Machine Learning

FYS-STK 4155

Project 2

Kari Eriksen

Abstract

Contents

1 Introduction

In this project we will deal with different methods within the field of machine learning to solve different types of problems, such as classification and regression. We will be looking at the Ising model in both 1D and 2D. We begin with estimating the coupling constant J for the 1 dimensional case using linear regression methods and evaluating the calculations with bootstrap. In 2 dimension the Ising model experiances a phase transition when there is a change in temperature. This phase shift we can define as a classification problem and try to train the different models to identify these. We will be using both logistic regression and a neural network for this purpose.

2 Theory

2.1 Ising model

The Ising model is a mathematical model named after Ernst Ising and it describes a system of spins in a lattice where the energy of the system can be found through eq. 1. J represents the coupling constant, s_k and s_l are spins with value either $+1$ or -1 (up or don), N is the total number of spins in the system and B is the external magnetic field which in this project is zero. We therefore look at a system with energy equal to eq. 2. The spins themself represents magnetic dipole moments and the lattice allow each spin to interact with its neighbors, indicated by the symbol $< kl >$ in the sum.

$$E = -J \sum_{<kl>}^{N} s_k s_l - B \sum_{k}^{N} s_k \tag{1}$$

$$E = -J \sum_{<kl>}^{N} s_k s_l \tag{2}$$

In 1 dimension there is no phase transition and we will use regression in order to determine the coupling constant between the spins in the system.
In 2 dimensions or higher however the system goes through a phase transition with change in temperature. Below the critical temperature, $T_C \approx 2.269$, the system is what we will call an ordered state. The spins tend to be aligned which causes a net magnetization of the system, also described as a ferromagnet. Above the critical temperature the coupling constant is smaller than zero and the spins interact in an antiferromagnetic way. We wish to train our model to classify a configurations of spins as an ordered or disordered system. To do so we begin with the 1D Ising model with nearest-neighbor interactions, eq. 3.

$$E[s] = -J \sum_{j=1}^{L} s_j s_{j+1} \tag{3}$$

Now we have L number of spins in a one dimensional array. If we where to have a data set $i = 1, ..., n$ of different configurations on the form $\{(E[s^i], s^i)\}$ we could use this in training a linear model to find the coupling constant. That way we could use much of the code from project 1. To do so we use the all-to-all Ising model and notice that the energy is linear in J.

$$E_{model}[s^i] = -\sum_{j=1}^{L}\sum_{k=1}^{L} J_{j,k} s_j^i s_k^i \tag{4}$$

We can now recast this problem to a linear regression model, rewriting all two-body interactions $\{s_j^i s_k^i\}_{j,k}^L$ as a vector $\mathbf{X}^i$.

$$E_{model}^i \equiv \mathbf{X}^i \cdot \mathbf{J} \tag{5}$$

3 Methods

3.1 Linear Regression

Linear regression is a method in statistics that predicts the response of one or several explanatory variables. It assumes a linear relationship between the dependent and independent variables. At its simplest form we could try to find the stright line between two points. This equation is fairly simple.

$$\hat{y} = \hat{\beta_0} + \hat{\beta_1} x + \hat{\epsilon} \tag{6}$$

Here $\hat{y}$ is a dependent variable, the outcome, x is an independent variable, or the predictor, and $\hat{\beta_0}$ and $\hat{\beta_1}$ the intercept and slope respectively. ϵ is the error in our prediction. The solution for $\hat{\beta_0}$ and $\hat{\beta_1}$ in this problem is best found with least square and is also fairly easy. Calculating the mean over both variables ($\bar{x}$ and $\bar{y}$) we can find the parameters that give the prediction that differs the least from the exact solution.

$$\beta_1 = \frac{\sum^n (x_i - \bar{x})(y_i - \bar{y})}{\sum^n (x_i - \bar{x})^2} \tag{7}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x} \tag{8}$$

