Machine Learning

NUS SoC, 2017/2018, Semester I
Lecture Theatre 19 (LT 19) / Tuesdays 14:00-16:00

Last Updated: September 25, 2017 - First porting.

Homework #2 » Support Vector Machines

Due: 10 Oct 2017, 11:59pm SGT. In this homework assignment, you will be using a soft margin support vector machine.

We will be utilizing the Scikit Learn SVM package for this homework, as you had practiced before in Tutorial. You may find it helpful to review the Scikit Learn's SVM documentation (http://scikit-learn.org/stable/modules/svm.html).

When evaluating E_{in} and E_{out} of the resulting classifier, use binary 0/1 classification error. The error should be the average 0/1 loss over the appropriate (train|test) set; i.e., $1/N \sum^{N} E(\boldsymbol{x}_n)$, where $E(\boldsymbol{x}_n)$ is pointwise 0/1 loss for the single instance $\boldsymbol{x}_n$).

Compute and output the classification accuracy given the predicted labels obtained with SVM classification versus the true labels of the test data in the following parts of the question. Note that training accuracy is $1 - E_{in}$, and testing accuracy is $1 - E_{out}$. You need to execute the experiments in your `.ipynb` file so that your answers can be validated against your output.

In particular, you should ensure to run the below code snippet, per problem part, for Problems 1A and 2D. This code creates an output file per question, which our autograding system will use to check against our model answer. You may decide to use your own function prototypes instead of these, but make sure you output the same strings. Also, for all parameters that are not explicitly stated or underdetermined, you should use scikit learn's default settings.

```
### The output of your program should be written in a file as follows.
#   for question 'i', write the output in 'problem-i.txt' file (e.g., 'problem-
fo = open('problem-i.txt', 'w')

# train your svm
# (n.b., svmTrain, svmPredict are not previously defined;
# you will have to supply code to implement them)
svmModel, totalSV  = svmTrain(dataTrain, labelTrain, cost, kernel, gamma, degre

# test on the training data
trainAccuracy = svmPredict(dataTrain, labelTrain, svmModel)

# test on your test data
testAccuracy = svmPredict(dataTest, labelTest, svmModel)

# report your results in the file
fo.write("Kernel: "+ str(kernel)+"\n")
fo.write("Cost: "+ str(cost)+ "\n")
fo.write("Number of Support Vectors: "+ str(totalSV)+"\n")
fo.write("Train Accuracy: "+ str(trainAccuracy)+"\n")
fo.write("Test Accuracy: " + str(testAccuracy)+"\n")
```

Problem 1 - US Postal Service Zip Code dataset (hw2-1.ipynb; 40 marks)

This is a problem from our textbook's homework set: Homework #8. You may find hints and discussions on the homework on the book's forum (http://book.caltech.edu/bookforum/) for the homework, as well as in IVLE. Feel free to quote or repost threads from the book's forum to IVLE; we will grant you participation marks for finding and propagating useful discussions on the assignment to the class via IVLE. Answers without corresponding experimentation to validate your findings will be assigned minimal partial credit.

We apply soft-margin SVM to handwritten digits from the processed US Postal Service Zip Code data set. The 2D data represents extracted features of intensity and symmetry for training and testing. We will train a one-versus-one (one digit is class +1 and another digit is class -1) classifier for the digits '1' (+1) and '5' (-1).

a. Consider the linear kernel $K(\boldsymbol{x}_n, \boldsymbol{x}_m) = \boldsymbol{x}_n^T \boldsymbol{x}_m$. Train and test using all of the points, writing the output to an output file `problem-1a.txt`. In addition to using all of the training examples, try subsets of the training data and print out accuracy over the testing set (1 - E_{out} (over all test examples), and the number of support vectors. Try with the first {50, 100, 200, 800} points with the linear kernel. The output of these experiments should be written in Markdown cells, as opposed to output to a file. Only the first part (with all points) should be written to the file `problem-1a.txt`.
b. Consider the polynomial kernel $K(\boldsymbol{x}_n, \boldsymbol{x}_m) = (1 + \boldsymbol{x}_n^T \boldsymbol{x}_m)^Q$, where Q is the degree of the polynomial. Comparing $Q = 2$ with $Q = 5$, which of the following statements is correct?

i. When $C = 0.0001$, E_{in} is higher at $Q = 5$.
ii. When $C = 0.001$, the number of support vectors is lower at $Q = 5$.
iii. When $C = 0.01$, E_{in} is higher at $Q = 5$.
iv. When $C = 1$, E_{out} is lower at $Q = 5$.
v. None of the above

c. Consider the radial basis function (RBF) kernel $K(\mathbf{x}_n, \mathbf{x}_m) = e^{(-||\mathbf{x}_n - \mathbf{x}_m||^2)}$ in the soft-margin SVM approach. Which value of $C \in \{0.01, 1, 100, 10^4, 10^6\}$ results in the lowest E_{in}? The lowest E_{out}?

