University of Caen

Machine learning and data analysis

GENARD Tom, DAUMAS-TSCHOPP Sacha

February 20, 2020

Abstract

Machine learning has been put on the spotlight this past few years due to recent advancements. It is now widely used in a variety of subjects, including physics. This project is the result of a two week master class on computer science applied to physics problems. It will use the various techniques that have been presented in the lectures and performed during the practice sessions. This report will be focus on few points:

- The purpose of the project
- The different programs to handle it
- The result we obtained and the discussion of the results
- The conclusion

In this first project, we will study the Ising model.

Key words: Machine learning, Neural network, Data Analysis

Contents

1 Acknowledgements

We would like to firstly thank Christopher Jacquiot from the Deep Learning and Computer Science Master at the University of Caen for his help in understanding some concepts, and helping us for what was a very long debugging process for the neural network.

We also want to thank Paul Guibourg and Tristan Le Cornu with which we discussed ideas and concepts, and with which we could discuss and compare our methodologies.

Finally, we would like to thank Morten Hjorth-Jensen for this opportunity to learn machine learning, and also Stian Bilek, Lucas Charpentier and Hanna Svennevik for their help during the laboratory sessions.

2 Introduction

2.1 Goal of the project

In this first project, we were interested in a system which contains a configuration of particles each with a different spins. This system was studied both in a one-dimensional, and a two-dimensional context, with the latter being an analysis under multiple starting conditions. We observed that below a certain temperature, called critical temperature (T_c), the system had ferromagnetic properties. This means that each particle of the system will have the same spin. We have at our disposal a series of data files, generated by a simulation by Mehta et al[3]. which gives a series of 10000 configuration for each temperature. The goal of the first project was to train a model from the data files and compare it with our own fit to which presented in a recent article[3]. To do so, we used several techniques such as linear regression, or logistic regression on which we will get back later (see chapter 3).

3 Programs

3.1 Producing data for the one-dimensional Ising model

In this part, we were looking to generate data using the Ising model. The Ising model is a statistical physics model. It has been used to model some phenomenons in which the collective effects were produced by local interactions between particles with two states. This model is used in many fields of study like physics, of course, but also mathematics, biology, chemistry and even social sciences. The main example of its application is the use it to model ferromagnetism. It is based on a lattice of spins in which particle are always oriented following the same spatial axis. They can only take two values $+J$ and $-J$. This model is the one that was studied in this project.

The energy of the system is defined as :

$$E = -J \sum_{<kl>}^{N} s_k s_l \tag{3.1}$$

with $s_k = \pm 1$, N the total number of spins, J a coupling constant expressing the strength of the interaction between neighboring spins. The sum is limited to the contribution of nearest neighbors of each particle in the lattice. In our code, we loop over all particles on which we calculate the contribution of it and its next neighbor. Each pair of particle can only be counted once, which means that we can simply calculate that for each particle, and using the BVK condition we are able to do this calculation for the last particle. To do so, we took inspiration from the code given in the worksheet of the project. With that code, we were able to rethink the problem as a linear regression model allowing us to write the problem as :

$$E^i_{\text{model}} \equiv X^i.J \tag{3.2}$$

with X^i the design matrix and J represents the non-local coupling strength.

3.2 Coupling constant using linear regression

From this, we can apply a linear regression using different methods such as Lasso [1] or Ridge regression [2] methods. The Least Absolute Shrinkage and Selection Operator (or LASSO) is a regression analysis method. Originally used for least square model, this method was employed to improve the accuracy of regression models. The Ridge method, also known as the Tikhonov regularization, is useful to mitigate the problem of multicollinearity in linear regression.

We implement the Lasso and Ridge regression to the previous code (see **??**). We got the following results (see 3.1 and 6.2). For this regression, we used an hyperparameter called lambda. It is used as a way to prevent our matrix from being singular. This would prevent us from inverting it, which is needed to fit our parameters.

