fairness reasons any questions about the exam must be posted on the Absalon forum.

WARNING: The goal of the exam is to evaluate your personal achievements in the course. We believe that take-home exams are most suitable for this evaluation, because they allow to test both the theoretical and practical skills. However, our ability to give take-home exams strongly depends on your honesty. Therefore, any suspicion of cheating, in particular collaboration with other students, will be directly reported to the head of studies and prosecuted in the strictest possible way. Be aware that if proven guilty you may be expelled from the university. Do not put yourself and your fellow students at risk.

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 Contradiction between Upper and Lower Bound

This is a multiple-choice question. Correct answers will give you positive points and incorrect answers will give you negative points. If you are unsure about an answer it may be better to skip the question (or write "I do not know") rather than make a blind and potentially incorrect guess.

In the course we have studied the Occam's razor bound, which provides a high-probability upper bound on the expected loss of all classifiers h within a countably infinite hypothesis class $\mathcal{H}$ (Theorem 3.3 in Yevgeny's lecture notes). We have also studied the VC lower bound, which states that if the VC dimension of a hypothesis class $\mathcal{H}$ is infinite, then it is impossible to obtain a high-probability upper bound smaller than 0.125 that holds for all classifiers h (Corollary 3.11 of Theorem 3.10; if you cannot follow the formal argument in Theorem 3.10 it is sufficient to think about the intuitive explanation we gave in class). The two results may sound contradictory. For each of the following statements write whether they are true or false.

1. The VC dimension is only defined for uncountably infinite hypothesis classes and, therefore, there is no contradiction between the two results.

2. The Occam's razor bound may be larger than 0.125 for some hypotheses in $\mathcal{H}$ and, therefore, there is no contradiction between the two results.

3. The Occam's razor bound holds for any data distribution $p(X, Y)$, whereas the data distribution in the construction of the lower bound was picked in a very specific way. Therefore, there is no contradiction between the two results.

4. It is possible to construct a data distribution $p(X, Y)$ and a hypothesis class $\mathcal{H}$ with infinite VC-dimension, such that for any sample S of more than 100 points with probability at least 0.95 we will have $L(h) \leq \hat{L}(h, S) + 0.01$ for all h in $\mathcal{H}$.

2 Validation

You have to estimate the expected errors of 100 classification rules, so that with probability at least 99% all the estimates will be accurate up to 0.01. What is the minimal size of a sample you have to collect to do the estimation? Provide a complete derivation, answers without derivations will not be accepted.

Figure 1: Examples from different datasets (just one is used in this assignment). Left to right: outfielder team 1/2, goalkeeper 1/2, referees, group, error.

3 Kick it!

Application Background

Computer vision and image analysis are becoming more and more important in sports analytics, the science of analyzing and modeling processes underlying sporting events. Sports with a high media coverage create a demand for systematic review and objective evaluation of the performance of individual athletes as well as of teams. Across almost all sports, management and coaches make use of statistics and categorized video material to support their strategies.

In this assignment, we consider classification of the protagonists in soccer. A pattern detection algorithm has extracted regions of interest (ROIs) from a live video recording. Now each ROI has to be classified into one out of seven main classes. We have the obvious five classes outfielder (two classes, one for each team), goalkeeper (two classes) and referees. Further, there is a category that groups ROIs that contain at least one outfielder from each team. Finally, there is a class for irrelevant objects (i.e., false detections by the ROI detection algorithm). Sample ROIs are shown in Figure 1, the classes are summarized with the corresponding class index (label) in Table 1.

Because teams and referees can be identified based on the color of the clothing, color histogram features are extracted from the ROIs. The color model is HSV (hue, saturation, value), considering 3, 3, and 2 bits per channel, respectively. The data acquisition and preprocessing is described in more detail by Schlipsing et al. (2017).

Table 1: Mapping from class number to class.

class index	class
0	player team 1
1	player team 2
2	goalkeeper team 1
3	goalkeeper team 2
4	referee
5	group
6	error

The Exercises

Question 1 (data understanding and preprocessing)**.** Download and extract the data.