Problem 2 - Gisette (hw2-2.ipynb; 20 marks)

GISETTE is a handwritten digit recognition problem. The problem is to separate the highly confusible digits '4' and '9'. This dataset is one of five datasets of the NIPS 2003 feature selection challenge. The dataset for this problem is large, so please budget time accordingly for this problem.

The above description URLs are for your information only; do not use the actual datasets featured there, use the ones available in IVLE as part of the .zip package, as there are some changes that we have made to the challenge dataset for classroom purposes.

d. Standard run: Use all the 6000 training samples from the training set to train the model, and test over all test instances, using the linear kernel.
e. Kernel variations: In addition to the basic linear kernel, investigate two other standard kernels: RBF (a.k.a. Gaussian kernel; set γ = 0.001), Polynomial kernel (set degree=2,coef0=1; e.g, $(1+x^Tx)^2$). Which kernel yields the lowest E_{in}?

Optional Bonus (hw2-3.ipynb)

For a small amount of optional bonus marks (1 or 2; the assignment is worth 100 marks), you can also complete work to implement the support vector machine algorithm itself. Note: This is more work that may not be commensurate to the amount of effort you need to do to accomplish the task. You can do any of the optional exercises, we will assign partial credit if you only complete 1.

f. Implement the support vector machine with soft margin by solving:

   Use a QP solver library (such as `cvxopt` or `libsvm`) to solve the SVM dual problem. You may find the hard margin solution on Github by phisart a useful starting point. You'll need to construct the appropriate arrays to pass to cvxopt and after invoking the solver to solve the QP, compute w and b to find the hyperplane.

   Use the implemented SVM to re-run your experiments for Problem 1, Part a. Do you get similar results?

g. Plot the decision boundary of your RBF SVM from Problem 1, Part c. Use a mesh to plot in, such that you assign a color (red, blue) to each the points in the mesh. You

may find the model code from Scikit Learn's page useful: http://scikit-learn.org/stable/auto_examples/svm/plot_iris.html.

h. Additionally, add to Part g), to include a plot of the margin and explicitly marking of the support vectors.

Format of the input files

You'll find these files in the sample submission .zip.

- `hw2-1-(train|test).txt:` Contains (1561|424) samples of 2-d data, with the class `1` or `-1` as the first column, all separated by spaces. This is a dense data format, as subsequent two columns are floating point numbers that indicate the (greylevel) intensity and symmetry from the original dataset.
- `hw2-2-(train|test).txt:` Contains (6000|1000) samples of 5000-dimensional data, with the class `1` or `-1` as the first column, all separated by spaces. Note that this is a sparse data format, as the remaining feature columns have a X:Y format, where the first 'X' indicates the feature number and the second 'Y' indicates the value of the feature (scaled to the [-1:1] interval). The samples will not always have all 5000 dimensions, so you can assume a 0.0 value for the missing features (if you are using a dense vector format).

Essay questions:

You are also asked to answer the following essay questions. These are to test your understanding of the lecture materials. Note that you may need to write math equations or notation in the files, so please familiarize yourself with this. You can either use a word processor or do a dump (i.e., print to .pdf) of another program that can display math (i.e., iPython notebook, R). Submit it as part of your .zip submission, exactly named as `essay2.pdf`.

1. Consider a binary classification problem. Let us denote the input set as $\{\boldsymbol{x}_n, y_n\}$ where $\boldsymbol{x}_n$ is the data point and $y_n \in \{-1,+1\}$ is the corresponding class label. Further, misclassifying a positive class data point costs k times more than misclassifying a negative class data point. Rewrite the SVM optimization function to model this constraint.
2. Consider the following training data:

1	1	+
2	2	+
2	0	+

0	1	–
1	0	–
0	0	–

a. Plot these six training points. Are the classes {+, –} linearly separable?
b. Construct the weight vector of the maximum margin hyper-plane by inspection and identify the support vectors.
c. If you remove one of the support vectors, does the size of the optimal margin decrease, stay the same, or increase? Explain.
d. Is your answer to (c) also true for any dataset? Provide a counterexample or give a short proof.

