Supervised Learning (COMP0078) – Coursework 2

Due : 03 January 2024.

Submission

You may work in groups of up to two. You should produce a report (this should be in .pdf format) about your results. You will not only be assessed on the **correctness/quality** of your answers but also on **clarity of presentation**. Additionally make sure that your code is *well commented*. Please submit on moodle i) your report as well as a ii) zip file with your source code. Finally, please ensure that if you are working in a group both of your student ids are on the coversheet. Regarding the use of libraries, you should implement regression using matrix algebra directly. Likewise any machine learning routines such as for example cross-validation should be implemented directly. Otherwise libraries are okay.

Note. Each coursework part is divided in a series of sub-parts. If you believe you are not able to prove an individual sub-part (e.g. 1.1), do not give up on the subsequent ones (e.g. 1.2)! In these cases, you are allowed to assume the results in the previous sub-parts to be true, even if you were not able to prove them.

1 PART I [20%]

Rademacher Complexity of finite Spaces

In this problem we will find an upper bound on the Rademacher complexity of finite space of hypotheses that depends only logarithmically on the cardinality of the space. This will enable us to improve the bound on the generalization error seen in class for finite spaces of Hypotheses.

We will first show an intermediate result: for any collection $X_1, \dots, X_m$ of centered random variables (namely $\mathbb{E}X_i = 0$ for all $i = 1, \dots, m$) taking values in $[a, b] \subset \Re$, we will show that

$$\mathbb{E} \max_{i=1,\dots,m} X_i \leq \frac{b-a}{2}\sqrt{2\log(m)}$$

1.1 [2 marks]. Let $\bar{X} = \max_i X_i$. Show that for any $\lambda > 0$

$$\mathbb{E}\bar{X} \leq \frac{1}{\lambda}\log \mathbb{E}e^{\lambda \bar{X}}$$

1.2 [5 marks]. Show that

$$\frac{1}{\lambda}\log \mathbb{E}e^{\lambda \bar{X}} \leq \frac{1}{\lambda}\log m + \lambda\frac{(b-a)^2}{8}$$

Hint: use ***Hoeffding's Lemma:*** *for any random variable X such that $X - \mathbb{E}X \in [a, b]$ with $a, b \in \Re$, and for any $\lambda > 0$, we have*

$$\mathbb{E}\ e^{\lambda(X-\mathbb{E}X)} \leq e^{\lambda^2(b-a)^2/8}$$

1.3 [3 marks]. Conclude that by choosing λ appropriately,

$$\mathbb{E} \max_{i=1,\dots,m} X_i \leq \frac{b-a}{2}\sqrt{2\log(m)}$$

as desired.

We are almost ready to provide the bound for the Rademacher complexity of a finite set of hypotheses. Let S a fomote set of points in $\Re^n$ with cardinality $|S| = m$. We can define the Rademacher complexity of S similarly to how we have done for the Rademacher complexity of a space of hypotheses:

$$\mathcal{R}(S) = \mathbb{E}_\sigma \max_{x \in S} \frac{1}{n} \sum_{j=1}^{n} \sigma_j x_j,$$

With $\sigma_1, \ldots, \sigma_n$ Rademacher variables (independent and uniformly sampled from $\{-1, 1\}$.

1.4 [3 marks]. Show that

$$\mathcal{R}(S) \leq \max_{x \in S} \|x\|_2 \frac{\sqrt{2\log(m)}}{n}$$

with $\|\cdot\|_2$ denoting the Euclidean norm.

1.5 [7 marks]. Let $\mathcal{H}$ be a set of hypotheses $f : \mathcal{X} \to \Re$. Assume $\mathcal{H}$ to have finite cardinality $|\mathcal{H}| < +\infty$. Let $S = (x_i)_{i=1}^n$ be a set of points in $\mathcal{X}$ an input set. Use the reasoning above to prove an upper bound for empirical Rademacher complexity $\mathcal{R}_S(\mathcal{H})$, where the cardinality of $\mathcal{H}$ appears logarithmically.