If we have several predictors we can extend our problem to a more general case.

$$\hat{y} = f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j \tag{9}$$

Now X is a vector containing all predictors, $X^\top = \{X_0, X_1, X_2, X_3, ..., X_p\}$, β_0 is the intercept and β_j is a vector keeping all coefficients for each predictor, the parameters we are searching for. $\hat{y}$ is the predicted values of $y = f(X)$. Moving β_0 to the $\beta -$ vector and adding an extra column with 1's to the design matrix $X^\top$ we can reduce the problem to vector form and get the following. We will make use of this notation when finding solutions using least square etc.

$$\hat{y} = \hat{X}\hat{\beta} + \epsilon \tag{10}$$

3.1.1 Ordinary Least Square

The least square method selects the parameters β so that residual sum of squares (RSS) is minimized.

$$\text{RSS}(\beta) = \sum_{i=1}^{N}(y_i - x_i^\top \beta)^2 \tag{11}$$

y_i is still the independent variable, and $x_i^\top \beta$ represents the prediction of outcome given the calculated parameter β. And the difference between these variables squared gives us the RSS of the parameter β. β is a vector $p+1$ long, the number of features (plus the intercept) in the design matrix.

This can be expressed in matrix notation, using eq. 10. To find an expression for the β-parameter we look for the minimum of the RSS, meaning we take its derivative wrt. β.

$$\text{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^\top(\mathbf{y} - \mathbf{X}\beta) \tag{12}$$

$$\frac{\partial \text{RSS}(\beta)}{\partial \beta} = 0 = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\beta) \tag{13}$$

$$\mathbf{X}^\top\mathbf{y} - \mathbf{X}^\top\mathbf{X}\beta = 0 \tag{14}$$

$$\hat{\beta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \tag{15}$$

This is the expresion we use in the ordinary least square-method in order to find the optimal β-values. This method depends on the propertie $\mathbf{X}^\top\mathbf{X}$ being postive definit in order to be able to calculate its inverse. In case it is not we must use other method.

3.1.2 Ridge Regression

As mentioned in the section above we may come across problems where the columns in X are not linear independent, often an issue for problems in high dimesions. Then the coefficients in β are not uniquely defined through least square. This was the motivation for what would be the Ridge regression, an ad hoc solution to the singularity of $\mathbf{X}^\top\mathbf{X}$ introduced by Hoerl and Kennard (1970). They suggested adding a tuning parameter λ, i.e. a penalty to the sizes of the coefficients.

$$\mathbf{X}^\top\mathbf{X} \;\rightarrow\; \mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I} \tag{16}$$

By doing the replacement above we are able to calculate the inverse and can again find the expression for β but this time through minimizing the penalized RSS. The solution for β is eq. 15, which is now dependent on the parameter λ.

$$\mathrm{PRSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^\top(\mathbf{y} - \mathbf{X}\beta) + \lambda||\beta||^2 \tag{17}$$

$$\frac{\partial \mathrm{PRSS}(\beta)}{\partial \beta} = 0 = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\beta) + 2\lambda\beta \tag{18}$$

$$\hat{\beta}(\lambda) = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y} \tag{19}$$

$\mathbf{I}$ is the identity matrix, a $p \times p$ matrix, and $\lambda \in [0, \infty]$. The tuning parameter λ determines the regularization of the problem and different λ will give different solution to the regression problem. Our task will be to find an optimal parameter for our case.

$$\hat{\beta}^{ridge} = \mathrm{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \right\} \tag{20}$$

We can see from eq. 20 that the method assumes our design matrix is centered, the intercept does not depend on the tuning parameter. β_0 is instead found by calculating the mean of y.

$$\beta_0 = \bar{y} = \frac{1}{N}\sum_{i}^{N} y_i \tag{21}$$

3.1.3 Lasso Regression

In 1996 Tibshirani suggested a new penalty, the Lasso. Similar to ridge regression but the difference lies in the last part.

$$\hat{\beta}^{lasso} = \mathrm{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \right\} \tag{22}$$

As for the ridge regression the intercept is given by the mean of y.

