Hyperparameters Tuning in MNIST Dataset

Jingbang Chen, Tim Chen, Yulin Song

December 8th 2022

1. Introduction

The MNIST dataset is a large dataset of labeled images of handwritten digits. The dataset was first published in 1998 by Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. It is widely used in machine learning research as a simple, yet effective, way to evaluate and compare the performance of different image recognition algorithms. The dataset consists of a training set of 60,000 images and a test set of 10,000 images. Each image is a 28x28 grayscale image of a handwritten digit between 0 and 9. The images are labeled with the correct digit, and the goal of a machine learning algorithm trained on this dataset is to learn to recognize the correct digit in new images.

2. Full factorial experiment

In our first experiment, we used a 2^5 full factorial design to test the effects of different hyperparameters and interactions. The outcome we use is the accuracy of the test set after training the images for 1 epoch. The factors we used are shown as follows:

2.1 Factor and Level Selection of First experiment

- Optimizer (level: SGD, Adam)

SGD(Stochastic Gradient Descent) is a simple optimization algorithm that updates the model parameters by moving in the direction of the negative gradient of the loss function with respect to the parameters. Adam(Adaptive Momentum Estimation), on the other hand, uses an adaptive learning rate that automatically adjusts the learning rate based on the historical gradient, first and second order momentum estimates. This means that Adam can adapt to the changing landscape of the loss function and use a larger learning rate when the gradients are consistent and a smaller learning rate when the gradients are noisy. It might improve the convergence rate of the algorithm and help it find the optimal solution more quickly.

- Batch size (level: 32 128)

Batch size is a hyperparameter that when the optimizer conducts one step of gradient descent, the number of images that input into the network. Batch sizes of power 2 are the most common choice. Using a larger batch size can reduce the variance of the gradients, which can improve the stability of the training process and help the model converge faster. On the other hand, using a smaller batch size can provide more accurate gradients, which can lead to better performance of the model. Thus, batch size is an important hyperparameter that can have a significant impact on the performance of a machine learning model. Finding the optimal batch size for a given model and dataset can require experimentation and careful tuning.

- Architecture of neural networks (level: CNN, MLP)

CNN(Convolutional Neural Networks) and MLP(Multi-Layer Perceptron) are two kinds of commonly used network structure in computer vision. (MLP) is a type of feedforward neural network that consists of multiple layers of artificial neurons. In comparison, CNN contains convolutional layers, pooling layers and fully-connected layers. One of the key features of CNNs is that they use convolutional layers, which apply a convolution operation to the input data, creating a set of feature maps that capture spatial relationships in the data. MLP is a combination of fully connected layers.

- Learning rate (level: 0.01, 0.001)

The learning rate is an important factor in the performance of a machine learning model, as a larger learning rate can allow the algorithm to make faster progress but may be less stable, while a smaller learning rate can be more stable but may take longer to converge. This hyperparameter is often used in log-scale, ie. 0.1, 0.01, 0.001.

- Number of feed forward layers (level: 1, 2)

In general, adding more feedforward layers to a network can allow the network to learn more complex, non-linear relationships between the input and output data, leading to improved performance. However, adding too many layers can also cause the network to overfit, resulting in poor performance on new, unseen data. In the MLP setting, the number of feed forward layers is just the number of layers. On the other hand, in the CNN setting, this hyperparameter determines the number of feed forward layers after the flattening process, which are after convolutional and pooling layers.

2.2 Experimental Design

We are interested in the effect of the five factors we choose, and their interactional impact on the overall accuracy of the model. We conduct a 2^5 design with 2 replicates, and take the average of 2 replicates as the outcome accuracy.

Table I. Factors and levels of full factorial experiment

	Levels	
Factor	-1	1
A = Optimizer	SGD	Adam
B = Batch Size	32	128
C = Number of Layer(s)	1	2
D = Learning Rate	0.01	0.001
E= Network Architecture	CNN	MLP

The experiment results are shown in Fig. 1.1

2.3 Analysis

Firstly, we run a model with all the first order and 2-factor interaction effects. The model is expressed as below:

$$y = \beta_0 + \sum_{i=1}^{5} \beta_i x_i + \sum_{i=1}^{5} \beta_{ii} x_i^2 + \sum_{i=1}^{5} \sum_{j=i+1}^{5} \beta_{ij} x_i x_j + \epsilon$$

We find out (Fig 1.1) that the main effect of C (Number Of Layers) and 2-factor interactions involving C are insignificant. Though not a necessary step, we still check the half-normal plot of the model to make sure the results accords with the parameter estimates. Then, we decided to remove C and rerun the model. As a result, the multiple R-squared value increases.