2 PART II [40%]

Bayes Decision Rule and Surrogate Approaches

In (binary) classification problems the classification or "decision" rule is a binary valued function $c : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y} = \{1, -1\}$. The quality of a classification rule can be measured by the misclassification error

$$R(c) = \mathbb{P}_{(x,y)\sim\rho}(c(x) \neq y)$$

assuming to sample an input-output pair (x, y) according to a distribution ρ on $\mathcal{X} \times \mathcal{Y}$.

2.1 [4 marks]. Let $\mathbf{1}_{y=y'}$ be the 0-1 loss such that $\mathbf{1}_{y=y'} = 1$ if $y \neq y'$ and 0 otherwise. Show that the misclassification error corresponds to the *expected risk* of the 0-1 loss, namely

$$R(c) = \int_{\mathcal{X}\times\mathcal{Y}} \mathbf{1}_{c(x)\neq y} \, d\rho(x, y)$$

(Surrogate Approaches) Since the 0-1 loss is not continuous, it is typically hard to address the learning problem directly and in practice one usually looks for a real valued function $f : \mathcal{X} \to \Re$ solving a so-called *surrogate problem*

$$\mathcal{E}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(f(x), y) \, d\rho(x, y)$$

where $\ell : \Re \times \Re \to \Re$ is a "suitable" convex loss function that makes the surrogate learning problem more amenable to computations. Given a function $f : \mathcal{X} \to \Re$, a classification rule $c_f : \mathcal{X} \to \{-1, 1\}$ is given in terms of a "suitable" map $d : \Re \to \{-1, 1\}$ such that $c_f(x) = d(f(x))$ for all $x \in \mathcal{X}$. Here we will look at some surrogate frameworks.

A good surrogate method satisfies the following two properties:

> **(Fisher Consistency)**. Let $f_* : \mathcal{X} \to \Re$ denote the expected risk minimizer for $\mathcal{E}(f_*) = \inf_{f:\mathcal{X}\to\Re} \mathcal{E}(f)$, we say that the surrogate framework is Fisher consistent if
>
> $$R(c_{f_*}) = \inf_{c:\mathcal{X}\to\{-1,1\}} R(c)$$

(Comparison Inequality). The surrogate framework satisfies as *comparison inequality* if for any $f : \mathcal{X} \to \Re$

$$R(c_f) - R(c_{f_*}) \leq \sqrt{\mathcal{E}(f) - \mathcal{E}(f_*)}$$

In particular, if we have an algorithm producing estimators f_n for the surrogate problem such that $\mathcal{E}(f_n) \to \mathcal{E}(f_*)$ for $n \to +\infty$, we automatically have $R(c_{f_n}) \to R(c_{f_*})$.

2.2 [4 marks]. (**Assuming to know ρ**), calculate the closed-form of the minimizer f_* of $\mathcal{E}(f)$ for the:

- **a)** squared loss $\ell(f(x), y) = (f(x) - y)^2$
- **b)** exponential loss $\ell(f(x), y) = \exp(-yf(x))$,
- **c)** logistic loss $\ell(f(x), y) = \log(1 + \exp(-yf(x)))$,
- **d)** hinge loss $\ell(f(x), y) = \max(0, 1 - yf(x))$.

(hint: recall that $\rho(x, y) = \rho(y|x)\rho_{\mathcal{X}}(x)$ with $\rho_{\mathcal{X}}$ the marginal distribution of ρ on $\mathcal{X}$ and $\rho(y|x)$ the corresponding conditional distribution. Write the expected risk as

$$\mathcal{E}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(f(x), y) d\rho(x, y) = \int_{\mathcal{X}} \left(\int_{\mathcal{Y}} \ell(f(x), y) \; d\rho(y|x) \right) \; d\rho_{\mathcal{X}}(x)$$

you can now solve the problem in the inner integral point-wise $\forall x \in \mathcal{X}$).

2.3 [4 marks]. The minimizer c_* of $R(c)$ over all possible decision rules $c : \mathcal{X} \to \{-1, 1\}$ is called *Bayes decision rule*. Write explicitly the Bayes decision rule (again **Assuming ρ known a priori**).

