# Predicting the Price of Diamonds using a Random Forest Model

Rory Murphy

STU44003

16317570

1. Introduction

The purpose of this report is to develop a model using the Random Forest ensemble technique to predict the price of diamonds. This model was developed using a dataset containing 53,940 cases of diamonds with ten variables measured on each diamond (described below). This dataset was assessed and adjusted before being used to build the model and this report describes this initial analysis of the data, the model constructed and an evaluation of the model’s effectiveness. After this, the model is interpreted to better understand the process by which the model is determining diamond prices. Finally, the Extremely Random Trees ensemble technique is examined in comparison to the Random Forest technique, with an Extremely Random Trees model being built for the same dataset.

2. Variables

Below is a list of the variables recorded on each diamond in the dataset used:

Price: The price of the observed diamonds, this variable is the outcome variable of the dataset, or the variable that the model will attempt to predict using the other variables.

Carat: The carat weight of a diamond.

Cut: An ordinal variable describing the quality of the diamond cut. The scale used in increasing order was Fair, Good, Very Good, Premium, Ideal.

Colour: The colour of a diamond. Recorded as a letter from D to J, with D being the best colour and J being the worst.

Clarity: An ordered ranking of how obvious inclusions are within the diamonds. The rankings ordered from best to worst are FL, IF, VVS1, VVS2, VS1, VS2, SI1, SI2, I1, I2, I3.

Depth %: The height of the diamond, measured form the culet to the table, divided by its average girdle diameter (given as a percentage).

Table %: The width of the diamond’s table as a percentage of its average diameter.

Length: The length of the diamond, measured in millimetres.

Width: The width of the diamond, measured in millimetres.

Depth: The depth of the diamond, measured in millimetres.

3. Cleaning of the Data

Before the model was developed, the data received was cleaned. This was done to prepare the dataset for analysis, removing any inaccurate or incorrect data which would otherwise reduce the effectiveness of the model being built.

3.1 Duplicate Entries

First, the data was examined to ensure no duplicate entries were observed. Each entry should come from a different diamond. Should the model be built with many duplicate entries, the model would be skewed towards predicting these types of diamonds. In other words, the model would believe diamonds of this type to be more common than they are and would be bias towards predicting the prices of these diamonds at the cost of effectively predicting the prices of other diamonds. No duplicate entries were found in the dataset, so this was not a concern.

3.2 Missing Data

Cases and variables with missing data should be examined, as they can cause problems when the model is being built. Furthermore, if a variable or case has a high percentage of missing values, this may give us information about the nature of our dataset. For example, if the depth variable was missing for many diamonds, this may indicate that it is expensive or difficult to record this measurement. At first, it appeared that there was no missing data, as no variables were marked as missing or empty. However, upon examining the length, width and depth measures, it was found that 7 cases had entries of 0 for length, width and depth, another case had 0 for length and depth and 12 other cases had entries of 0 for depth.

Since it does not seem possible for a diamond to have no length, width or depth, all 20 of these entries were removed, as it is likely that they were dirty data, potentially accidental entries into whatever system was used to collect the data. Due to the low volume of irregularities in length, width and depth occurring in the data relative to the size of the dataset, it is unlikely that there is an underlying issue with recording the variable itself, so no further action was deemed necessary. Ideally, the data providers would be consulted on this result. However, they were not available during the undertaking of this project.

3.3 Near Zero Variance

If a variable has a high percentage of identical values in the dataset, consideration towards removing the variable should be made. Such variables give relatively little information to the model, only adding to the model’s complexity. Variables where the ratio of the most frequently occurring value to the second most frequently occurring value was 95:5 or higher and variables where the percentage of unique values was below 10% of the total number of cases were considered to have near zero variance. In other words, such variables were considered too similar between diamonds to provide enough useful information about an individual diamond’s price. These figures are based on the default values given by the “nearZeroVar” function in the “Caret” package in the programming language “R”. Since no variables fell into this category, all were sufficiently varied to be considered in this model. The frequency ratio and proportion of unique values is available in appendix 1.

3.4 Outliers

Outliers can reduce the effectiveness of a model by distorting results. Furthermore, Regression Trees (which the Random Forest for predicting diamond prices is made up of) can be heavily influenced by outliers, distorting the results produced. Because of this, the cases containing one or more outlier were removed from the dataset on these grounds.

Cases with an outlier in one or more quantitative variables were removed. Each quantitative variable (excluding price, the outcome variable) was assessed for outliers, with any values greater than the upper quartile + two times the interquartile range and any values below the lower quartile – two times the interquartile range being considered outliers. This technique is commonly used to classify cases as outliers, however 1.5 times the interquartile range is more often used as the range for classifying observations as outliers. When using 1.5 as the coefficient on the interquartile range, 5022 cases were classified as containing one or more outlier, accounting for 9.31% of the data. This was considered too large an amount of the data to treat as outliers, so a coefficient of two was used instead.

With this method, 1490 cases containing one or more outliers were found, accounting for 2.76% of the data (having already removed the 20 dirty cases in section 3.2). Ideally, an expert would be consulted as to the nature of these outliers, especially for the two variables with the highest number of outliers (depth % which contained 1130 outliers and carat weight which had 250). However, such a consultation was outside the scope of this report. Instead, these 1490 cases were excluded from the dataset.

3.5 Table % Rounding

When examining the table % variable, it was found that almost all values for table % were whole numbers (only 924 of the 53,940 observations (1.71%) were not). Since there is nothing about the table % variable that would indicate values should be whole numbers, this was likely a result of many of the cases being rounded at some point during the data collection process. Consideration was made towards rounding the remaining data points for table % and treating it as an ordinal variable, however this ultimately would not have improved the model. It is more desirable for the model to be able to handle non-rounded values for table % when making predictions, so the values were not changed. Ideally, this would be discussed with those responsible for providing the data, however they were not accessible during the process of this analysis.