3.2 Singular value decomposition

As mentioned in 3.1.1 we are dealing with a design matrix that is singular meaning $\mathbf{X}^\top\mathbf{X}$ is not invertible. One way to solve this problem is with the use of singular value decomposition (SVD). We can rewrite $\mathbf{X}$ as the factorization of a $n \times p$ unitary matrix $\mathbf{U}$, a $p \times p$ diagonal matrix $\mathbf{\Sigma}$ and the conjugate transpose of a $p \times p$ unitary matrix $\mathbf{V}$.

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \tag{23}$$

Now the inverse of the matrix product $\mathbf{X}^\top\mathbf{X}$ can be written as the inverse of the matrix product of the SVD.

The Moore-Penrose pseudoinverse is defined as $\mathbf{X}^+ = (\mathbf{X}^\top\mathbf{X})^{-1}$. Using the SVD we get eq. 24.

$$\mathbf{X}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\top \tag{24}$$

Now we can rewrite the solution of β as eq. 15.

$$\beta = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\top\mathbf{y} \tag{25}$$

3.3 Logistic regression

$$\mathbf{x}_i^\top\mathbf{w} + b_0 \equiv \mathbf{x}_i^\top\mathbf{w} \tag{26}$$

$$f(\mathbf{x}_i^\top\mathbf{w}) = \frac{1}{1+e^{-\mathbf{x}_i^\top\mathbf{w}}} \tag{27}$$

$y_i = \{0, 1\}$

$$C(\beta) = \sum_{i=1}^{N} -y_i \log(f(X_i^\top\beta) - (1-y_i)\log[1 - f(X_i^\top\beta)]) \tag{28}$$

$$P(y_i = 1|\mathbf{x}_i, \theta) = \frac{1}{1+e^{-\mathbf{x}_i^\top\mathbf{w}}} \tag{29}$$

3.4 Neural Network

$$\Delta_j^l = \frac{\partial E}{\partial z_j^l} = \frac{\partial E}{\partial a_j^l}\sigma'(z_j^l) \tag{30}$$

$$\Delta_j^l = \frac{\partial E}{\partial z_j^l} = \frac{\partial E}{\partial b_j^l}\frac{\partial b_j^l}{\partial z_j^l} = \frac{\partial E}{\partial b_j^l} \tag{31}$$

$$\Delta_j^l = \frac{\partial E}{\partial z_j^l} = \sum_k \frac{\partial E}{\partial z_k^{l+1}}\frac{\partial z_j^{l+1}}{\partial z_j^l} \tag{32}$$

$$= \sum_k \Delta_k^{l+1}\frac{\partial z_k^{l+1}}{z_k^l} \tag{33}$$

$$= \left(\sum_k \Delta_k^{l+1}\beta_{kj}^{l+1}\right)\sigma'(z_j^l) \tag{34}$$

$$\frac{\partial E}{\partial \beta_{jk}^l} = \frac{\partial E}{\partial z_j^l}\frac{\partial z_j^l}{\partial \beta_{jk}^l} = \Delta_j^l a_k^{l+1} \tag{35}$$

3.5 Stochastic gradient descent

Gradient descent
  minimizing the cost function

$$E(\theta) = \sum_{i=1}^{N} e_i(X_i, \theta) \tag{36}$$

$$\mathbf{v_t} = \eta_t \nabla_\theta E(\theta_t) \tag{37}$$

$$\theta_{t+1} = \theta_\mathbf{t} - \mathbf{v_t} \tag{38}$$

$$\nabla_\theta E(\theta) = \sum_{i}^{n} \nabla_\theta e_i(\mathbf{x}_i, \theta) \longrightarrow \sum_{i \in B_k} \nabla_\theta e_i(\mathbf{x}_i, \theta) \tag{39}$$

