

Assignment 1: Regression, Nonparametric Methods

Due February 4 at 11:59pm
100 marks total

This assignment is to be done individually.

Important Note: The university policy on academic dishonesty (cheating) will be taken very seriously in this course. You may not provide or use any solution, in whole or in part, to or by another student.

You are encouraged to discuss the concepts involved in the questions with other students. If you are in doubt as to what constitutes acceptable discussion, please ask! Further, please take advantage of office hours offered by the instructor and the TA if you are having difficulties with this assignment.

DO NOT:

- Give/receive code or proofs to/from other students
- Use Google to find solutions for assignment

DO:

- Meet with other students to discuss assignment (it is best not to take any notes during such meetings, and to re-work assignment on your own)
- Use online resources (e.g. Wikipedia) to understand the concepts needed to solve the assignment

Question 1 (10 marks)

Show that the minimizer for least-squares linear regression with L_2 regularization is $\mathbf{w} = (\lambda \mathbf{I} + \Phi^T \Phi)^{-1} \Phi^T \mathbf{t}$, as in Eqn. 3.28 in PRML.

Question 2 (5 marks)

Show that the K -nearest neighbour density model defines an improper distribution whose integral over all space is divergent. (Question 2.61 in PRML)

Question 3 (20 marks)

Question 3.11 in PRML.

Question 4 (40 marks)

In this question you will implement linear basis function regression with polynomial and Gaussian bases.

Start by downloading the code and dataset from the website. The dataset is the **AutoMPG** dataset from the UCI repository. The task is to predict fuel efficiency (miles per gallon) from 7 features describing a car.

Functions are provided for loading the data¹, 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.

You may also find the provided function `designMatrix.m` useful.

Polynomial basis functions

Implement linear basis function regression with polynomial basis functions. Perform the following experiments:

1. Using the first 100 points as training data, and the remainder as testing data, fit a polynomial basis function regression for degree 1 to degree 10 polynomials. Do not use any regularization. Plot training error and test error (in RMS error) versus polynomial degree. **Put this plot, along with a brief comment on what you see, in your report.** For the basis functions:
 - (a) Include the bias function as always.
 - (b) For each input variable $x_i, i = 1, \dots, 7$, and for each power $k = 1, \dots, degree$, there is a basis function $\Phi_{i,k}$ that returns x_i^k . So you should have $7 \cdot degree$ basis functions, plus the bias function, and hence that many weights. In words, treat each input variable as separate single variable and then follow the treatment in the book and use the single-variable powers up to the degree. The degree (maximum power) should run from 1 to 10.
2. It is difficult to visualize the results of high-dimensional regression. Instead, only use one of the features (use `X_n(:,3)`) and again perform polynomial regression. Produce plots of the training data points, learned polynomial, and test data points. The code `visualize_1d.m` may be useful. **Put 2 or 3 of these plots, for interesting (low-order, high-order) results, in your report. Include brief comments.**

¹Note 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.

3. Implement L_2 -regularized regression. Again, use the first 100 points, and only use the 3rd feature. Fit a degree 8 polynomial using $\lambda = \{0, 0.01, 0.1, 1, 10, 100, 1000\}$. Use 10-fold cross-validation to decide on the best value for λ . Produce a plot of validation set error versus regularizer value. Use a `semilogx` plot, putting regularizer value on a log scale. **Put this plot in your report, and note which regularizer value you would choose from the cross-validation.**

Gaussian basis functions

Implement linear basis function regression with Gaussian basis functions. You may use the supplied `dist2.m` function. For the centers $\boldsymbol{\mu}_j$ use randomly chosen training data points (use `randperm` in MATLAB). Set $s = 2$. Perform the following experiments:

1. Using the first 100 points as training data, and the remainder as testing data, fit a Gaussian basis function regression using 5, 15, 25, \dots , 95 basis functions. Do not use any regularization. Plot training error and test error (in RMS error) versus number of basis functions. **Put this plot, along with a brief comment on what you see, in your report.**
2. Implement L_2 -regularized regression. Again, use the first 100 points (do **not** only use the 3rd feature, use them all). Fit a regression model with 90 basis functions using $\lambda = \{0, 0.01, 0.1, 1, 10, 100, 1000\}$. Use 10-fold cross-validation to decide on the best value for λ . Produce a plot of validation set error versus regularizer value. Use a `semilogx` plot, putting regularizer value on a log scale. **Put this plot in your report, and note which regularizer value you would choose from the cross-validation.**

Question 5 (25 marks)

We developed the technique of kernel density estimation in lecture. Kernel density estimation can also be used for regression. Consider the kernel density estimate of the joint distribution:

$$p(\mathbf{x}, t) = \frac{1}{N} \sum_{n=1}^N f(\mathbf{x} - \mathbf{x}_n, t - t_n)$$

A regression estimate can be created by considering the expectation $y(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}]$.

This expectation is computed as:

$$y(\mathbf{x}) = \frac{\sum_{n=1}^N t_n g(\mathbf{x} - \mathbf{x}_n)}{\sum_{m=1}^N g(\mathbf{x} - \mathbf{x}_m)}$$

where $g(\mathbf{x})$ is the marginal distribution of the kernel function $f(\cdot, \cdot)$ used above:

$$g(\mathbf{x}) = \int f(\mathbf{x}, t) dt$$

This regression estimate is known as the Nadaraya-Watson model, kernel regression, and locally-weighted regression. The derivation is provided in Sec. 6.3.1 of PRML.

1. Implement the Nadaraya-Watson regression model using Gaussian kernels $f(\cdot, \cdot)$. Use $g(\mathbf{x}) = \exp(-\|\mathbf{x}\|^2/2h^2)$.

Again, use the first 100 points (do **not** only use the 3rd feature, use them all). Fit a model using $h = \{0.01, 0.1, 0.25, 1, 2, 3, 4\}$. Use 10-fold cross-validation to decide on the best value for h . Produce a plot of validation set error versus kernel width. **Put this plot in your report, and note which kernel width you would choose from the cross-validation.**

2. If $f(\cdot, \cdot)$ is a Gaussian, why is $g(\mathbf{x})$ a Gaussian? (One line answer, read Sec. 2.3 if you don't know.)

Submitting Your Assignment

You must create your assignment in electronic form in **PDF format**. Submit your assignment using the *new* online assignment submission server at: <https://courses.cs.sfu.ca>. **Check early that you are set up for this, especially that you can log in to the server.**

You should create a report with the answers to questions and figures described above. Make sure it is clear what is shown in each figure. **DO NOT INCLUDE SOURCE CODE.**