
# Returned Order Prediction Report

**March 9th, 2020**

**By Team 6:**

**Alyson Chen, Sumner Crosby, Fairy Gandhi,**

**Alan Market, Wenbin Yang**

# Table of Contents

# 1. Summary

## A. Product Category & Plant:

For this report, data from Hunter Douglas was examined to attempt predicting which orders would be returned within 90 days of an order being fulfilled. More specifically, only Honeycomb Shades produced in Plant B were used in this study.

## B. Model metrics

A. To evaluate our models, we used sensitivity, specificity, precision, and accuracy to measure each model's ability to make accurate and economically viable predictions on our data set. After examining all of the models, one model stood out due to its diagnostic metrics. The Weighted KNN model shown in Section 6 exhibits the following metrics, (table below). The higher precision and accuracy values make this the best-fit model.

- Sensitivity: 15.2%
- Specificity: 98.8%
- Precision: 24.5%
- Accuracy: 96.8%

## C. Economic impact summary TCO, Benefit, NPV, Time to Payback, ROI

TCO: For this model, the initial cost of development is $2,450, and the annual cost of maintenance is $246,876 including inspections. By the fifth year, the accumulated TCO is 1,224,580.

Benefit: This model can reduce the cost of unit repairs by $475,200 within 5 year, and reduce the cost of remade orders by $1,083,000. Total 5-year benefit is 1,558,200.

NPV: The NPV for a five year horizon is $119,702.

Time to payback: 0.04 years to payback investment.

ROI: The return on investment for implementing this model is 2743%

## D. Recommendations

B. From the predictive analysis, the best-fit model was found to be the model highlighted in Section 6, the Weighted KNN model, and this group will recommend that Hunter Douglas implement this model given its economic viability and comparatively robust predictive power. This model exhibited the best model diagnostics, with about 20% precision. As noted above, the ROI was high and time to payback low, indicating this model could prove useful in catching orders that may have issues or eventually be returned while saving the Company money. Given the long run-time for the model, it was difficult to tune the parameters, but with more time and resources the model's predictive power could be further increased.

# 2. Data

## A. Summary

The data used in this report was provided by Hunter Douglas, which outlined all orders made in the calendar 2018 year. Prior to any data cleaning or manipulation, the dataset includes 3,159,252 data points across 25 columns. The table below shows the column fields included in the original dataset. As mentioned in the above Section, the specific production plant and product line used for this report will be Honeycomb shades from production plant B.

| | | | |
|---|---|---|---|
| ORIGINAL_ORDER | ORIGINAL_ORDER_LINE | SALES_ORDER | SALES_ORDER_LINE |
| HEIGHT | PRODUCT_CATEGORY | ORIGINAL_PLANT | ALLIANCE_LEVEL_ID |
| ORDER_REASON_ID | REGION_STATE_ID | NET_SALES_UNITS | REMAKE_UNITS |
| RESPONSIBILITY_CODE_ID | FABRIC_ID | COLOR_ID | SLAT_SIZE |
| SO_CREATED_DATE | ORIGINAL_MATERIAL_ID | OPERATING_SYSTEM_ID | OPERATING_SYS_OPT_ID |
| REGIONAL_SALES_MGR_ID | REASON_CODE_ID | REASON_CODE | SOLD_TO_ID |
| WIDTH | | | |

## B. Cleaning approach

As with most modeling applications, data preparation was a crucial part of forming the following models. In the data initialization code, which loads the data for further analysis by performing some primary cleaning, such as simple character vector restructuring and character to numeric transformations. Given the large size of the dataset, there were some outliers, especially for the height and width categories, so the data was trimmed to ensure greater normality of distributions. Data values for unused production plants, productlines, and other columns were dropped, leaving about 750,000 rows in the dataset. The final step of the initialization process is to remove certain NA values.

C. In addition to the data cleaning tasks outlined above, the biggest data preparation challenge was to correctly flag orders that were returned. The first step was to combine the order and order line numbers into a single index. Next, the original and sales indexes were compared and orders that were not original orders were considered to be returned orders. Returns that occur after 90 days were assumed to be for reasons other than defects or order issues, so for the purpose of this report, they were not considered to be returns. The final step of return flagging was to create binary control columns for if each row was a original order that was returned, a return order, and whether or not the return was a remake or remodel. In addition to some model specific data cleaning, each model will subset the dataset to include only original orders that were returned or return orders to avoid double counting the number of real returns.

