Screening AlexNet Hyperparameters with MNIST

STATS 201A Project

Adam Rohde
Ashley Chiu

December 6, 2019

MOTIVATION, BACKGROUND, and OBJECTIVE

Convolutional Neural Networks and AlexNet

The use of neural networks has become increasingly popular for many applications. This class of algorithms typically uses pre-labeled training data to learn patterns that are consistently correlated with given labels. Subsequently, the developed knowledge is used to properly classify new observations.

More specifically, Convolutional Neural Networks (CNN) have proven particularly effective in image recognition. As the name implies, such networks contain at least one convolutional layer[1,I]which reduce overall complexity, yielding simpler, more efficient, and easier to train models. While many architectures have been developed on the widely used training databases ImageNet and MNIST[II] , we choose to explore a seminal CNN architecture called AlexNet.

A deep CNN developed at the ImageNet LSVRC-2012 contest, AlexNet significantly outperformed competitor models in classifying high-resolution images in the ImageNet database. In brief, AlexNet utilizes 5 convolutional layers and 3 fully connected layers with final softmax combined with a variety of "new and unusual features [to] improve its performance."[III]

Hyperparameters

Neural networks require the tuning of a number of different hyperparameters, pre-set (i.e. before training) variables which determine the network structure and/or the way a network is trained.[IV,V] Such hyperparameters include, but are not limited to, activation functions, learning rate, batch size, number of layers, regularization methods like dropout, number of epochs, and normalization. Optimizing hyperparameters can become a substantial task due to the sheer number of possibilities. Even more, hyperparameter settings drastically affect the performance of the network: researchers employ a number of techniques, such as random or grid searches, to find an optimal combination hyperparameter settings.

Objective

We deploy an experimental approach to hyperparameter tuning in the context of AlexNet. We investigate the "new and unusual features" embedded within AlexNet, amongst other hyperparameters known to influence accuracy *to screen for the most important hyperparameters in an AlexNet-type CNN.*

EXPERIMENT

Neural Network ("The Model")

Using TensorFlow and the Keras [2] functional API in Python, we built a version of the AlexNet CNN, which follows the guidance of the actual AlexNet authors and a few additional web sources.[VI,VII,VIII] Since our objective is to identify which hyperparameters are most important in AlexNet-type CNNs, we attempt to emulate the original structure of AlexNet fairly closely.

It is important to note, however, that the AlexNet-type CNN we utilized in our data collection varies from the original application of AlexNet on the ImageNet dataset. AlexNet utilized a highly efficient GPU implementation and took six days to train a single model – due to time and hardware limitations, this was not a feasible implementation for our screening experiment. Instead, we utilize a smaller AlexNet-type CNN with max 3 convolutional layers and 2 fully connected layers with a final softmax on the more manageable MNIST dataset, compared to the original 8 layers with a final softmax. As the MNIST dataset also involves images and is sufficiently large, we expect this implementation will still allow us to collect and analyze useful data to reach our experiment's objective. We also chose a practical, specific set of hyperparameters to tune and test for importance.

[1] Convolutional layers derive their name from the convolution operation in mathematics and can be thought of as a "many-to-one" type mapping. The use of convolutional layers, thus, allows each layer to consider local information (i.e. dependencies such as neighboring pixels in an image).

[2] Keras provides a high-level interface with which to quickly develop and test neural networks.

Variables [3]

As mentioned, the original AlexNet authors incorporated "new and unusual features" to improve model accuracy, including the ReLU activation function, normalization, dropout, and additional convolutional layers (a deeper network). The authors claim that without these features, their model's performance significantly decreases. We test the importance of these and three other hyperparameters[IX]:

1. *Learning Rate (A)*: Refers to step size for the underlying gradient descent algorithm. The authors used an "equal learning rate for all layers, which [was] adjusted manually throughout training. The heuristic [they] followed was to divide the learning rate by 10 when the validation error rate stopped improving with the current learning rate. The learning rate was initialized at 0.01 and reduced three times prior to termination."[X] Learning rate is one of the most important tuning parameters in a neural network, as without a properly tuned learning rate, neural networks can often fail to converge. [XI] *We test learning rate settings of 0.01 and 0.0001 in our initial runs.*[4]

2. *Number of Epochs (B)*: Refers to the number of times the entire training dataset is shown to the network while training. Intuitively, number of epochs are generally maximized, allowing for more opportunities to learn. However, too large a number can result in overfitting. *We test our model with 2 and 10 epochs, based on prior domain knowledge and experience with the MNIST dataset.*

3. *Batch Size (C)*: Refers to the number of training examples given to the network at each pass before network parameter are updated. Similar to number of epochs, this is not a main tuning parameter for the authors; however, we know inherently that tuning batch size can improve model accuracy. *For simplicity, we test batch sizes of 50 and 100.*

4. *Dropout (D)*: Forces a more robust model and refers to the dropping of neurons throughout the learning process. The authors claim that "without dropout, [the] network exhibits substantial overfitting. Dropout roughly doubles the number of iterations required to converge." [XII] *We test the model with and without the dropout in the fully connected layers.*

5. *Activation Function (E)*: Refers to the introduction of nonlinearity. AlexNet was the first CNN to use the ReLU function in this capacity.[XIII] Until this introduction, the standard activation function was tanh. The authors "applied [ReLU] to the output of every convolutional and fully-connected layer" [XIV] and claim that "deep convolutional neural networks with ReLUs train several times faster than their equivalents with tanh units." [XV] *We test the effects of ReLU and tanh activations.*

6. *Additional Convolutional Layer (F)*: Refers to additional depth, which the authors write "seems to be important…removing any convolutional layer…resulted in inferior performance." [XVI] *As such, we tested the model with 2 and 3 convolutional layers.*

7. *Normalization (G)*: Rescales data by normalizing the output of each activation layer (subtracts batch mean and divides by batch standard deviation). The authors claim normalization reduced their error rates by 1-2%.[XVII] *As such, we test the model without and with batch normalization.*

Recorded Response

To evaluate the effect(s) of our variables, we recorded test accuracy.[5]

[3] The two levels listed for each variable correspond to -1 and 1 in our model matrix, respectively. See Appendix B.

[4] Analysis of our initial experiment's data indicated that factor settings for learning rate needed to be further tuned (similar to the AlexNet authors' approach). Refer to Analysis portion for further detail and revised factor settings.

[5] Accuracy is a percentage calculated as the number of correctly identified digits over the total number of digits in the test data.

Design

It is clear from our discussion of variables, that we are employing a 2^k factorial design ($k = 7$), as each variable (factor) was tested at two levels. Due to computing limitations, we ultimately settled on a 2^{7-2}_{IV} randomized block design with 2 replicates, blocked by experimenter (student) with defining relationship $I = ABCDF = ABDGE = CEFG$ (Appendix B). Overall, the experiment contains 32 runs and 64 total observations. Since we are using a resolution IV design, none of our main effects are aliased with two-factor interactions ($me = 3fi$, $2fi = 2fi$). This is a simpler aliasing structure and should allow us to more clearly and easily see distinct results.[6,XVIII]

Note that the findings from our initial runs/experiment led us to run a follow-up experiment with tuned factor levels for learning rate. *All design, randomization, and blocking principles discussed herein apply throughout the various iterations of our experiment.*

Blocking and Randomization

We utilize a randomized blocked design to reduce/control variability that could alter the reliability of our results (nuisance variables, e.g. version of Python, TensorFlow, machine processing). Each student conducted experiments and collected data independently using the same model, yielding two replicates. Run orders were randomized separately for each iteration[7] based on different random seeds.

Data Collection Procedures

We executed our AlexNet CNN Python scripts[8] overnight on the same dates. After each run, test accuracy and training time data were automatically written to CSV via Python on our respective machines. Each script execution took approximately 3 hours.

ANALYSIS

Part I - Initial Experiment

Exploratory Data Analysis

We read our collective data into R, first performing initial exploratory data analysis by constructing boxplots of accuracy as a function of each factor (Appendix A, Fig. 1). From these basic plots, we saw learning rate (A) to be the most obvious, significant factor influencing test accuracy. In this preliminary stage, we also see that normalization (G) may affect accuracy. Interestingly, we also see a large amount of variation between blocks. Our half normal plot of main effects verifies these results (Appendix A, Fig. 2). Lastly, our data appears to have some significant outliers and influential points.

Main Effects Model

We proceeded by running an initial linear model on only the main effects, confirming that Learning Rate (A) is significant at $\alpha = 5\%$. While block and normalization (G) appeared to be significant factors in our exploratory analysis, the main effects model does not necessarily support these results (Appendix A, Fig. 3). Further, a quick residual analysis of this main effects model showed poor results (Appendix A, Fig. 4), as 1) the data do not appear to satisfy the condition of homoscedasticity and 2) the Normal Q-Q plot, while showing no obvious patterns, re-highlighted a potential issue with outliers.

StepAIC – ME + Two-Factor Interactions

To follow, based on the poor residual plots, we concluded that we should incorporate some interaction effects to improve the model. We run both forward and backward step-wise regression capturing all main effects and their two-factor interactions (we assume that three-factor and higher-level interactions

[6] Our model matrix can be found in Appendix B. The full alias structure can be found in source XVIII.

[7] We were required to run two iterations of our experiment, as our initial factor settings for learning rate needed to be further tuned. As such, there were four total instances of data collection (two per block/researcher).