The normal qq plot follows the general requirements, but the residual vs fitted plot have some outliers. We proceed to run the Breusch-Pagan test and results in Fig 1.4 indicates a high enough p value, indicating that the null hypothesis which proposes that the error variance in the regression model is constant cannot be rejected.

2.4 Discussion

The simple full factorial design gives us enough evidence to rule out number of layers in the hyperparaemeters tuning. Besides, according to the signs of the coefficients, we can reasonably infer that using CNN model and adam as optimizers are the overall superior level settings when we are trying to maximize the accuracy of the MNIST dataset prediction. However, in order to precisely find the level settings to tune the hyperparameters such as batch size and learning rate, we need to conduct follow up experiments, and take a closer look at the quantitative factors while fixing the qualitative levels that we are confident about.

3. Follow up experiment

In order to further analyze the quantitative effects of different factors on the performance of neural networks and find the optimal level settings, we will conduct a central composite design. This will allow us to study the first-order, second-order, and interaction effects of the quantitative factors, such as the learning rate and batch size. For the fixed qualitative factors, we will choose the levels that have the highest accuracy, such as using the Adam optimizer and a CNN architecture with two layers. Additionally, we will introduce a new quantitative factor, the number of hidden nodes in the feedforward network, to study the effect of network width on performance.

3.1 Central Composite Design (CCD)

One of the main uses of a Central Composite Design (CCD) is to study the relationship between the variables and the outcome in a quantitative manner, and using a contour plot to visualize the change, allowing us to study both linear and nonlinear relationships between the Neural Network factors we choose and the model prediction accuracy.

We designed the experiment to be a rotatable design by setting alpha value to be $\sqrt[4]{8} = 1.68$. A rotatable design specifically allows for rotation around the design axes, which can make the experimental deisgn more efficient and improve the accuracy of the final results.

Table II. Factors and levels of follow up experiment					
	Levels				
Factor	-1.68	-1	0	1	1.68
A = Batch Size	13	40	80	120	147
B = Hidden Layer Nodes	10	30	60	90	110
C = Learning Rate	0.003	0.01	0.02	0.03	0.037

We use 8 cube points, 6 star points and 6 center points in our 20 run experiment. We have 5 runs for each factor setting combination and the experiment results are demonstrated in Appendix Fig. 2. As learnt in class, we can use the two-level design (runs 1–8) to estimate the three linear effects and 3 two-factor interactions and use the 6 (runs 9–14) plus the 6 replicates at the center point (runs 15–20) to estimate the three quadratic effects.

We run the linear model to evaluate the significant main effect, second-order effect and interaction effect from the above design(Appendix Fig. 2.1). Realizing that B(Hidden Layer Nodes) are not significant, we quickly rerun the model with only A and C terms. The results are attached in Appendix Fig. 2.6, and we proceed with the result of parameter estimates, which are all significant in the model.

3.2 Optimization

Using the coefficient from the CCD model, we can draw the corresponding contour plot (Fig. 2.7), and by observing the plot and doing derivations, we can confirm that the stationary point we have is a saddle point. Even though the left bottom and the right top region have higher accuracy, after converting axes to real values (Fig. 2.8), those ares represents impractical values, such as extremely small learning rate and too large of a batch size causes overfitting, thus impossible for the corresponding factors to chosen at such level for out computer experiment.

