Indian Institute of Technology Kanpur
CS771 Introduction to Machine Learning

Instructor: Purushottam Kar
Date: August 20, 2019
Total: 80 marks

ASSIGNMENT 1

1 What should I submit, where should I submit and by when?

Your submission for this assignment will be one PDF (.pdf) file and one python (.py) file. Instructions on how to prepare and submit these files is given below.

Assignment Package:
`http://web.cse.iitk.ac.in/users/purushot/courses/ml/2019-20-a/material/assn1.zip`
Deadline for all submissions: 07 September, 9:59PM IST
There is no provision for "late submission" for this assignment

1.1 How to submit the PDF file

1. The PDF file must be submitted using Gradescope in the *group submission mode*. Note that this means that auditors may not make submissions to this assignment.
2. Make only one submission per assignment group on Gradescope. Gradescope allows you to submit in groups - please use this feature to make a group submission.
3. To clarify the above, there are currently 49 assignment groups. We wish to have only 49 submissions on Gradescope, i.e. one submission per assignment group, not one submission per student.
4. Link all group members in your group submission. If you miss out on a group member while submitting, that member may end up getting a zero since Gradescope will think that person never submitted anything.
5. You may overwrite your group's submission (submitting again on Gradescope simply overwrites the old submission) as many times as you want before the deadline.
6. Do not submit Microsoft Word or text files. Prepare your PDF file using the style file we have provided (instructions on formatting given later).

1.2 How to submit the Python file

1. Upload the Python file on your IITK (CC or CSE) website (for CC, log on to `webhome.cc.iitk.ac.in`, for CSE, log on to `turing.cse.iitk.ac.in`).
2. Fill in the following Google form to tell us the exact path to the file
   `https://forms.gle/5VTTP5ty3qQ2YMph8`
3. Do not submit Jupyter notebooks or else files in other languages such as C/Matlab/Java. We will use automated scripts to evaluate your code and will not bother to run code in other formats or other languages (we will simply give such submissions a zero).

4. The name of your file should preferably be "submit.py". Submit only one file. We will not download multiple files from your website even if you ask us to. We will download only one file which we expect to be a Python (.py) file (only one .py file will be accepted and not .ipynb or .m or .c or .cc or .java or .zip or .gz etc).

5. Do not host your Python code submission file on file-sharing services like Dropbox or Google drive. Host it on IITK servers only. We will be using a script to autodownload your submissions and if not careful, Dropbox or Google Drive URLs may send us an HTML page (instead of your submission) when we try to autodownload your file. Thus, it is best to host your code submission file locally within IITK servers.

6. While filling the form, give the complete URL to the file, not just to the directory that contains that file. The URL should contain the filename as well.

   (a) Example of a proper URL:
   `https://web.cse.iitk.ac.in/users/purushot/mlassn1/submit.py`

   (b) Example of an improper URL (file name missing):
   `https://web.cse.iitk.ac.in/users/purushot/mlassn1/`

   (c) Example of an improper URL (incomplete path):
   `https://web.cse.iitk.ac.in/users/purushot/`

7. We will use an automated script to download all your files. If your URL is malformed or incomplete, or if you have hosted the file outside IITK and it is difficult for us to download automatically, then we will not bother to make any efforts to manually locate your file or else contact you for clarifications. We will simply give your group a zero in these cases.

8. Make sure you fill in the Google form with your file link before the deadline. We will close the form at the deadline.

9. Make sure that your file is actually available at that link at the time of the deadline. We will run a script to automatically download these files after the deadline is over. If your file is missing, we will simply assign your group zero marks.

10. We will entertain no submissions or mercy appeals over email, Piazza etc. All submissions must take place before the stipulated deadline over the Gradescope and the Google form. The PDF file must be submitted on Gradescope at or before the deadline and the Python file must be available on the link specified on the Google form at or before the deadline.