Consider the training data `trainInput.csv` and the corresponding labels `trainTarget.csv`. Report the class frequencies, that is, for each of the 7 classes report the number of data points divided by the total number of data points) in the training data.

The ith row in `trainInput.csv` are the features of the ith training pattern. The class label of the ith pattern is given in the ith row of `trainTarget.csv`.

Deliverables: description of software used; frequency of classes

Question 2 (principal component analysis)**.** Perform a principal component analysis of the training data `trainInput.csv`. Plot the eigenspectrum (see the plot on slide 28 of the *PCA* slides for an example). How many components are necessary to "explain 90 % of the variance"? Visualize the data by a scatter plot of the data projected on the first two principal components. Use different colors for the different classes in the plot.

Deliverables: description of software used; plot of the eigenspectrum; number of components necessary to explain 90 % of variance; scatter plot of the data projected on the first two principal components with different colors indicating the 7 different classes

Question 3 (clustering)**.** Perform 7-means clustering of `trainInput.csv`. After that, project the cluster centers to the first two principal components of the training data. Then visualize the clusters by adding the cluster centers to the plot from the previous exercise. Briefly discuss the results: Did you get meaningful clusters? Initialize the cluster centers with training data points from different classes. Take the *first data point* from each class you can find in `trainInput.csv` (i.e., the first class center is the data point in the very first line in the file).

Deliverables: description of software used; one plot with cluster centers and data points; short discussion of results

Question 4 (Overfitting)**.** John Langford, who is "Doctor of Learning at Microsoft Research", maintains a very interesting blog (web log). Read the very true blog entry: "Clever methods of overfitting," `http://hunch.net/?p=22`, 2005.[1]

Choose three of the different types of overfitting and discuss if and how they can occur when applying machine learning techniques to the sport analytics task. Ignore the last type of overfitting and issues related to reviewing of scientific papers (still, it is good to keep them in mind).

Deliverables: Short discussion addressing three "methods of overfitting" listed in the blog entry

Question 5 (multi-class classification)**.** Use a non-linear classification method (picking from the methods presented in the course) for classifying the 7 image classes. Use `trainInput.csv` and `trainTarget.csv` for training. After you trained a model, use the test data in `testInput.csv` and `testTarget.csv` to evaluate it. Report the classification error on both training and test set.

Deliverables: description of software used; arguments for your choice of classification methods; a short description of how you proceeded and what training and test results you achieved

Question 6 (binary classification using support vector machines)**.** Now we consider binary classification using support vector machines (SVMs). To this end, we reduce the problem to distinguishing between referees and regions not containing persons of interest. Please use the data files `trainInputBinary.csv`, `trainTargetBinary.csv`, `testInputBinary.csv`, and `testTargetBinary.csv`. These are real-world data. The splitting into training and testing data has been done by the people providing the images. The class frequencies in training and testing data set differ considerably. This could be bad luck, but could also indicate a sampling bias (i.e., a violation of the i.i.d. assumption). The differences between training and test set make the problem difficult for some learning algorithms. Please ignore this phenomenon in the assignment, that is, do *not* use asymmetric loss functions, class-dependent regularization parameters (i.e., different "C-values" depending on the class), etc.[2]

For this exercise, use standard C-SVMs as introduced in the lecture. Employ radial Gaussian kernels of the form

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{z}\|^2) \ . \tag{1}$$

[1] If the link is not working, try: `http://www.kdnuggets.com/2015/01/clever-methods-overfitting-avoid.html`

[2] Be careful, because, for instance, the SVM from the Matlab Bioinformatics Toolbox may by default use different regularization parameters depending on the class and the class frequency – at least that was the case in the past.

Here $\gamma > 0$ is a bandwidth parameter that has to be chosen in the model selection process. Note that instead of γ often the parameter $\sigma = \sqrt{1/(2\gamma)}$ is considered.

