Project on Machine Learning

Data Analysis and Machine Learning FYS-MAT3155/FYS4155

Department of Physics, University of Oslo, Norway

Fall semester 2017

Using results from Monte Carlo models for machine learning

Introduction. The aim of this project is to use an already developed Monte Carlo program (either the ising Model or a variational Monte Carlo code) to produce, in case of the Ising model, the energy as function of temperature. For the variational Monte Carlo calculation of interacting electrons in an oscilaltor trap, the data are represented by the ground state energies as functions of the variational parameters.

In its simplest form the energy of the Ising model is expressed as, without an externally applied magnetic field,

$$E = -J \sum_{<kl>}^{N} s_k s_l$$

with $s_k = \pm 1$. The quantity N represents the total number of spins and J is a coupling constant expressing the strength of the interaction between neighboring spins. The symbol $< kl >$ indicates that we sum over nearest neighbors only. We will assume that we have a ferromagnetic ordering, viz $J > 0$. We will use periodic boundary conditions and the Metropolis algorithm only. Alternatively we can use the supplied variational Monte Carlo program which solves Schroedinger's equation for two interacting electrons in a harmonic oscillator trap. Both codes can be found at the webpage of the course under programs.

Part a): Producing the data. If we opt for the Ising model code, we need to generate for every span over the lattice and output of the total energy and the magnetic moment (if we wish to study this quantity as well), that is compute and write to file as function of temperature $\langle E \rangle$ and $\langle |M| \rangle$. We will use a fixed lattice siste of $L \times L = 40 \times 40$. Make sure the calculations have been equilibrated and compute these expectation values for for $T \in [1.0, 3.0]$ with a step in temperature $\Delta T = 0.1$ or smaller. This are the data you will use to

estimate the standard deviation in the next part of the project. You should keep a file for the different temperatures as you will need to compute the standard deviation for each temperature.

Alternatively you can run the variational Monte Carlo program for two interacting electrons confined to move in a harmonic oscillator trap. Here you will produce the variational expectaction value of the energy as a function of the variational parameters α and β. You could single out only a set of such parameters after you have found the minimum energy.

These two cases form then our training data which we will use for our estimates of the standard deviation.

Part b): Estimating the standard deviation. This part deals with widely used resampling methods to find the standard deviation on our data sets. The methods we can use are the Blocking method, the Bootstrap method and the Jackknife method. The latter two are examples of so-called resampling methods. The functions provided under the program folder encode all three methods. The typical situation is to generate data using a C++ or Fortan code, as in part a) above. These data are then used by a post-analysis program in order to perform a statistical analysis.

In case of the Ising model, the final product will be a table of the expectation values of the energy (or the magnetic moment) as function of temperature and with a proper standard deviation. These data will then enter our analysis in parts c-e). We will call these data the training data.

If you opt for the variational Monte Carlo program, you need to make a table of the ground state energy as function of various variational parameters (you can limit yourself to α only) with proper standard deviation estimations.

Your task here is thus to use the data from part a) and generate the standard deviation using one (or more) of the methods discussed above. In general, the Bootstrap method is the most widely used one.

These data will then be used in our next step, where we will try to obtain a functional approximation to our data sets.

Part c): Fitting the data using regression analysis and other methods. With the data from part b), with a proper determination of the standard deviation, the task here is to find a fit to the data using regression methods. Here we will focus on the least square method with a straight line fit first and thereafter using singular value decomposition to fit a function that reproduces the data.

Since obtaining these data points may not be trivial, we want to use these data to fit a function which can allow us to make predictions for values of y which are not in the present set. The perhaps simplest approach is to assume we can parametrize our function in terms of a polynomial of degree $n-1$ with n

points, that is

$$y = y(x) \rightarrow y(x_i) = \tilde{y}_i + \epsilon_i = \sum_{j=0}^{n-1} \beta_i x_i^j + \epsilon_i,$$

where ϵ_i is the error in our approximation.

