Statistical Mechanics of Complex Systems

Supervised Learning & Bayesian Inference

Alberto Chimenti - 1210360; Anna Braghetto - 1205200; Davide Maniscalco - 1212063

1 2D Ising Model

The Hamiltonian for the classical Ising model is given by

$$H = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j \quad \text{with} \quad \sigma_i = \pm 1$$

The lattice site indices i, j run over all nearest neighbours of a $2D$ square lattice, and J is some arbitrary interaction energy scale. We adopt periodic boundary conditions. Onsager proved that this model undergoes a phase transition in the thermodynamic limit from an ordered ferromagnet with all spins aligned to a disordered phase at the critical temperature

$$\frac{T_c}{J} = \frac{2}{log(1+\sqrt{2})} \simeq 2.26 \, .$$

2 Example of phases classification

Given an Ising state without any additional information, we want to classify if it belongs to the ordered or the disordered phase. This categorical machine learning problem is well suited for logistic regression. To see how it works, we consider the $2D$ Ising model on a 40×40 square lattice, and use Monte-Carlo sampling to prepare 10^4 states at every fixed temperature T out of a pre-defined set.
The available dataset is made up of ordered ($T/J < 2.0$) and disordered ($T/J > 2.5$) states, and of a critical region around the critical temperature T_c ($2.0 < T/J < 2.5$). We furthermore assign a label to each state according to its phase: 0 if the state is disordered, and 1 if it is ordered.

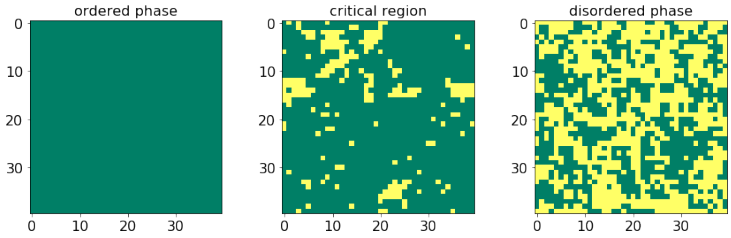

Figure 1: Three examples of states classification, one for each phase.

For all the models (Deep neural network and Random forest) the data has been divided into 3 subsets. Train and test set are constructed using the samples from ordered and disordered phase, successively the performance of the algorithm has been tested using data samples in the critical region.

This might seem a good way to check the performance since the distinguishability of the two phases in the critical region is less sharp than the one provided in the training stage.

2.1 Classification with Neural Network

At first, the two states of the 2D Ising Model are classified with Deep Neural Network according to their phases: ordered and disordered.

The logistic regression is performed by Deep Neural Network that are built using TensorFlow backend. A very simple network structure has been chosen with one single hidden layer with a ReLu rectifier. The number of the hidden neurons was allowed to vary and same holds for SGD learning rate. Then, the best hyperparameters are chosen in order to maximize the accuracy of the model with grid search.

The results obtained for the training set are shown in the Fig. 2.

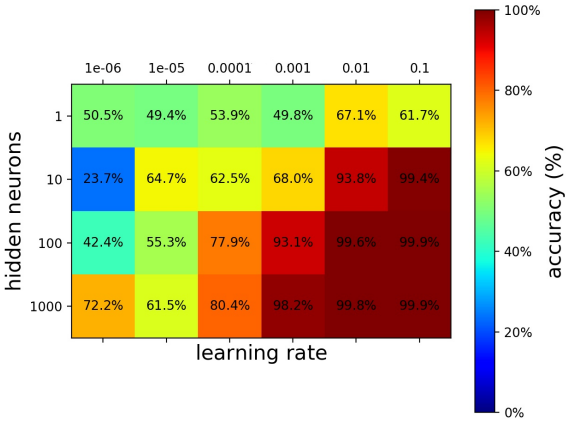

Figure 2: Accuracy as a function of the number of hidden neurons and learning rate for the **training** dataset.

