CS8803: STR, Spring 2018. Lab 4 (EXTRA CREDIT): Learning with Kernels

You may work in groups of up to 3.

Due: Wednesday, May 2nd, beginning of exam period

Assignment

This assignment gives you a chance to implement some of the kernel-based learning algorithms discussed in class on real data.

As you may remember from Lab 2, automated interpretation of ladar data is crucial for outdoor vehicle operation. Just like in Lab 2, you will be building a system to classify ladar points into one of five categories: ground (supporting surface), vegetation, facade, pole, and wire. You must **implement** 2 learners we discussed in class. (Talk to me if you're interested in implementing other variants instead.)

- Gaussian Process Regression with a Gaussian RBF kernel (choose two classes)
- Kernelized Logistic Regression (with your choice of nonlinear kernel, via gradient descent, choose two classes)
- **Extra Credit**: research and implement a kernel method of your choosing (Kernelized SVMs?, Gaussian process regression with a different nonlinear kernel?)

Data

You will use the same dataset that you used in Lab 2: 3D point-clouds of Oakland in Pittsburgh (see `http://www.cs.cmu.edu/~vmr/datasets/oakland_3d/cvpr09/doc/`). Features are provided courtesy of Dan Munoz, but you are welcome to come up with new features to use (please explain if you do). You should not distribute the data without permission.

There are five classes, their labels values are:

- 1004: Veg
- 1100: Wire
- 1103: Pole
- 1200: Ground
- 1400: Facade

Run each algorithm on both datasets and report on the performance.

What to turn in

I am interested in a short performance summary and nice visualizations. You should turn in your *code* and a 2-3 page *report* to the T.A. via email. For each learner you implemented, report on the following:

1. How well did each algorithm perform? Compare the two algorithms to each other and compare them both to the online learning algorithms that you implemented in Lab 2. How does each algorithm perform on the held-out data?
2. How easy was the learner to implement?
3. How long does the learner take (in terms of data points, dimensions, classes, etc...) for training and prediction?
4. Show images/movies of the classified data.
5. How did you choose (hyper)parameters (priors, learning rate, etc. . .)?