Lambda value	MSE regression	R squared Linear	MSE Lasso	R squared Lasso	MSE Ridge	R squared Ridge
1e-4	8.94191942746125e-12	0.9999999999997758	5.057353893470236e-07	0.9999999873228625	7.802785920111012e-17	1.0
1e-2	8.94191942746125e-12	0.9999999999997758	1.9423885945083355e-05	0.9999995131064974	7.802743548297867e-13	0.9999999999999805
1	8.94191942746125e-12	0.9999999999997758	0.19423658568758542	0.9951311219700557	7.801778594918692e-09	0.9999999998044349
1e+2	8.94191942746125e-12	0.9999999999997758	4.420935056186113	0.8891815695242054	7.706119916777709e-05	0.9999980683269413
1e+4	8.94191942746125e-12	0.9999999999997758	12.136833701589042	0.6957691428029869	0.29547387957690485	0.992593433026009

Table 3.1: Array gathering the result of the train data for the study of the lambda parameter.

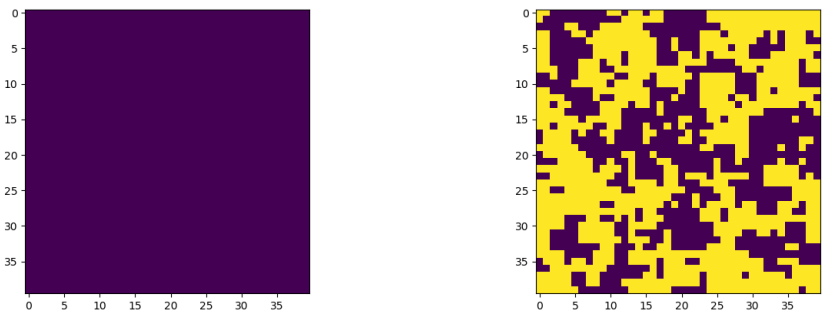
Lambda value	MSE regression	R squared Linear	MSE Lasso	R squared Lasso	MSE Ridge	R squared Ridge
1e-4	9.864617468246669e-12	0.9999999999997566	7.318936929229967e-06	0.9999998194798457	1.2329114387669582e-15	1.0
1e-2	9.864617468246669e-12	0.9999999999997566	1.9352731387363197e-05	0.9999995226686483	1.2329068172430457e-11	0.9999999999999696
1	9.864617468246669e-12	0.9999999999997566	0.1935413609345173	0.995226340015822	1.232304209787824e-07	0.9999999969605456
1e+2	9.864617468246669e-12	0.9999999999997566	4.451788946500688	0.8901974924155555	0.0011742045772606686	0.9999710384727241
1e+4	9.864617468246669e-12	0.9999999999997566	12.02804974325915	0.7033304949915857	1.024508636012752	0.9747306939686472

Table 3.2: Array gathering the result of the study of the lambda parameter for the test data.

3.3 Producing data for the two-dimensional Ising model

The Ising model is also used to generate a system in 2D, a lattice of particles which will again have spins. This time however, instead of calculating the energy like for the 1D system, we will focus on the properties of generated lattices depending on the temperature. This cannot be simulated by a simple algorithm, thus we will use data already available. A paper published by Mehta et al.[3] provides data for simple 40 by 40 lattices using this method, using various temperatures as starting parameters. Temperature adds energy to every particle, giving it the ability to swap its spin if it has enough energy. Each particle tends to follow the spin direction of its neighbors, meaning that this disturbance creates a variation in an otherwise homogeneous system.

We will, in this analysis, use different methods to classify our data, with the aim being predicting if a certain generated lattice is ordered or disordered, meaning ferromagnetic or not. We begin with a script (given in the lecture) which allows us to read the data and fills an array. This gives us the ability to draw a map of the system, showing the spin state (up or down) of every particles in each configuration. One of the main ways of classifying data, is to use a logistic regression. We will use this regression first on the data for all particles of a configuration, then, we will do another regression this time on an *homogeneity factor* which gauges the overall homogeneity of a configuration.



In this example, we can see on the left an ordered phase, and on the right a disordered phase.

4 Results and analysis

4.1 Results of the 1D linear regression

From the arrays seen just above (see 3.1 and 3.2), we can plot the behavior of the fit according to different values of lambda (see 4.1). The first thing we can see is that up to lambda at 1, all the methods seem to have similar behavior. Furthermore, we can also notice that how fast the Lasso method is decreasing comparing to the others. Starting from lambda at 100, the train and test data of the ridge method are diverging. The first conclusion we can take of our model is that the Ridge method seems to be really efficient for a high lambda value. However, the Lasso method is as accurate as Ridge or the OLS method for a low value of lambda.