2.4 [4 marks]. Are the surrogate frameworks in problem (2.2) Fisher consistent? Namely, can you find a map $d : \Re \to \{-1, 1\}$ such that $R(c_*(x)) = R(d(f_*(x)))$ where f_* is the corresponding minimizer of the surrogate risk $\mathcal{E}$? If it is the case, write d explicitly.

(Comparison Inequality for Least Square Surrogates) Let $f_* : \mathcal{X} \to \Re$ be the minimizer of the expected risk for the surrogate least squares classification problem obtained in problem (2.2). Let sign : $\Re \to \{-1, 1\}$ denote the "sign" function

$$\text{sign}(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases} .$$

Prove the following comparison inequality

$$0 \leq R(\text{sign}(f)) - R(\text{sign}(f_*)) \leq \sqrt{\mathcal{E}(f) - \mathcal{E}(f_*)},$$

by showing the following intermediate steps:

2.5.1 [8 marks]. $|R(\text{sign}(f)) - R(\text{sign}(f_*))| = \int_{\mathcal{X}_f} |f_*(x)| d\rho_{\mathcal{X}}(x)$,
Where $\mathcal{X}_f = \{x \in \mathcal{X} \mid \text{sign}(f(x)) \neq \text{sign}(f_*(x))\}$.

2.5.2 [8 marks]. $\int_{\mathcal{X}_f} |f_*(x)| d\rho_{\mathcal{X}}(x) \leq \int_{\mathcal{X}_f} |f_*(x) - f(x)| d\rho_{\mathcal{X}}(x) \leq \sqrt{\mathbb{E}(|f(x) - f_*(x)|^2)}$.
Where $\mathbb{E}$ denotes the expectation with respect to $\rho_{\mathcal{X}}$

2.5.3 [8 marks]. $\mathcal{E}(f) - \mathcal{E}(f_*) = \mathbb{E}(|f(x) - f_*(x)|^2)$.

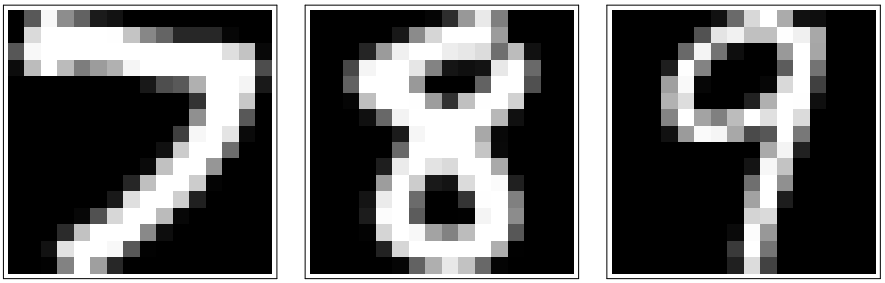

Figure 1: Scanned Digits

3 PART III [40%]

Kernel perceptron (Handwritten Digit Classification)

Introduction: In this exercise you will train a classifier to recognize hand written digits. The task is quasi-realistic and you will (perhaps) encounter difficulties in working with a moderately large dataset which you would not encounter on a "toy" problem.

You may already be familiar with the perceptron, this exercise generalizes the perceptron in two ways, first we generalize the perceptron to use *kernel* functions so that we may generate a nonlinear separating surface and second, we generalize the perceptron into a majority network of perceptrons so that instead of separating only two classes we may separate k classes.

Adding a kernel: The *kernel* allows one to map the data to a higher dimensional space as we did with basis functions so that class of functions learned is larger than simply linear functions. We will consider a single type of kernel, the polynomial $K_d(\boldsymbol{p}, \boldsymbol{q}) = (\boldsymbol{p} \cdot \boldsymbol{q})^d$ which is parameterized by a positive integer d controlling the dimension of the polynomial.