In order to find the optimal parameters β_i instead of solving the above linear algebra problem, we define a function which gives a measure of the spread between the values y_i (which represent hopefully the exact values) and the parametrized values $\tilde{y}_i$, namely

$$Q(\hat{\beta}) = \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2 = \left(\hat{y} - \hat{\tilde{y}}\right)^T \left(\hat{y} - \hat{\tilde{y}}\right),$$

or using the matrix $\hat{X}$ as

$$Q(\hat{\beta}) = \left(\hat{y} - \hat{X}\hat{\beta}\right)^T \left(\hat{y} - \hat{X}\hat{\beta}\right).$$

The function

$$Q(\hat{\beta}) = \left(\hat{y} - \hat{X}\hat{\beta}\right)^T \left(\hat{y} - \hat{X}\hat{\beta}\right),$$

can be linked to the variance of the quantity y_i if we interpret the latter as the mean value of for example a numerical experiment. When linking below with the maximum likelihood approach below, we will indeed interpret y_i as a mean value

$$y_i = \langle y_i \rangle = \beta_0 x_{i,0} + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_{n-1} x_{i,n-1} + \epsilon_i,$$

where $\langle y_i \rangle$ is the mean value. Keep in mind also that till now we have treated y_i as the exact value. Normally, the response (dependent or outcome) variable y_i the outcome of a numerical experiment or another type of experiment and is thus only an approximation to the true value. It is then always accompanied by an error estimate, often limited to a statistical error estimate given by the standard deviation discussed earlier. In the discussion here we will treat y_i as our exact value for the response variable.

Introducing the standard deviation σ_i for each measurement y_i, we define now the χ^2 function as

$$\chi^2(\hat{\beta}) = \sum_{i=0}^{n-1} \frac{(y_i - \tilde{y}_i)^2}{\sigma_i^2} = \left(\hat{y} - \hat{\tilde{y}}\right)^T \frac{1}{\hat{\Sigma}^2} \left(\hat{y} - \hat{\tilde{y}}\right),$$

where the matrix $\hat{\Sigma}$ is a diagonal matrix with σ_i as matrix elements.

In order to find the parameters β_i we will then minimize the spread of $\chi^2(\hat{\beta})$ by requiring

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \beta_j} = \frac{\partial}{\partial \beta_j} \left[\sum_{i=0}^{n-1} \left(\frac{y_i - \beta_0 x_{i,0} - \beta_1 x_{i,1} - \beta_2 x_{i,2} - \cdots - \beta_{n-1} x_{i,n-1}}{\sigma_i} \right)^2 \right] = 0,$$

which results in

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \beta_j} = -2\left[\sum_{i=0}^{n-1}\frac{x_{ij}}{\sigma_i}\left(\frac{y_i-\beta_0 x_{i,0}-\beta_1 x_{i,1}-\beta_2 x_{i,2}-\cdots-\beta_{n-1}x_{i,n-1}}{\sigma_i}\right)\right]=0,$$

or in a matrix-vector form as

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \hat{\beta}} = 0 = \hat{A}^T\left(\hat{b}-\hat{A}\hat{\beta}\right).$$

where we have defined the matrix $\hat{A}=\hat{X}/\hat{\Sigma}$ with matrix elements $a_{ij}=x_{ij}/\sigma_i$ and the vector $\hat{b}$ with elements $b_i=y_i/\sigma_i$.

We can rewrite

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \hat{\beta}} = 0 = \hat{A}^T\left(\hat{b}-\hat{A}\hat{\beta}\right),$$

as

$$\hat{A}^T\hat{b} = \hat{A}^T\hat{A}\hat{\beta},$$

and if the matrix $\hat{A}^T\hat{A}$ is invertible we have the solution

$$\hat{\beta} = \left(\hat{A}^T\hat{A}\right)^{-1}\hat{A}^T\hat{b}.$$

If we then introduce the matrix

$$\hat{H} = \hat{A}^T\hat{A},$$

we have then the following expression for the parameters β_j (the matrix elements of $\hat{H}$ are h_{ij})

$$\beta_j = \sum_{k=0}^{p-1}h_{jk}\sum_{i=0}^{n-1}\frac{y_i}{\sigma_i}\frac{x_{ik}}{\sigma_i} = \sum_{k=0}^{p-1}h_{jk}\sum_{i=0}^{n-1}b_i a_{ik}$$

We state without proof the expression for the uncertainty in the parameters β_j as

$$\sigma^2(\beta_j) = \sum_{i=0}^{n-1}\sigma_i^2\left(\frac{\partial \beta_j}{\partial y_i}\right)^2,$$

resulting in

$$\sigma^2(\beta_j) = \left(\sum_{k=0}^{p-1}h_{jk}\sum_{i=0}^{n-1}a_{ik}\right)\left(\sum_{l=0}^{p-1}h_{jl}\sum_{m=0}^{n-1}a_{ml}\right) = h_{jj}!$$