We can question this comparison however, the OLS method is completely independent from the hyperparameter lambda. Here, we took this method as our reference model. That is why a given value of lambda will be chosen for the rest of the project as a constant $\lambda = 10^{-4}$.

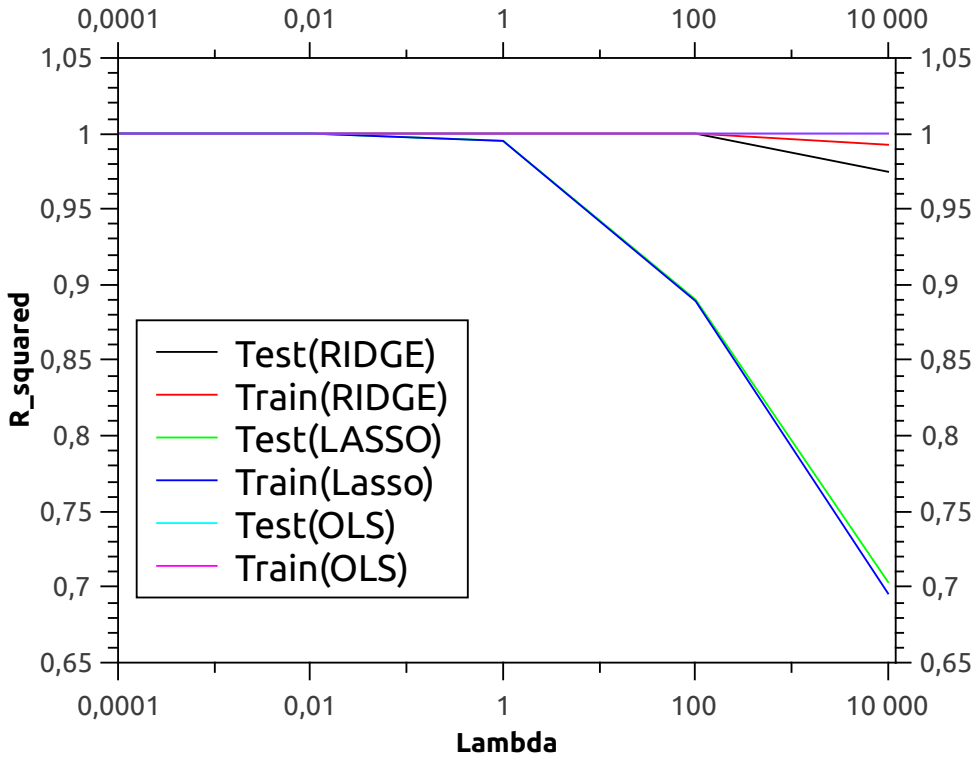


Figure 4.1: Evolution of the R^2 with different value of the parameter lambda

4.2 Results of the 2D classification using a logistic regression

The aim of this part being a classification of our multiple configurations, we used both a logistic regression method and a neural network to determine its properties.

For the logistic regression, we thought about a way to quickly determine how homogeneous a configuration is. Ferromagnetic properties appears when all spins are aligned, which means when

the system is completely homogeneous. On the other hand, a very heterogeneous system will not have this property. That is why, in our program, we calculate what we call an *homogeneity factor*. This factor is simply calculated by doing a sum of all spins of the configuration, giving us the total spin. We divide that value by the total number of particles, 1600, which gives us a value between -1 and 1. If we take the absolute value of that factor, we obtain a value between 0 and 1 that gives us a metric which we can use in our training. If this value is close to 1, the system is deemed homogeneous. On the other hand, if it's close to 0, it is considered heterogeneous. We also expect that in the critical phase, around a temperature of 2.25, that this value will be around 0.5. We will still keep these systems in our training.