# 3. Model I: Decision Tree

## A. Description

Decision tree can deal with both regression and classification problems. In our dataset, we're trying to predict the outcome-FlagReturnOrig which is a binary variable. We think by using trees could help us to precisely predict our model. Moreover, there are several reasons for us to go with Tree. First of all, Trees can deal with large and complicated dataset, In Hunter Douglas dataset which contains 730100 rows and 25 columns. In this case, a tree would be the efficient algorithm. Second, Compared to other algorithms, trees require less effort for data preprocessing. Moreover, trees don't require scale data or normalization data. In this way, our process in generating data cleaning would be more efficient. Especially dealing with such large dataset like Hunter Doulglas.

## B. Parameters

After we finished our data exploration and data preprocessing, we chose different variables for each model to see what would be the factor that will influence the return rate. In the tree model, we chose five different variables such as " responsibility ID", "orderReasonID" , "remakeUnits" ,"height" and "fabricID". We excluded the variables which have large levels and only include the smaller level of variables.

## C. Confusion matrix

The table below is the confusion matrix for the tree model. Three of them accuracy rates are

between 96% to 97%. However, the precision rates aren't looking good. To sum up, the tree model performances aren't good as we expected.

***Table 1: Remake Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 659 | 1894 |
| Observed No Return | 7959 | 295454 |

Precision: 7.6%
Accuracy: 96.8%

***Table 2: Remodel Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 552 | 1743 |
| Observed No Return | 6530 | 293990 |

Precision: 7.8%
Accuracy: 96.8%

***Table 3: All Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 659 | 2044 |
| Observed No Return | 5659 | 289940 |

Precision: 10.4%
Accuracy: 97.4%

## D. Economic impact model

The annual ROI of the model is around -141%. As we can tell, the decision tree model doesn't work for the strategies from Hunter Douglas. Therefore, we'll not recommend the company to make this model to be their target model.

## E. Data prep

There are several preparation steps before building the tree model. First of all, I created a dataframe to only include the useful variables from the original dataset. The criteria for me to choose these variables to add into the new dataset is based on the level of the variable. In the new dataset, it only contains lower level columns. Second, I wrote a for loop to make all the character types change into factors. Next, I split the dataset into training and testing sets and their percentage is 60% vs 20%. Last but not least, I used the Tree package called “rpart” in R to build a tree model.

## F. Learning

In this project, it has taught several things toward data cleaning and prediction model build. First of all, by manipulating real world data, it was a handy experience for us before landing our first job after graduation. However, the process of data cleansing wasn’t a smooth ride, it took us a couple of weeks to clean up the data before we move onto the next steps. Second thing I’ve learned is to decide the most useful variables among these 25 columns and change them into dummy variables. Moreover, the process of building a tree isn't as easy as I expected. In the end, I overcame these challenges by looking at many researches and articles. Most important of all, I learned by trial and error in the process of building my model.

# 4. Model II: Generalized Boosted Regression

## D. Description

Model 2 implements a Generalized Boosted Regression Model (GBM). GBMs are a popular machine learning algorithm with proven success across many domains and are a good alternative to generalized linear models when predictieve power is low. GBM models are similar to a random forest regression, with the main difference being that where a random forest builds a group of deep independent trees, GBMs build a system of shallow and weak successive trees with each tree learning and improving on the previous. When combined, this group of successive trees produce a powerful "committee" that is often hard to beat with other algorithms. After testing some preliminary logistic generalized linear models that exhibited low predictive power, the GBM model was explored to create a more useful predictive model. A number of modeling parameters were tested, with the below model parameters creating the most robust model.

