Homework 1: Perceptron

Due: 9/12/17

1. *Exploratory data analysis.*

   (a) Pick a dataset you are interested in exploring.
   We have listed some datasets on the course project page that you might consider:

   `https://people.orie.cornell.edu/mru8/orie4741/projects.html`

   but you are free to find your own dataset in a field that interests you. We encourage you to pick a dataset you might want to use for your project.

   (b) Pose a question about the dataset.

   (c) Create four visualizations of the data that help answer your question. For each visualization, describe briefly what conclusions you drew from it. (Remember to label your axes!)

   For this problem, you may use any programming language. You do not need to submit your code; just submit your visualizations. If you use Julia, we recommend trying the plotting tools in the package `Plots.jl`, following the syntax used in the exploratory data analysis demo to get started:

   `https://github.com/ORIE4741/demos/blob/master/eda.ipynb`

2. *Vectors, matrices, and inner products.*

   What is a vector? Is it just a matrix with one column, or is it something different? This problem explores this question using the Julia programming language. You will find Julia distinguishes between vectors (one-dimensional arrays) and matrices (two-dimensional arrays), even when the matrix has only one column. If you'd like, you may try out (translations of) these code examples in any other programming language for full credit; but the answers will be nicer in Julia. This code uses the `Random` package, which you can load with the command

   ```
   using Random
   ```

(a) The inner product, or dot product, of two vectors is the sum of the products of their elements. Here, let's use the transpose function `'` to compute inner products. Compare the following code

```
u = rand(3,1)
v = rand(3,1)
u'*v
```

with

```
x = rand(3)
y = rand(3)
x'*y
```

Are the shapes of the results the same or different? Use the `size` function to check. What rule do you think Julia is using?

(b) There are many other ways to compute inner products so that the result is always a number and not an array. For example, try the following:

```
dot(u,v)
sum(u.*v)
```

What results do these functions have when applied to u and v, and to x and y? What is the size of the result?

(c) Generate a random matrix `A = rand(5,5)`. Consider the first column `A[:,1]`, and the first row `A[1,:]`. What shape do these have? Are they vectors or matrices?

(d) How would you take the inner product between the first column `A[:,1]` and the first row `A[1,:]`? State at least two methods. Your result should be a number, not an array.

3. *Perceptron.*

(a) Code the perceptron algorithm by filling in the appropriate portion of the notebook `perceptron.ipynb` found at

`https://github.com/ORIE4741/homework`

The easiest way to access this notebook is to use JuliaBox and sync the above repository. You can then open the notebook and work directly in JuliaBox. This is explained in detail in the Julia tutorial section materials.

Note that any unicode symbol (*e.g.*, ♮) is valid in Julia; you can type them in a Jupyter notebook as you would in LaTeX (*e.g.*, `\natural`).

You are also welcome to use any other language (that your TAs can read) to solve the problem; you'll just need to rewrite the starter code from the notebook in your favorite language.

Make sure your algorithm runs in time linear in n. This means you cannot check whether each point is correctly classified at every iteration.

Feel free to increase the maximum number of iterations.

(b) How many updates does the algorithm take before converging? Plot the datapoints, the true vector $w\natural$, and the final hypothesis of the Perceptron algorithm. You can use the plotting function we've provided: `plot_perceptron(X,y,w)`.

(c) Repeat (b) with a randomly generated data set of size 20, 100, and 1000. Compare your results with (b).

(d) Randomly generate a linearly separable data set of size 1000 with $x_i \in \mathbb{R}^{10}$ instead of $\mathbb{R}^2$. How many updates does the algorithm take to converge?

(e) Repeat the algorithm on the same data set as (d) for 100 experiments. In the iterations of the perceptron algorithm for this part, pick your next consideration point $x(t)$ randomly instead of deterministically. Plot a histogram for the number of updates that the algorithm takes to converge.

(f) Summarize your conclusions with respect to accuracy and running time as a function of n (size of data set) and d (dimension).

(g) Go back to $n = 50$ and $d = 2$. Add an outlier to your dataset. What happens when we run the perceptron algorithm?

(h) How could we fix the perceptron algorithm? Try out at least one idea and report your results. *Note: this question has no right answer.*

You should submit 1) your notebook (.ipynb) and 2) a PDF of the notebook, which you can generate by printing the notebook to PDF from your browser. Make sure to clearly mark each problem section.

4. *Calibration.* How long did you spend on each problem in this homework assignment, and on the homework assignment, in total?