Here's the result of our calculations of the homogeneity factor, and the result of our logistic classification using these values :

Temperature	Homogeneity factor	Accuracy
0.25	1.0	
0.50	0.99999975	1.0
0.75	0.99994875	1.0
1.00	0.99926525	1.0
1.25	0.988936	0.9917
1.50	0.986469625	1.0
1.75	0.96399	1.0
2.00	0.908942625	0.9967
2.25	0.6792085	0.8896
2.50	0.139016375	0.9984
2.75	0.087678	1.0
3.00	0.06525775	1.0
3.25	0.055608	1.0
3.50	0.049606	1.0
3.75	0.04428975	1.0
4.00	0.04030325	1.0

As we can see, the results are overall very good, expect around the critical point which was expected. However, we quickly notice a peculiar response for the temperature 1.25. We decided to investigate this case further, and try to find the origin of such an unexpected result.

The first thing we did was to draw the configuration of these cases where the logistic classification did not work. We notice that, out of all 10000 generated event for this temperature, only 91 exhibits a peculiar patterned configuration. This was found using the program *pattern_ analysis.py* gracefully provided by C. Jacquiot. This program detects how many groups (or "blobs") of similar spins are drawn in the configuration. This allowed us to have a general count on all of the temperatures.

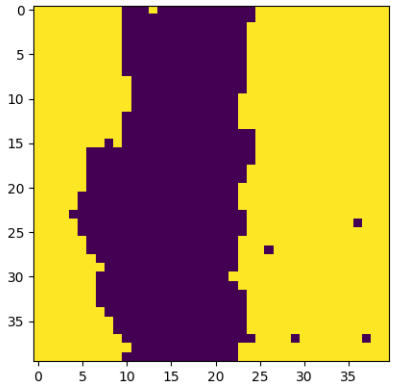


Figure 4.2: One of the configuration showing this "stripe pattern" phenomenon

An obvious observation we can make, in this configuration appears "stripes" of alternating spins. This meant that our very naive homogeneity factor might wrongly identify this as a disordered state, while it clearly shows a pattern of homogeneity. This brought up a question, if a system has alternating pattern of homogeneous spins, can we consider it ferromagnetic? Or, should it be categorized as an heterogeneous system, like our classification did? This was a great topic of discussion in this first project, mainly because this phenomena might have been the result of the simulation, without any physical backing. Or, it could be a physical effect that might happen at key temperature where the boost in energy allow spins to rotate once, making its neighbors do the same, but not enough for the entire system to swap spins.

We ended up deciding that these system were not ferromagnetic, mainly due to the fact that these alternative stripes might cancel each other at great distances.

4.3 Results of the 1D regression using neural networks

In this last analysis, we used a neural network to predict the energy of a system, given its configuration. We used 3 hidden layers, each with 1024 neurons. To determine the ideal lambda value to use, we first launched a grid search using only 20 epochs :

Lambda	Accuracy	MSE
10^{-5}	0.993	0.009
10^{-4}	0.999	0.004
10^{-3}	0.952	0.078
10^{-2}	0.952	0.078
10^{-1}	0.952	0.078
1	0.952	0.078
10	0.952	0.078

In this analysis, we use both the accuracy and the MSE as our metric to rank our neural networks. The energy of our system is a discontinuous integer value, meaning that we can reduce our problem to a classification one, which allows us to use the accuracy.

Using this "ideal" lambda value with which we have a minimal MSE and maximum accuracy, 10^-4, we launch a larger neural network. It will work on a total number of 200 epochs, but will be stopped early when the optimization of the MSE gives a variation of less than 0.001 for 10 calculations in a row to prevent overfitting.

We end up with an accuracy of 0.9998 and an MSE of 0.0011. This is a very good result, but an expected one. Our system is quite simple, and our calculations are very straight-forward. It is not surprising that a neural network is able to reproduce the data. However, this means that we could reduce the number of epoch without fearing loosing too much accuracy.

5 Summary

To summarize our result, in this first project we studied how well a regression problem can be solved by a linear regression.

For the regression problem, we used three methods, and we noticed that the Ridge and Lasso regressions aren't as efficient as the OLS method. This was only tested on the R^2 value. A study for all metrics would give us a more complete picture of the accuracy of these regressions.

For both classification problems, we have used and shown that both logistic regressions and neural networks can accurately classify data. We noted that the neural network gave better results, but at the cost of an increased computing time. These methods worked best with very low values of lambda, under 10^4, and quickly became less and less efficient as it increases.