[8] See Appendix C.

are negligible). We do this to identify the best model that incorporates two-factor interactions without overfitting. Note, in our resolution IV design, no main effects are aliased with 2-factor interactions.

StepAIC created a final model with A, $B : D$, $C : E$, $C : F$, and $C : G$ significant at $\alpha = 10\%$ (Appendix A, Fig. 6). Block was not significant at this α, but appears to be significant in the half normal plot (Appendix A, Fig. 5). Based on these results, we build a final model (Appendix A, Fig. 7)

$$Accuracy \sim A + B + C + D + E + F + G + B : D + C : E + C : F + C : G + Block$$

R^2 and Adjusted R^2 are not particularly high in this model at 0.4389 and 0.3069, respectively. However, as this is a screening experiment to determine which hyperparameters are most important in a neural network, we are not overly concerned with fit/prediction capabilities. We proceeded by analyzing the residuals of this "final" model, which still showed poor results (Appendix A, Fig. 8). Specifically, the plot of residuals vs. fitted values shows a distinct linear relationship, with a negative slope. In addition, the Normal Q-Q plot shows a slightly s-shaped curve with tails. After seeing these results in the residual analysis, we attempted to transform the response variable, accuracy. However, no transformation, including log, sqrt, or power, appeared to improve the residuals[9].

Outliers

Upon closer inspection of the pattern in the plot of residuals vs. fitted values (Appendix A, Fig. 8), we saw that most of the unusual variability was clustered towards the lower accuracy rates (fitted values of approx. 0.5 to 0.8) – based on our domain knowledge of neural networks, we developed a suspicion that runs with the higher learning rate ($A = -1$) might be exhibiting a convergence issue.[10,XIX]

To address this suspicion, we removed outliers from our dataset.[11] In reviewing the outlier dataset, we saw that all of the observations were indeed run with the higher learning rate of 0.01, confirming our initial suspicion. Following the removal of these outliers, we re-ran the entire analysis discussed above (Appendix A, Fig. 9-14). This time, our StepAIC model showed almost all main effects and two-factor interactions to be significant at $\alpha = 5\%$ (Appendix A, Fig. 10-11), learning rate (A) still being the most significant. Block was no longer significant.

Once again, we reviewed the residual plots (Appendix A, Fig.12) and saw much improvement, as there was no longer a negative sloping line in the plot of residuals vs. fitted values. However, the residuals still do not satisfy the condition of homoscedasticity. In addition, the s-shaped pattern still remained on the Normal Q-Q plot. Again, we tried a log-transform on the response, which improved the Normal Q-Q plot. Still, we see some minor deviations at the tails (Appendix A, Fig. 13-14).

We acknowledge that we cannot reasonably rely on the subset data, as we have no way of knowing whether we truly only removed noise and not signal. As such, *based on these initial findings and residual analysis, we ran a follow-up experiment.*

Part II - Follow-Up Experiment (Learning Rate factor settings adjusted)

We re-ran our entire experiment with better-tuned factor settings on learning rate (reduced the lower factor setting by a factor of 10), a similar methodology employed by the authors of AlexNet ($A = -1$ corresponds to learning rate = 0.001 and $A = +1$ corresponds to learning rate = 0.0001)[12].

In this second iteration, we continued with the same analysis and re-ran a main effects model and StepAIC with all main effects and two-factor interactions (note that we maintained the log transform on our response variable, per our first round of analysis). Our new main effects model showed that learning rate, again, was significant, along with Number of Epochs (B) (Appendix A, Fig. 15-16). Still,

[9] R output and plots from this exercise intentionally omitted.

[10] Too large a step size can cause the gradient algorithm to consistently miss the local optima, causing training to diverge.

[11] Outliers were defined as observations with accuracy greater than or less than $1.5 \pm IQR$.

[12] See Appendix B.

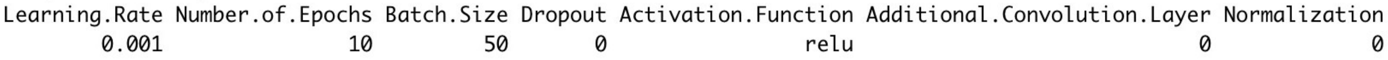
we believe that important two-factor interactions are not included in this model, so consistent with our initial experiment, we run a StepAIC (Appendix A, Fig. 17-18). The results showed that almost all main effects (except the additional convolutional layer factor) and many two-factor interactions were significant at $\alpha = 5\%$.[13] This time, when performing the residual analysis (Appendix A, Fig. 19), we noticed no obvious pattern in the plot of residuals vs. fitted values. Further, the data did not benefit from any additional transformations. The normal Q-Q plot also shows data mostly on a straight line, with minor deviations at the tails. We acknowledge that there are leverage/influential points in the data, but do not feel we have majorly violated the residual assumptions of homoscedasticity and normally distributed residuals. As such, the final model through StepAIC is as follows:

$$\begin{aligned} log(Accuracy) \sim & A + B + C + D + E + F + G + \\ & A:B + A:C + A:D + A:E + A:F + A:G + \\ & B:C + B:D + B:G + C:D + C:E + C:G + D:E + D:F + D:G \end{aligned}$$

See Appendix A, Fig. 18 for the full results of our final model. This includes the relative significances of the various factors. Based on the final two levels for each factor in our experiment, the optimal settings for this particular implementation of the AlexNet CNN would thus be:

$$A = -1, B = 1, C = -1, D = -1, E = -1, F = -1, G = -1$$

In terms of actuals variables:



```
Learning.Rate Number.of.Epochs Batch.Size Dropout Activation.Function Additional.Convolution.Layer Normalization
        0.001               10         50       0                relu                            0             0
```

CONCLUSION AND POSSIBLE FUTURE WORK

Our experiments support several of the AlexNet authors' findings on important factors for improved image recognition. The clearest result is that learning rate is a crucial hyperparameter that requires continuous refinement throughout the training process: a poor choice of learning rate can outweigh the positive effect(s) of other well-chosen hyperparameters. In addition, we confirmed that longer training time (in terms of epochs) and ReLU activations tend to improve performance.

However, our results also contradict some of the authors' conclusions. First, adding an additional convolutional layer was not a significant factor. Further, dropout and normalization were actually found to lead to poorer performance within the context of our experiment. We suspect this may be because our CNN implementation used the smaller MNIST dataset, and these particular features (focused on overfitting) may not have been as useful for this less complex dataset. We also see that there are several important interaction effects that have significant consequences for performance. Some such interactions might even be more important than the main effects of the factors themselves (e.g., the interaction between dropout and activation function might be more influential for accuracy than the main effect for either of these factors individually). This suggests that a great deal of care must be taken in the selection of hyperparameter settings in the development of CNNs.

Further investigation could test these hypotheses, but even more, we might test whether these same factors have significant effects on training time, as practitioners constantly seek a balance of accuracy and training time (i.e. overall efficiency). It might be that long training time generally improves accuracy, as we observed here, but that real-world constraints limit this luxury. Similarly, ReLU activation functions might prove to be more important than other factors within a broader context of accuracy and training time. It seems clear that the ultimate key to achieving a network with optimal performance is balancing all of these important hyperparameters in the context of the complexity of the data at hand.

[13] Interaction plots have been excluded from this report due to the large number of plots.

Sources

[I] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12), F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 1. Curran Associates Inc., USA, 1097-1105. [AlexNet]

[II] https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d

[III] AlexNet, P.1

[IV] https://arxiv.org/pdf/1705.08520.pdf

[V] https://arxiv.org/pdf/1803.09820.pdf

[VI] https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d

[VII] https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py

[VIII] https://towardsdatascience.com/a-walkthrough-of-convolutional-neural-network-7f474f91d7bd

[IX] https://arxiv.org/pdf/1803.09820.pdf

[X] AlexNet, p. 6

[XI] https://arxiv.org/abs/1206.5533

[XII] AlexNet, p. 6

[XIII] https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951

[XIV] AlexNet, p. 4

[XV] AlexNet, p. 3

[XVI] AlexNet, p. 2

[XVII] AlexNet, p. 4

[XVIII] Appendix A-18 in Douglas C. Montgomery. 2020. Design and Analysis of Experiments. 10th Edition, Wiley.