Training and testing the kernel perceptron: The algorithm is *online* that is the algorithms operate on a single example $(\boldsymbol{x}_t, y_t)$ at a time. As may be observed from the update equation a single kernel function $K(\boldsymbol{x}_t, \cdot)$ is added for each example scaled by the term α_t (may be zero). In online training we repeatedly cycle through the training set; each cycle is known as an *epoch.* When the classifier is no longer changing when we cycle thru the training set, we say that it has converged. It may be the case for some datasets that the classifier never converges or it may be the case that the classifier will *generalize* better if not trained to convergence, for this exercise the choice is left to you to decide how many epochs to train a particular classifier (alternately you may research and choose a method for converting an online algorithm to a batch algorithm and use that conversion method). The algorithm given in the table correctly describes training for a single pass through the data (*1st epoch*). The algorithm is still correct for multiple epochs, however, explicit notation is not given. Rather, latter epochs (additional passes thru the data) is represented by repeating the dataset with the $\boldsymbol{x}_i$'s renumbered. I.e., suppose we have a 40 element training set $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_{40}, y_{40})\}$ to model additional epochs simply extend the data by duplication, hence an m epoch dataset is

$$\underbrace{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{40}, y_{40})}_{\text{epoch 1}}, \underbrace{(\boldsymbol{x}_{41}, y_{41}), \ldots, (\boldsymbol{x}_{80}, y_{80})}_{\text{epoch 2}}, \ldots, \underbrace{(\boldsymbol{x}_{(m-1)\times 40+1}, y_{(m-1)\times 40+1}), \ldots, (\boldsymbol{x}_{(m-1)\times 40+40}, y_{(m-1)\times 40+40})}_{\text{epoch m}}$$

where $\boldsymbol{x}_1 = \boldsymbol{x}_{41} = \boldsymbol{x}_{81} = \ldots = \boldsymbol{x}_{(m-1)\times 40+1}$, etc. Testing is performed as follows, once we have trained a classifier $\boldsymbol{w}$ on the training set, we simply use the trained classifier with only the *prediction* step for each example in test set. It is a mistake when ever the prediction $\hat{y}_t$ does not match the desired output y_t, thus the test error is simply the number of mistakes divided by test set size. Remember in testing the *update* step is never performed.

	Two Class Kernel Perceptron (training)
Input:	$\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\Re^n, \{-1, +1\})^m$
Initialization:	$\boldsymbol{w}_1 = \mathbf{0}$ $(\alpha_0 = 0)$
Prediction:	Upon receiving the tth instance $\boldsymbol{x}_t$, predict $\hat{y}_t = \text{sign}(\boldsymbol{w}_t(\boldsymbol{x}_t)) = \text{sign}(\sum_{i=0}^{t-1} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_t))$
Update:	`if` $\hat{y}_t = y_t$ `then` $\alpha_t = 0$ `else` $\alpha_t = y_t$ $\mathbf{w}_{t+1}(\cdot) = \mathbf{w}_t(\cdot) + \alpha_t K(\boldsymbol{x}_t, \cdot)$

Generalizing to k classes: Design a method (or research a method) to generalise your two-class classifier to k classes. The method should return a vector $\boldsymbol{\kappa} \in \Re^k$ where κ_i is the "confidence" in label i; then you should predict either with a label that maximises confidence or alternately with a randomised scheme.

I'm providing you with mathematica code for a 3-classifier and a demonstration on a small subset of the data. First, however, my mathematica implementation is flawed and is relatively inefficient for large datasets. One aspect of your goals are to improve my code so that it can work on larger datasets. The mathematical logic of the algorithm should not change, however either the program logic and/or the data structures will need to change. Also, I suspect that it will be considerable easier to implement sufficiently fast code in Python (or the language of your choice) rather than Mathematica.

Files: From `http://www0.cs.ucl.ac.uk/staff/M.Herbster/SL/misc/`, you will find files relevant to this assignment. These are,

`poorCodeDemoDig.nb`	demo mathematica code
`dtrain123.dat`	mini training set with only digits 1,2,3 (329 records)
`dtest123.dat`	mini testing set with only digits 1,2,3 (456 records)
`zipcombo.dat`	full data set with all digits (9298 records)

each of the data files consists of records (lines) each record (line) contains 257 values, the first value is the digit, the remaining 256 values represent a 16×16 matrix of grey values scaled between -1 and 1. In attempting to understand the algorithms you may find it valuable to study the mathematica code. However, remember the demo code is partial (it does not address model selection, and though less efficient implementations are possible it is not particularly efficient.) Improving the code may require thought and observation of behaviour on the given data, there are many distinct types of implementations for the kernel perceptron.

