MiniProject 1 - COMP551

Jérôme Genzling, Felipe Fontes and Sarah Al Taleb

April 8, 2023

Abstract

In this project, we investigated the performance of two regression models on two different data sets. The first machine learning model implemented was linear regression to predict different buildings' heating and cooling load. We also worked on a logistic regression model to study the classification of potential bankruptcy. The models achieved excellent results predicting the values from the built testing data sets. We confirmed our results by comparing them with the sklearn library models as they were nearing each other. Moreover, we studied the influence of different hyper-parameters, such as the training data size, the mini-batch size, and the maximum number of iterations of our gradient descent approach on our models' accuracy. Additionally, we implemented for both models a mini-batch algorithm to optimize the computations, but the results remained similar. As a result, the computed weights and bias informed us on the relevance of certain features. Continuing in this direction, we tried optimizing the efficiency of the models by removing unimportant features and duplicated data in the experience, which improved the machine learning models. We also studied the impact of Ridge Regularization on our regression model and implemented a grid-search for our hyper-parameters using 10-fold cross-validation.

1 Introduction

Linear regression and logistic regression models were implemented in this project. We used linear regression to work on the first data set since this algorithm works for continuous values. It allows us to find the best model to fit the data set values and then accurately predict the output to any given input in a linear fashion. We implemented a closed-form and a gradient descent approach to find the most suitable set of parameters.

For the second data set, we used logistic regression since the output is binary. Using our gradient descent implementation, our task is to calculate the weights and the bias to understand better which parameter influences the predictions. We then decided to use one-hot encoding to represent the features and the cross-entropy cost function.

A mini-batch algorithm was added for both methods to optimize the models' training time. We also tried to remove certain features in the linear regression model to see if better predictions could be made and obtained good performance for both models. We compared our results with the built-in libraries from sci-kit learn and observed the same mean squared error for the linear regression model and a similar accuracy for the logistic regression model.

Furthermore, we implemented some hyper-parameters optimization. First, looking at each hyper-parameter independently for both models, we coded a grid search using a 10-fold cross-validation algorithm to test their correlation with the models' accuracy. We also tried to implement a Ridge Regularization, but this did not help to get a lower Mean Squared Error (MSE) for our linear regression task. Finally, we removed some features of the first data set to simplify our model (dimension reduction) and tried to run our binary classifier without the duplicates in our second data set to analyze their influence.

Existing research has used these data sets. Tsanas and Xifara 2012 implemented a linear regression model and a random forest method to compare the performances on the building data set, whereas Kim and Han 2003 tested three different data mining techniques, such as inductive learning methods, neural networks, and

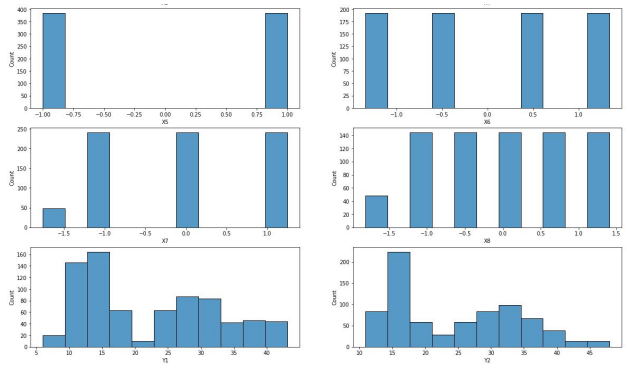
genetic algorithms, to discover bankruptcy rules. On the first data set, they got an MSE of 3.81 using the Random Forest technique and 10.67 using an iteratively reweighted least squares regression. In the second data set, the authors observed that the genetic algorithms had a better performance (94% accuracy vs 90 for the inductive learning and neural network) compared to the other methods. Their methodology was interesting as it gave them explanations about the decisions taken (called rules) in this case to decide about the output.