## E. Parameters

After testing a number of GLM and GBM models, five input variables were used for the final model, which are as follows: height, width, fabricIDD, colorBin, and operatingSysID. Height and width indicate blind size and general dimensions, which were shown to play a significant role in returned order trends. FabricIDD is a statistically significant control for blind fabric type and colorBin is a binned variable to control for the color designation of the blind. The formation of both variables is described in further detail in Part E - Data Preparation. OperatingSysID is the operating system used in each order, which surprisingly increases the predictive robustness of the model. The GBM model takes a number of inputs, which are highlighted in the below table.

| Parameter | Value |
|---|---|
| Distribution | Bernoulli |
| n.trees | 1000 |
| interaction.depth | 1 |
| shrinkagE | 0.01 |
| keep.data | TRUE |

## F. Confusion Matrix

The following tables, (next page), show the results of the predictive modeling, as constructed in Parts A and B. Tables 1 and 2 show the predictive result when subsetting only for remade and remodel returned orders. While the predictive accuracy was high, this may be misleading as the number of returns as a percentage of the entire dataset is quite small, (about 2.7%) so even a low number of true positives will force a high accuracy.

The low precision value indicates the lack of positive predictive power in the model, however it may still be useful in this business application. Interestingly, predicting remake vs. remodel separately seems to have less predictive power as using all data points, which is highlighted in Table 3 by a higher precision output.

***Table 1: Remake Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 1,613 | 6,347 |
| Observed No Return | 27,465 | 299,520 |

Precision: 5.5%
Accuracy: 89.9%

***Table 2: Remodel Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 417 | 452 |
| Observed No Return | 32,915 | 293,990 |

Precision: 1.3%
Accuracy: 89.8%

***Table 3: All Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 1,716 | 7,187 |
| Observed No Return | 26,428 | 300,468 |

Precision: 6.1%
Accuracy: 90.0%

## G. Economic Impact Model

After performing some economic analysis on the financial viability and usefulness of this model, it does not seem that Hunter Douglas should implement this model. As shown in the above tables, the GBM model used for the predictions was somewhat successful, being able to accurately predict about 90% of outcomes, yet the precision is low, creating a large number of false positives. This severely affects the ability to use this model at Hunter Douglas, as the calculated ROI rate is incredibly negative. Despite being able to predict some returns, which would allow Hunter Douglas to find some problem orders prior to shipping, the cost of inspecting a little over 27,000 orders per year would cost a lot more than it would save. For this reason, it cannot be recommended that Hunter Douglas implement this model in Factory B for Honeycomb Shades.

## H. Data Preparation

The main data cleaning and preparation is described in the initialization steps discussed in Section 2. For the generalized boosted regression, two columns were binned for easier analysis: fabric type and color type. These two variables were indicated to have significant differences in manufacturing, so they were expected to play a role in determining if an order was returned or not. Fabric type, (FabricIDD) was binned by removing fabric types under 1000 orders and combining all fabric types under 2000 orders. Color type, (colorBin) was binned by the percentage of orders that were returned into 5 risk categories: low, low-mid, mid, mid-high, and high. A number of bin thresholds were tested, but a custom threshold that uses varying sample sizes and return rate ranges was implemented for this model given its higher predictive power when tested.

## I. Learning

This process has taught me a lot about real-world data cleaning, manipulation, modeling, and predictive analysis. The data cleaning and manipulation prior to modeling was difficult, but nothing out of the ordinary. I learned that really examining the data helps inform cleaning decisions and can make the rest of the modeling process easier. After data cleaning, the modeling aspects of this project were pretty difficult and I learned that data manipulation is more important to modeling than I expected. Throughout the modeling process, I constantly had to examine the data and determine how the data could be further manipulated in an appropriate way to create the best model. I learned a lot about regression modeling large datasets and the architecture of both LOGIT and generalized boosted modeling. As expected, a large part of this project was predictive analysis, and constructing a good predictive model proved to be a challenge. Lastly, this project taught me about time management and efficient coding. Some of the modeling and cleaning was very time intensive and I had to create hundreds of lines of code, so code organization and structure became very important.

# 5. Model III: Neural Network Model

## A. Description

In the Neural network model, the basic idea is to combine input information in a flexible model, which is called the neural net. Neural network is represented by nodes and arcs. The nodes can be classified into three layers. The output of one layer becomes the input of the next layer. The input layer consists of nodes that accept input variables. The last layer is called the output layer. The layers between input and output layers are called hidden layers. The arcs are directed so that the flow in the network is one-way from left to right. A neural network model attempts to capture more complex relationships. So how does the neural network work exactly? First, it begins by setting the value of the coefficients (or the weights) randomly. An iterative procedure is then used to modify the initial value guided by the prediction made in the existing network. In each step, it uses the existing neural network to make predictions for each observation. It then distributes error from the output node to the hidden layers and modifies weights by small steps in the direction of reducing errors.