The maximum value of the accuracy for the training set is very high; almost 100% of the states are well classified for learning rate set to 0.1 and 100 or 1000 neurons and learning rate set to 0.01 and 100 neurons.

The performance of the DNN are shown for the test set in Fig. 3.

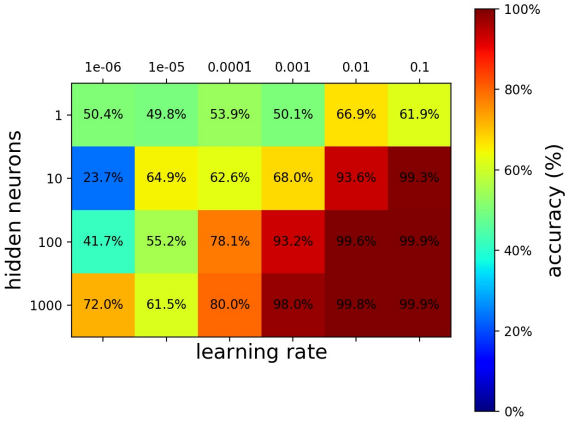

Figure 3: Accuracy as a function of the number of hidden neurons and learning rate for the **test** dataset.

The behaviour of the accuracy for the test set is quite similar to the one observed for the training set, shown in Fig. 2. The best parameters are equal to the train ones (learning rate=0.1, neurons=100,1000 and learning rate=0.01, neurons=100).

The good results obtained for the training and the test dataset show how the ordered and disordered phases of the Ising Model are well classified with Deep Neural Network.

The behaviour of the accuracy obtained for the critical dataset, containing just the critical region, is shown in Fig. 4.

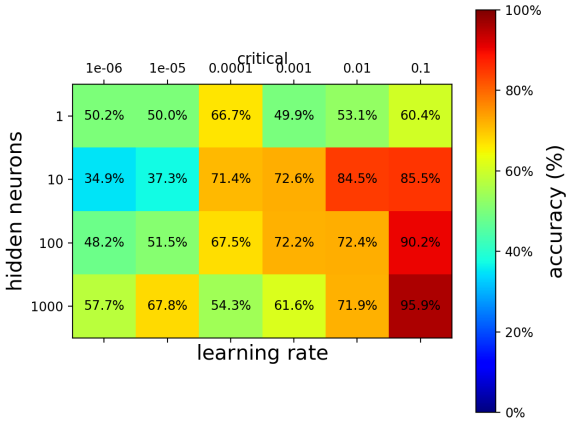

Figure 4: Accuracy as a function of the number of hidden neurons and learning rate for the critical dataset.

For the critical region, the maximum value of the accuracy (96%) is lower than the ones obtained for training and test dataset (99.9%): the classification is more difficult around the critical region because the ordered and disordered configurations are quite similar. This might lead us to think that there was an overfitting problem in some of the parameters selection that performed well on the non critical data. Moreover we can underline the fact that since we have 40*40 neurons in the input layer, the DNN performs better with a comparable number of neurons in the hidden one which probably prevents it to loose some information in the "encoding" procedure.

The computational time is reported in order to study how the complexity of the model affects the time needed to train the DNN.

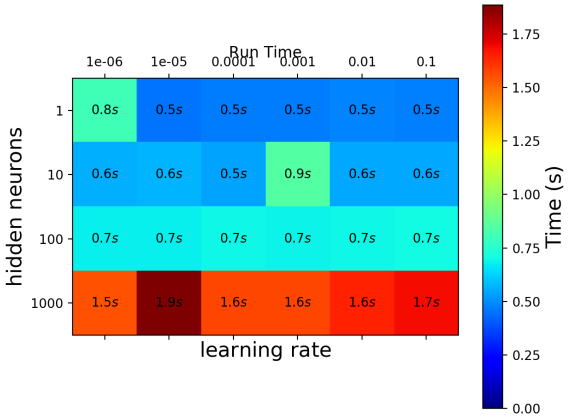

Figure 5: Needed time as a function of the number of hidden neurons and learning rate for the critical dataset.