3. A kernel is valid if it corresponds to a scalar product in some (perhaps infinite dimensional) feature space. Remember a necessary and sufficient condition for a function $K(\mathbf{x}, \mathbf{x})$ to be a valid kernel is that associated Gram matrix, whose elements are given by $k(x_n,x_m)$, should be positive semi-definite for all possible choices of the set **x**. Show whether the following are also kernels:
   1. $K(\mathbf{x},\mathbf{x}') = c\ \langle \mathbf{x},\mathbf{x}' \rangle$
   2. $K(\mathbf{x},\mathbf{x}') = \langle \mathbf{x},\mathbf{x}' \rangle^2 + \mathbf{e}^{(-||\mathbf{x}||^2)}\mathbf{e}^{(-||\mathbf{x}'||^2)}$

   where <–,–> indicates an inner product. *Hint: refer back to the properties needed to solve Tutorial 3A, Problem 4.*

It can be useful to repeat the question in your essay submission, for readability purposes, but it's not mandatory.

Submission Formatting

For us to grade this assignment in a timely manner, we need you to adhere strictly to the following submission guidelines. Following the submission guideliness will help us grade the assignment in an appropriate manner. You will be penalized if you do not follow these instructions. Your matric number in all of the following statements should not have any spaces and all letters should be in CAPITALS (inclusive of the ending check letter). You are to turn in the following files:

- Your source code files `hw2-<x>.ipynb` for this homework assignment. We will be reading your code, so please do us a favor and format it nicely. To expedite grading, we need to require all of the assignments to use Python 3.4 (3.5 should ok as well).

  In your source code `.ipynb`, describe any information you want us to know about your submission, inclusive of your **suitably filled out independent work statement**. You should not include any identifiable information about your assignment (your

name, phone number, etc.) except your matric number and email (we need the email to contact you about your grade, please use your *[a|e]*******@nus.edu.sg* address, not your email alias). This is to help you get an objective grade in your assignment, as we won't associate matric numbers with student names. You can include this information at the top of your source code in Markdown cell(s) in the (just the first `-1.ipynb`, when you have multiple problems) `.ipynb` notebook source file. Make sure to run your code and save the state after running your working code upon submission.

At the end of your (first) notebook source file, you need to put your **Statement of Independent Work**. Use a Markdown cell and follow the instructions given in tutorial about how to fill this section out. Please be honest and if you do not follow the course policy, and suggest a plausible alternative evaluation for us to consider.

Also put pointers or text to any (online or offline) references you have utilised in making your submission in an appropriate **References** cell.

- A .PDF file `essay2.pdf` that contains your answers to the essay questions (no word processing files, please export to PDF with any unusual fonts embedded within the document if needed.) You may choose to use Markdown cells to answer the essay questions directly in the `hw2-1.ipynb` notebook file, and in this case you will not need this separate `essay2.pdf` file. Again, do not disclose any identifiable information in your essay answers.
- All of the created output `problem-<x>.txt` files, that are programmatically generated by your code, following the sample code exactly to open and print out the necessary metrics for our auto grading. Note that you may also output these in your code as regular STDOUT print statements as well; but you must at least generate the `problem-<x>.txt` files for our auto-grader.
- Any ancillary files that your submission uses. We may or may not read them, so please put the core part of your programming answers in the `hw2-<x>.ipynb` file.

Please do not submit the training and testing files, as this will significantly blow up your submission's size and cause unnecessary bandwidth consumption.

These files will need to be suitably zipped in a single file called `<matric number>.zip`. Please use a zip archive and not tar.gz, bzip, rar or cab files. Make sure when the archive unzips that all of the necessary files are found in a directory called `<matric number>` (note to Mac users please test your zip archives on the command line; as OSX "automagically" creates the parent folder but it is often actually not contained in the archive). Upload the resulting zip file to the IVLE workbin by the due date: 10 Oct 2017, 11:59:59 pm SGT. There absolutely will be no extensions to the deadline of this assignment. Read the late policy if you're not sure about grade penalties for lateness.

Grading Guidelines

The grading criteria for the assignment is shown below (subject to change, but should be final). Note that these percentages differ from Homework #1.

- 60% Programming exercises.
- 30% Essay questions.
- 10% Documentation quality, inclusive of correctly following these submission guidelines.

Hints

- We are not providing the skeleton `.ipynb` files from now on; please do this yourself. We suggest that you copy Homework #1's template and adjust it accordingly. Remember to include any high level documentation at the top of the file, any references and your filled-out statement of indepedent work.
- Do not scale the data when you use libsvm or other packages, otherwise you may inadvertently change the (effective) kernel and get different results;
- In some packages, you need to specify double precision;
- Some packages have software parameters whose values affect the outcome. ML practitioners have to deal with this kind of added uncertainty.
- Reading the fine manual for SVM on Scikit Learn, especially 1.4.5 Tips on Practical Use