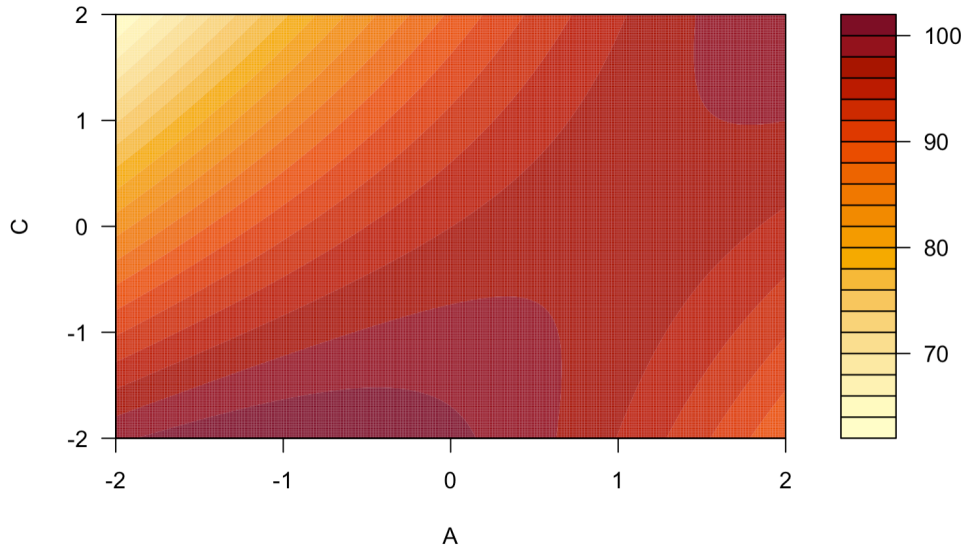

Fig. 2.7 Filled Contour Plot of Response Surface from CCD

3.3 Residual Analysis and Conclusion

Final Equation we derived from the above CCD is the following:

$Accuracy = 95.9752 + 3.0882\,A - 3.0375C - 1.6699A^2 + 2.9835AC - 0.3944C^2$

which have an Rsquared accuracy of 90.84%. The C squared term is kept because the interaction effect of Aand C is important, and keeping the second order term enables us to perform the response surface methodology, and therefore provide us with a mathematical model with stationary point to possibly explain the accuracy with respect to the factor settings and find the optimal settings.

As we can see since out x, y are provided within the [-2, 2] range, the intercept of 95.97 guarantees a high accuracy of any of the possible observations based on our model. Since the B matrix is neither Positive definite or negative definite, we can easily infer that the stationary point derived from the model equation is in fact a saddle point. This observation is in accordance with the contour plot shown above, the point we get A = 1.0573768, C = 0.1485594 is not a local or global maximum, but a saddle point. So in this case, we cannot get the theoretical maximum value as desired due to the shape of our model.

4. Shortcomings

The images in the MNIST dataset are pre-processed and normalized to make them easier for machine learning algorithms to process. For example, the images are centered and the background is removed. This preprocessing makes the dataset easier to work with, but it also means that the performance of machine learning algorithms trained on the MNIST dataset may not be directly applicable to real-world applications where the images are not pre-processed in the same way.

In our experiment, we used an empirical approach to tackle the prediction problem. However, we found that the stationary point derived from the response surface method was a saddle point, rather than a maximum. This means that it did not provide us with any

additional insight or useful information. As a result, we were unable to gain a better understanding of the relationship between the variables and the outcome using this method.

5. Summary and Future Work

Despite the above limitations, the MNIST dataset remains a popular choice for machine learning research because it is well-understood and widely used, making it easy to compare the performance of different algorithms. Additionally, the simplicity of the dataset means that it can be used to test and develop new machine learning techniques without the need for complex or specialized data.

Overall, experimental design has shown, again and again, to be a valuable tool for tuning hyperparameters in machine learning algorithms, as it allows us to systematically explore different combinations of hyperparameter values and identify the ones that lead to the best performance in a methodological manner.

Appendix

Fig. 1 Initial Screening Model Matrix

Run	Optimizer	Learning rate	Batch size	Layer	Model	outcome 1	outcome 2	ybar
1	1	1	1	1	-1	92.62	92.73	92.675
2	1	1	1	-1	-1	93.51	95.37	94.44
3	1	1	-1	1	-1	92.62	92.73	92.675
4	1	1	-1	-1	-1	95.2	96.07	95.635
5	-1	1	1	1	-1	80.21	83.45	81.83
6	-1	1	1	-1	-1	67.84	65.73	66.785
7	-1	1	-1	1	-1	90.61	84	87.305
8	-1	1	-1	-1	-1	23.97	60.89	42.43
9	1	-1	1	1	-1	90.83	91.77	91.3
10	1	-1	1	-1	-1	91.91	91.01	91.46
11	1	-1	-1	1	-1	91.09	92.34	91.715
12	1	-1	-1	-1	-1	93.92	93.38	93.65
13	-1	-1	1	1	-1	88.57	90.15	89.36
14	-1	-1	1	-1	-1	89.37	93.66	91.515
15	-1	-1	-1	1	-1	90.42	90.58	90.5
16	-1	-1	-1	-1	-1	76.67	93.64	85.155
17	1	1	1	1	-1	97.38	97.82	97.6
18	1	1	1	-1	-1	98.01	97.8	97.905
19	1	1	-1	1	-1	96.74	96.85	96.795
20	1	1	-1	-1	-1	97.1	97.48	97.29
21	-1	1	1	1	-1	80.64	79.3	79.97
22	-1	1	1	-1	-1	45.35	44.49	44.92
23	-1	1	-1	1	-1	20.23	21.03	20.63
24	-1	1	-1	-1	-1	7.06	22.75	14.905
25	1	-1	1	1	-1	97.72	97.45	97.585
26	1	-1	1	-1	-1	97.07	96.33	96.7
27	1	-1	-1	1	-1	97.59	97.2	97.395
28	1	-1	-1	-1	-1	97.61	98.23	97.92
29	-1	-1	1	1	-1	95.87	95.34	95.605
30	-1	-1	1	-1	-1	95.37	94.15	94.76
31	-1	-1	-1	1	-1	89.08	90.9	89.99
32	-1	-1	-1	-1	-1	86.46	80.98	83.72