[XIX] https://arxiv.org/abs/1206.5533

Appendix A – Figures

Note: Interaction plots intentionally omitted due to volume

Appendix A - Figures

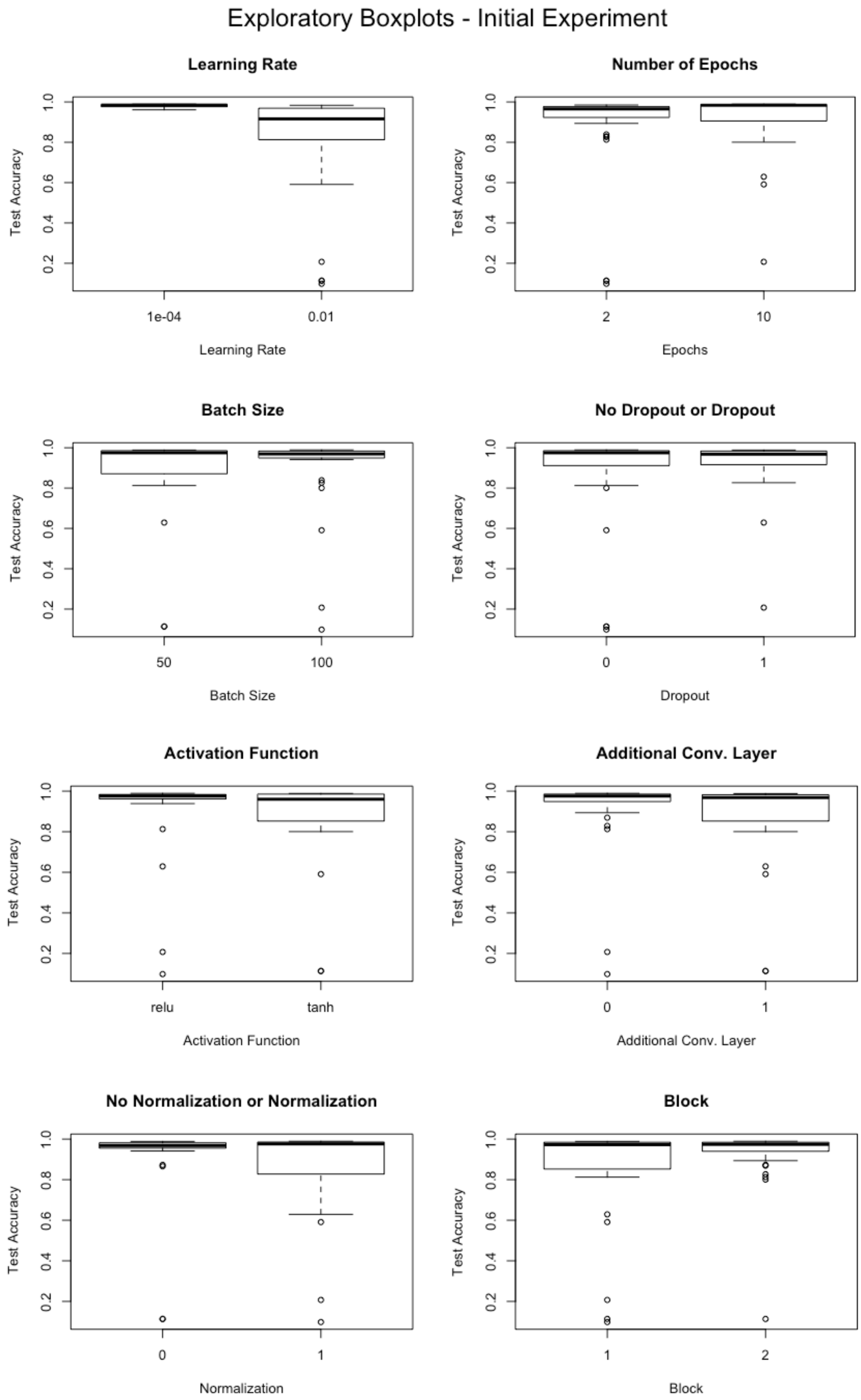


Figure 1: *Exploratory Data Analysis - Boxplots (Initial Experiment)*

Appendix A - Figures

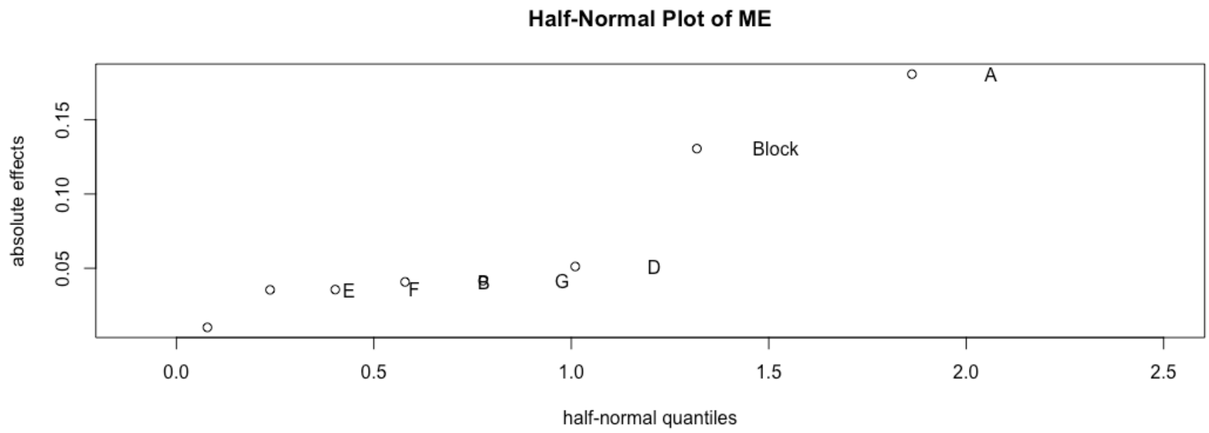
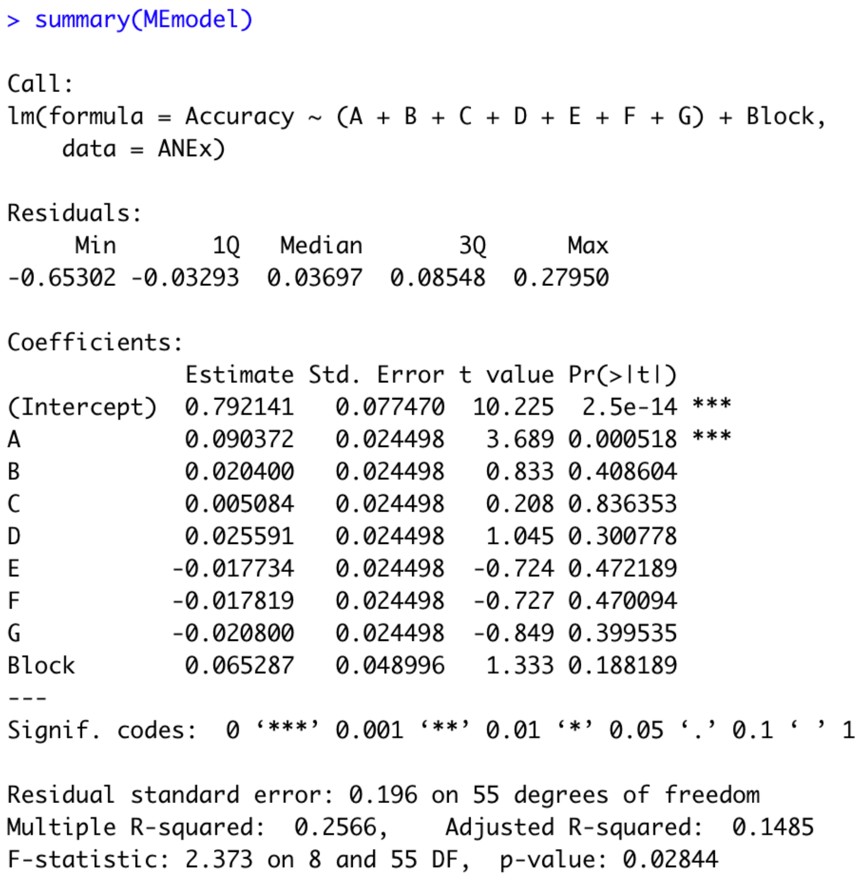


Figure 2: *Half-Normal Plot - Main Effects Model (Initial Experiment)*



```
> summary(MEmodel)

Call:
lm(formula = Accuracy ~ (A + B + C + D + E + F + G) + Block,
    data = ANEx)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.65302 -0.03293  0.03697  0.08548  0.27950 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.792141   0.077470  10.225  2.5e-14 ***
A            0.090372   0.024498   3.689 0.000518 ***
B            0.020400   0.024498   0.833 0.408604    
C            0.005084   0.024498   0.208 0.836353    
D            0.025591   0.024498   1.045 0.300778    
E           -0.017734   0.024498  -0.724 0.472189    
F           -0.017819   0.024498  -0.727 0.470094    
G           -0.020800   0.024498  -0.849 0.399535    
Block        0.065287   0.048996   1.333 0.188189    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.196 on 55 degrees of freedom
Multiple R-squared:  0.2566,	Adjusted R-squared:  0.1485 
F-statistic: 2.373 on 8 and 55 DF,  p-value: 0.02844
```