6 Annex

Lambda value	Linear regression train	MSE regression	R squared Linear	Lasso method train	MSE Lasso	R squared Lasso	Ridge method train	MSE Ridge	R squared Ridge
1e-4	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	8.94191942746125e-12	0.9999999999997758	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	5.057353893470236e-07	0.9999999873228625	**-3.67199568e-12 -9.99999994e-01 -5.81583533e-13 -2.26405768e-10 -2.95052116e-14 2.32391078e-12 2.78151369e-16 -8.69893476e-15 -5.03093492e-19 1.05189988e-17**	7.802785920111012e-17	1.0
1e-2	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	8.94191942746125e-12	0.9999999999997758	**-0.00000000e+00 -9.97726056e-01 -0.00000000e+00 -4.74012034e-05 5.99099841e-08 2.64179905e-07 -5.24951554e-10 -4.68041998e-10 9.44913719e-13 1.65916312e-13**	1.9423885945083355e-05	0.9999995131064974	**-3.67076575e-10 -9.99999376e-01 -5.81750938e-11 -2.26404919e-08 -2.95018687e-12 2.32390051e-10 2.78133082e-14 -8.69889255e-13 -5.03065981e-17 1.05189449e-15**	7.8027435482978 67e-13	0.9999999999999805
1	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	8.94191942746125e-12	0.9999999999997758	**-0.00000000e+00 -7.72606310e-01 -0.00000000e+00 -4.74010686e-03 5.96419059e-06 2.64180550e-05 -5.22860383e-08 -4.68052824e-08 9.41305630e-11 1.65937659e-11**	0.19423658568758542	0.9951311219700557	**-3.66921719e-08 -9.99937582e-01 -5.81782345e-09 -2.26390919e-06 -2.94992198e-10 2.32375681e-08 2.78112475e-12 -8.69835469e-11 -5.03030498e-15 1.05182945e-13**	7.801778594918692e-09	0.9999999998044349
1e+2	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	8.94191942746125e-12	0.9999999999997758	**0.00000000e+00 -0.00000000e+00 0.00000000e+00 -1.72633308e-02 0.00000000e+00 7.60185463e-05 -1.89461207e-08 -6.91651810e-08 4.61871944e-11 -7.28125440e-11**	4.420935056186113	0.8891815695242054	**-3.51889564e-06 -9.93796571e-01 -5.84820711e-07 -2.24998316e-04 -2.92354313e-08 2.30946050e-06 2.76056343e-10 -8.64483585e-09 -4.99488169e-13 1.04535748e-11**	7.706119916777709e-05	0.9999980683269413
1e+4	**-1.87212048e-07 -9.99999215e-01 1.00950656e-08 -1.48976292e-08 -3.81164691e-11 5.95106836e-11 1.22818422e-15 -8.86435023e-14 1.11022302e-16 0.00000000e+00**	8.94191942746125e-12	0.9999999999997758	**0.00000000e+00 -0.00000000e+00 0.00000000e+00 -0.00000000e+00 0.00000000e+00 -1.25102394e-04 2.60248263e-08 6.48273836e-07 -6.56663134e-11 -8.75998850e-10**	12.136833701589042	0.6957691428029869	**-4.99127701e-05 -6.15874226e-01 -4.45864650e-05 -1.39296495e-02 -1.70563680e-06 1.42965664e-04 1.66493843e-08 -5.35126044e-07 -3.03468928e-11 6.47070052e-10**	0.2954738795769048 5	0.992593433026009

Table 6.1: Array gathering the result of the study of the lambda parameter.