## B. Parameters

Initially, the parameters I used for the model were only 'height' and 'width' as my input parameters, with 'flagReturnOrig' as my output variable. The number of hidden layers selected was 7. Keeping a linear output in the final layer of a neural network is usually used in regression settings only; in classification settings, we have to apply the activation function to the output neuron(s) as well. Hence, we set linear.output = 'F' as a parameter too.

The final model had 'fabricbin', 'colorbin', 'height', and 'width' as parameters. FabricID and ColorID columns were binned, and the top 10% was considered for modeling. The number of hidden layers selected was 10.

## Confusion matrix

Confusion Matrix is a performance measurement for the classification problem where output can be two or more classes. It is a table with four different combinations of predicted and actual values. It is beneficial for measuring Precision, Specificity, Accuracy, etc.

The table below shows that the precision rate is comparatively low which is creating a large number of false positives.

***Table 1: All Returns***

| Input/Output | Positive | Negative |
|---|---|---|
| Positive | 216 | 19,362 |
| Negative | 7454 | 702,379 |

Precision: 2.8%
Accuracy: 96.3%

Sensitivity: 1.1%

### C. Economic impact model

Economic Impact Model for Hunter Douglas consists of Confusion Matrix, detailed Financial Analysis, Financial Assumptions, Benefits ( Total Five-year benefits, etc.), Costs. Economic impact analysis provides a quantitative method to estimate the economic benefits that a particular project or industry brings to the economies. In the Hunter Douglas economic Impact model, Analytics expenses Inspection cost is calculated. Based on the calculations and analysis, one can conclude that the model is not an excellent fit for Hunter Douglas. In other words, it's recommended not to implement the model.

### D. Data prep

For the hunter Douglas data set, before implementing the first phase of the Neural network model, I split the dataset into 60:40 ratio. 60% for the trained data and 40% for the test data. Training the neural network with sample data is always recommended, since, full training with all data can take a considerable amount of time. Then installed the neural net library in R. Because this is a classification problem, I set the err.fct = 'ce,' which stands for cross-entropy, and set linear.output=FALSE in the neural net function.

### E. Learning

The learnings from the neural network model is that it's always good to use multiple starting values to build the network. If we implement bagging and boosting to the neural network model it can reduce the error to a lot of extent. The smaller the validation error, the fitter is the model. I also learned that the higher number of nodes in the hidden layer captures the complexity, but it also increases the chances of overfitting.

## 6. Model IV: K K-nearest Neighbors (KKNN)

### A. Description

J. One of the models we chose is the KKNN model. KKNN is a Supervised Machine Learning algorithm that classifies a new data point into the target class, depending on the features of its neighboring data points. The reason we selected the KKNN model is that we already had removed some irrelevant columns from the dataset, but we still can not decide the variables to be included as predictors. The KNN model isn't required to choose predictors.

### B. Parameters

After testing several times of the parameters for the KNN model, I decided to use 'fabricBin', 'colorBin' and the height and width as the parameters. And also, add an interaction term between height and width, which can improve the accuracy, because the size often determines whether this order is suitable or not. It is the main factor for the returned order.

## C. Confusion matrix

***Table 1: Remake Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 1,083 | 5,683 |
| Observed No Return | 4,362 | 247,213 |

Precision: 19.9%
Accuracy: 95.3%

***Table 2: Remodel Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 792 | 4,774 |
| Observed No Return | 1,430 | 250,345 |

Precision: 35.6%
Accuracy: 97.6%

***Table 3: All Returns***

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 1,875 | 10,457 |
| Observed No Return | 5,792 | 497,558 |

Precision: 24.5%
Accuracy: 96.8%

## D. Economic impact model

The annual ROI of the model is around 2,743%. So, we can say the KKNN model works well for the strategies from Hunter Douglas. Therefore, we'll recommend the company to make this model to be their target model.