Figure 3: *R Output - Main Effects Model (Initial Experiment)*

Appendix A - Figures

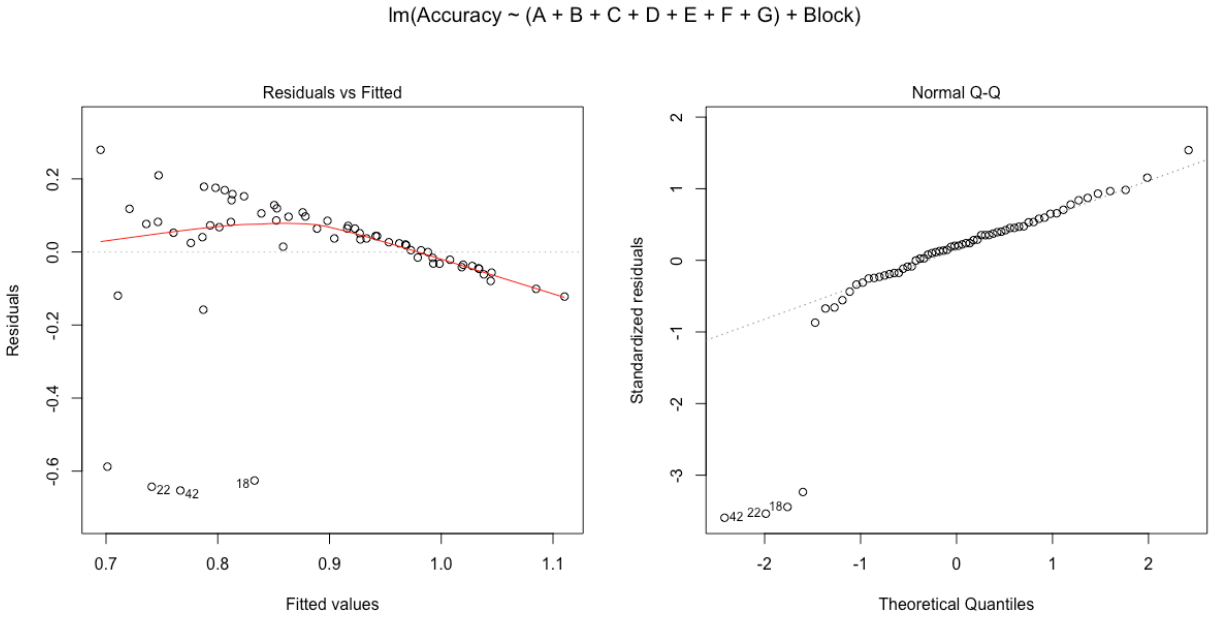


Figure 4: *Residual Plots - Main Effects (Initial Experiment)*

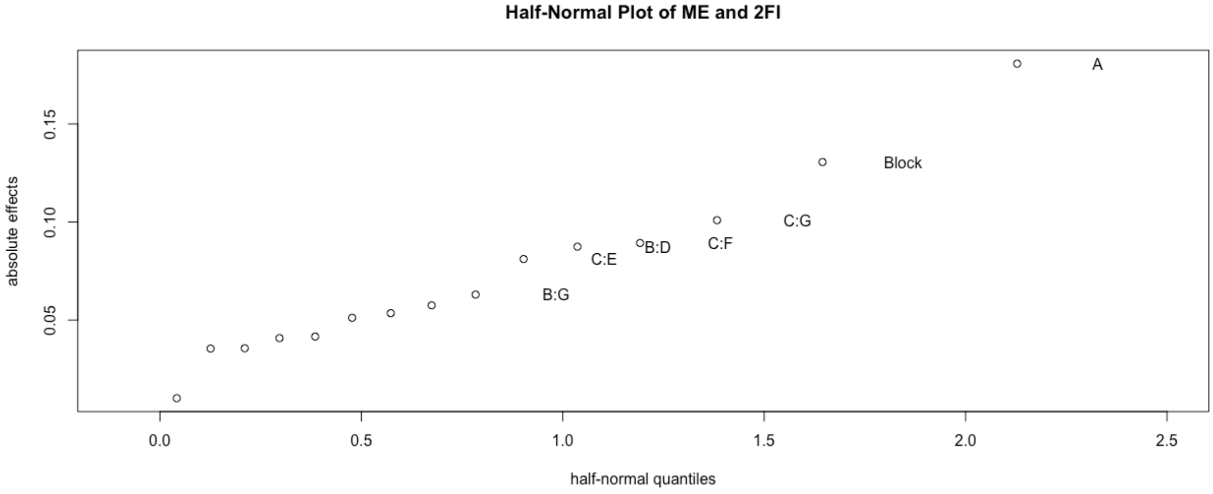
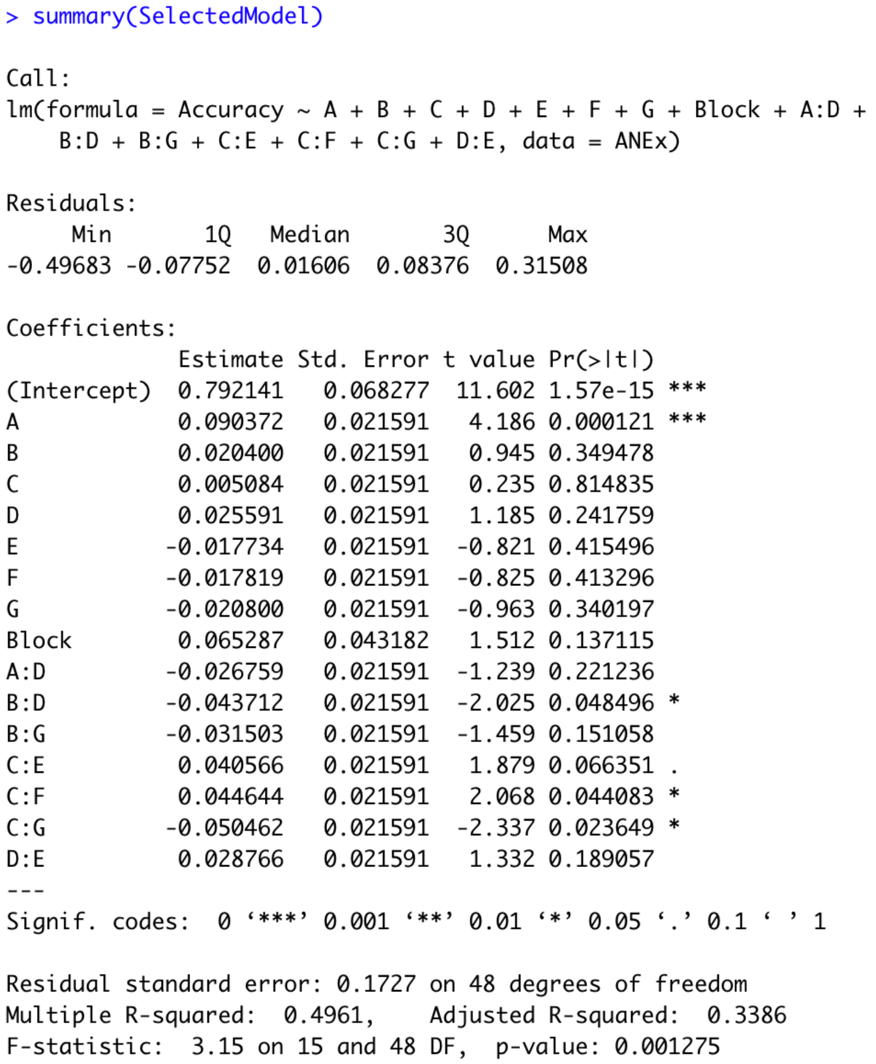
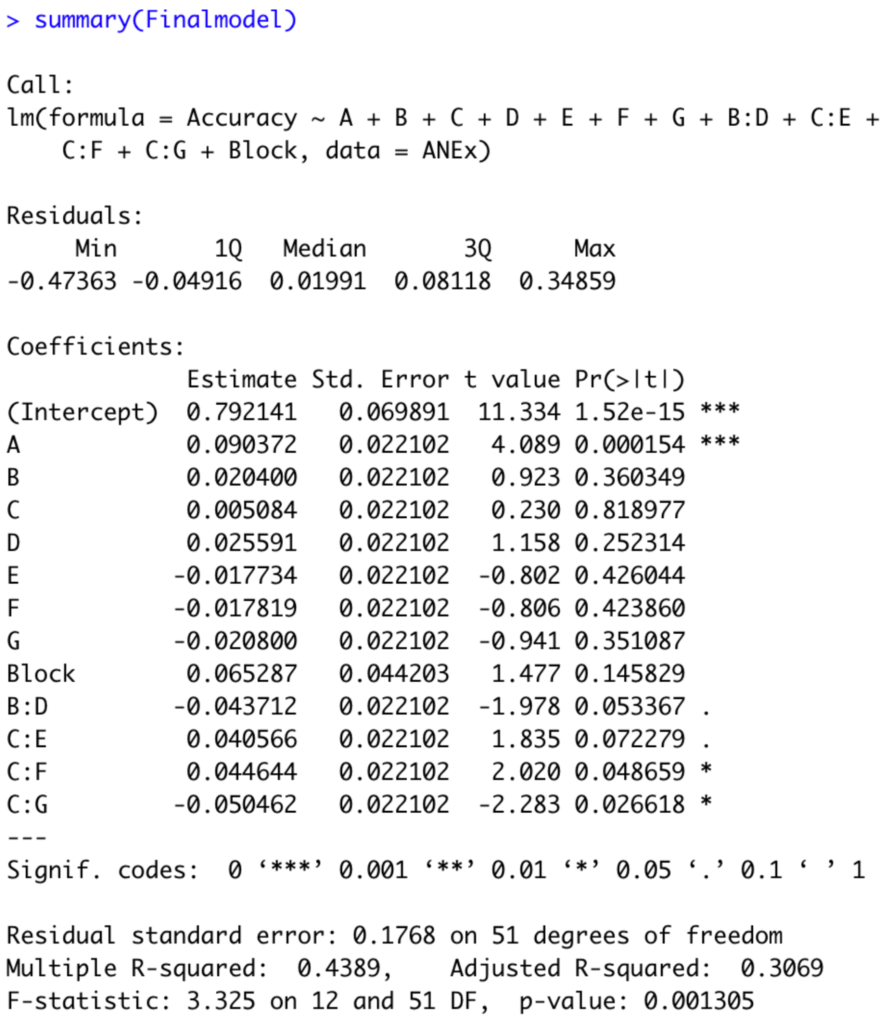