2 Data sets

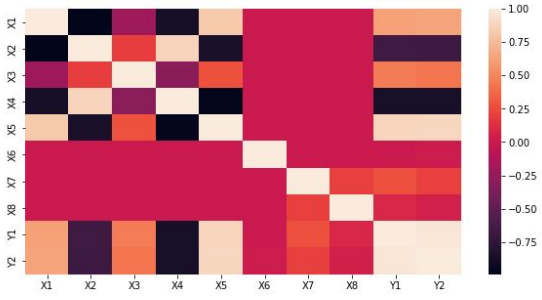
2.1 Data set 1: Energy

The first data set is composed of 8 different features and two outputs. The inputs are Relative Compactness, Surface Area, Wall Area, Roof Area, Overall Height, Orientation, Glazing Area, and Glazing Area Distribution will be named X1, X2, X3, X4, X5, X6, X7, and X8 throughout the report. The outputs, Heating Load and Cooling Load, will be addressed as Y1 and Y2, respectively. They describe the main features that a building can have and that experts might take into account when defining the energetic performance of a building. There are 768 instances in this data set, and no missing values or duplicates were found when cleaning the data. We have normalized our input values because the difference in scale was up to 100 times. We also wanted our algorithm to be insensitive to outliers.

When analyzing the data, we see that Y1 and Y2 follow a similar distribution. We can also notice that some of the inputs present very symmetrical distributions, such as X5, X6, X7, and X8. This raised a concern as to whether they were relevant to the prediction models. With further testing, we concluded that the model would perform better if X6 and X8 were dropped, especially regarding the feature correlation matrix.



(a) Some data visualization of features and outputs from Data set 1



(b) Features Correlation Matrix of Data set 1

Figure 1: Data Exploration of Data set 1

2.2 Data set 2: Bankruptcy

The second data set is composed of 250 instances with six ternary features and one binary output. The inputs Industrial Risk, Management Risk, Financial Flexibility, Credibility, Competitiveness, and Operating Risk will be called IR, MR, FF, CR, CO, and OP, respectively, and the only output will remain as Class.

The data set represents the risk of bankruptcy for businesses based on certain financial and organizational parameters. During the data cleaning, we encoded the values into integers for further computations. There are two ways of handling the features: one-hot encoding to prevent multicollinearity or ordinal encoding (putting numbers). The latter often assumes that there is an ordinal relationship. Based on common practice in binary classification, we decided to use the one-hot encoding method for more accurate precision.

Our main concern was due to a large number of duplicate rows. In total, there were 147 duplicates which correspond to over 50% of the total data. With the entire dataset, the inputs and outputs were quite balanced. However, once the duplicates were removed, the previous balance shifted into a 25/78 ratio for B/BN (possible values for the output variables). Faced with this problem, we calculated the performance with and without the duplicates. However, it is important to note that the duplicates are not necessarily bad or good for the dataset. On the one hand, it could show that the real data follows a similar pattern of highly repetitive scenarios. On another, it could make the model ungeneralizable if this repetition was caused by a poor/non-representative data-collection method.

IR	MR	FF	CR	CO	OP	Class
P 80	P 62	A 74	A 77	A 56	P 79	NB 143
N 89	N 119	P 57	P 79	P 91	N 114	B 107
A 81	A 69	N 119	N 94	N 103	A 57	

IR	MR	FF	CR	CO	OP	Class
P 35	P 29	A 36	A 42	A 26	P 38	NB 78
N 32	N 42	P 34	P 39	P 53	N 41	B 25
A 36	A 32	N 33	N 22	N 24	A 24	

Figure 2: Composition of the second data set with and without the duplicates

2.3 Ethical aspects

Ethical issues arise when working with data sets; it is essential to keep them in mind to build reliable models. The datasets could be biased if the data were collected only from a particular community or ethnicity, for instance, and it would result in wrong and biased predictions. For example, suppose banks used the second data set to decide which companies should receive loans based on logistic regression bankruptcy prediction. In that case, it could unfairly ruin the future of those companies. Another possible scenario could be the safety and accommodation of families in residential buildings. If there is an unjustified bias in the data towards predicting a lower cooling or heating load than in reality, then some tenants may pay way higher charges to heat and cool down their apartments than planned by this model.

3 Results

3.1 Data set 1: Linear Regression

For our first data set, we implemented two types of linear regression models: an analytical closed-form and a gradient descent based one, using an L2 loss. Under the same circumstances (learning rate = 0.005, maximum number of iterations = 100,000, and test size = 20%), they presented very similar performances. For closed-form, we found MSE to be 9.529, while for the gradient descent one, 9.540 on the test set (respectively 9.300 and 9.319 on the train set). Compared to the sci-kit closed-form, there was a 0.006 difference, showing that our model is very accurate. Note that the MSE values seem high because we chose not to normalize the output variables. We observe that the analytical "best solution" is indeed the one with the lowest MSE. Furthermore, in both cases, the weights of X6 and X8 were the lowest out of all input variables nearing 0. This consolidates our initial suspicion to remove them from the model. After the initial test, we decided to create some variations regarding our model settings. We started by changing the maximum number of iterations. We found that most of the improvement in the MSE came from the initial 10,000 iterations. After that, the training and test MSE values plateau around 9.5. We then decided to keep the maximum iteration number to 50.000 going on.