As expected as the complexity of the model (i.e. the number of neurons increases) increases, the training time increases. The behaviour of time with respect to the learning rate does not follow any rule.

2.2 Classification with Random Forest

The classification is performed also with a Random Forest Classifier. There are two important hyperparameters left to vary which affect the performance of the algorithm: the number of estimators in the ensemble and the depth of the trees used.

At first the classification is performed with just extremely fine trees (2 leaves) or extremely coarse trees (=10000) and varying the number of trees from 10 to 100.

The results obtained for the training, test and critical sets are shown below.

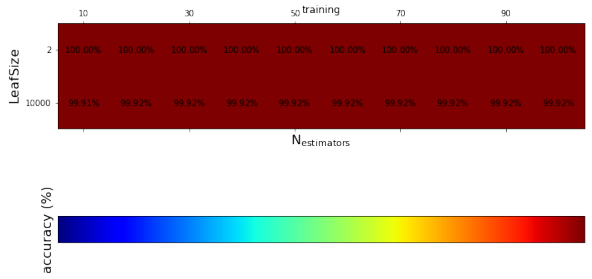

Figure 6: Accuracy as a function of the number of trees and leaf size for the **training** dataset.

As shown in Fig. 6 the accuracy is 100% for the fine configurations while it is 99.9% for the coarse configurations.

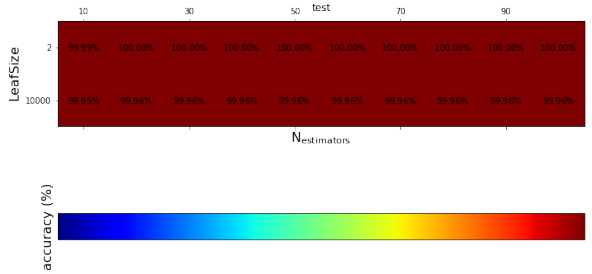

Figure 7: Accuracy as a function of the number of trees and leaf size for the **test** dataset.

The results for the test set shown in Fig.7 are equal to the ones obtained for the train dataset. Any choice on the hyperparameters performs well.

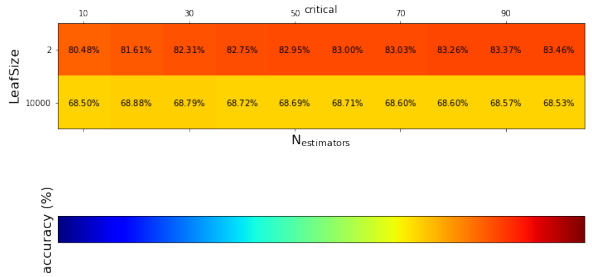

Figure 8: Accuracy as a function of the number of trees and leaf size for the training dataset.

As in for the DNN the perfomance on the critical set are lower: for the critical dataset we observe overfitting. In particular for the fine configurations the accuracy is around 80% while for the course configurations the accuracy is around 68%.

From the results obtained for the critical dataset the best hyperparameters to chose for the Ising classification are: leaf size equals to 2 and 100 estimators.

Also the computational time is analyzed in order to show how the complexity of the model affects the computational cost.

As expected, the more the complexity of the model, the more is the time needed to train it.

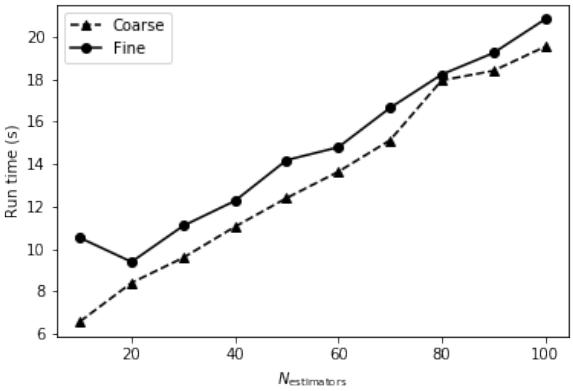

Figure 9: Needed time for fine and coarse configuration as a function of the number of estimators.