Figure 5: *Half-Normal Plot - Main Effects and Two-Factor Interactions (Initial Experiment)*



```
> summary(SelectedModel)

Call:
lm(formula = Accuracy ~ A + B + C + D + E + F + G + Block + A:D +
    B:D + B:G + C:E + C:F + C:G + D:E, data = ANEx)

Residuals:
     Min       1Q   Median       3Q      Max
-0.49683 -0.07752  0.01606  0.08376  0.31508

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.792141   0.068277  11.602 1.57e-15 ***
A            0.090372   0.021591   4.186 0.000121 ***
B            0.020400   0.021591   0.945 0.349478
C            0.005084   0.021591   0.235 0.814835
D            0.025591   0.021591   1.185 0.241759
E           -0.017734   0.021591  -0.821 0.415496
F           -0.017819   0.021591  -0.825 0.413296
G           -0.020800   0.021591  -0.963 0.340197
Block        0.065287   0.043182   1.512 0.137115
A:D         -0.026759   0.021591  -1.239 0.221236
B:D         -0.043712   0.021591  -2.025 0.048496 *
B:G         -0.031503   0.021591  -1.459 0.151058
C:E          0.040566   0.021591   1.879 0.066351 .
C:F          0.044644   0.021591   2.068 0.044083 *
C:G         -0.050462   0.021591  -2.337 0.023649 *
D:E          0.028766   0.021591   1.332 0.189057
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1727 on 48 degrees of freedom
Multiple R-squared:  0.4961,    Adjusted R-squared:  0.3386
F-statistic:  3.15 on 15 and 48 DF,  p-value: 0.001275
```

Figure 6: *R Output - StepAIC Model with Main Effects and 2fi (Initial Experiment)*



```
> summary(Finalmodel)

Call:
lm(formula = Accuracy ~ A + B + C + D + E + F + G + B:D + C:E +
    C:F + C:G + Block, data = ANEx)

Residuals:
     Min       1Q   Median       3Q      Max
-0.47363 -0.04916  0.01991  0.08118  0.34859

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.792141   0.069891  11.334 1.52e-15 ***
A            0.090372   0.022102   4.089 0.000154 ***
B            0.020400   0.022102   0.923 0.360349
C            0.005084   0.022102   0.230 0.818977
D            0.025591   0.022102   1.158 0.252314
E           -0.017734   0.022102  -0.802 0.426044
F           -0.017819   0.022102  -0.806 0.423860
G           -0.020800   0.022102  -0.941 0.351087
Block        0.065287   0.044203   1.477 0.145829
B:D         -0.043712   0.022102  -1.978 0.053367 .
C:E          0.040566   0.022102   1.835 0.072279 .
C:F          0.044644   0.022102   2.020 0.048659 *
C:G         -0.050462   0.022102  -2.283 0.026618 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1768 on 51 degrees of freedom
Multiple R-squared:  0.4389,    Adjusted R-squared:  0.3069
F-statistic: 3.325 on 12 and 51 DF,  p-value: 0.001305
```

Figure 7: *R Output - Final Model (Initial Experiment)*

Appendix A - Figures

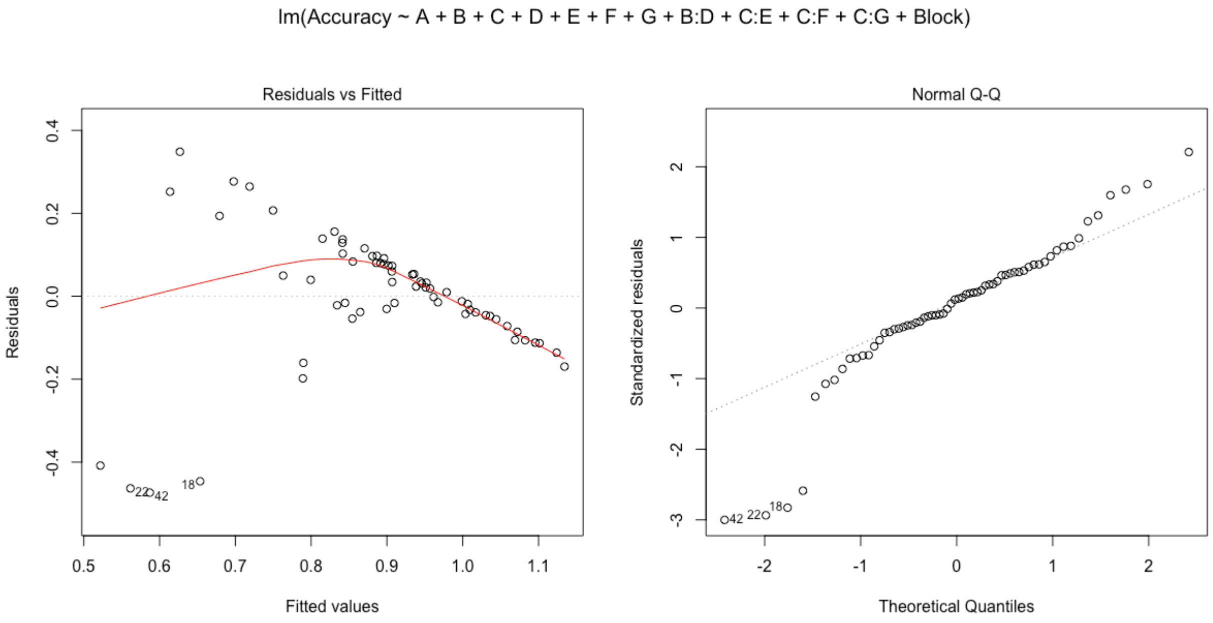


Figure 8: *Residual Plots - Final Model (Initial Experiment)*

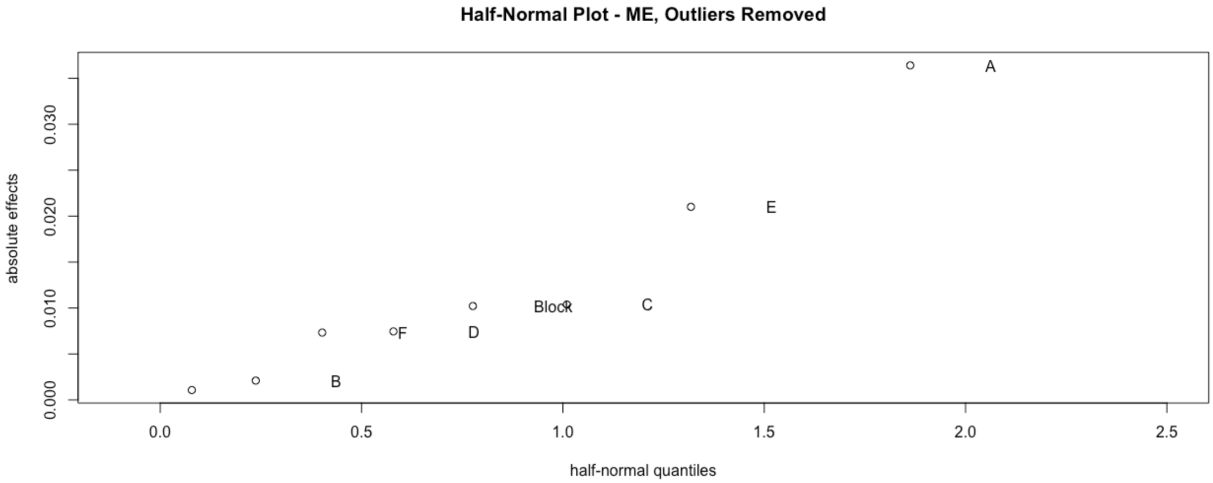


Figure 9: *Half-Normal Plot - Main Effects, Outliers Removed (Initial Experiment)*

