ECSE 551 Mini Project 1

Isabel Lougheed, 260989364 **Mathieu Mailhot, 260989370**

Frank-Lucas Pantazis, 260986139

Abstract

This project involves implementing a logistic regression linear classifier from scratch. This report presents the results of the logistic regression classifier model when performed on two datasets, a Chronic Kidney Disease (CKD) dataset and a battery dataset. INCLUDE IMPORTANT FINDINGS.

1 Introduction

2 Datasets

The first dataset used is a CKD dataset that is comprised of 28 numerical features which each represent a medical measurement of a patient. There is one target variable indicating whether the patient was diagnosed with CKD ('CKD') or not diagnosed with CKD ('Normal'). WRITE MORE ABOUT OUR FINDINGS ABOUT THE DATASET.

The second dataset used in this project is a battery dataset comprised of 32 real-valued features which represent specific battery attributes. There is a target variable to classify whether the battery is normal ('Normal') or defective ('Defective'). WRITE MORE ABOUT OUR FINDINGS ABOUT THE DATASET.

3 Results

3.1 Hyper Parameter Analysis

The optimization of our models was achieved by testing different Gradient descent variations as well as fine tunning each model's hyperparameters. For starters, two variations of gradient descent were implemented on the normalized CDK and battery datasets, using all available features. Since most of the models that we developed were trained on similar datasets (variations on the normalized dataset with all features), it seemed reasonable to assume that the best-performing gradient descent variation on this dataset would also perform well on the others.

These two variations are:

- Gradient Descent with a Decaying Learning Rate
- Momentum Based Gradient Descent

Although both models performed similarly in terms of accuracy, the gradient descent with a decaying learning rate converged with a much lower tolerance than momentum-based gradient descent. This suggests that it may be more precise.

Following this, the step size for the decaying gradient descent for each model was fine-tuned by iterating through the following range of values [0.1, 0.011] and plotting its accuracy against its speed. Since our gradient descent method's learning diminishes after each iteration, starting with larger step size values such as 0.1 was a better approach.

By analyzing the performance of each model, it was determined that they all worked best with a starting step size of 0.03 and a tolerance of 10^{-6}. As shown in the figures below, the 1-accuracy value remained relatively constant across a wide range of step sizes. To distinguish the best-performing step size, convergence speed became the key criterion. Indeed, when using larger step sizes, the convergence speed was slower, and during the first iterations, the values tended to oscillate much less compared to smaller step sizes.

3.2 Linear Model With All Features

3.3 Removed Feature Model

3.4 Quadratic Model

One of the models explored in this project is a quadratic model. This was implemented by adding a quadratic term to the learned function for the more important features, the features with the highest weights attributed to them from the regular linear model. For the chosen important features x_i,

$$f_w(x) = w_0 + w_1 x_i + w_2 x_i^2 + \dots \tag{1}$$

A cross validation was performed on the quadratic model for each of the datasets for different amounts of quadratic features. As seen in Figure !!!!, as the number of quadratic terms increases, the error increases. Therefore, for both the CKD quadratic model and the battery quadratic model, only one feature has a quadratic term in the learned function. Since the weights of the CKD linear model are

$$\begin{aligned}[&-1.0179, 2.4871, -0.0185, 0.1225, 0.0509, 0.1222, -0.2118, -0.2312, -0.3058, -0.1395,\\ &0.1897, 0.4275, 0.1648, -0.1032, -0.2050, 0.2870, 0.2827, 0.0557, 0.1185, 0.1652,\\ &-0.0089, -0.3447, -0.2880, 0.0708, 0.0002, -0.0845, -0.0942, 0.1653, -0.7693],\end{aligned}$$

the feature that has the largest weight magnitude, excluding the bias weight is feature 2 and therefore feature 2 has a quadratic term. Since the weights of the battery linear model are

$$\begin{aligned}[&5.5288, 5.4114, -0.0814, 0.0036, 0.0838, -0.3571, -0.9275, -0.8126, -0.2352, -0.0839,\\ &-0.1379, -0.2578, -0.0698, -0.5288, -0.3349, -0.5354, -0.2542, -0.2727, -0.0385, -0.1533,\\ &-0.1472, 0.0454, 0.1609, 0.1094, 0.0360, -0.7155, 0.0190, -0.6298, -0.4767, -0.3815,\\ &-0.3547, -0.3608, -1.6986],\end{aligned}$$

the feature that has the largest weight magnitude, excluding the bias weight is feature 1 and therefore feature 1 has a quadratic term.

3.5 Cubic Model

To further expand the model, a cubic model was also tested. The implementation of the cubic model is very similar to the quadratic model, except the more important features have polynomial terms of degree three in the learned function. For the chosen important features x_i,

$$f_w(x) = w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 + \dots \tag{2}$$

A cross validation was performed on the cubic model for each of the datasets for different amounts of cubic features. As seen in Figure !!!, as the number of cubic terms increases, the error increases. Therefore, for both the CKD cubic model and the battery cubic model, only one feature has a cubic term in the learned function. This means that for the CKD cubic model, feature 2 has a cubic term and for the battery cubic model, feature 1 has a cubic term.