We then moved to different training data sizes. The split values used were 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, and 0.8. We found a linear MSE improvement up to 0.5, which showed the best performance of 9.316. However, past 0.5, the model predictions became worse. This is due to the over-fitting of weights to the training set

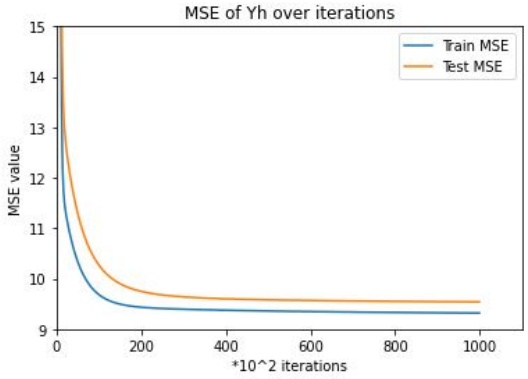
```
Weight : [[-6.37656796e+00 -7.67703348e+00]
 [ 7.53042927e+13 -6.92245669e+13]
 [-3.72960175e+13  3.42849069e+13]
 [-7.72242003e+13  7.09894700e+13]
 [ 7.30364213e+00  7.00183044e+00]
 [-2.10079138e-02  5.32197154e-02]
 [ 2.68385593e+00  1.97467632e+00]
 [ 3.33441236e-01  2.79972215e-02]]
Bias : [22.30494753 24.47353701]
Mean Squared Error: 9.529294087144553
```

(a) Weights, biases and MSE of the analytical closed form model 1

```
Weight : [-2.14099212e-01  5.58165977e-01 -9.51440780e-01 -6.50870023e-01
  1.82289082e-01 -1.38793075e-01 -1.07616068e+00  1.97417055e+00
 -1.50538388e+00 -1.77760219e+00  1.80111932e+00 -6.30891148e-01
 -2.86411774e+00  3.47841688e+00 -1.22167315e+00 -7.01240898e-01
 -6.21298003e-04  9.44881799e-02]
Bias : [-0.60737402]

Test Cross-Entropy Cost: 0.009057453217231084
```

(b) Weights, biases and CE Cost of model 2



(c) Learning curves of our Gradient Descent model over the number of iterations

Figure 3: Analysis of the weights and biases of our models, including the maximum number of iterations

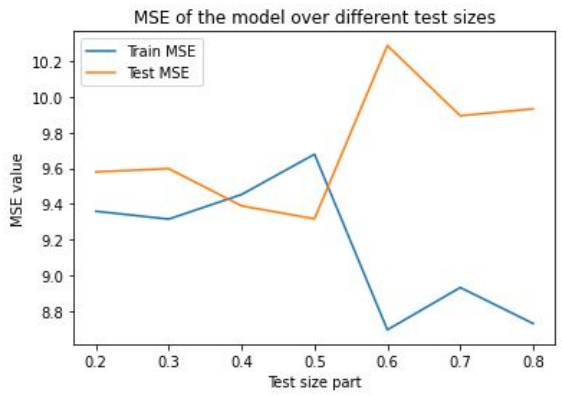
data when the latter is not large enough to have a significant representation of all possible inputs potentially given to the model. The MSE for the training set dramatically decreased after 0.5. Our next experiment was about different batch sizes. We chose the values 8, 16, 32, 64, and 128 as our batch sizes. The two smallest sizes resulted in the highest MSEs, while 128 resulted in the best MSE of 9.324. For different learning rates (0.1, 0.05, 0.01, 0.005), they all presented very similar results (cf. Jupyter Notebook plot). The best one was 0.1. We then simplified our model by removing features X6 and X8 since both the correlation matrix and the weights seemed to lead us that way. According to the results, our hypothesis was justified as the MSE without them was lower than with them, 9.220 and 9.529, respectively. Apart from the optimization in the computation -memory and calculations needed being reduced-, the analytical linear regression model and the mini-batch stochastic gradient descent based linear regression model gave similar performance. This inclines us to SGD for models where we have large number of instances and features and where the inverse of $\mathbf{X}^\top$*X is hard to compute. After trying these variations, we added Ridge regularization, and we used 0, 0.01, 0.1, 0.5, 1, and 5 as the regularization rates. However, as we increased the rate, the cost function also increased. Therefore, adding regularization made the performance worse. This could be explained since our model is quite simple with a limited amount of features.