Experimental Protocol: Your report on the main results should contain the following (errors reported should be percentages not raw totals):

1. Basic Results: Perform 20 runs for $d = 1, \ldots, 7$ each run should randomly split `zipcombo` into 80% train and 20% test. Report the mean test and train error rates as well as well as standard deviations. Thus your data table, here, will be 2×7 with each "cell" containing a mean$\pm$std.

2. Cross-validation: Perform 20 runs : when using the 80% training data split from within to perform 5-fold cross-validation to select the "best" parameter d^* then retrain on full 80% training set using d^* and then record the test errors on the remaining 20%. Thus you will find 20 d^* and 20 test errors. Your final result will consist of a mean test error$\pm$std and a mean d^* with std.

3. Confusion matrix: Perform 20 runs : when using the 80% training data split that further to perform 5-fold cross-validation to select the "best" parameter d^* retrain on the full "80%" training set using d^* and then produce a *confusion matrix*. Here the goal is to find "confusions" thus if the true label (on the test set) was "7" and "2" was predicted then a "error" should recorded for "(7,2)"; the final output will be a 10×10 matrix where each cell contains a confusion error *rate* and its standard deviation (here you will have averaged over the 20 runs). Note the diagonal will be 0. In computing the error rate for a cell use

$$\frac{\text{"Number of times digit } a \text{ was mistaken for digit } b \text{ (test set)"}}{\text{"Number of digit } a \text{ points (test set)"}}.$$

4. Within the dataset relative to your experiments there will be five hardest to predict correctly "pixelated images." Print out the visualisation of these five digits along with their labels. Is it surprising that these are hard to predict? Explain why in your opinion that is the case.

5. Repeat 1 and 2 (d^* is now c and $\{1, \ldots, 7\}$ is now S) above with a Gaussian kernel

$$K(\boldsymbol{p}, \boldsymbol{q}) = e^{-c\|\boldsymbol{p}-\boldsymbol{q}\|^2},$$

   c the width of the kernel is now a parameter which must be optimised during cross-validation however, you will also need to perform some initial experiments to a decide a reasonable set S of values to cross-validate c over.

6. Choose (research) an alternate method to generalise the kernel perceptron to k-classes then repeat 1 and 2.

Assessment: In your report you will not only be assessed on the correctness/quality of your experiment (e.g., sound methods for choosing parameters, reasonable final test errors) but also on the clarity of presentation and the insightfulness of your observations. Thus the aim is that your report is sufficiently detailed so that the reader could largely repeat your experiments based on the description in your report alone. The report should also contain the following.

- A discussion of any parameters of your method which were not cross-validated over.
- A discussion of the two methods chosen for generalising 2-class classifiers to k-class classifiers.
- A discussion comparing results of the Gaussian to the polynomial Kernel.
- A discussion of your implementation of the kernel perceptron. This should at least discuss how the sum $\boldsymbol{w}(\cdot) = \sum_{i=0}^{m} \alpha_i K(\boldsymbol{x}_i, \cdot)$ was i) represented, ii) evaluated and iii) how new terms are added to the sum during training.
- Any table produced in 1-6 above should also have at least one sentence discussing the table.

Note: (further comments on assessment) : Your score in Part I is not the "percentage correct," rather it will be a qualitative judgement of the report's *scientific excellence*. Thus a report that is merely correct/good as a baseline can expect 24-32 points out of 40. An excellent report will receive 32-40 points. Regarding page limits the expectation is that an excellent report will be approximately no more of three pages of text (this does not include tables, extra blank space on page, repetition of text from the assignment). There is no strict limit, however, as some writers are more or less concise than others (thus one page may be sufficient) and there are a range of formatting possibilities.