2.3 Classification with Extremely Randomized Trees

The classification is also performed with an Extremely Randomized Trees Classifier. The new method introduces more randomness "into the ensemble by randomly choosing features to split on as well as randomly choosing thresholds to split on."

Such as in Random Forest, the hyperparameters left to vary are the leaf size (2 or 1000) and the number of estimators (from 10 to 100).

The results obtained for the training, test and critical sets are shown below.

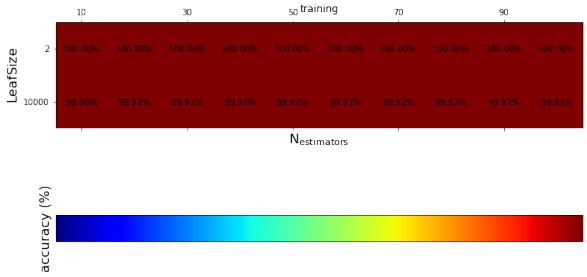

Figure 10: Accuracy as a function of the number of trees and leaf size for the **training** dataset.

The results obtained for the Extremely Randomized Trees are equals to Random Forest: 100% of accuracy for the fine configurations and 99% of accuracy for the fine configurations.

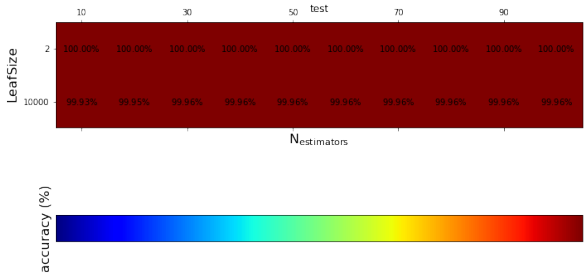

Figure 11: Accuracy as a function of the number of trees and leaf size for the **test** dataset.

Since the performances on the test sets are equals to the training set, there is not overfitting, such as for the Random Forest Classifier: for the fine configurations the accuracy is 100% and it is 99% for the coarse ones.

The results here are quite similar to the ones obtained before with the Random Forest classifier. However we can see an improvement since the performance diminishes less by decreasing the number of trees (especially in the fine configuration).

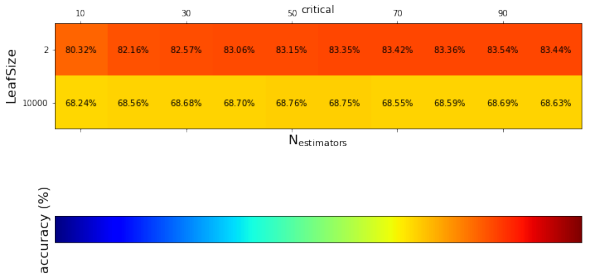

Figure 12: Accuracy as a function of the number of trees and leaf size for the critical dataset.

Again, like in the previous case we observe overfitting on the non-critical data : the accuracy is around 82% and 68% for the fine and the course configurations respectively.

By looking to the perfomance on the critical dataset, the best hyperparamenters of the models are a leaf size set to 2 and 90 estimators.

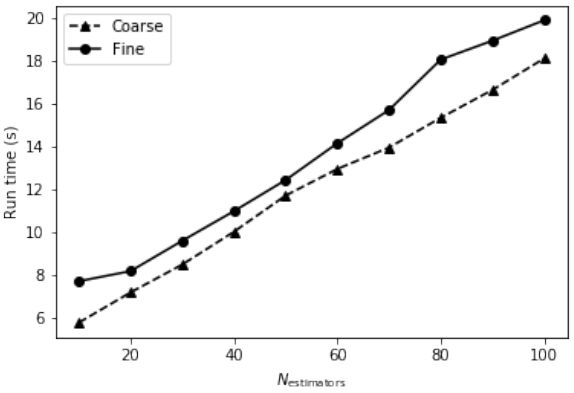

Figure 13: Needed time for fine and course configuration as a function of the number of estimators.