Table 1: Cross validation of different CKD models

CKD Model	**MSE**	**Error**
Linear model with all normalized features	0.21073183686528782	0.16896551724137931
Linear model with all standardized features	0.21760510560452745	0.1724137931034483
Feature removed model		
Quadratic model	0.20873582936775956	0.15862068965517245
Cubic model	0.2078159785211106	0.15517241379310348

Table 2: Cross validation of different battery models

Battery Model	**MSE**	**Error**
Linear model with all normalized features	0.0886492622549294	0.05846153846153843
Linear model with all standardized features	0.0886183051677138	0.05846153846153843
Feature removed		
Quadratic model	0.08897472474905832	0.05846153846153843
Cubic model	0.08894956726027695	0.05846153846153843

3.6 Cross Validation

4 Discussion and Conclusion

5 Statement of Contributions

6 Appendix

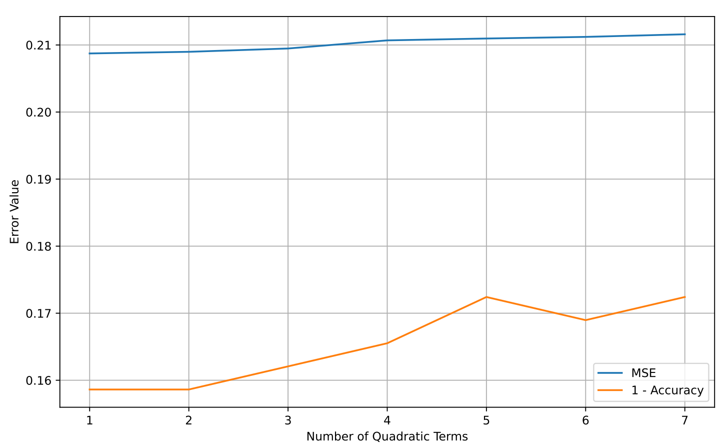

Figure 1: Number of quadratic terms vs. error for the CKD quadratic model

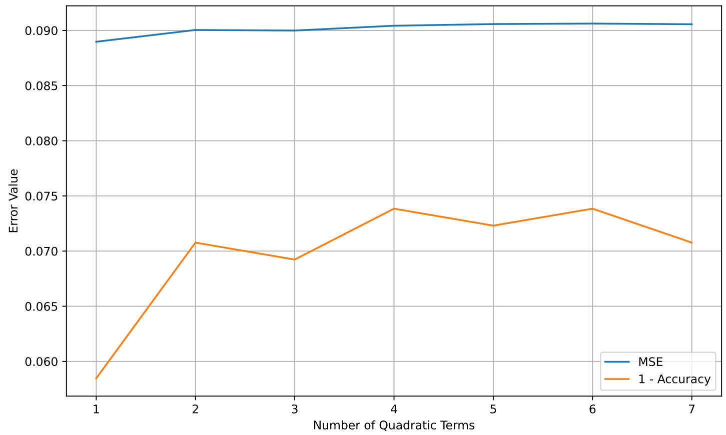

Figure 2: Number of quadratic terms vs. error for the battery quadratic model

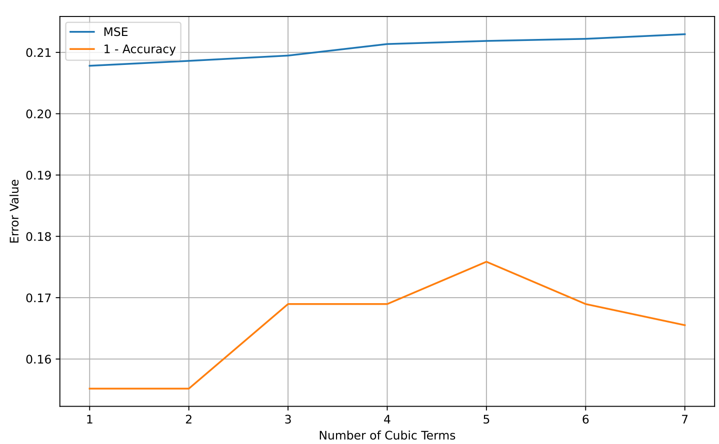

Figure 3: Number of cubic terms vs. error for the CKD cubic model

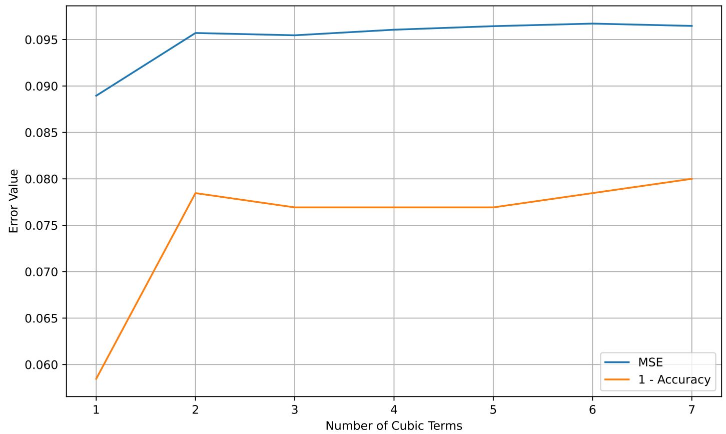

Figure 4: Number of cubic terms vs. error for the battery cubic model

References

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. **Note that the Reference section does not count towards the eight pages of content that are allowed.**

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.