Jaakkola's heuristic provides a reasonable initial guess for the bandwidth parameter σ or γ of a Gaussian kernel. To use Jaakkola's heuristic to estimate a good value for σ, consider for every training example $\boldsymbol{x}_i$ the distance to the closest training example $\boldsymbol{x}_j$ having a different label (i.e., $y_i \neq y_j$). The median of these distances can be used as a measure of scale and therefore as a guess for σ. More formally, compute

$$G = \left\{ \min_{(\boldsymbol{x}_j, y_j) \in S \wedge y_i \neq y_j} \{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\} \mid (\boldsymbol{x}_i, y_i) \in S \right\}$$

based on your training data S. Then set σ_{Jaakkola} equal to the median of the values in G:

$$\sigma_{\text{Jaakkola}} = \text{median}(G)$$

Compute the bandwidth parameter γ_{Jaakkola} from σ_{Jaakkola} using the identity given above.

Use grid-search to determine appropriate SVM hyperparameters γ and C. Look at all combinations of

$$C \in \{b^{-1}, 1, b, b^2, b^3\}$$

and

$$\gamma \in \left\{ \gamma_{\text{Jaakkola}} \cdot b^i \mid i \in \{-3, -2, -1, 0, 1, 2, 3\} \right\} \ ,$$

where the base b can be chosen to be either 2, the base e of the natural logarithm (Euler's number), or 10. Feel free to vary this grid. For each pair, estimate the performance of the SVM using 5-fold cross validation. Pick the hyperparameter pair with the lowest average 0-1 loss (classification error) and use it for training an SVM with the complete training dataset. Only use `trainInputBinary.csv` and `trainTargetBinary.csv` in the model selection and training process.

Report the values for C and γ you found in the models selection process. Compute the classification accuracy based on the 0-1 loss on the training data as well as on the test data `testInputBinary.csv` and `testTargetBinary.csv`. An accuracy on the test set significantly larger than 75 % can be expected.

Deliverables: description of software used; a short description of how you proceeded; initial γ or σ value suggested by Jaakkola's heuristic; optimal C and γ found by grid search; classification accuracy on training and test data

4 Decision Trees and Random Forests

Decision trees and random forests are important learning models that we have studied in the course. In this question we develop some theoretical understanding of this model.

Figure 2: An example of a decision tree with 3 leaves.

1. We start with the simplest example of one-dimensional data ($\mathcal{X} = \mathbb{R}$) and the class of one-level decision trees, also known as decision stumps. A decision stump $h_{\theta,y}$ is defined by a parameter θ and a label $y \in \{\pm 1\}$, so that if $x \geq \theta$ then $h_{\theta,y}(x) = y$ and otherwise $h_{\theta,y}(x) = -y$. What is the VC-dimension of decision stumps in one dimension? Prove your answer. (You have to provide both upper and lower bound.)

2. Now consider two-dimensional data ($\mathcal{X} = \mathbb{R}^2$) and the class $\mathcal{H}$ of one-level decision trees, which are allowed to pick any one out of the two parameters for making a decision. Show that $d_{VC}(\mathcal{H}) \geq 3$.

3. Let $\mathcal{H}$ be the class of two-level incomplete decision trees with three leaves (one leaf at the first level and two leaves at the second level, see Figure 2 for an example) and $\mathcal{X} = \mathbb{R}$ (we are back to one-dimensional input). Provide a lower bound on $d_{VC}(\mathcal{H})$. Justify your answer.

4. Let $d_{k,m}$ be the VC-dimension of a hypothesis class $\mathcal{H}_{k,m}$ of decision trees with k leaves for m-dimensional input data.

   (a) Write down a bound on $L(h)$ in terms of $d_{k,m}$ and other necessary quantities that holds for all $h \in \mathcal{H}_{k,m}$ with probability at least $1 - \delta$. (You do not have to calculate $d_{k,m}$.)

   (b) What terms in the bound are non-decreasing when the size of the trees k grows? Which terms are non-increasing? Justify your answer. (You should only discuss terms that depend on k.)