The time needed to train the ETC model is similar to RF.

2.4 Conclusion

By looking to the results obtained for the DNN, RF and ETC, the best model to classify the states of 2D Ising Model is the DNN: the accuracy is higher in the critical set and the computational time to train the model is lower.

3 Bayesian Approach

It might be possible to get information about the system using the following approach.

We can describe the system using the Hamiltoninan:

$$\mathcal{H} = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j$$

where the sum runs over the neighbors.

We might set up the Bayesian approach by means of the Bayes rule:

$$P(model|data) = \frac{P(data|model)P(model)}{P(data)}$$

where we have respectively:

- the **likelihood** P(data|model)
- the **prior** P(model)
- the **posterior** P(model|data)

- P(data) which can simply be considered as a normalization factor

Now the likelihood can be written as:

$$P(data|model) = P(\sigma|\{J,\beta\}) = \frac{1}{\mathcal{Z}(J,\beta)} e^{+\beta J \sum_{\langle ij \rangle} \sigma_i \sigma_j} \quad (1)$$

with

$$\mathcal{Z}(J,\beta) = \sum_{\{\sigma\}} e^{\beta J \sum_{\langle ij \rangle} \sigma_i \sigma_j}$$

For simplicity we can now rewrite the bayesian approach by means of the defining parameters mentioned above; writing $J\beta = k$ as an adimentional quantity and neglecting the normalization factor

$$P(k|\sigma) \simeq P(\sigma|k)P(k)$$

Such an equation can be exploited for an optimization problem for the model parameters k which maximize the probability in the posterior distribution.

We can further expand the equation by substituting the likelihood in (1)

$$\frac{\partial ln P(k|\sigma)}{\partial k} = \Big[\sum_{\langle ij \rangle} \sigma_i \sigma_j - \frac{\partial ln \mathcal{Z}(k)}{\partial k} \Big] + \frac{\partial ln P(k)}{\partial k}$$

We can expand the partition function term like:

$$\frac{\partial ln \mathcal{Z}(k)}{\partial k} = \frac{1}{\mathcal{Z}} \frac{\partial Z}{\partial k} = \big\langle \sum_{\langle ij \rangle} \sigma_i \sigma_j \big\rangle \simeq \big\langle \sum_{\langle ij \rangle} \sigma_i \sigma_j \big\rangle_{data}$$

We could therefore get an estimation of the mean over the ensamble with the mean over our dataset.

Summarizing, this approach leads us to estimate the rescaled temperature of the system; which is not the real temperature but a adimentional quantity relative to the configurations provided in the dataset.

The final optimization problem takes the following form:

$$\Big[\big(\sum_{\langle ij \rangle} \sigma_i \sigma_j \big)_{chosen} - \big\langle \sum_{\langle ij \rangle} \sigma_i \sigma_j \big\rangle_{data} \Big] + \frac{\partial ln P(k)}{\partial k} = 0 \quad (2)$$

In this case we would have to choose a prior such as a Gaussian distribution leading to the final equation for the temperature of the system. For simplicity reason we set the variance of the Gaussian distribution equal to one and therefore obtain:

$$k = \Big[\big(\sum_{\langle ij \rangle} \sigma_i \sigma_j \big)_{chosen} - \big\langle \sum_{\langle ij \rangle} \sigma_i \sigma_j \big\rangle_{data} \Big] \quad (3)$$

It is worth to underline now that this equation should give us the "rescaled temperature" of the chosen configuration. Such a quantity would represent the "distance" of the chosen configuration from the mean over the whole provided dataset.

In this approach, it would be necessary to have an evenly distributed set of configurations, meaning that the average over the data would fall in the critical region. Taking a sample which falls in that region would mean to have a estimated temperature close to zero.

Therefore the classifier would label the chosen sample as ordered or disordered depending on the sign of k.

This framework takes also into account the Z2 symmetry of the system and would give the same results even with shuffled data.

Obviously the sum over the pairs $< i, j >$ would run over the interacting neighbors.