## E. Data prep

To build the KNN model, we first needed to determine the K value. What is K value? K is the neighbor we considered for the model. The different K we choose, the different accuracy we get. So, the first step is using train function to find the K value. There are more than 0.7 million rows in our dataset, so we tried to make Length equals to 100, 200 and so on. Finally, we chose 500 as Length, Which can get the highest accuracy, and K equals 867. And then, because the KKNN model doesn't allow any null values, the next step is to use na.omit function to eliminate all NA, then convert all factor variables to numeric to make sure every variable can be included in KKNN model. Training and testing were divided into 50% respectively. Finally, the model looks good and works as well as expected. It can be seen from the figure that all the points have an ordered distribution. Below 800, as the number of points increases, the accuracy of the prediction model gradually increases, and at about 850, the accuracy of the model tends to 96%.

## F. Learning

I learned a lot from this project, such as how to deal with an unstructured dataset, how to clean it, how to pick the useful information from it and how to do a project with team members. And also this course project allowed me to work more hands on with different predictive models. I tried several models for this, and finally, I chose the KKNN model, because it works perfect and with the highest accuracy.

# 7. Model V: Random Forest

## A. Description

To test the efficacy of a random forest classifier for predicting failures of Hunter Douglas orders, we built and evaluated several different variants of random forest and finally settled on using a generalized boosted random forest model for prediction. The boosted tree regression performed marginally better than the traditional random forest on the same set of parameters, but the performance of both models was underwhelming. The initially boosted tree model was trained on 10000 trees, and then pruned back to 2987 for predictions as recommended by the package specific pruning function. The distribution specified to the model was a Bernoulli distribution as our target variable was binary, and the model itself was trained on 40% of the data. The internal model training set/test set split was that the model was trained on 80% of the given data set. The interaction depth of the model which dictates the maximum number of nodes per tree in the random forest was set to 5 after testing of different levels. We kept the learning rate of the boosted tree model at the standard rate of .001 to avoid large changes that could disrupt the model's accuracy. The final cutoff for the predictor cutoff was set at .14 for the full data set, and .121 for the training set, and these values were chosen to optimize the ratio of true positives to false positives in the predictions.

## B. Parameters

The left-hand side target variable for the model was a dummy variable where 1 indicated that an order was a failure per Hunter Douglas' definition of what qualifies as a failure. The right-hand side predictor variables were: Order state, width, height, fabric decile ranks, color decile ranks, material ID decile ranks, vendor alliance ID, and operating system ID indicator.

## C. Confusion matrix

*Test Set Confusion Matrix:*

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 211 | 7506 |
| Observed No Return | 624 | 291,607 |

*Main Confusion Matrix: All Returns*

| Input/Output | Predicted Return | Predicted No Return |
|---|---|---|
| Observed Return | 235 | 19,434 |
| Observed No Return | 857 | 729,270 |

Sensitivity: 1.2%

Specificity: 99.9%

Precision: 21.5%

Accuracy: 97.3%

### D. Economic impact model

The economic impact of employing this tree model would be very negative. The high upfront cost of development as well as the very low sensitivity means that the model will not catch enough failures to be profitable. The annual ROI of this model is -626%. Given the poor performance of this model, we cannot recommend that Hunter Douglas employ this model. With that said, they can use this model to see what strategies do not work for forecasting product failures of honeycomb blinds. This could help to focus their efforts for subsequent model development.

### E. Data prep

The model specific data preparation for the boosted model mainly revolved around reducing the number of factors within the fabric, color, material, and state variables. The model only accommodates factor variables with under 53 levels, so for the state variable which included more than this, I found the top 49 states in terms of failures, and then binned the remaining state factors into a catchall category to bring the total to 50. For the rest of the variables that required binning, I calculated the failure rate in percentages, and then used these percentages to bin all the color, fabric, and material IDs into deciles by failure ranks. The only other changes that need to be made to the data was to convert the failure indicator variable from logical to numeric in order to have it function as a true dummy variable.

### F. Learning

While building and testing different versions of random forest models, I did find that some performed better than others. The traditional random forest model found in the randomForest package in R did not perform as well as the boosted tree model. The process of boosting is typically better for detecting anomalies in unbalanced datasets which better describes the situation with the Hunter Douglas Data as there are relatively few order failures relative to the number of successes. The biggest challenge with building the boosted tree model was tuning the parameters of the model to try and strike a balance between overfitting the data and being able to catch a meaningful number of the failures. In the end, I do not think that I was able to truly find a good balance as my best performing model found a much smaller proportion of failures in the test set than it did in the training set.