Fig. 1.1 Initial Screening Model Parameter Estimates:

```
Call:
lm(formula = Accuracy ~ (A + B + C + D + E)^2, data = project_new)

Residuals:
     Min       1Q   Median       3Q      Max
-19.1713  -4.6175  -0.5181   3.7525  19.4737

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  83.8162     1.8952  44.226  < 2e-16 ***
A            11.3550     1.8952   5.992 1.88e-05 ***
B             3.9594     1.8952   2.089 0.053022 .
C             3.2419     1.8952   1.711 0.106469
D            -8.5794     1.8952  -4.527 0.000344 ***
E             2.3356     1.8952   1.232 0.235597
A:B          -4.1725     1.8952  -2.202 0.042709 *
A:C          -3.6956     1.8952  -1.950 0.068918 .
A:D           9.0350     1.8952   4.767 0.000210 ***
A:E          -4.5631     1.8952  -2.408 0.028475 *
B:C          -0.2769     1.8952  -0.146 0.885671
B:D           2.8194     1.8952   1.488 0.156280
B:E          -2.6906     1.8952  -1.420 0.174875
C:D           2.7062     1.8952   1.428 0.172526
C:E           0.2762     1.8952   0.146 0.885927
D:E           4.1494     1.8952   2.189 0.043731 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.72 on 16 degrees of freedom
Multiple R-squared:  0.8764,    Adjusted R-squared:  0.7606
F-statistic: 7.565 on 15 and 16 DF,  p-value: 0.0001136
```

Fig. 1.2 Initial Screening Model Residuals vs Fitted:

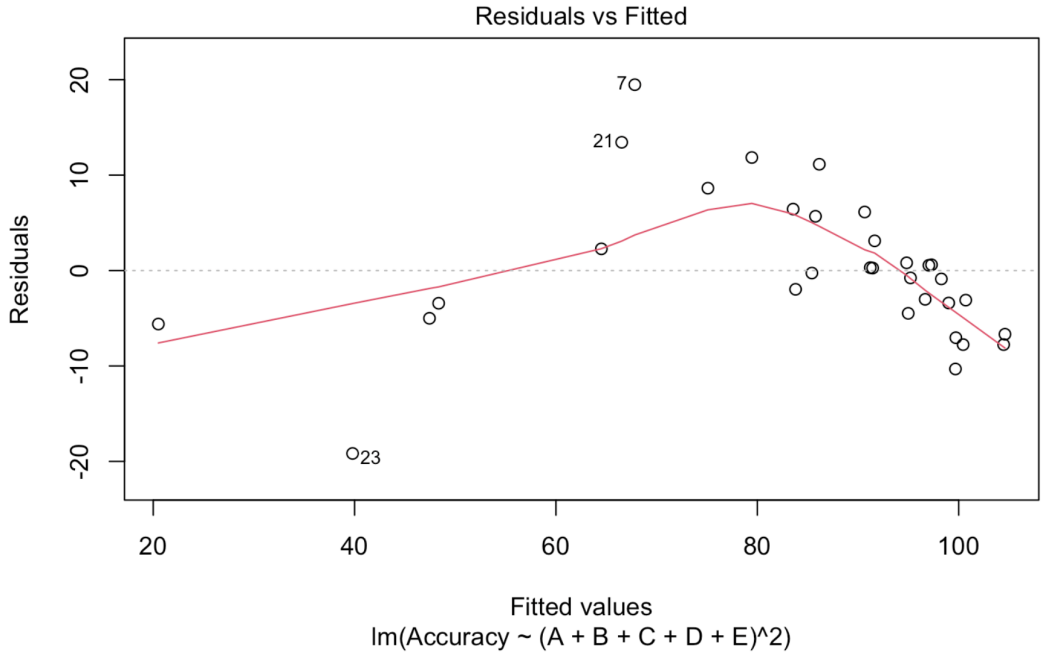

Fig. 1.3 Initial Screening Model Residuals QQ plot:

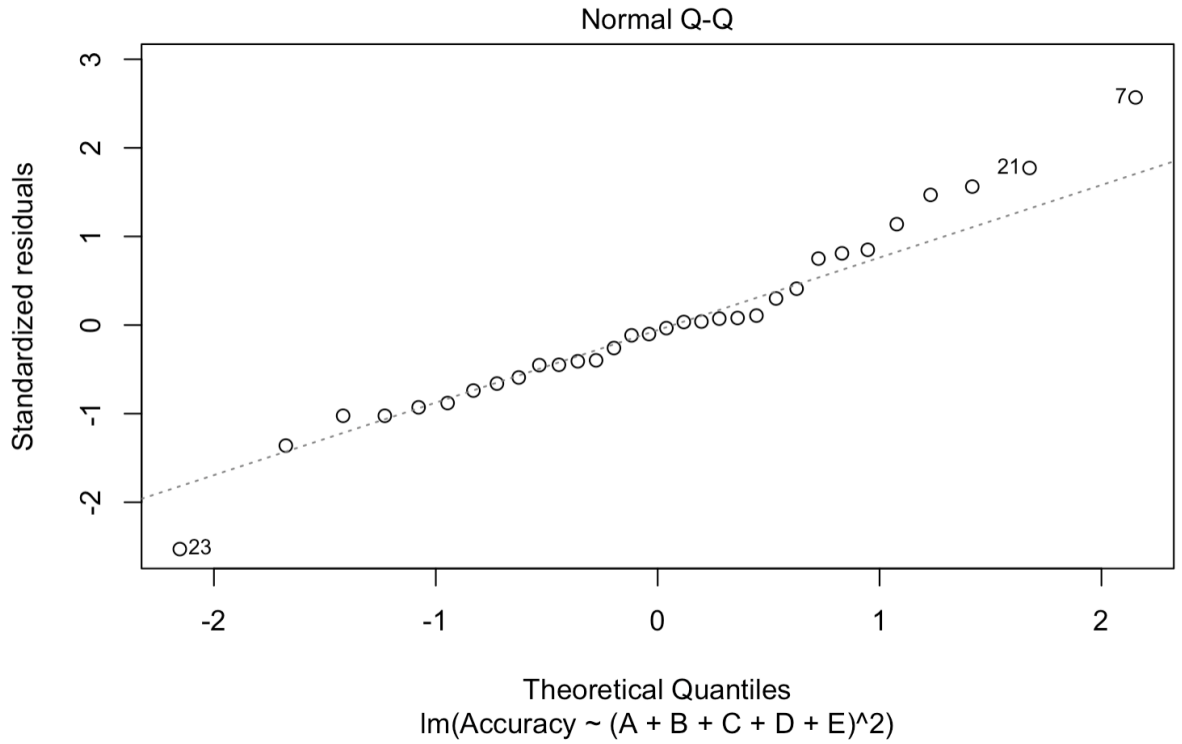

Fig. 1.4 Initial Screening Model studentized BP test

```
        studentized Breusch-Pagan test

data:  mod_2
BP = 13.193, df = 10, p-value = 0.2131
```