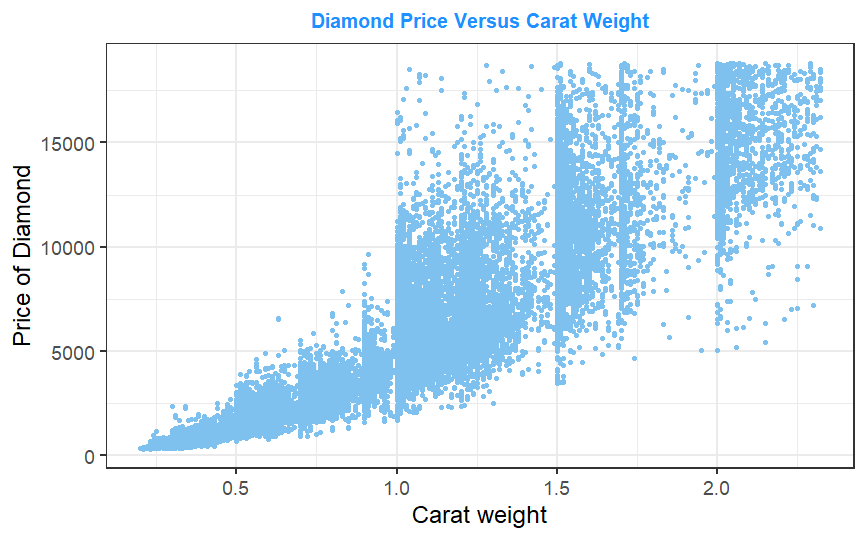
4. Initial Analysis

In this section, an exploration of the data is conducted. Data exploration is an important step in producing a model, as having a good understanding of the underlying dataset before attempting to build a model generally results in a more effective model being built. The relationship between each of the recorded variables was examined, with any findings of note and changes to the dataset discussed throughout this section.

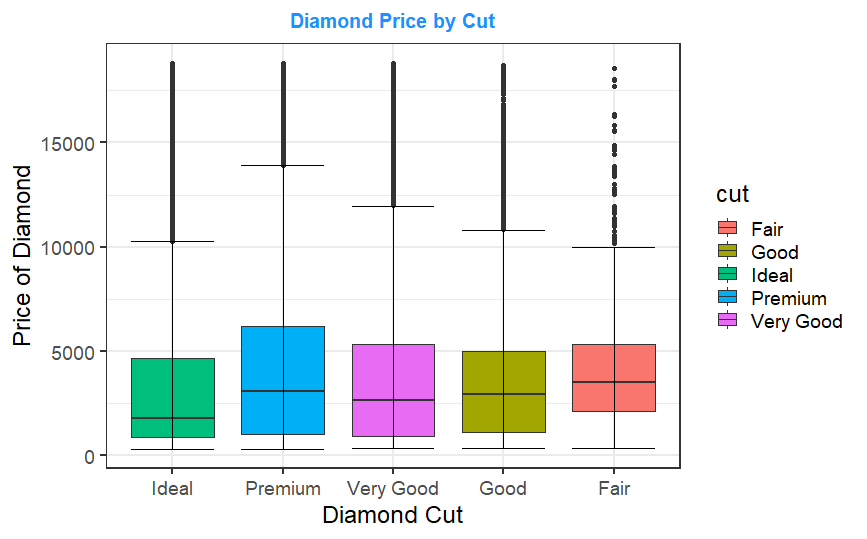
4.1 Relationship Between Price and Other Variables

As the objective of this model is to predict the price of diamonds, the relationships between price and the other recorded variables has been examined. Each variable was graphed against diamond price and the observed relationship explored to get a better idea of how each variable related to pricing. The results of this exploratory assessment are given below.

4.1.1 Diamond Price and Carat Weight

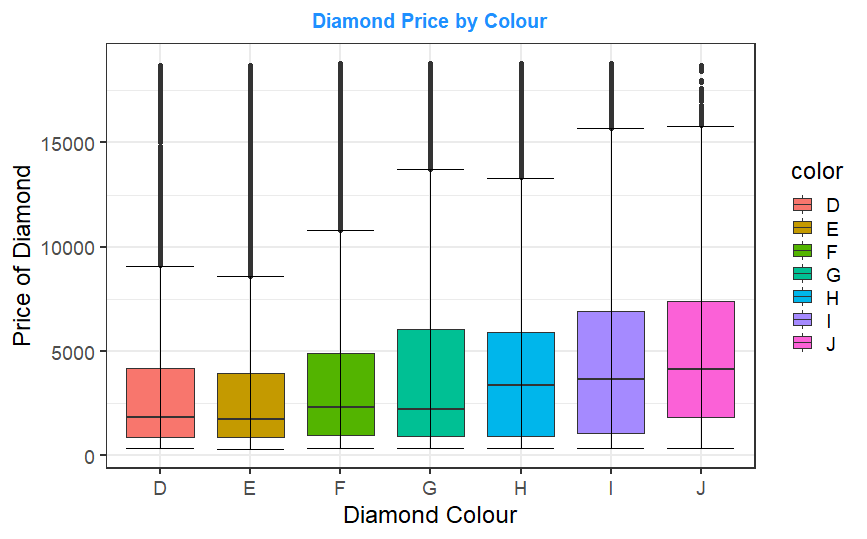
**From the graph, it is clear that diamonds with larger carat weight tend to have higher prices. The Pearson’s correlation coefficient of 0.92 indicates a strong positive linear relationship between these two variables. However, the prices of higher weighted diamonds, especially for diamonds with carat weight above 1, tend to have a much greater spread. It is possibly the case that, for heavier diamonds, other factors have a greater impact on the value of a diamond, leading to greater fluctuations appearing for greater carat weights.

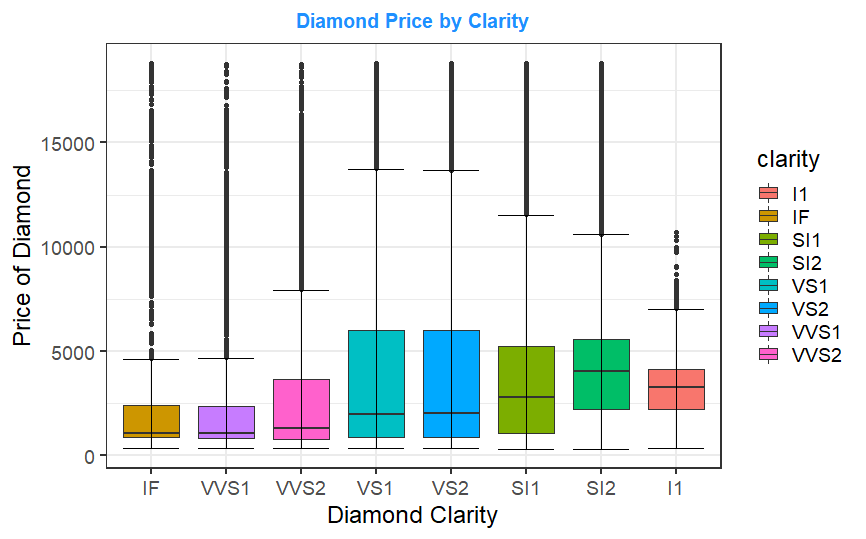
4.1.2 Diamond Price and Cut

From the graph, there does not appear to be a particularly strong relationship between diamond price and cut quality. Despite being described as ordinal data, the boxplots show a fall in median diamond price between “Fair”, “Good” and “Very Good” diamonds. This rises again for “Premium” but is lowest for “Ideal” diamonds, which are supposedly the most desirable. While the upper quartile does increase with cut (excluding “Ideal” diamonds), evidence suggests that there is fairly small relationship between a diamond’s cut quality and its price.