We also experimented with creating a cross-validated random forest using an AWS EC2 platform to expedite the process of training the model. Even running on AWS with 32 CPU cores and 128GB of ram, the training time for the models on 40% of the data as a training set took around 10 hours. I ran three different models, but between debugging and blatant overfitting I was not able to build a usable cross-validated random forest model with the time I had left. With that said it was a great learning experience to go through the process of setting up an RStudio instance on AWS.

## 8. CRR analysis

**Top Five Order Reasons for CRR orders:**

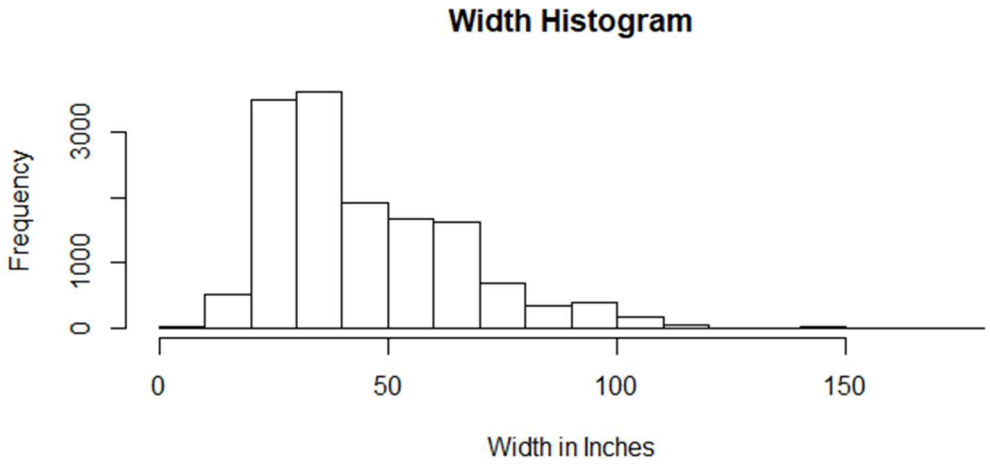
I. Width too Wide
II. Product/Model Incorrect
III. Width too Narrow
IV. Fabric/Slat Style Wrong
V. Dealer/Decorator Error