Fig. 2 Follow up Model Matrix

Run	Batch size	Middle layer	Learning rate	Accuracy from 5 Runs					Mean	SD	ln(s^2)
1	-1	-1	-1	97.51	97.43	97.1	96.94	97.64	97.324	0.2620381652	-2.678530235
2	-1	-1	1	91.97	90.63	88.14	91.98	90.34	90.612	1.407173053	0.6831655292
3	-1	1	-1	97.31	97.3	97.33	96.72	97.51	97.234	0.2682237872	-2.63186724
4	-1	1	1	89.71	92.54	92.15	93.19	11.35	75.788	32.24050893	6.946447406
5	1	-1	-1	97.29	97.95	98.17	98.13	97.97	97.902	0.3178930638	-2.29208046
6	1	-1	1	94.73	95.64	96.52	96.53	96.21	95.926	0.6797823181	-0.771965305
7	1	1	-1	98.02	97.89	97.18	97.48	97.66	97.646	0.2981006541	-2.420648168
8	1	1	1	95.48	95.17	96.72	94.26	95.03	95.332	0.8018827845	-0.4415856717
9	-1.68	0	0	89.56	77.59	88.55	88.63	90.86	87.038	4.796946529	3.135959151
10	1.68	0	0	96.71	96.96	97.23	95.87	96.9	96.734	0.4629730014	-1.540173078
11	0	-1.68	0	96.57	95.73	94.62	96.21	97.05	96.036	0.8297373078	-0.3732922498
12	0	1.68	0	97.16	96.38	96.27	96	95.86	96.334	0.452707411	-1.585018508
13	0	0	-1.68	98.55	98.25	98.2	97.95	97.87	98.164	0.2407986711	-2.847588169
14	0	0	1.68	91.19	93.15	93.23	93.72	92.75	92.808	0.8658036729	-0.2881942036
15	0	0	0	95.42	97.09	95.52	95.18	95.28	95.698	0.7056458035	-0.6972837243
16	0	0	0	96.13	96.82	95.19	96.63	95.33	96.02	0.6616947937	-0.8259017321
17	0	0	0	96.51	97.2	95.8	96.38	96.68	96.514	0.4529724053	-1.583848141
18	0	0	0	95.84	96.79	94.79	96.09	96.58	96.018	0.7009536361	-0.7106270675
19	0	0	0	96.66	96.71	96.24	95.97	92.74	95.664	1.487368145	0.7940164255
20	0	0	0	96.34	96.95	96.45	96.67	96.31	96.544	0.2391317628	-2.86148114

Fig. 2.1 Follow-up Model Parameter Estimates

```
Call:
lm(formula = ybar ~ (A + B + C)^2 + I(A^2) + I(B^2) + I(C^2),
    data = followup)

Residuals:
    Min      1Q  Median      3Q     Max
-4.1912 -0.7108 -0.0891  0.9352  3.4343

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  96.1081     1.0317  93.153 4.98e-16 ***
A             3.0882     0.6849   4.509  0.00113 **
B            -1.1186     0.6849  -1.633  0.13345
C            -3.0375     0.6849  -4.435  0.00126 **
I(A^2)       -1.6859     0.6675  -2.526  0.03008 *
I(B^2)       -0.1628     0.6675  -0.244  0.81227
I(C^2)       -0.4104     0.6675  -0.615  0.55235
A:B           1.7580     0.8944   1.965  0.07773 .
A:C           2.9835     0.8944   3.336  0.00755 **
B:C          -1.8840     0.8944  -2.106  0.06142 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.53 on 10 degrees of freedom
Multiple R-squared:  0.8728,    Adjusted R-squared:  0.7583
F-statistic: 7.624 on 9 and 10 DF,  p-value: 0.001926
```