This relationship may have been observed due to the existence of a relationship between cut and another variable that caused diamonds with higher quality cuts to be worth less. For example, if diamonds with higher cut quality tended to have low carat weights, due to the relationship between carat weight and diamond price, diamonds with higher quality cuts would thus have lower carat weights and lower prices. However, comparing cut quality to the other variables recorded did not indicate any such relationship. Ultimately, diamonds with higher cut quality do not appear to be priced higher than those with lower cut qualities. Based on this data, those with lower cut qualities tended to have slightly higher prices. This appears to oppose the order given to cut quality by the data providers. Ideally, this would be discussed with them, however they were not accessible while this analysis was being conducted.

4.1.3 Diamond Price and Colour

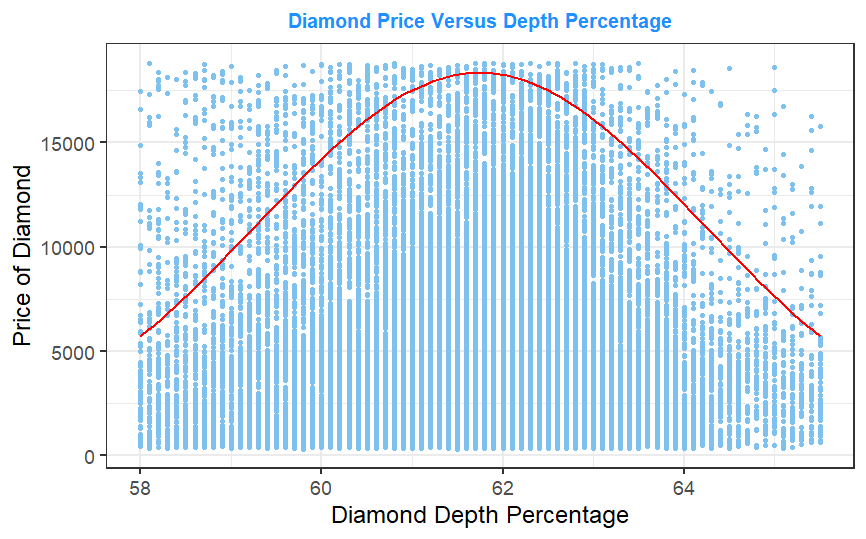
The graph shows a relationship between a diamond’s colour and price, however, while the data description indicated that the colours were ordinal, with D being the best and J the worst, the relationship with price appears to increase as the colour rating decreased. Based on the data, diamonds with colours associated with lower letters tended to have higher prices than those associated with higher letters. As with the relationship between price and cut, the relationship between colour and other recorded variables was examined, with a potential reason for this relationship being explored in section 4.4. If it were possible, this relationship between colour and price would be raised with the providers of this data, but they were not available during the conducting of this analysis.

4.1.4 Diamond Price and Clarity

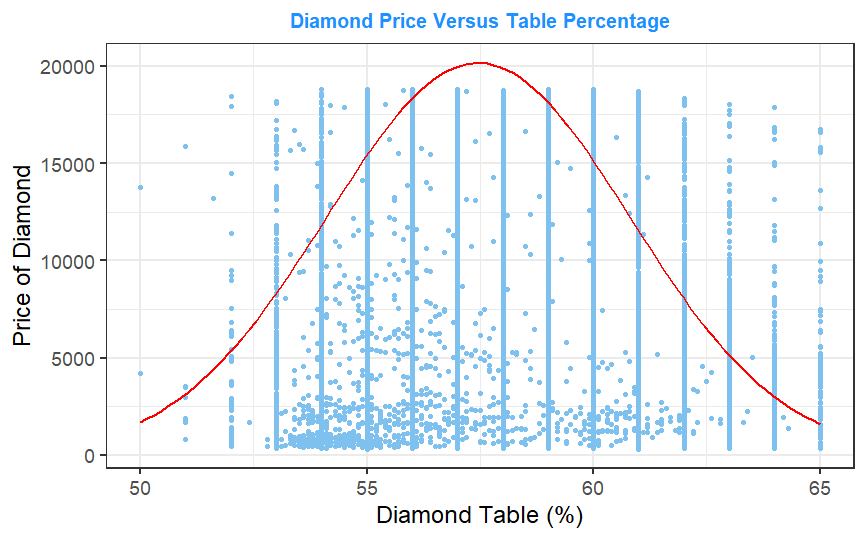
It should be noted that no diamonds with the clarity classification of FL, I2 or I3 were recorded in this dataset. This means that diamonds of these types cannot be assessed by the model provided. The model is not built to capture the relationship these values have with diamond prices due to their absence, and predictions made would likely be inaccurate.

Like with colour, there is a relationship between diamond price and clarity indicated by the data, and once again this relationship contradicts the scale given by the data providers. All else equal, lower clarity diamonds tended to have higher prices. The relationship between clarity and the other variables was explored to determine a potential cause for this discrepancy, with one being explored in section 4.5. Once again, the data provider would be consulted on this relationship were they available during the conducting of this project.

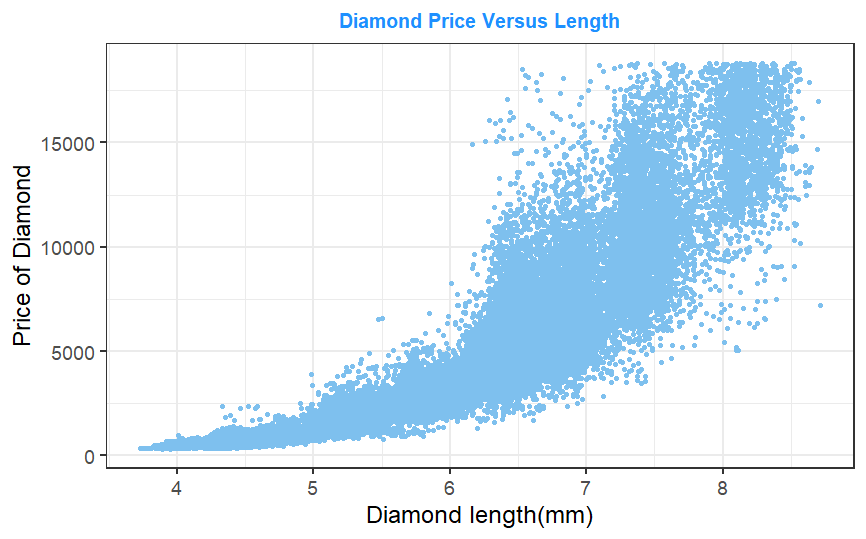
4.1.5 Diamond Price and Depth %

**At first, there appears to be no discernible relationship between diamond price and depth %. However, for higher and lower depth %, there does tend to be a larger number of diamonds with lower prices, indicating that diamonds with very high and low depth % tended to have lower prices (to make this easier to see, the red line has been added). Since depth % measures diamond height relative to girdle diameter, this indicates that diamonds with a height being roughly 62% of the girdle diameter could be valued higher. This is possibly considered a rare or desirable shape for diamonds to have, with diamonds deviating from this shape being worth less. However, it should be noted that this is a very loose relationship, as there are still many diamonds with the ideal depth % and far lower prices as well as a significant number of diamonds without the ideal depth % and higher prices. Ultimately, there is some evidence that a relationship between the depth % and price exists, but it appears to be a very weak relationship.