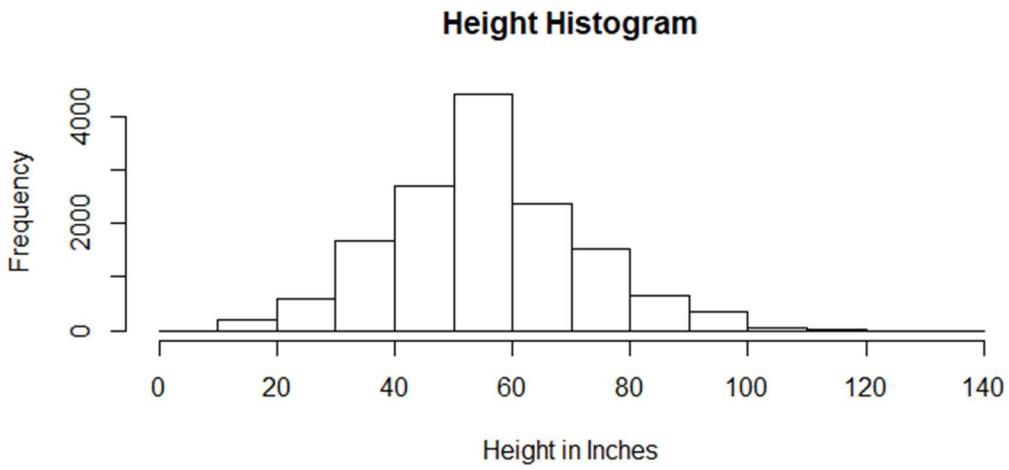
**Top Three CRR Order Return States:**

I. Arizona
II. California
III. Texas

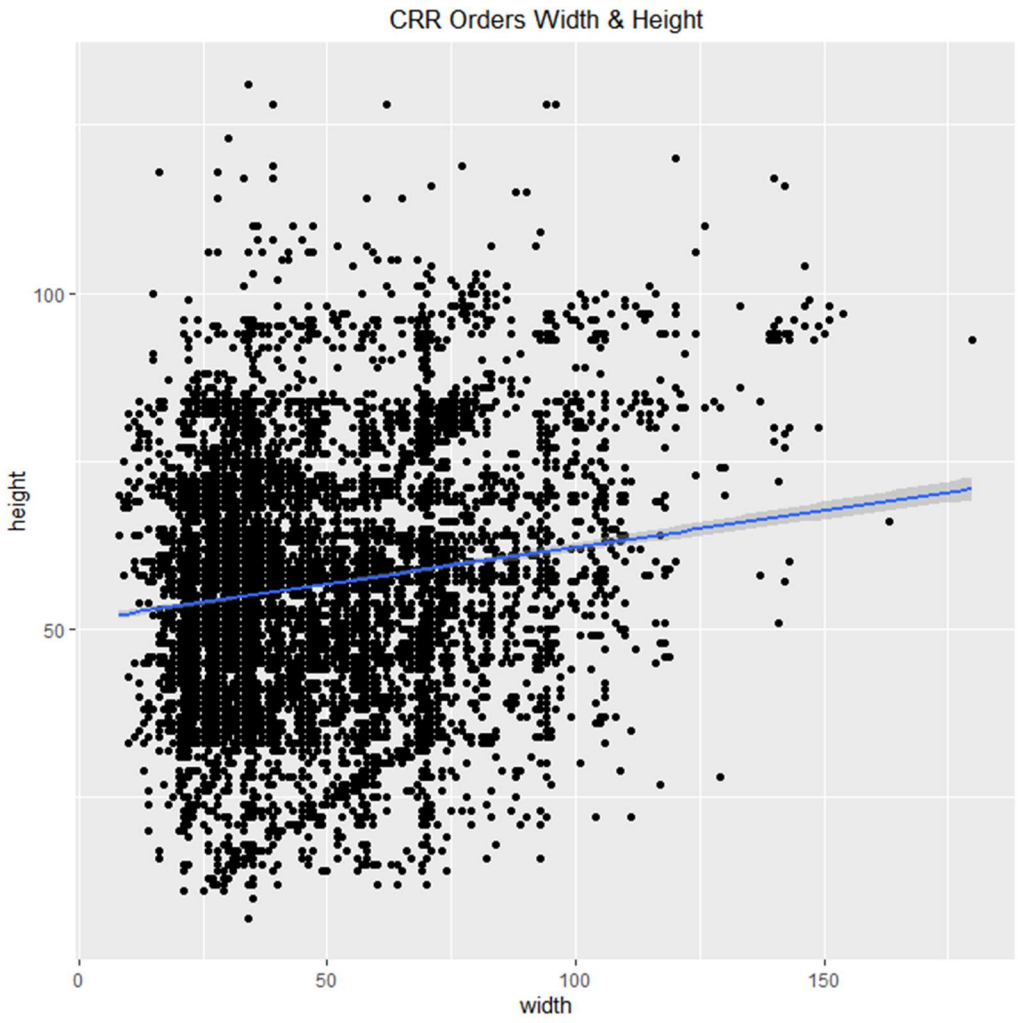
**Width & Height summary Statistics:**

I. Width Mean: 45.76”
II. Width Median: 38”
III. Height Mean: 56.29”
IV. Height Median: 58”




When focusing on Customer Requested Remake orders, there is a clear trend that width plays the largest role. The first and third reasons by count of CRR order returns are the width being too wide or narrow. Furthermore, the distribution of CRR order widths is right skewed when compared with the relatively normal distribution of CRR order heights. Hunter Douglas could work to mitigate CRR orders by working with their prefered dealers to inform installers on double checking order dimensions before submitting them to Hunter Douglas. Hunter Douglas could use Arizona, California, and Texas as test markets for the strategy as these states experience far more CRR orders than the rest of the country.


CRR Orders Width & Height
height
100
50
0
50
100
150
width


The two plots above illustrate that very different trends dimensions exist across material fail rank bins. This could help to further inform which orders need to be double checked. Hunter Douglas could see If the order has an extreme width or height and belongs to a material bin with a high failure percentage.

Unfortunately for Hunter Douglas, the majority of CRR orders seem to be the result of human error. While they can try and take steps to add in extra checkpoints to catch errors before a product is manufactured, they will not be able to eliminate human error entirely.