Finally, a Hyper-Parameter Grid Search using a 10-fold cross-validation to further optimize our model. Indeed, the different hyper-parameters may not be independent and may have some correlation. We found that a batch size of 8 with a learning rate of 0.1 seems to be the best hyper-parameter set for this model.

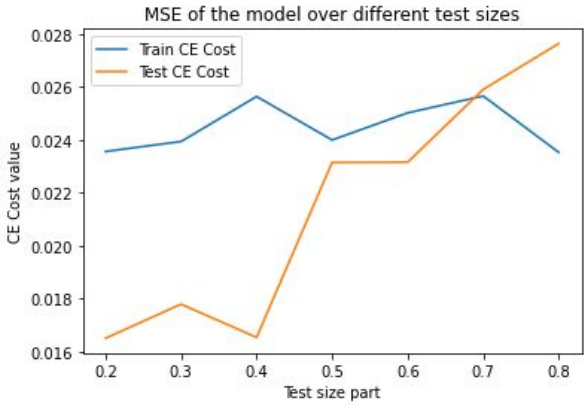
3.2 Data set 2: Logistic Regression

For the second data set (with the duplicates), we ran similar experiments and variations. Starting with a full batched logistic regression, we found the model to be 100% accurate under standard conditions (learning rate = 0.005, maximum number of iterations = 100,000, and test size = 20%) with a cross-entropy cost of 0.0090 on the test set and 0.0140 on the train set. The explanations we had for the linear regression model seem to be still coherent for this binary classifier. Looking at the importance of the weight (Fig.3b), we can see that all features seem to play an important role to get that final prediction. The last feature, Operating Risk (represented by the three last weights), may have a lower importance in that model and could be removed to simplify it.

As mentioned before, we noticed many duplicates in the data set. When running the experiments without removing the duplicates, the results were always precisely/nearing 100% accuracy for different iteration limits, test set sizes, batch sizes, and learning rates. There is, however, a slight improvement in the cross-entropy



(a) Train and Test performances of model 1 in function of test size



(b) Train and Test performances of model 2 in function of test size

cost. For iteration limits, most of the improvement for the test set happens up to nearly 10,000. For the data split ratio, the best cost of 0.01650 was at an 80/20 split. When comparing batch sizes, size 8 was the best for our model. Similarly to the linear regression model, 0.1 was the best learning rate. All of these values have to be tackled with caution as it may be said that this model may only memorize the duplicate output values and not really learn a pattern out of them. That is why we also decided to remove the duplicates to see how well the model would perform. The accuracy and precision were still very high, 0.952 and 0.941, respectively. However, compared to not removing the duplicates, it was the lowest value. The same thing happened to the cost-entropy, with a value of 0.183, the highest we ever had. Still, the recall was at 1 (since we only have a False Positive in the test set), which is the most important metric here as for both the bank and the client, we do not want our model to predict a non-bankruptcy process, whereas there will be one.

4 Discussion and Conclusion

In the end, we successfully implemented a linear regression model and a logistic regression model to perform binary classification. These models' performances reach the same accuracy levels as those published in the literature on the same data sets. Apart from testing the different optimizers algorithms (i.e. Full-Batch/Mini-Batch SGD), we also saw the influence of hyper-parameters and tried to improve our models using more advanced techniques such as Ridge Regularization, Hyper-Parameters Grid Search and Dimension Reduction. Other techniques that could have been tried would be to use a non-linear basis set on top of the features we had to maybe have a better fit, or to improve even more our optimizers, adding some Momentum as in the famously used Adam optimizer. Finally, we could also try to use another encoding for the second data set. This could be even more useful when we tackle more complex Machine Learning models such as Random Forests of Deep Neural Networks.

5 Statement of Contributions

Jérôme, Felipe, and Sarah shared the coding and writing parts equally.

References

[KH03] Myoung-Jong Kim and Ingoo Han. "The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms". In: *Expert Systems with Applications* 25.4 (2003), pp. 637–646.

[TX12] Athanasios Tsanas and Angeliki Xifara. "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools". In: *Energy and buildings* 49 (2012), pp. 560–567.