The first step here is to approximate the function y with a first-order polynomial, that is we write

$$y = y(x) \rightarrow y(x_i) \approx \beta_0+\beta_1 x_i.$$

By computing the derivatives of χ^2 with respect to β_0 and β_1 show that these are given by

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \beta_0} = -2\left[\sum_{i=0}^{1}\left(\frac{y_i - \beta_0 - \beta_1 x_i}{\sigma_i^2}\right)\right] = 0,$$

and

$$\frac{\partial \chi^2(\hat{\beta})}{\partial \beta_0} = -2\left[\sum_{i=0}^{1} x_i\left(\frac{y_i - \beta_0 - \beta_1 x_i}{\sigma_i^2}\right)\right] = 0.$$

Define then

$$\gamma = \sum_{i=0}^{1}\frac{1}{\sigma_i^2},$$

$$\gamma_x = \sum_{i=0}^{1}\frac{x_i}{\sigma_i^2},$$

$$\gamma_y = \sum_{i=0}^{1}\left(\frac{y_i}{\sigma_i^2}\right),$$

$$\gamma_{xx} = \sum_{i=0}^{1}\frac{x_i x_i}{\sigma_i^2},$$

$$\gamma_{xy} = \sum_{i=0}^{1}\frac{y_i x_i}{\sigma_i^2},$$

and show that

$$\beta_0 = \frac{\gamma_{xx}\gamma_y - \gamma_x\gamma_y}{\gamma\gamma_{xx} - \gamma_x^2},$$

$$\beta_1 = \frac{\gamma_{xy}\gamma - \gamma_x\gamma_y}{\gamma\gamma_{xx} - \gamma_x^2}.$$

Find these parameters for your data set and compare the fitted function $y = y(x) \rightarrow y(x_i) \approx \beta_0 + \beta_1 x_i$ and find the error in the parameters β_0 and β_1 as well. How does your parametrization compare with the input data y_i? Give a critical discussion of the method.

Develop now a program where you extract the parameters β_i and their corresponding errors using the general Least Square method (LSM), with polynomials of degree $p = 2$ and $p = 3$. Which polyomials gives the best fit to your data? Here you should prepare an input file with the number of data points in your training data set, the corresponding standard deviation σ_i as well as the degree p of the polynomial you want to fit.

The LSM suffers often from both being underdetermined and overdetermined in the unknown coefficients β_i. A better approach is to use the Singular Value Decomposition (SVD) method discussed in the lecture notes on regression analysis. Here you should use the SVD programs and use the same input file you used for the general LSM. In the main functions you should specifiy

which function you wish to approximate the date with. The example programs discussed above show you examples on how to use these codes as well as how to write the functional form of the polynomial or fitting function.

Part d): Introducing Bayesian statistics. More text to come

Part e): Studying the Ising model or the VMC results with Neural networks. More text to come

Background literature

If you wish to read more about the Ising model and statistical physics here are three suggestions.

- M. Plischke and B. Bergersen, *Equilibrium Statistical Physics*, World Scientific, see chapters 5 and 6.
- D. P. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*, Cambridge, see chapters 2,3 and 4.
- M. E. J. Newman and T. Barkema, *Monte Carlo Methods in Statistical Physics*, Oxford, see chapters 3 and 4.

Introduction to numerical projects

Here follows a brief recipe and recommendation on how to write a report for each project.

- Give a short description of the nature of the problem and the eventual numerical methods you have used.
- Describe the algorithm you have used and/or developed. Here you may find it convenient to use pseudocoding. In many cases you can describe the algorithm in the program itself.
- Include the source code of your program. Comment your program properly.
- If possible, try to find analytic solutions, or known limits in order to test your program when developing the code.
- Include your results either in figure form or in a table. Remember to label your results. All tables and figures should have relevant captions and labels on the axes.
- Try to evaluate the reliabilty and numerical stability/precision of your results. If possible, include a qualitative and/or quantitative discussion of the numerical stability, eventual loss of precision etc.

- Try to give an interpretation of you results in your answers to the problems.
- Critique: if possible include your comments and reflections about the exercise, whether you felt you learnt something, ideas for improvements and other thoughts you've made when solving the exercise. We wish to keep this course at the interactive level and your comments can help us improve it.
- Try to establish a practice where you log your work at the computerlab. You may find such a logbook very handy at later stages in your work, especially when you don't properly remember what a previous test version of your program did. Here you could also record the time spent on solving the exercise, various algorithms you may have tested or other topics which you feel worthy of mentioning.