Computational Physics III
Machine Learning
WS-18

Distribution: 25/10/18, Due: 31/10/18

Overview

In this exercise we will go through some basic Polynomial (including Linear) Regression that will be helpful in understanding some terminology used later on in Machine Learning and is intended to give the students some intuition about the major challenges that any machine learning algorithm faces. The task involves fitting data with polynomials of different order. Formally, this goes under the name of polynomial regression. It will be seen how our ability of prediction depends on the number of data points we have, the "noise" in the data, and our knowledge about relevant features.

The Prediction Problem

Consider a probabilistic process that gives rise to labeled data (x, y). The data is generated by drawing samples from the equation

$$y_i = f(x_i) + \eta_i,$$

where $f(x_i)$ is some fixed, but (possibly unknown) function, and η_i is a Gaussian, uncorrelated noise variable such that

$$\langle \eta_i \rangle = 0$$

$$\langle \eta_i \eta_j \rangle = \delta_{ij} \sigma$$

We will refer to the $f(x_i)$ as the **true features** used to generate the data.

To make prediction, we will consider a family of functions $g_\alpha(x; \theta_\alpha)$ that depend on some parameters θ_α. These functions represent the **model class** that we are using to try to model the data and make predictions. The $g_\alpha(x; \theta_\alpha)$ encode the class of **features** we are using to represent the data.

To learn the parameters $\boldsymbol{\theta}$, we will train our models on a **training data set** and then test the effectiveness of model on a *different dataset*, the **test data set**. The reason we must divide our data into a training and test dataset is that the point of machine learning is to make accurate predictions about new data we have not seen. As we will see below, models that give the best fit to the training data do not necessarily make the best predictions on the test data. This will be a running theme that we will encounter repeatedly in machine learning.

The task in this tutorial is to model the data with polynomials and make predictions about the new data that we have not seen. We will consider two qualitatively distinct situations.

1. In the first case, the process that generates the underlying data is in the model class we are using to make predictions. For polynomial regression, this means that the functions $f(x_i)$ are themselves polynomials.

2. In the second case, our data lies outside our model class. In the case of polynomial regression, this could correspond to the case where the $f(x_i)$ is a 10-th order polynomial but $g_\alpha(x;\theta_\alpha)$ are polynomials of order 1 or 3.

In the exercises we consider 3 model classes:

1. the case where the $g_\alpha(x;\theta_\alpha)$ are all polynomials up to order 1 (linear models),
2. the case where the $g_\alpha(x;\theta_\alpha)$ are all polynomials up to order 3,
3. the case where the $g_\alpha(x;\theta_\alpha)$ are all polynomials up to order 10.

To measure our ability to predict, we will learn our parameters by fitting our training dataset and then making predictions on our test data set. One common measure of predictive performance of our algorithm is to compare the predictions,$\{y_j^{\text{pred}}\}$, to the true values $\{y_j\}$. A commonly employed measure for this is the sum of the average square-error on the test set

$$ASE = \sum_{j=1}^{N_{\text{test}}} (y_j^{\text{pred}} - y_j)^2$$

We will return to this in later tutorials. For now, we will try to get a qualitative picture by examining plots on test and training data.

Fitting vs. predicting when data is in the model class

We start by considering the case:

$$f(x) = 2x.$$

Then the data is clearly generated by a model that is contained within all three model classes we are using to make predictions (linear models, third order polynomials, and tenth order polynomials).

Run your code for the following cases:

1. For $f(x) = 2x$, $N_{\text{train}} = 10$ and $\sigma = 0$ (noiseless case), train the three classes of models (linear, third-order polynomial, and tenth order polynomial) for a training set when $x_i \in [0, 1]$. Make graphs comparing fits for different order of polynomials. Which model fits the data the best?
2. Do you think that the data that has the least error on the training set will also make the best predictions? Why or why not? Can you try to discuss and formalize your intuition? What can go right and what can go wrong?
3. Check your answer by seeing how well your fits predict newly generated test data (including on data outside the range you fit on, for example $x \in [0, 1.2]$) using your code. How well do you do on points in the range of x where you trained the model? How about points outside the original training data set? Make graphs for this.
4. Repeat the exercises above for $f(x) = 2x$, $N_{\text{train}} = 10$, and $\sigma = 1$. What changes?
5. Repeat the exercises above for $f(x) = 2x$, $N_{\text{train}} = 100$, and $\sigma = 1$. What changes?
6. Summarize what you have learned about the relationship between model complexity (number of parameters), goodness of fit on training data, and the ability to predict well.

Fitting vs. predicting when data is not in the model class

Thus far, we have considered the case where the data is generated using a model contained in the model class. Now consider $f(x) = 2x - 10x^5 + 15x^{10}$. *Notice that for the linear and third-order polynomial the true model $f(x)$ is not contained in model class $g_\alpha(x)$* .

1. Repeat the exercises above, fitting and predicting for $f(x) = 2x - 10x^5 + 15x^{10}$ for $N_{\text{train}} = 10, 100$ and $\sigma = 0.1$. Record your observations.
2. Do better fits lead to better predictions?
3. What is the relationship between the true model for generating the data and the model class that has the most predictive power? How is this related to the model complexity? How does this depend on the number of data points N_{train} and σ?
4. Summarize what you think you learned about the relationship of knowing the true model class and predictive power.

Note: Each of the ten parts in the exercise carry 3-points. Please write comments in your code wherever appropriate, make a text file where you give explanation for each part and (importantly) include graphs wherever appropriate.