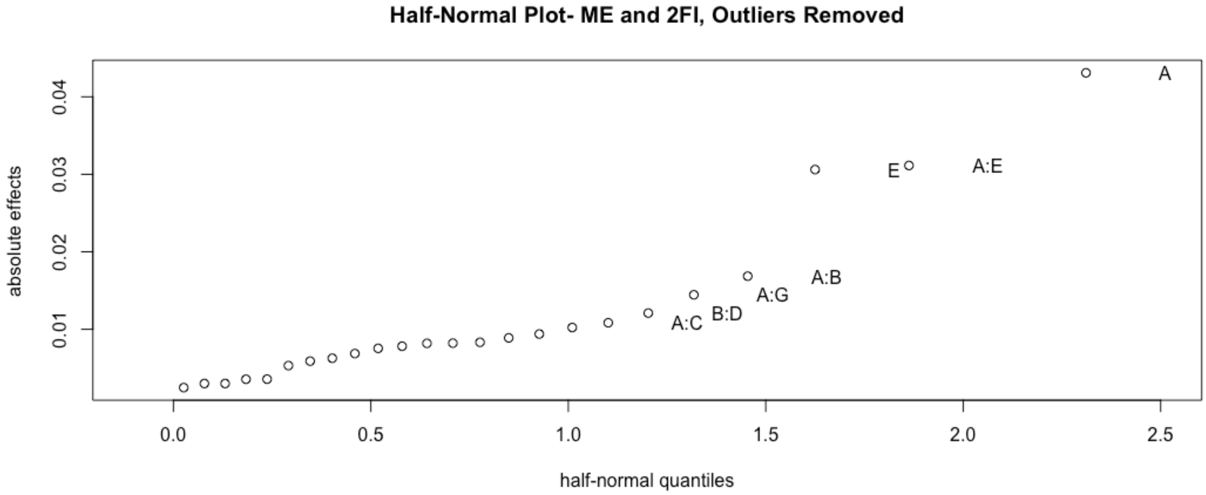
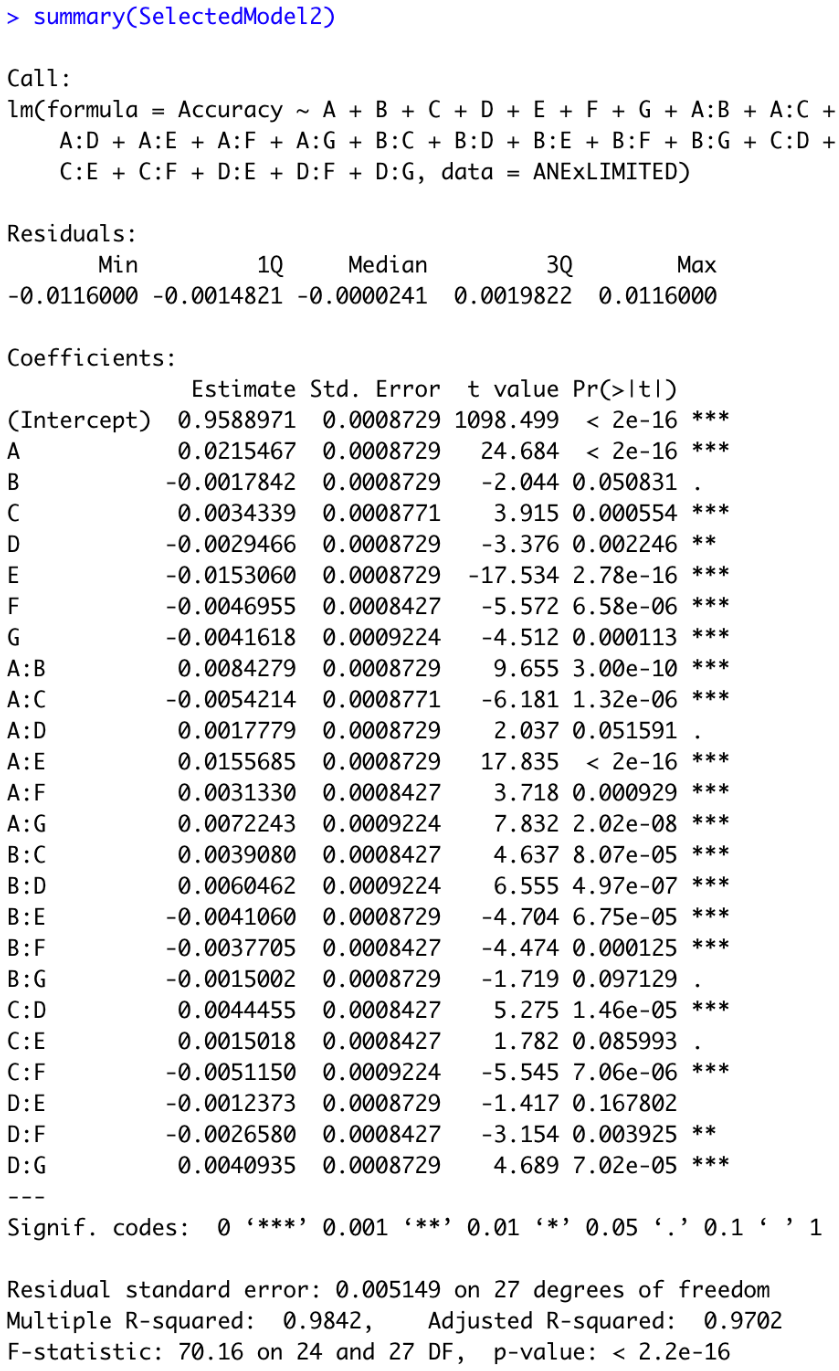


***Figure 10:** Half-Normal Plot - Main Effects and Two-Factor Interactions, Outliers Removed (Initial Experiment)*

Appendix A - Figures



```
> summary(SelectedModel2)

Call:
lm(formula = Accuracy ~ A + B + C + D + E + F + G + A:B + A:C +
    A:D + A:E + A:F + A:G + B:C + B:D + B:E + B:F + B:G + C:D +
    C:E + C:F + D:E + D:F + D:G, data = ANExLIMITED)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0116000 -0.0014821 -0.0000241  0.0019822  0.0116000

Coefficients:
              Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.9588971  0.0008729 1098.499  < 2e-16 ***
A            0.0215467  0.0008729   24.684  < 2e-16 ***
B           -0.0017842  0.0008729   -2.044 0.050831 .
C            0.0034339  0.0008771    3.915 0.000554 ***
D           -0.0029466  0.0008729   -3.376 0.002246 **
E           -0.0153060  0.0008729  -17.534 2.78e-16 ***
F           -0.0046955  0.0008427   -5.572 6.58e-06 ***
G           -0.0041618  0.0009224   -4.512 0.000113 ***
A:B          0.0084279  0.0008729    9.655 3.00e-10 ***
A:C         -0.0054214  0.0008771   -6.181 1.32e-06 ***
A:D          0.0017779  0.0008729    2.037 0.051591 .
A:E          0.0155685  0.0008729   17.835  < 2e-16 ***
A:F          0.0031330  0.0008427    3.718 0.000929 ***
A:G          0.0072243  0.0009224    7.832 2.02e-08 ***
B:C          0.0039080  0.0008427    4.637 8.07e-05 ***
B:D          0.0060462  0.0009224    6.555 4.97e-07 ***
B:E         -0.0041060  0.0008729   -4.704 6.75e-05 ***
B:F         -0.0037705  0.0008427   -4.474 0.000125 ***
B:G         -0.0015002  0.0008729   -1.719 0.097129 .
C:D          0.0044455  0.0008427    5.275 1.46e-05 ***
C:E          0.0015018  0.0008427    1.782 0.085993 .
C:F         -0.0051150  0.0009224   -5.545 7.06e-06 ***
D:E         -0.0012373  0.0008729   -1.417 0.167802
D:F         -0.0026580  0.0008427   -3.154 0.003925 **
D:G          0.0040935  0.0008729    4.689 7.02e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.005149 on 27 degrees of freedom
Multiple R-squared:  0.9842,	Adjusted R-squared:  0.9702
F-statistic: 70.16 on 24 and 27 DF,  p-value: < 2.2e-16
```

Figure 11: *R Output - StepAIC Model with Main Effects and 2fi, Outliers Removed (Initial Experiment)*

Appendix A - Figures

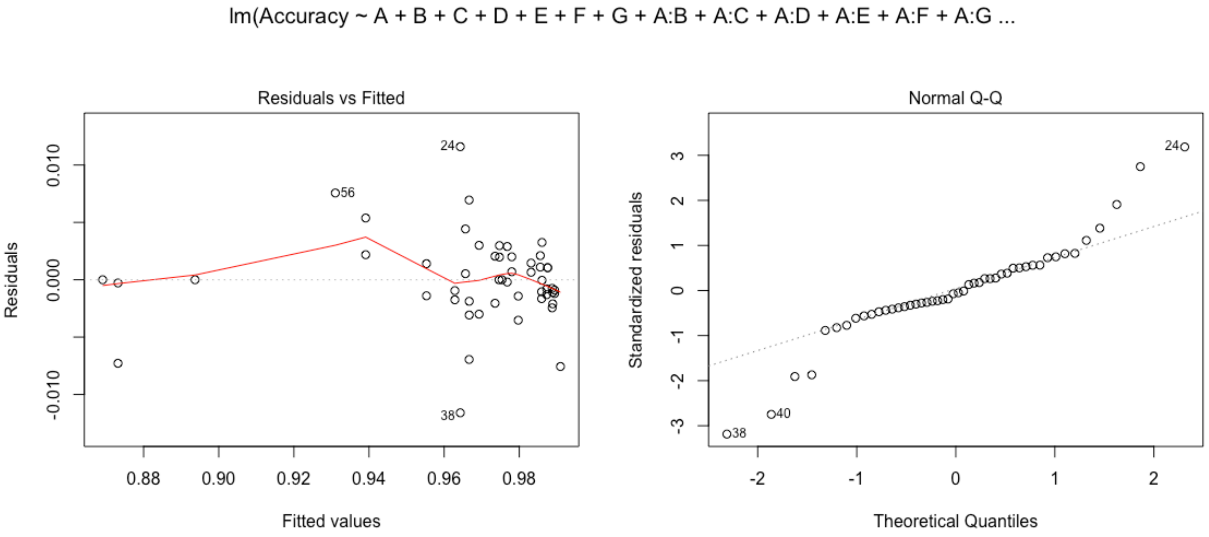


Figure 12: Residual Plots - StepAIC model, Outliers Removed (Initial Experiment)