Fig. 2.2 Follow-up Model Half-Normal Plot

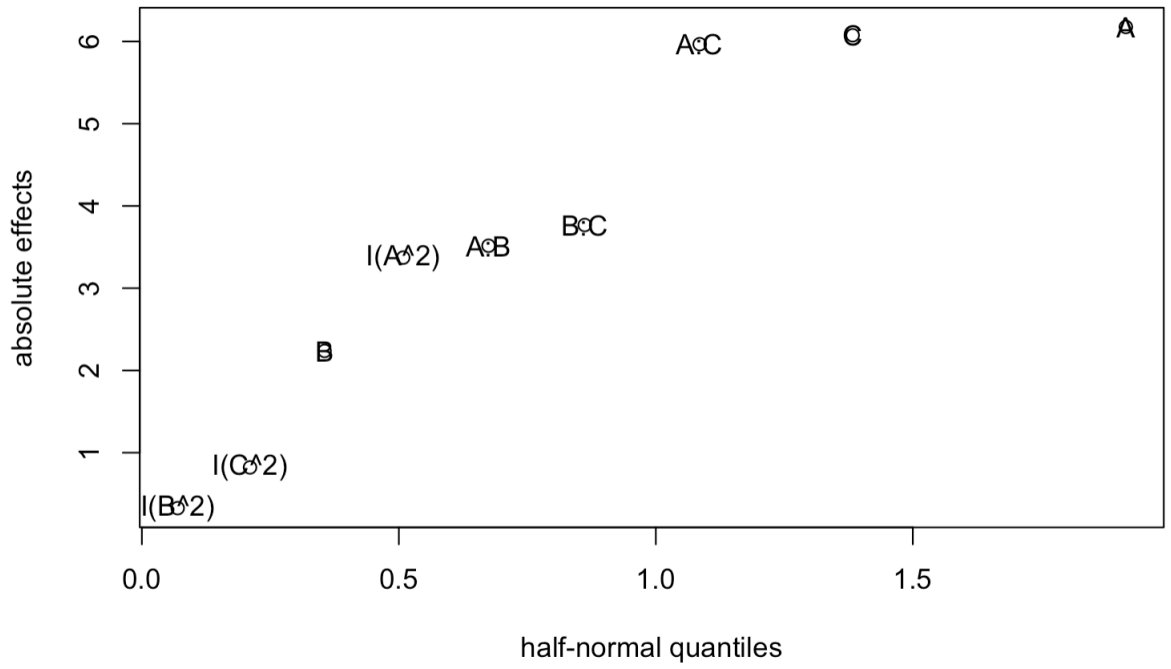

Fig. 2.3 Follow-up Model Residuals vs Fitted Plot

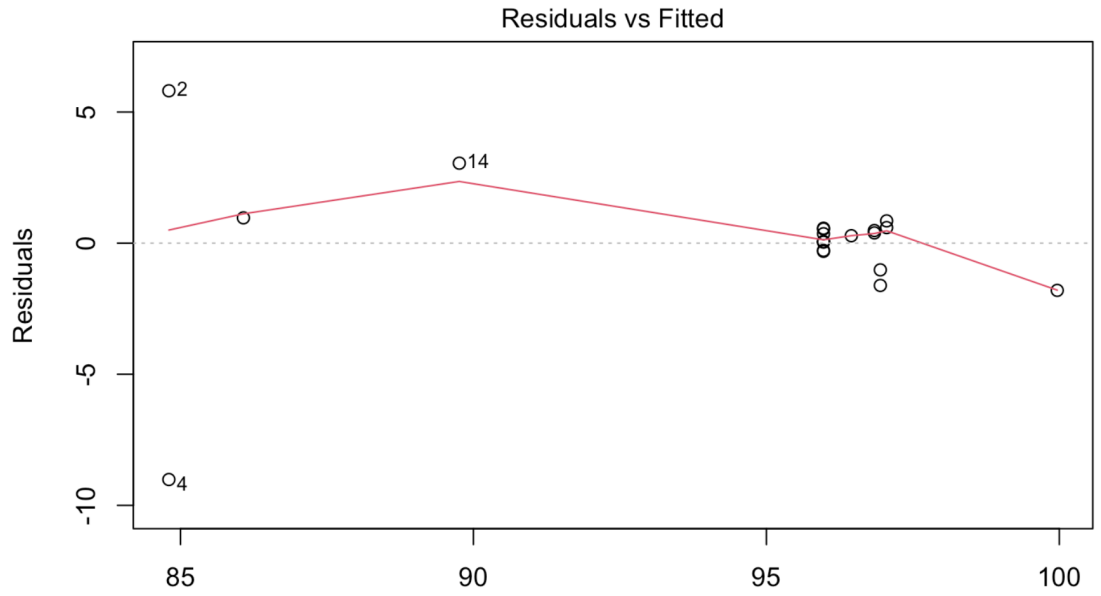

Fig 2.4 Follow-up Model Normal QQ plot

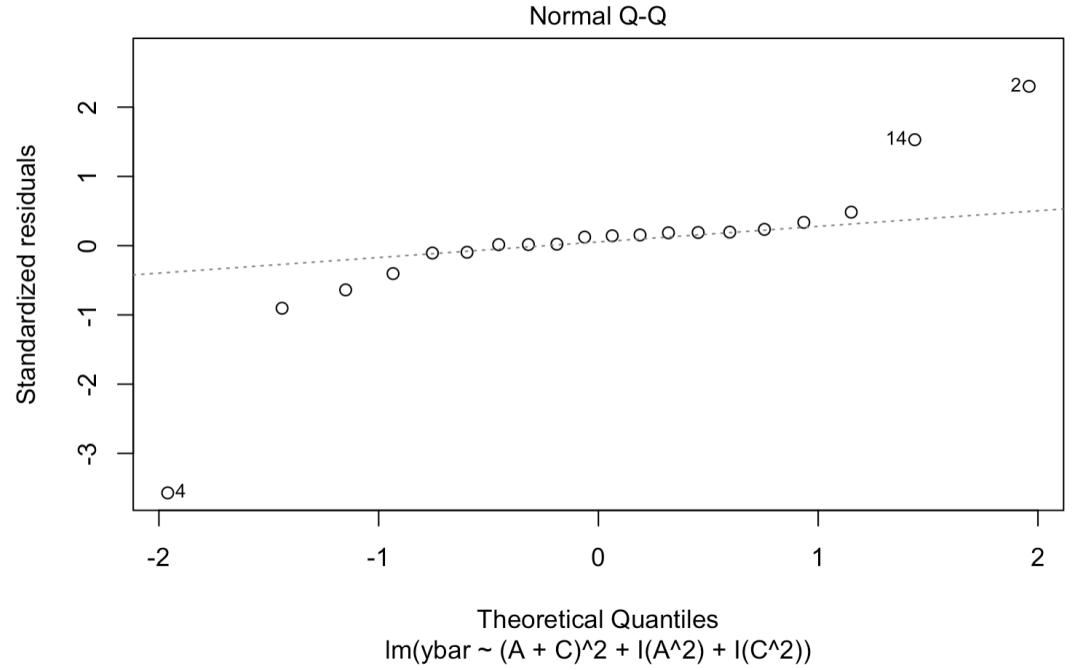

Fig 2.5 Interaction Plot (Significant)

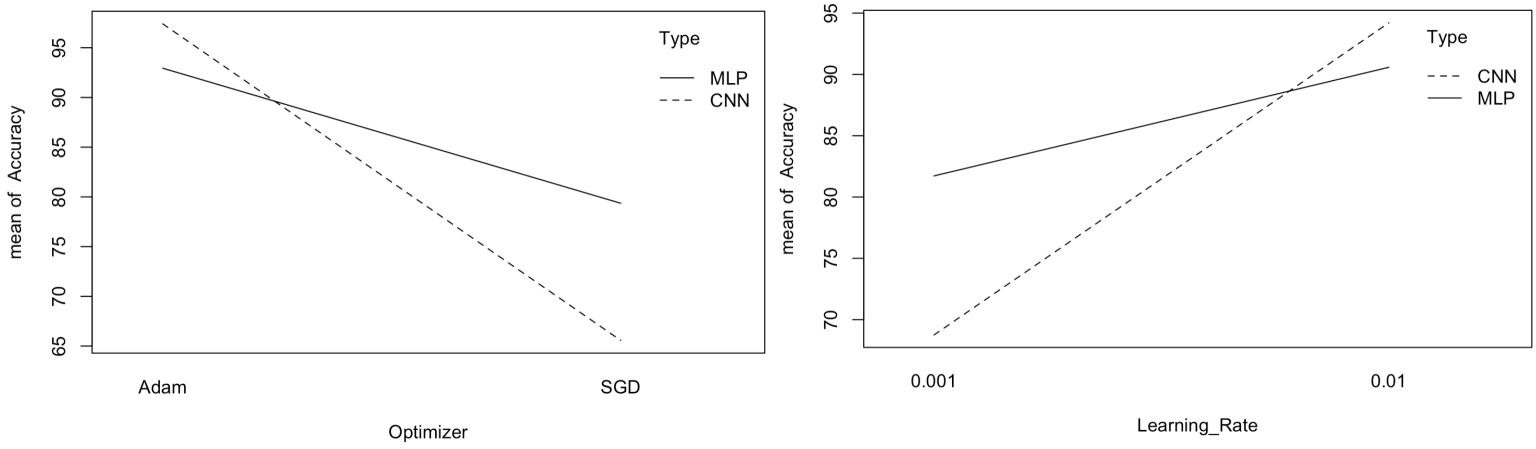

Fig 2.6 Parameter from Central Composite Design after selection

```
Call:
lm(formula = ybar ~ (A + C)^2 + I(A^2) + I(C^2), data = followup)

Residuals:
    Min      1Q  Median      3Q     Max
-9.0137 -0.2857  0.3214  0.5749  5.8103

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  95.9752     1.0737  89.391  < 2e-16 ***
A             3.0882     0.8393   3.679  0.00248 **
C            -3.0375     0.8393  -3.619  0.00279 **
I(A^2)       -1.6699     0.8140  -2.051  0.05943 .
I(C^2)       -0.3944     0.8140  -0.485  0.63550
A:C           2.9835     1.0962   2.722  0.01654 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.1 on 14 degrees of freedom
Multiple R-squared:  0.7325,    Adjusted R-squared:  0.637
F-statistic: 7.668 on 5 and 14 DF,  p-value: 0.00117
```

Fig 2.8 Contour plot with actual values