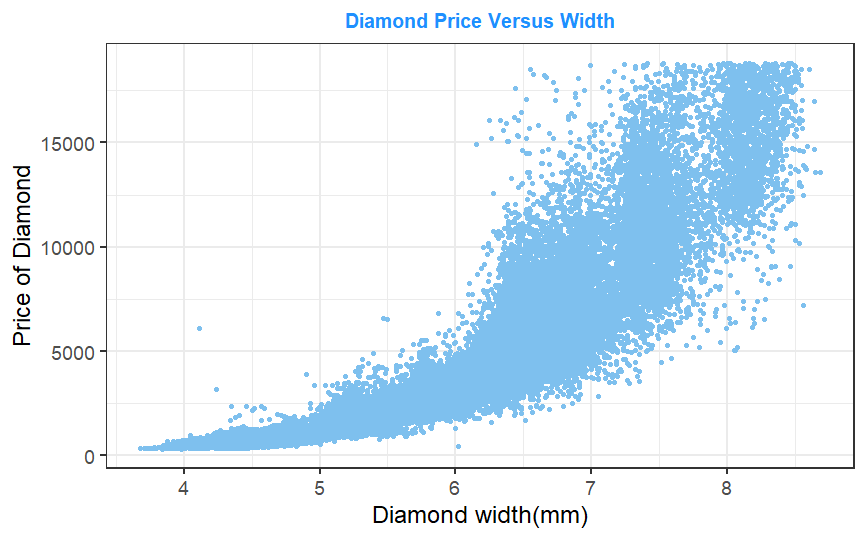
4.1.6 Diamond Price and Table %

Most table % values occur at fixed intervals (as discussed in section 3.5) which can be seen in the graph, with many cases occurring at fixed intervals along the x-axis. Ignoring this, like with depth %, we see a vague indication that diamonds with their width over average diameter (or table %) being roughly between 53% and 62% tended to be priced higher than those with a table % outside this range (the red line has been added to make this relationship clearer). This indicates that diamonds with a specific shape tended to be priced higher. However, again like with depth %, this relationship appears to be relatively weak, with many diamonds in the ideal table % range having lower prices and a large number outside this range having higher prices.

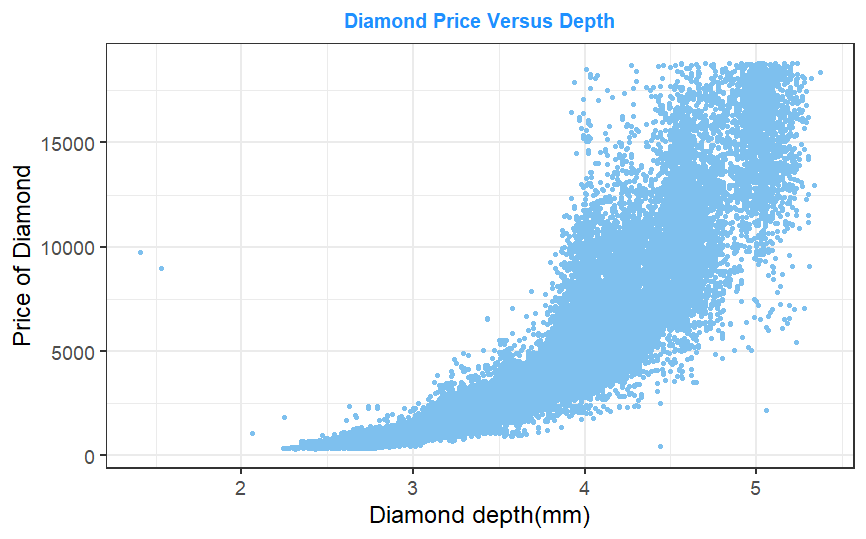
4.1.7 Diamond Price and Length

The graph shows a very strong exponential relationship between diamond price and length. As length increases, the value of the diamond rises. Intuitively, this makes sense, as the larger a diamond is, the more valuable it would be. Like with carat weight, after a certain point (length of 6mm in this case), the variation in prices at different lengths tended to rise noticeably, however even for larger length diamonds the exponential growth in prices is still clearly visible.