Problem 1.1 (The Squared SVM Solver)**.** Consider the following problem formulation for n binary labelled data points $(\mathbf{x}^i, y^i)_{i=1,\ldots,n}$ where $\mathbf{x}^i \in \mathbb{R}^d$ and $y^i \in \{-1, +1\}$ that uses the squared hinge loss instead of the hinge loss

$$\arg\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{w}\|_2^2 + C \cdot \sum_{i=1}^{n} \left(\left[1 - y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle\right]_+ \right)^2 \qquad (P1)$$

The above optimization problem (P1) can be rewritten as a new optimization problem (P2)

$$\begin{aligned} \arg\min_{\mathbf{w}\in\mathbb{R}^d, \boldsymbol{\xi}\in\mathbb{R}^n} \ & \frac{1}{2}\|\mathbf{w}\|_2^2 + C \cdot \sum_{i=1}^{n} \xi_i^2 \\ \text{s.t. } & y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle \geq 1 - \xi_i, \text{ for all } i \in [n] \qquad (P2) \end{aligned}$$

We have also provided you with data on a 20 dimensional binary classification problem with 20000 data points in the assignment package (in a file called `data`). You may create (held out/k-fold) validation sets out of this data in whichever way you like.

The following enumerates 7 parts to the question. Parts 1-5 need to be answered in the PDF file. Part 6 needs to be answered in the Python file. Part 7 is a bonus part and needs to be answered in the PDF file itself.

1. Introduce a dual variable α_i for each of the n constraints in (P2) and write the expression for the Lagrangian. Remember, the Lagrangian has no $\max, \min$ terms and is simply a function of the form $\mathcal{L}(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha})$ where $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_n] \in \mathbb{R}^n$ and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n] \in \mathbb{R}^n$. Also take care that in this case, the primal problem (P2) has not only a $\mathbf{w}$ variable that represents the model, but also a $\boldsymbol{\xi}$ variable that encodes the slacks. (5 marks)

2. Find out the dual problem by eliminating the primal variables $\mathbf{w}, \boldsymbol{\xi}$ using the first order optimality trick i.e. by setting $\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{0}$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\xi}} = \mathbf{0}$. Write down the dual optimization problem that you get as a result – let us call this problem (D2). Note that (P1), (P2), (D2) are all the same problem written differently. (10 marks)

3. Implement at least 3 methods to solve the problem of learning with the squared hinge loss. Coordinate maximization on D2 (similar to the method we saw in class to solve the dual of the CSVM problem) must be one of the 3 methods you implement. The other 2 can be any one of (a)-(e) methods given below

   (a) SGD on P1 (or mini-batch variant)
   (b) Coordinate descent on P1
   (c) Coordinate minimization on P1
   (d) Projected stochastic gradient ascent on D2 (or mini-batch variant)
   (e) Coordinate ascent on D2
   (f) Coordinate maximization on D2 (**compulsory**)

   Please do not implement gradient descent on P1 or projected gradient ascent on D2. We wish to implement only stochastic or else coordinate methods in this assignment. For each of the methods you implement, tell us the name of the method and write down all details about how you implemented the method. For example, if you used mini-batch SGD, write down the gradient expression and tell us what batch size and step lengths did you choose. For another example, if you used coordinate minimization, write down the formula for the minimizer along a chosen coordinate as well as which method you used to select the next coordinate along which to minimize. (3 x 3 = 9 marks)

4. Also tell us, for each of these 3 methods you implemented, how did you arrive at the best values for the hyperparameters mentioned above – for example, you might say "*We used step length at time t to be $\eta/\sqrt{t}$ where we checked for $\eta = 0.1, 0.2, 0.5, 1, 2, 5$ using held out validation and found $\eta = 2$ to work the best*". For another example, you might say, "*We tried random and cyclic coordinate selection choices and found cyclic to work best using 5-fold cross validation*". Thus, you must tell us among which hyperparameter choices did you search for the best and how. (3 x 2 = 6 marks)

5. For your chosen 3 methods, plot the convergence curves offered by those 3 methods in a manner as we do in lecture notebooks. The x axis in the graph should be time taken and the y axis should be the value of the objective function in (P1) (warning: do not use the objective function in (P2) or (D2) but rather the objective function in (P1) for all 3 methods). Plot all 3 curves on the same graph (not on 3 different graphs) so that they can be compared. Include this graph in your PDF file submission as an image. Write down which method you think performs the best.
   Warning: this is not a toy 2D dataset like we have in class. Any method (GD/SDCA etc) may take several seconds/minutes to run before giving decent results. You may have to run these methods for several tens of thousands of iterations as well. (10 marks)

6. Of the methods you experimented with above, choose the one you think is the best method (i.e. which achieves the smallest objective value of (P1) in the fastest time in the most reliable manner) and submit code for that method in `submit.py` (we do not need code for the other 2 methods, just the one you thought was best). We will evaluate your method on a slightly different dataset than the one we have given you and check how good is the method you submitted. (40 marks)

7. *Bonus*: Notice that (P2) does not have the positivity constraints $\xi_i \geq 0$ that the CSVM formulation had. Can you show that the inserting these additional constraints into (P2) does not change the optimization problem at all (i.e. the solution remains the same)?

