Homework 3: Due March 29 at 3:00pm Montreal time

General guidelines for homeworks:

- You are encouraged to meet with other students to discuss the homework, but all write-ups must be done on your own. Do not take notes from those meetings. You should know how to work out the solutions by yourself.
- Please acknowledge other students with whom you discussed the problems and what resources (other than the instructor/TAs, lecture notes, and textbook) you used to help you solve the problem. This won't affect your grade.
- Homework grades will be based not only on getting the "correct answer", but also on good writing style and clear presentation of your solution. It is your responsibility to make sure that the marker can easily follow your line of reasoning.
- For programming questions, please make sure your code is clearly commented and easy to read. If the marker cannot understand your code and it does not run correctly, you might not be able to get any partial marks.
- Follow the instructions exactly. We reserve the right to refuse to grade the homework or deduct marks if the instructions are not followed.
- Your code should not rely on advanced external libraries (e.g. `scikit-learn`), but you are free to use `Numpy` (for numeric operation) and `Matplotlib` (for plotting). You can also use `scipy.io` for loading the data.
- This will be the last HW for this course. Feel free to use all your remaining grace days.

1 Kernels (5 points)

For any two documents x and z (note that x and z are not vectors), define a function $k(x, z)$ to be the number of unique words that occur in both x and z (i.e. the size of the intersection of the sets of words in the two documents). Is this function a kernel? Justify your answer. You can assume that the size of the vocabulary is D. *Hint:* $k(x, z)$ *is a kernel if there exists* $\phi(x)$ *such that* $k(x, z) = \phi(x)^T\phi(z)$.

2 Programming: AdaBoost (10 points)

In this question you will implement AdaBoost for discriminating between 4s and other digits. You will need the code/data in the folder "adaboost". A code skeleton `boost_digits.py` is provided. You need to fill in the details of AdaBoost.

In particular, a function `findWeakLearner` is provided. This function chooses the best decision stump (feature, threshold, parity) to minimize weighted 0-1 loss.

Each decision stump is returned from `findWeakLearner` as d, p, θ. This represents the weak learner:

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad p\mathbf{x}(d(1), d(2)) > p\theta \\ -1 & \text{otherwise} \end{cases}$$

where $\mathbf{x}(d(1), d(2))$ means the pixel value at $(d(1), d(2))$ of the image $\mathbf{x}$.

In your report, provide the final plot of training error and test error produced using AdaBoost. Also include the visualization of the final classifier produced using `visualizeClassifier` in `utils.py`

Note that a second data file `digits10000.mat` is provided if you wish to experiment with more data than the 1000 in `digits.mat`.

Hint: if $b = 0$, the expression "a/b" will have numerical problem. A common trick is to use "$a/(b + np.finfo(float).eps)$". In Python, "$np.finfo(float).eps$" will generate a very small number.

3 Programming: Kernel regression (10 points)

In this question, you will implement the nonparametric kernel regression using two different kernel functions. You will need the code and data in the folder "regression".

Given a set of n training data $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), ..., (\mathbf{x}_n, t_n)\}$, the regression function is:

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{n} t_i g(\mathbf{x} - \mathbf{x}_i)}{\sum_{i=1}^{n} g(\mathbf{x} - \mathbf{x}_i)}$$

where $g(\cdot)$ is the kernel we have to choose.

Functions are provided for loading the data[1], and normalizing the features and targets to have 0 mean and unit variance.

```
[t,X] = loadData();
X_n = normalizeData(X);
t = normalizeData(t);
```

For the following, use these normalized features `X_n` and targets. Use the first 100 data points as training data, and the remainder as testing data. Use only the 3rd feature `X_n(:,3)`.

Perform the following experiments:

(a) Create a Python script `gaussian_vis.py` for the following.

Perform kernel regression using the Gaussian kernel:

$$g(u) = \frac{1}{\sqrt{2\pi h^2}} \exp(-u^2/(2h^2))$$

Fit a regression model using $h = \{0.01, 0.1, 1, 2, 3, 4\}$. Produce plots of the training data points, learned regression function, and test data points. The code `visualize_1d.py` in HW1

[1] Note that loadData reorders the datapoints using a fixed permutation. Use this fixed permutation for the questions in this assignment. If you are interested in what happens in "reality", try using a random permutation afterwards. Results will not always be as clean as you will get with the fixed permutation provided.

might be useful. **Put 2 or 3 of these plots, for interesting (high values/low values of h) results, in your report. Include brief comments.**

(b) Create a Python script `gaussian_validate.py` for the following.

Again, perform the kernel regression using the Gaussian kernel with $h = \{0.01, 0.1, 0.25, 1, 2, 3, 4\}$. Use 10-fold cross-validation to decide on the best value for h. Produce a plot of validation set error versus h. Use a `semilogx` plot, putting h value on a log scale. **Put this plot in your report, and note which h value you would choose from the cross-validation.**

(c) Create a Python script `epanechnikov_vis.py` for the following.

Repeat the experiment in (a), but this time use the Epanechnikov kernel:

$$g(u) = \begin{cases} \frac{3}{4}(1 - u^2/h^2) & \text{if } |u|/h \leq 1; \\ 0 & \text{otherwise} \end{cases}$$

Include corresponding plots and brief comments in your report.

(d) Create a Python script `epanechnikov_validate.py` for the following.

Repeat the experiment in (b), but this time use the Epanechnikov kernel. **Put the corresponding in your report, and note which h value you would choose from the cross-validation.**

How to Submit

Please create a document in PDF format (named it as `hw3.pdf`) containing the following:

- Your name, student ID, email address
- Your solutions to the questions (including plots/figures)

You are strongly encouraged to type the solutions. But you really prefer, you can also write the solutions by hand, then scan them and create a PDF document. If you choose to write your solutions by hand, please make sure it is neat and legible. If the markers cannot understand your handwriting, you will lose marks.

Put your codes (including data files and others that are provided as part of the assignment) in a folder named `lastname_firstname` (replace with your last and first names). Create a ZIP file (name it `lastname_firstname.zip`, again, replace with your last and first names accordingly) containing the top-level directory.

For example, if I were to submit the homework, I would create a ZIP file named `wang_yang.zip`. After running "`unzip wang_yang.zip`", I should get the following:

```
wang_yang/
wang_yang/adaboost
wang_yang/adaboost/boost_digits.py
wang_yang/...... (other files)
```

If I run `python boost_digits.py` in the `wang_yang/adaboot` folder, I should get the plots in the report from the relevant question.

Go to this course in Moodle, then click "`HW3 submission`". Submit the following two files: 1) the PDF document; 2) the zip file containing your codes (see above). The submission server is configured to only accept PDF and ZIP files. If you try to submit other file format, the submission server will not accept it. So make sure you know how to create PDF and ZIP files ahead of time.

Note that you can make multiple submissions. But we will only consider the last submission. We will use the time stamp recorded on Moodle to calculate late days you have used for this assignment.