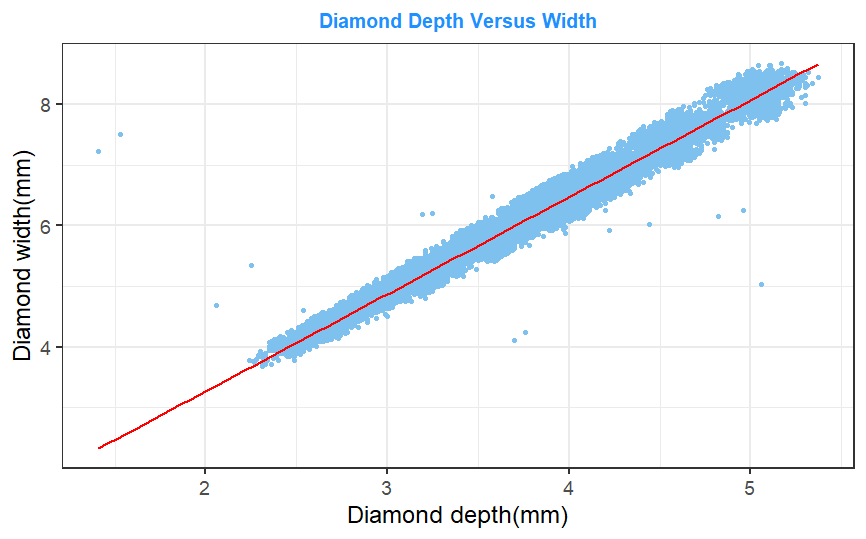
4.1.8 Diamond Price and Width

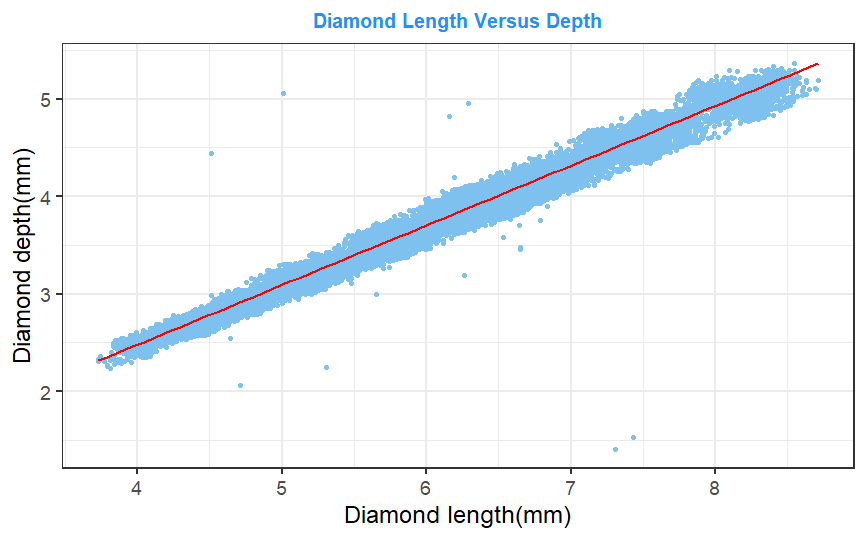
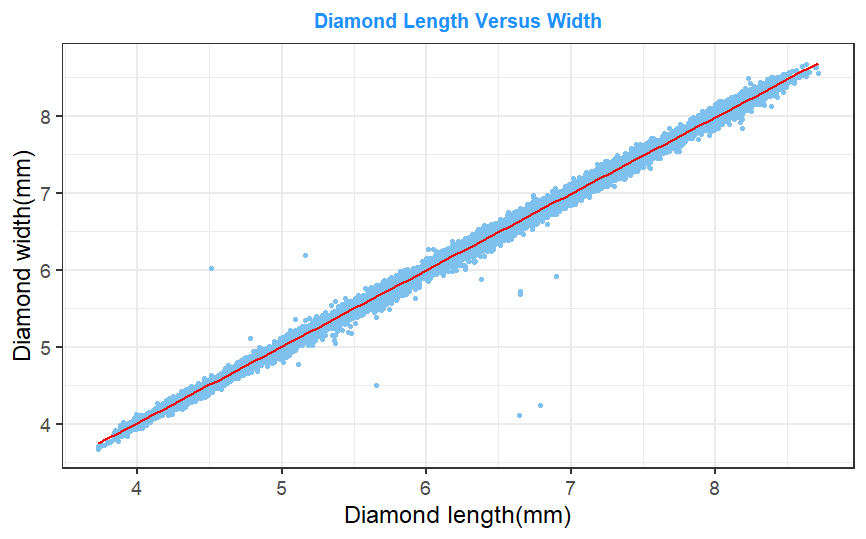
The relationship between diamond price and width and diamond price and length are almost identical, to the point where only a few differing data points make it clear that this graph is not the one examined in section 4.1.7. This indicates that length and width may be highly correlated variables, and this relationship is investigated in section 4.2. As with length, this relationship is intuitive, as the wider a diamond is, the more valuable it is likely to be.

4.1.9 Diamond Price and Depth

Once again, the relationship between diamond price and depth is very similar to that of price and length as well as that of price and width. However, the depths of diamonds tended to be closer together than the widths and lengths, and an increase in depth of one millimetre resulted in a greater increase in price relative to the effect of 1 millimetre of length or width. This can be seen by the scales of the x-axis, where the highest priced diamonds occurred for diamonds with depth of around 5mm while for length and width this occurs around 8mm. Other than that, the shape of the price-depth curve is nearly identical to that of the price-length and price-width curves. The potential correlation between all three of these variables is examined in section 4.2.

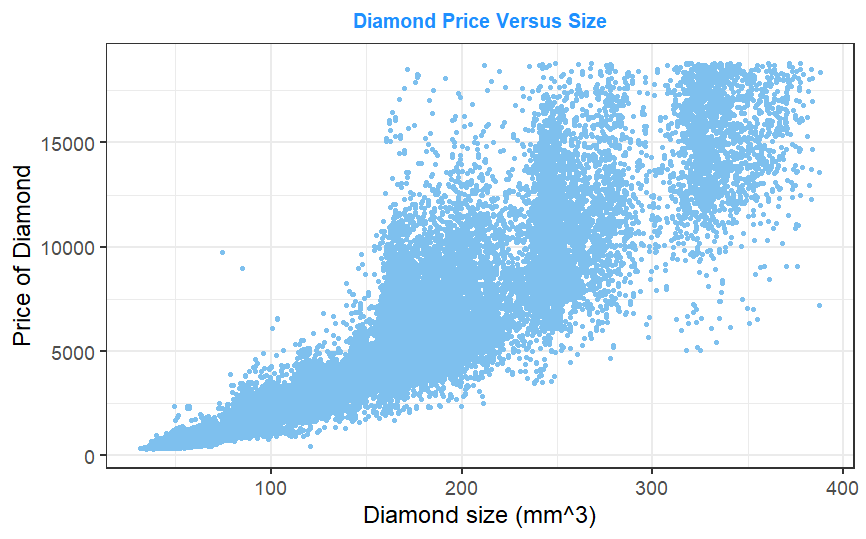
4.2 Relationship Between Length, Width and Depth

The similarities between the graphs comparing diamond price to length, width and height indicated a strong possibility that these three variables are strongly correlated. Thus, an exploration of each of length against width, length against depth and width against depth was conducted to explore this relationship further. The red line in each of the three graphs is a line of best fit between the variables recorded, and a large proportion of the datapoints are close to the line of best fit indicating a strong correlation.



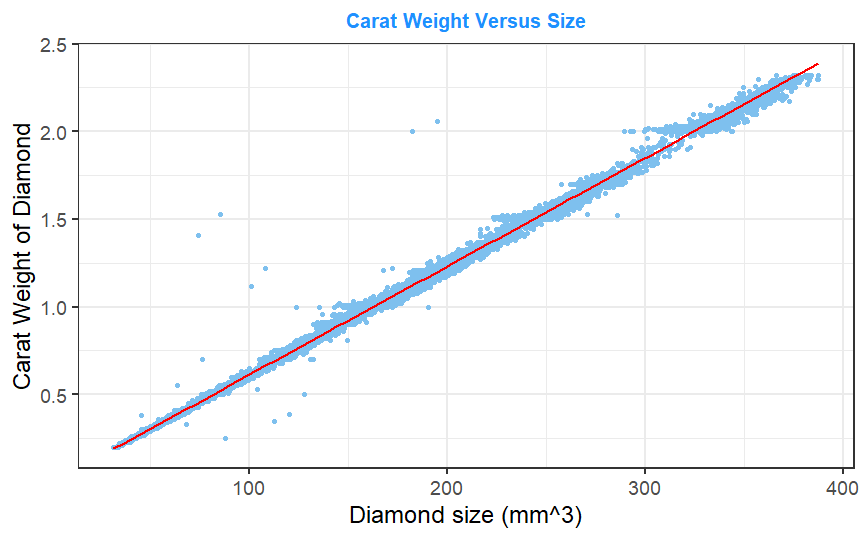
Furthermore, the Pearson’s correlation coefficients of all three relationships examined above were above 0.999. This indicates that for an increase in length, width or depth of a diamond, a near perfectly proportional rise in the other two size variables occurs. Based on the strong positive correlation between these variables, it appears to be the case that diamonds observed are a very similar shape, resulting in diamonds with greater length, width or depth also being larger in the other two dimensions (for example, diamonds with larger lengths, due to being the same shape as other diamonds observed, must also have larger widths and depths).