Note that parts 1-5 (and the bonus part 7 if you are attempting it) need to be answered in the PDF file. Part 6 needs to be answered in the Python file. You may want to check the references section of the course website and refer to [**DAU**] S. 7.7 S. 11.5-11.6, [**SSBD**] S. 15.3-15.5 for help in solving parts 1 and 2 of this problem. (80 + bonus marks)

2 How to Prepare the PDF File

Use the following style file to prepare your report.
`https://media.neurips.cc/Conferences/NeurIPS2019/Styles/neurips_2019.sty`

For an example file and instructions, please refer to the following files
`https://media.neurips.cc/Conferences/NeurIPS2019/Styles/neurips_2019.tex`
`https://media.neurips.cc/Conferences/NeurIPS2019/Styles/neurips_2019.pdf`

You must use the following command in the preamble

```
\usepackage[preprint]{neurips_2019}
```

instead of `\usepackage[final]{neurips_2019}` as the example file currently uses. Use proper LaTeX commands to neatly typeset your responses to the various parts of the problem. Use neat math expressions to typeset your derivations. Remember that parts 1-5 (and the bonus part

7 if you are attempting it) need to be answered in the PDF file. All plots must be generated electronically - no hand-drawn plots would be accepted. All plots must have axes titles and a legend indicating what the plotted quantities are. Insert the plot into the PDF file using proper LaTeX `\includegraphics` commands.

3 How to Prepare the Python File

The assignment package contains a skeleton file `submit.py` which you should fill in with the code of the method you think works best among the methods you tried. You must use this skeleton file to prepare your Python file submission (i.e. do not start writing code from scratch). This is because we will autograde your submitted code and so your code must have its input output behavior in a fixed format. Be careful not to change the way the skeleton file accepts input and returns output.

1. The skeleton code has comments placed to indicate non-editable regions. **Do not remove those comments**. We know they look ugly but we need them to remain in the code to keep demarcating non-editable regions.

2. We have provided you with data points in the file `data` in the assignment package that has 20000 data points, each being 20 dimensional. You may use this as training data in any way to tune your hyperparameters (e.g. step length) by splitting into validation sets in any fashion (e.g. held out, k-fold). You are also free to use any fraction of the training data for validation, etc. but your job is to do really well in terms of minimizing the objective function of (P1) and that too really fast.

3. The code file you submit should be self contained and should not rely on any other files (e.g. other .py files or else pickled files etc) to work properly. Remember, we will download only one Python (.py) file from your website. This means that you should store any hyperparameters that you learn (e.g. step length etc) inside that one Python file itself using variables/functions.

4. We will test your submitted code on a secret dataset which would be slightly different from the one we have provided you. Thus, your code should not assume it will be given 20000 data points (for example, our secret dataset may have 2000 or even 23000 data points) nor should it assume that each data point has 20 dimensions (for example, our secret dataset may have 15 or even 25 dimensions). However, we can assure you that our secret dataset looks similar to the dataset we have provided you so that hyperparameter choices that seem good to you during validation, should work decently on our secret dataset as well.

5. Certain portions of the skeleton code have been marked as non-editable. Please do not change these lines of code. Insert your own code within the designated areas only. If you tamper with non-editable code (for example, to make your code seem better or faster than it really is), we may simply refuse to run your code and give you a zero instead (we will inspect each code file manually).

6. You are allowed to freely define new functions, new variables, new classes in inside your submission Python file while not changing the non-editable code.

7. You are not allowed to use any machine learning libraries such as sklearn, scipy, pandas etc in your submission Python file. Do not use the sys library either. You are expected to implement your method yourself from scratch. You are allowed to use only basic Python

libraries such as numpy, time and random which have already been included for you. Do take care to use broadcasted and array operations as much as possible and not rely on loops to do simple things like take dot products, calculate norms etc.

8. The assignment package also contains a file called `eval.py` which is an example of the kind of file we will be using to evaluate the code that you submit. Before submitting your code, make sure to run `eval.py` and confirm that there are no errors etc.

9. You do not have to submit `eval.py` to us – we already have it with us. We have given you access to the file `eval.py` just to show you how we would be evaluating your code.