   (c) What terms in the bound are non-decreasing when the input dimension m grows? Which terms are non-increasing? Justify your answer. (You should only discuss terms that depend on m.)

5. Calculation of the VC dimension $d_{k,m}$ of decision trees is not an easy task. Instead, we develop an alternative way of assessing the performance of decision trees and random forests. In a typical way of training a random forest, the individual trees in the forest are trained on random subsamples of the data. Let S be a training set of size n, where n is an even number. Lets assume that each tree is trained on $n/2$ randomly selected points from S (without replacement). Let B denote the number of trees in the forest.

Let S_i for $i \in \{1, \ldots, B\}$ denote the subset of points used to train the i-th tree and $\bar{S}_i = S \setminus S_i$ the subset of remaining points.

(a) Provide a bound on the expected loss of the first tree, $L(h_1)$, in terms of $\hat{L}(h_1, \bar{S}_1)$. The bound should hold with probability at least $1 - \delta$. Justify your answer.

(b) Derive a bound on $L(h_i)$ in terms of $\hat{L}(h_i, \bar{S}_i)$ that holds for all trees $h_1, \ldots, h_B$ simultaneously with probability at least $1 - \delta$. Hint: even though there is dependence between loss estimates for different trees, the union bound holds even when applied to dependent events.

(c) A random forest predicts by taking a majority vote of the predictions of the individual trees. Let $L(\mathrm{RF})$ denote the expected loss a random forest RF. Show that $L(\mathrm{RF}) \leq \frac{2}{B} \sum_{i=1}^{B} L(h_i)$. Hint: the majority vote errs when at least half of the trees make an error. You should turn this observation into a formal argument. Consider losses on individual examples, $\ell(\mathrm{RF}(X), Y)$, and bound $\ell(\mathrm{RF}(X), Y)$ in terms of the losses $\ell(h_i(X), Y)$ in the case when $\ell(\mathrm{RF}(X), Y) = 0$ and $\ell(\mathrm{RF}(X), Y) = 1$.

(d) Derive a bound on $L(\mathrm{RF})$ in terms of the observed losses $\hat{L}(h_i, \bar{S}_i)$. The bound should hold with probability at least $1 - \delta$. Remark: the bound that you obtain in this question is likely to give non-trivial generalization guarantees in practice.

5 Rotation of the inputs

If we use PCA for preprocessing, we rotate the input to our machine learning algorithm. How does rotation affect different methods? Consider supervised learning for classification. Let the input space be $\mathbb{R}^d$. Now consider the transformation of the input data by a rotation matrix $\boldsymbol{R} \in \mathbb{R}^{d \times d}$, which only rotates the data. That is, instead of $\boldsymbol{x}$ the algorithm is provided $\boldsymbol{R}\boldsymbol{x}$. The transformation is applied to train and test data.

In mathematical terms, we say that $\boldsymbol{R}$ is a member of the d-dimensional *special orthogonal group* and write $\boldsymbol{R} \in \mathrm{SO}(d)$. The formal definition of a rotation matrix is that $\boldsymbol{R}$ is a rotation matrix if and only if $\boldsymbol{R}^{\mathrm{T}} = \boldsymbol{R}^{-1}$ and $\det \boldsymbol{R} = 1$.

Which classification methods are affected by the transformation in the sense that their classification performance may change if trained and tested on the transformed data compared to the original data? Consider three classifiers:

1. k-nearest neighbor algorithm with standard Euclidean metric
2. Support vector machine with radial Gaussian kernel as used for binary classification earlier in this exam, see Eq. (1)

3. Random forest

For each of them, provide a proof either showing that the classification performance will not change or may change. For the former, you may ignore numerical issues. For the latter, note that it is sufficient to give a single toy example where the classification changes.

References

M. Schlipsing, J. Salmen, M. P. Tschentscher, and C. Igel. Adaptive pattern recognition in real-time video-based soccer analysis. *Journal of Real-Time Image Processing*, 13(2):345–361, 2017.