It is desirable for models to be built using less variables, making predictions more straightforward and reducing model complexity. Furthermore, due to their high correlation, combining these three variables will result in the loss of very little information. As a result, the length, width and depth variables were combined into a derived variable; size. Since the shapes of the diamonds are not given, size will be determined by multiplying length, width and depth.

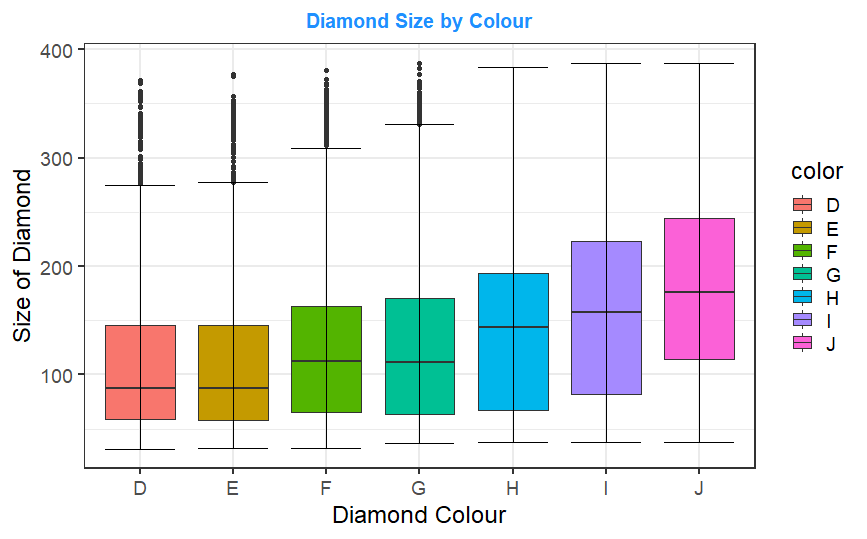
Comparing this new size measure to price, we can see a very similar relationship to those seen between price and length, width and height. The relationship between size and price appears to effectively captures the relationship between price and length, width and height.

The relationship between size and the other recorded variables has also been assessed, and any important results are described below.

4.3 Relationship Between Size and Carat Weight

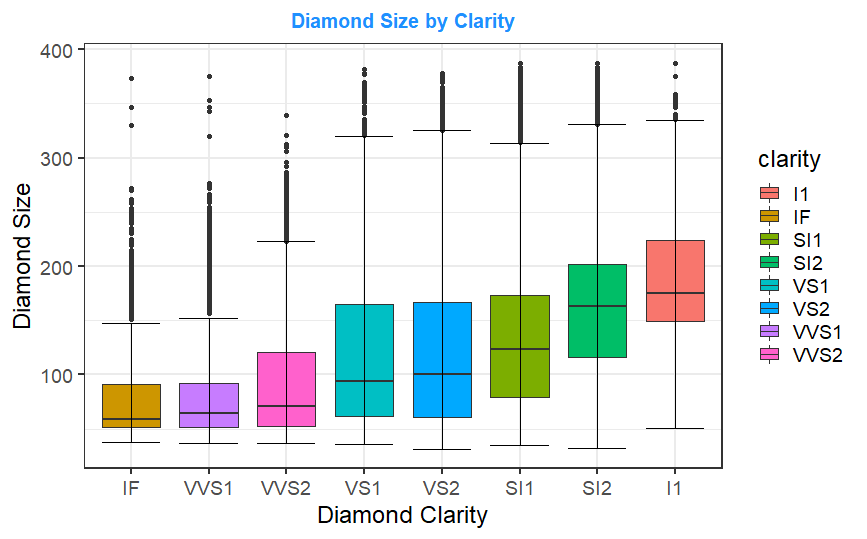
The Pearson’s correlation coefficient between carat and size was over 0.999, indicating near perfect positive linear relationship between size and carat weight. The similarity between the price-size graph seen in 4.2 and the price-carat graph in 4.1.1 is further evidence of such a relationship. Intuitively, this makes sense, as since all diamonds are made of the same material, their density will be the same. Thus, the mass (and as a result weight) of a diamond depends only on its volume (or size).

Due to the high correlation between these two variables, the removal of one of these variables will reduce the model’s complexity without the loss of much information on the price of a diamond. As a result, one of the variables will be removed from this analysis. It is recommended to remove the variable with the lowest variance, which in this case is carat weight, having a variance of 0.21 against size’s variance of 5448.54. Thus, the carat weight variable has been removed from the model.

4.4 Relationship Between Size and Colour

As indicated in section 4.1.3, contrary to the order indicated by the data description, diamonds with supposedly better colours tended to have lower prices than those with worse colours. The relationship between colour and the other recorded variables was examined, and as the graph on the right shows, a negative relationship between diamond size and colour exists. This indicates that larger diamonds, which were shown in section 4.2 to be associated with higher prices, tended to be of worse colours than smaller diamonds. This relationship is a potential cause for diamonds with better colours being priced lower, as those diamonds also tended to be smaller, resulting in them being considered less valuable.

4.5 Relationship Between Size and Clarity

As described in section 4.1.4, the relationship between diamond price and clarity seemed to contradict the scale indicated by the data providers. Diamonds with high clarity tended to be priced lower. The relationship between clarity and the other recorded variables was explored to find a potential cause for such an occurrence, and as with colour (see section 4.4), the variable’s relationship between size provides a potential explanation for this occurrence.

As seen by the graph, diamonds with lower clarity (all else held equal) tended to be much larger than those with higher clarity. This may be due to clarity being based on the obviousness on inclusions in the diamond. It is probable that inclusions are more obvious on larger diamonds, hence larger diamonds tend to receive lower clarity ratings. Due to the strong positive relationship observed between size and price (section 4.2), since diamonds with worse clarity ratings tended to be larger, their price would be higher. It may be this relationship which results in the negative relationship between price and clarity observed in section 4.1.4.