$$\nabla_\theta E^{MB}(\theta) = \sum_{i \in B_k}^{M} \nabla_\theta e_i(\mathbf{x}_i, \theta) \tag{40}$$

$$\mathbf{v_t} = \eta_t \nabla_\theta E^{MB}(\theta) \tag{41}$$

$$\theta_{t+1} = \theta_\mathbf{t} - \mathbf{v_t} \tag{42}$$

4 Resampling Methods

4.1 Bootstrap

The bootstrap is a resampling method suggested by Efron in 1979. It tell us how well our regression models assess the problem at hand. To say something about this it is commen to calculate the statistical properties given in the section above, particularly the MSE. As we can see from eq. 43 this is a measurment consisting of three properties. The variance, the bias and the irreducible error.
In machine learning phenomena such as overfitting and underfitting are highly important to be aware of and is connected with the MSE. Our goal is to predict the outcome $\hat{y}$ given some observed data. To do so we split our data into a training set and a test set and train the model with the training data. Depending on how well we fit our model to the training data we may overfit or underfit. Overfitting means that we fit the data so well that we have made the model to close and dependent on the training data. On the other hand we can underfit, meaning that we miss many important features of the data. In both cases trying the model out on the test set we get bad results. This is what is known as the bias-variance tradeoff.
The variance tells us how the predicted outcome differs from its mean, and a high variance

will correspond to overfitting. The bias says how much difference there is between the models predicted value and the true value. A high bias corresponds to underfitting.
Since our predictor is a random variable, how we draw the data will effect the estimate of the response. In bootstrap we draw samples with replacement from our dataset (the training data) and fit the model with this. Then we test the model with the test data and calculate the errors. Doing this many times we can examine the behavoir of the fit and get a more accurate estimate of the errors.

5 Statistical Properties

$$\begin{aligned}
\text{MSE} &= \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 && (43)\\
&= \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}}_i)^2 + \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{\hat{y}}_i)^2 + \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{\hat{y}}_i)(\bar{\hat{y}}_i - \hat{y}) && (44)\\
&= Var(\hat{y}) + Bias + \epsilon && (45)
\end{aligned}$$

$$\text{R}^2(y,\hat{y}) = 1 - \frac{\sum_{n}^{i=1}(y_i - \hat{y}_i)^2}{\sum_{n}^{i=1}(y_i - \bar{y})^2} \tag{46}$$

6 Implementation

6.1 Part b: 1D Ising with linear regression

6.2 Part c: 2D Ising with logistic regression

6.3 Part d: 1D Ising with neural network

6.4 Part e: 2D Ising with neural network

7 Results

7.1 Part b: 1D Ising with linear regression

7.2 Part c: 2D Ising with logistic regression

7.3 Part d: 1D Ising with neural network

7.4 Part e: 2D Ising with neural network

7.5 Part f: Critical evaluation

8 Discussion

9 Conclusion

References

[1] Trevor Hastie, *The Elements of Statistical Learning*, Springer, New York, 2nd edition, 2009.

[2] Gareth James, *An Introduction to Statistical Learning*, Springer, New York, 2013.

[3] https://compphysics.github.io/MachineLearning/doc/pub/Regression/html/._Regression-bs000.html, 08/10-18.

[4] https://ml.berkeley.edu/blog/2017/07/13/tutorial-4/, 08/10-18.

[5] https://arxiv.org/pdf/1803.08823.pdf,, 08/10-18.

[6] https://www.jstor.org/stable/pdf/2346178.pdf, 08/10-18.

[7] http://math.arizona.edu/~hzhang/math574m/Read/RidgeRegressionBiasedEstimationForNor
pdf, 08/10-18.

[8] http://statweb.stanford.edu/~tibs/sta305files/Rudyregularization.pdf, 08/10-18.