Lambda value	Linear regression test	MSE regression	R squared Linear	Lasso method test	MSE Lasso	R squared Lasso	Ridge method test	MSE Ridge	R squared Ridge
1e-4	**4.09170304e-08 -9.99999727e-01 -1.70623171e-08 -1.64008334e-09 2.65583226e-11 3.12622717e-12 1.62499788e-13 -3.39641509e-14 5.55111512e-17 0.00000000e+00**	9.864617468246669e-12	0.9999999999997566	**2.62531132e-03 -9.99896969e-01 -1.77508505e-04 -3.13854313e-06 1.32592003e-06 2.07720102e-08 -1.18404678e-09 -1.79111217e-11 -3.12329295e-12 -5.01698546e-14**	7.318936929229967e-06	0.9999998194798457	**3.37174166e-10 -9.99999975e-01 -1.32794963e-11 -8.92535332e-10 6.55379355e-13 9.14497210e-12 -4.32228787e-15 -3.42191972e-14 7.01121285e-18 4.13879404e-17**	1.2329114387669582e-15	1.0
1e-2	**4.09170304e-08 -9.99999727e-01 -1.70623171e-08 -1.64008334e-09 2.65583226e-11 3.12622717e-12 1.62499788e-13 -3.39641509e-14 5.55111512e-17 0.00000000e+00**	9.864617468246669e-12	0.9999999999997566	**0.00000000e+00 -9.97735838e-01 0.00000000e+00 -4.77648033e-05 -2.74950653e-07 2.72534727e-07 2.22298912e-09 -4.96549370e-10 -3.94861745e-12 1.87895480e-13**	1.9352731387363197e-05	0.9999995226686483	**3.37169814e-08 -9.99997534e-01 -1.32794417e-09 -8.92533909e-08 6.55380206e-11 9.14495931e-10 -4.32229647e-13 -3.42191540e-12 7.01122903e-16 4.13878919e-15**	1.23290681724304 57e-11	0.999999999999696
1	**4.09170304e-08 -9.99999727e-01 -1.70623171e-08 -1.64008334e-09 2.65583226e-11 3.12622717e-12 1.62499788e-13 -3.39641509e-14 5.55111512e-17 0.00000000e+00**	9.864617468246669e-12	0.9999999999997566	**0.00000000e+00 -7.73578819e-01 0.00000000e+00 -4.77670959e-03 -2.75874785e-05 2.72556379e-05 2.23042180e-07 -4.96609539e-08 -3.96170048e-10 1.87939302e-11**	0.1935413609345173	0.995226340015822	**3.36573683e-06 -9.99753478e-01 -1.32494052e-07 -8.92315684e-06 6.54886899e-09 9.14272253e-08 -4.31983978e-11 -3.42107818e-10 7.00769369e-14 4.13777637e-13**	1.232304209787824e-07	0.9999999969605456
1e+2	**4.09170304e-08 -9.99999727e-01 -1.70623171e-08 -1.64008334e-09 2.65583226e-11 3.12622717e-12 1.62499788e-13 -3.39641509e-14 5.55111512e-17 0.00000000e+00**	9.864617468246669e-12	0.9999999999997566	**0.00000000e+00 -0.00000000e+00 -0.00000000e+00 -1.73818735e-02 -3.72545709e-05 7.83140884e-05 2.95043708e-07 -7.68437655e-08 -5.20907769e-10 -6.67320457e-11**	4.451788946500688	0.8901974924155555	**2.85097550e-04 -9.75935992e-01 -1.06674857e-05 -8.71018858e-04 6.11084291e-07 8.92443770e-06 -4.09831536e-09 -3.33937725e-08 6.68637423e-12 4.03894052e-11**	0.0011742045772606686	0.9999710384727241
1e+4	**4.09170304e-08 -9.99999727e-01 -1.70623171e-08 -1.64008334e-09 2.65583226e-11 3.12622717e-12 1.62499788e-13 -3.39641509e-14 5.55111512e-17 0.00000000e+00**	9.864617468246669e-12	0.9999999999997566	**-0.00000000e+00 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00 0.00000000e+00 -1.31113190e-04 -9.87524431e-08 6.89878274e-07 2.79726935e-10 -9.42051419e-10**	12.02804974325915	0.7033304949915857	**5.96971195e-04 -2.89199437e-01 9.15818689e-05 -2.57091657e-02 1.29444612e-05 2.63313968e-04 -9.94299802e-08 -9.85055671e-07 1.69218001e-10 1.19124891e-09**	1.0245086360127 52	0.9747306939686472

Table 6.2: Array gathering the result of the study of the lambda parameter.

References

[1] Lasso (statistics): Wikipedia, the free encyclopedia.

[2] Ridge regression: Wikipedia, the free encyclopedia.

[3] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre G.R. Day, Clint Richardson, Charles K. Fisher, David J. Schwab. A high-bias, low-variance introduction to Machine Learning for physicists. *Cornell University*, pages 0–122, 2019.