Appendix A - Figures

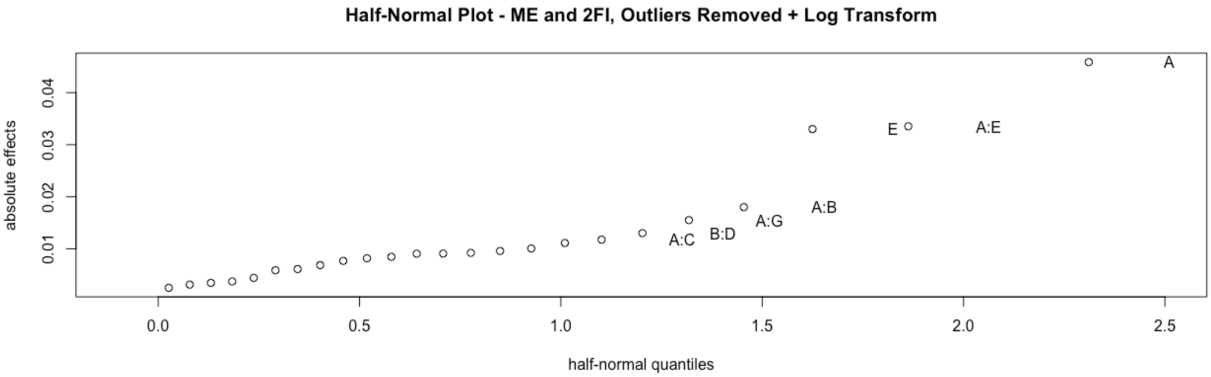


Figure 13: *Half-Normal Plot – Main Effects and 2FI with log transformation, Outliers Removed (Initial Experiment)*

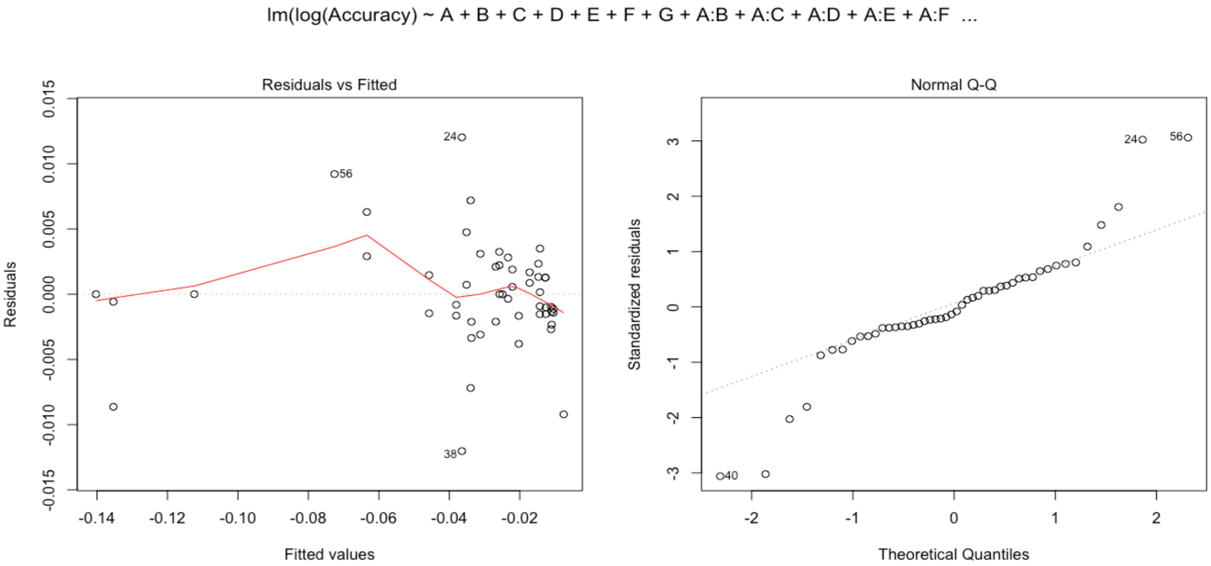


Figure 14: *Residual Plot - StepAIC model with log transformation, Outliers Removed (Initial Experiment)*

Appendix A - Figures

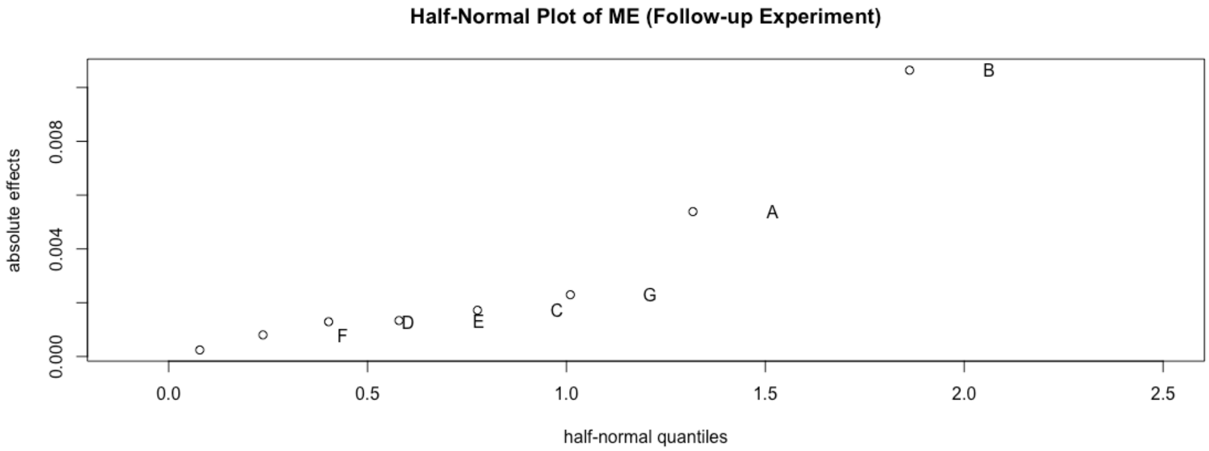
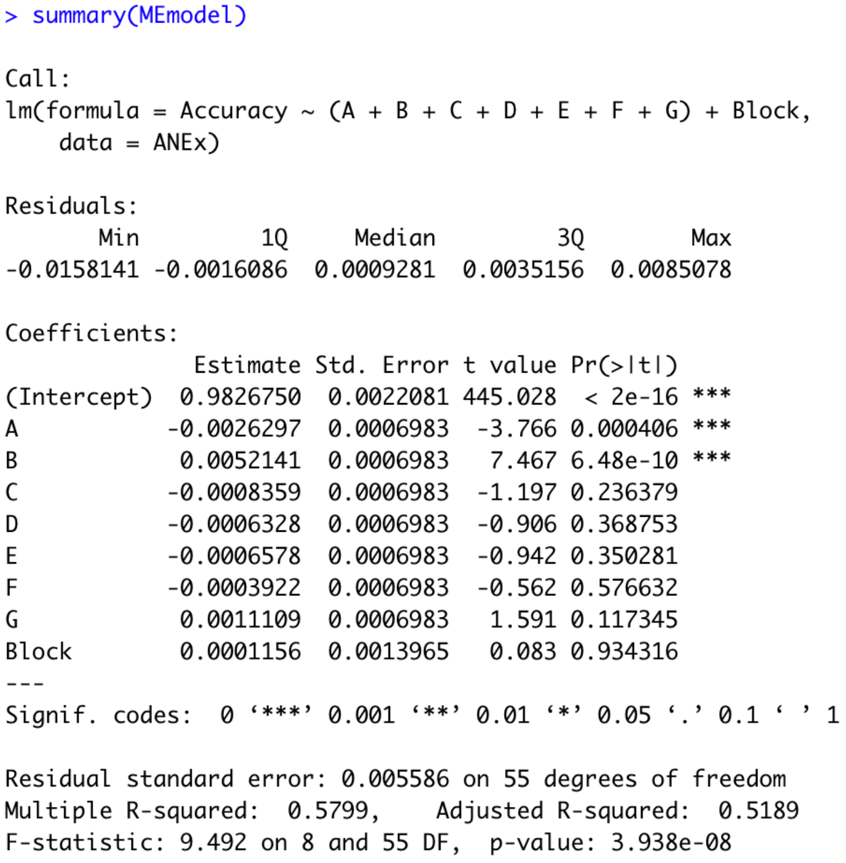


Figure 15: Half-Normal Plot – Main Effects (Follow-Up Experiment)



```
> summary(MEmodel)

Call:
lm(formula = Accuracy ~ (A + B + C + D + E + F + G) + Block,
    data = ANEx)

Residuals:
       Min         1Q     Median         3Q        Max 
-0.0158141 -0.0016086  0.0009281  0.0035156  0.0085078 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.9826750  0.0022081 445.028  < 2e-16 ***
A           -0.0026297  0.0006983  -3.766 0.000406 ***
B            0.0052141  0.0006983   7.467 6.48e-10 ***
C           -0.0008359  0.0006983  -1.197 0.236379    
D           -0.0006328  0.0006983  -0.906 0.368753    
E           -0.0006578  0.0006983  -0.942 0.350281    
F           -0.0003922  0.0006983  -0.562 0.576632    
G            0.0011109  0.0006983   1.591 0.117345    
Block        0.0001156  0.0013965   0.083 0.934316    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.005586 on 55 degrees of freedom
Multiple R-squared:  0.5799,	Adjusted R-squared:  0.5189 
F-statistic: 9.492 on 8 and 55 DF,  p-value: 3.938e-08
```