5. Adjusted Dataset and Expected Variable Importance

Having conducted an exploratory analysis of the data, the adjusted dataset resulting from this analysis consists of 52,430 observations of seven variables (including price), with length, width and depth described in section 1 being combined into size (described below) and carat weight being removed. It is with this data that the model was built and tested.

Size: Product of diamond length, width and depth. Measured in mm3.

Based on this initial analysis, predicting the price of a diamond is expected to depend a lot on size, with a very strong positive relationship observed in section 4.2. Diamond colour and clarity should also contribute to the prediction of a diamond’s price based on the relationships observed in section 4.1.3 and 4.1.4. The relationship between price and cut seen in section 4.1.2 may contribute somewhat to the prediction of diamond price, however this relationship does not appear to be as strong as those of price and size, colour and clarity. Table % and depth % are expected to contribute very little to the prediction of a diamond’s price based on section 4.1.5 and 4.1.6, where a very weak relationship was observed. Furthermore, stronger relationships between price and other variables such as price, colour and clarity can easily be expressed in linear terms as seen from the graphs of these relationships in sections 4.1 and 4.2. It may be the case that a Linear Regression model is effectively able to predict the price of diamonds, and this will be explored in section 7.2.

In section 4.4 and 4.5, relationships between size and colour and size and clarity were observed. These two relationships indicate that the importance of these variable to the prediction of a diamond’s price may be dependent on one another. Furthermore, it makes sense that the importance of a diamond’s colour or clarity would depend on the size of the diamond. If a diamond is very small, the colour and clarity may not be obvious to an observer, reducing their importance to the valuation of the diamond compared to a larger diamond, where they would be more obvious. Thus, important interaction terms may exist between these variables in the models built. This will be examined further in sections 7.2 and 10.2. Unlike with colour and clarity, no relationship was found between cut and another variable which could explain the fact that lower quality cut diamonds appeared to have higher prices.

6. Data Partition

The data was split before being used to build the Random Forest model. 41,946 of the recorded diamonds were used to build this model (training data), accounting for 80% of the sample while the remaining 10,484 (20%) were used to test the model’s effectiveness (test data). Assessing performance using data that was used to build a model will overestimate the performance of the model. By using a separate dataset to evaluate performance (that being the test data), an unbiased measure of the model’s effectiveness can be obtained.

The splits created were stratified to ensure (where possible) that the proportions of values contained in the full sample were reflected in both the training and test splits. For example, the below table shows the frequency of each colour of the diamonds in the full dataset, the training dataset and the test dataset.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset\Colour (% frequency) | D | E | F | G | H | I | J |
| Full Data | 12.67 | 18.26 | 17.17 | 21.07 | 15.31 | 9.92 | 5.06 |
| Training Split | 12.64 | 18.31 | 17.78 | 21.08 | 15.26 | 9.88 | 5.04 |
| Test Split | 12.74 | 18.07 | 17.46 | 21.04 | 15.50 | 10.08 | 5.10 |

The table shows that the proportions of each colour found are very similar between each of the three sets of data. This was done deliberately to ensure that the model would be built and tested with datasets containing the same proportions of values to that of the sample data (and thus, in theory, the entire population of diamonds). Furthermore, splitting the data in such a way reduces the chance of the training data missing a feature of a diamond, and thus not considering it when developing the model. Returning to our above example, suppose the training dataset contained no diamonds with the colour “J”. The model built would then not be able to account for any potential impacts this colour would have on the diamond price when making predictions. This would likely make the model considerably worse at predicting prices for diamonds of this colour.

7. Comparison Models

Before creating the Random Forest model, a Regression Tree and Linear Regression model were built from the adjusted training dataset. These models were used as a comparison for evaluating the performance of the Random Forest model (section 9.1).

7.1 Regression Tree

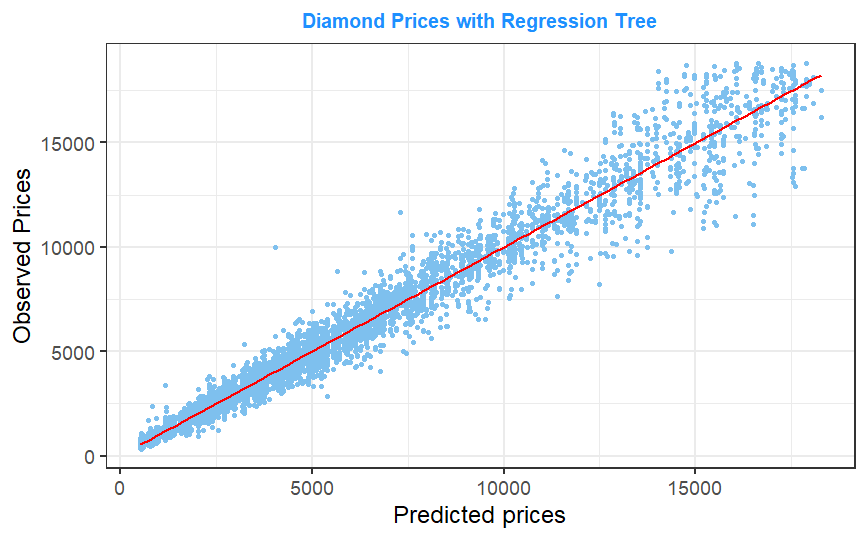
The Regression Tree was constructed using the ANOVA method to determine node impurity when selecting which splits to use. A bottom up approach was used to select the size of the tree, building a maximal tree and selecting the tree which minimised the cross-validation error (plus one standard error). The complexity parameter associated with this optimal tree was 7.1x10-6. This resulted in the construction of a very large tree. Large trees are needed to capture linear relationships effectively. As discussed in section 5, the relationship between price and other variables of importance tended to be linear in nature. Usually, large trees tend to overfit data. However, due to the size of the dataset, very large trees could be constructed without overfit reducing the accuracy of the model.

Regression Trees give a measure of variable importance. The improvement to node impurity across the entire tree caused by each variable is measured, with the higher improvement indicating a more important variable. Below is a table outlining the importance of each variable determined by the tree.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Size | Clarity | Colour | Depth % | Cut | Table % |
| Variable Importance (%) | 83.60 | 9.91 | 5.34 | 0.45 | 0.40 | 0.31 |

Size was by far the most important variable when determining diamond price in the Regression Tree. This aligns with our plotting of size against price, in which a strong positive relationship was observed. Diamond clarity and colour also contributed to the prediction of diamond price, both of which a relationship was again observed from plotting them against price. Depth %, cut and table % contributed very little to the prediction of diamond price in this model. While section 4.1.2 showed a relationship between price and cut, it was not as strong as the ones seen between price and size, clarity and colour. Furthermore, weak relationships was observed when graphing price against depth % and table %. The variable importance observed by the Regression Tree matches what we expected to observe based on our initial analysis (see section 5).