Figure 16: R Output - Main Effects Model (Follow-Up Experiment)

Appendix A - Figures

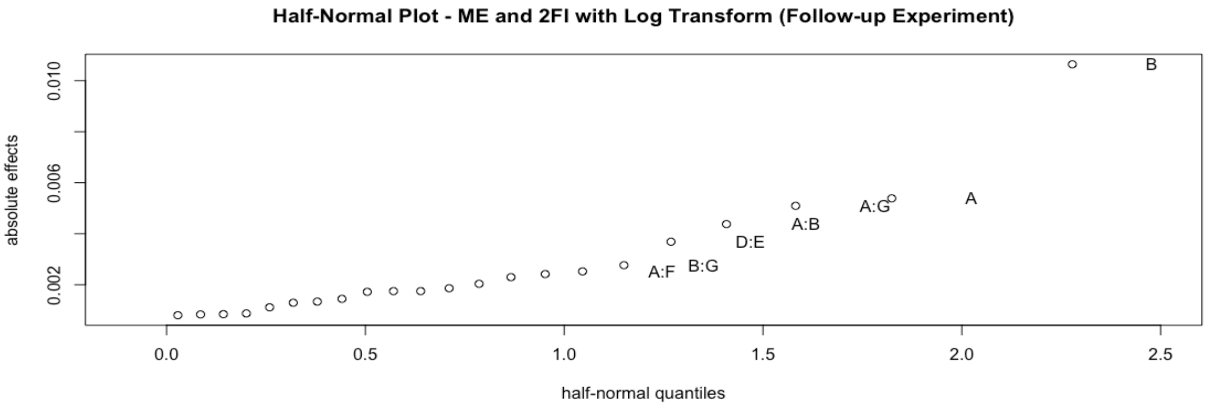
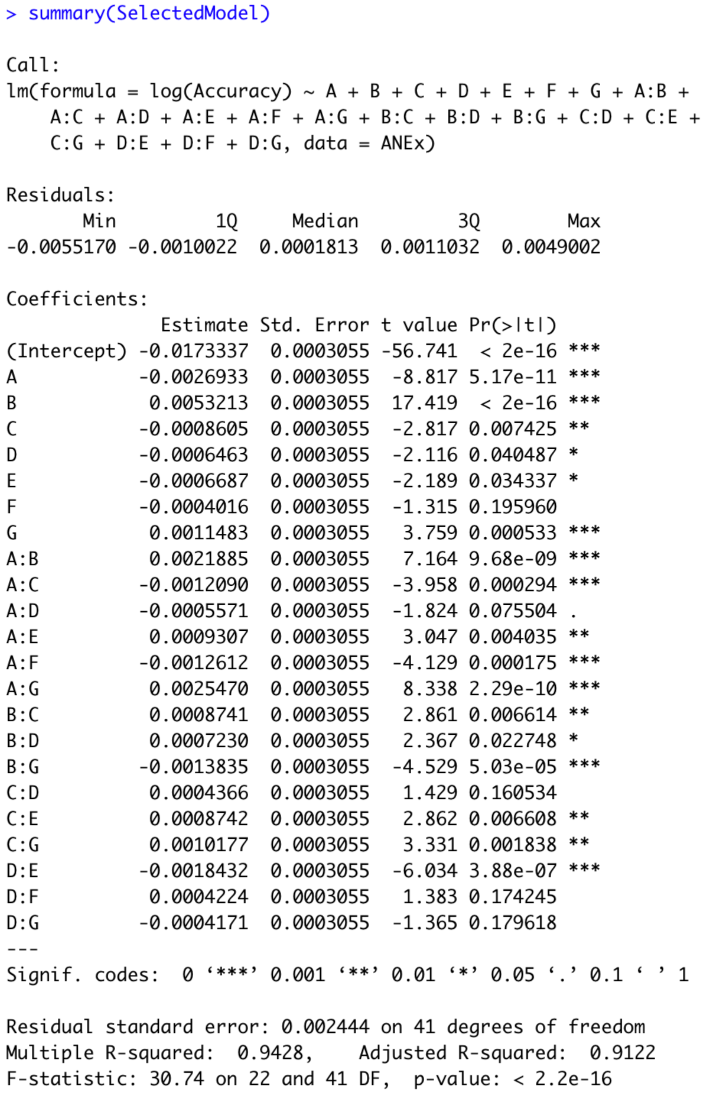


Figure 17: *Half-Normal Plot - Main Effects and Two-Factor Interactions with log transformation (Follow-Up Experiment)*



```
> summary(SelectedModel)

Call:
lm(formula = log(Accuracy) ~ A + B + C + D + E + F + G + A:B +
    A:C + A:D + A:E + A:F + A:G + B:C + B:D + B:G + C:D + C:E +
    C:G + D:E + D:F + D:G, data = ANEx)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0055170 -0.0010022  0.0001813  0.0011032  0.0049002

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.0173337  0.0003055 -56.741  < 2e-16 ***
A           -0.0026933  0.0003055  -8.817 5.17e-11 ***
B            0.0053213  0.0003055  17.419  < 2e-16 ***
C           -0.0008605  0.0003055  -2.817 0.007425 **
D           -0.0006463  0.0003055  -2.116 0.040487 *
E           -0.0006687  0.0003055  -2.189 0.034337 *
F           -0.0004016  0.0003055  -1.315 0.195960
G            0.0011483  0.0003055   3.759 0.000533 ***
A:B          0.0021885  0.0003055   7.164 9.68e-09 ***
A:C         -0.0012090  0.0003055  -3.958 0.000294 ***
A:D         -0.0005571  0.0003055  -1.824 0.075504 .
A:E          0.0009307  0.0003055   3.047 0.004035 **
A:F         -0.0012612  0.0003055  -4.129 0.000175 ***
A:G          0.0025470  0.0003055   8.338 2.29e-10 ***
B:C          0.0008741  0.0003055   2.861 0.006614 **
B:D          0.0007230  0.0003055   2.367 0.022748 *
B:G         -0.0013835  0.0003055  -4.529 5.03e-05 ***
C:D          0.0004366  0.0003055   1.429 0.160534
C:E          0.0008742  0.0003055   2.862 0.006608 **
C:G          0.0010177  0.0003055   3.331 0.001838 **
D:E         -0.0018432  0.0003055  -6.034 3.88e-07 ***
D:F          0.0004224  0.0003055   1.383 0.174245
D:G         -0.0004171  0.0003055  -1.365 0.179618
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.002444 on 41 degrees of freedom
Multiple R-squared:  0.9428,    Adjusted R-squared:  0.9122
F-statistic: 30.74 on 22 and 41 DF,  p-value: < 2.2e-16
```

Figure 18: *StepAIC Model with Main Effects and 2fi and log transformation (Follow-Up Experiment)*

Appendix A - Figures

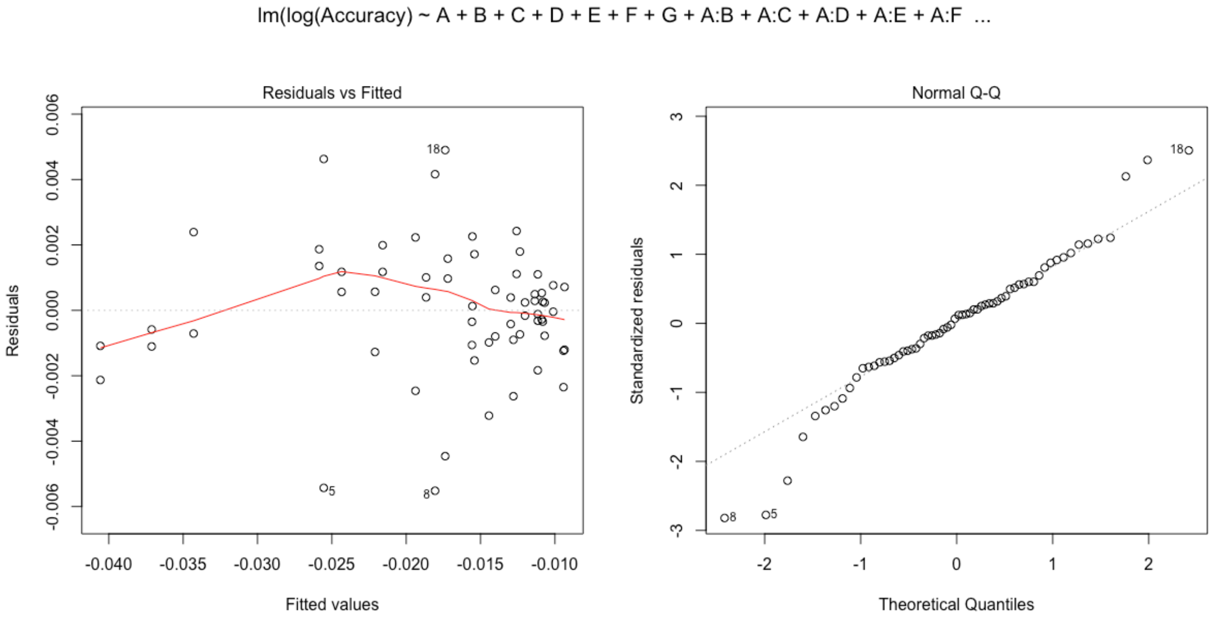


__Figure 19:__ Residual Plot - StepAIC "Final" Model (Follow-Up Experiment)