In terms of performance, the Regression Tree was effective at predicting diamond prices. This is likely due to the size of the tree allowing for linear relationships to be effectively captured by the model. Furthermore, trees are effective at capturing non-linear relationships as well. As a result, the tree built could effectively capture both linear and non-linear relationships between a diamond’s price and other variables.

 On the right is a graph of the Regression Tree’s predictions of the test data’s prices against the prices observed for these diamonds. There is a clear positive correlation between the two values. A Pearson’s correlation coefficient of 0.988 was observed, indicating a near perfect correlation between the predicted and observed prices. While the graph shows a dip in accuracy for higher priced diamonds, the effectiveness of this model indicates that a more complex model like a Random Forest may not be necessary to effectively predict diamond prices. This will be discussed further in section 9.1.

A root-mean-square error (RMSE) of 593 was found for this model. This metric measures the error between the observed and predicted prices, with a lower error indicating a smaller difference between observed and predicted values. RMSE will be used to compare the performance of the different models created in the prediction of diamond prices.

7.2 Linear Regression

The Linear Regression model was first built using all 6 independent variables as well as every interaction term between these variables. Since Linear Regression gives a measure of uncertainty to each term appearing in a model, any of these terms which has a p-value above 0.05 were removed. A p-value above 0.05 indicates a greater than 5% chance that the true value of a given coefficient is equal to 0. In other words, there is a good chance that the term is not important to the prediction made. 12 interaction terms had a p-value below 0.05, and as a result were considered significant to the performance of the model. These interactions were:

|  |  |  |  |
| --- | --- | --- | --- |
| cut \* colour | cut \* clarity | cut \* depth % | cut \* table % |
| cut \* size | **colour \* clarity** | **colour \* depth %** | **colour \* table %** |
| colour \* size | **clarity \* table %** | **clarity \* size** | **table % \* size** |

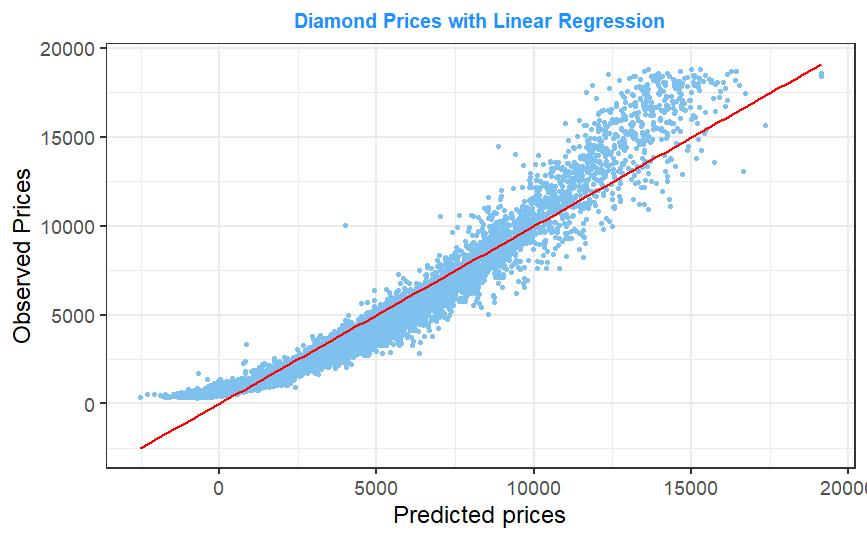
An interaction term indicates that the effect of a variable on the model’s prediction depends on the result of another variable. For example, the interaction between diamond colour and size was considered significant in determining the price of a diamond in the Linear Regression model. This indicates that the effect of a diamond’s colour on its price would be different depending on the size of the diamond. In section 5 we discussed the potential for important interaction terms to exist between a diamond’s size, colour and clarity. These three terms, amongst others, appear in this Linear Regression model as important to the predictions made.

To determine the importance of each of these interaction terms to the model’s accuracy, models excluding one of each of the the interaction terms were built, used to predict prices for the test diamonds and the RMSE between the observed and predicted prices were calculated. Interactions whose exclusion led to a large increase in the RMSE of the model (or greatly reduced the accuracy of the model’s predictions of prices) were considered important. Below these results are given in a table.

|  |  |  |
| --- | --- | --- |
| **Interaction Excluded** | **RMSE** | **Change from Complete Model’s RMSE** |
| **Cut - Colour** | 985 | <0.5 |
| **Cut - Clarity** | 985 | <0.5 |
| **Cut - Depth %** | 985 | <0.5 |
| **Cut - Table %** | 985 | <0.5 |
| **Cut - Size** | 985 | <0.5 |
| **Colour - Clarity** | 1007 | 22 |
| **Colour - Depth %** | 985 | <0.5 |
| **Colour - Table %** | 985 | <0.5 |
| **Colour - Size** | 1019 | 34 |
| **Clarity - Table %** | 985 | <0.5 |
| **Clarity - Size** | 1047 | 62 |
| **Table % - Size** | 987 | 2 |

\*Values have been rounded to the nearest whole number.

The exclusion of colour - clarity, colour – size and clarity – size interaction terms each resulted in a far greater reduction in RMSE than the exclusion of any other interaction. This indicates that these three interactions are the most important in the prediction of a diamond’s price. This aligns with our findings in section 5. As a result, it is likely that these three interactions will be of importance to the Random Forest model built.

The performance of the Linear Regression model was also evaluated using the test sample by comparing the diamond prices predicted by the model to the observed prices (see graph to the right). The linear model appears to overestimate lower diamond prices and underestimate higher priced diamonds. This can be seen most clearly at the tails of the graph. Despite this, the Pearson’s correlation coefficient of 0.967 shows that the model’s predictions are still very close to the observed prices. The RMSE of the model was 985, not as low as the Regression Tree model. Combined, these indicate that the Linear Regression model, while not as accurate as the Regression Tree model, can still predict the price of a diamond effectively. In other words, while the overall relationship was not linear, it can still be represented by a linear relationship with relatively high success.

The linear relationship observed between diamond prices and important variables such as size and colour indicate that the price of a diamond can be modelled effectively with a linear method. The effectiveness of this Linear Regression model acts as further evidence in support of this. However, it does not explain the relationships entirely, hence the linear model underperforming when compared to the Regression Tree model. As with the Regression Tree model, the effectiveness of this simple model calls into question the necessity of a complicated ensemble model such as a Random Forest. This is addressed in more detail in section 9.1.

8. Building the Random Forest Model

Random Forests are an ensemble model that combines the results of many regression or classification trees to make a prediction. To construct an effective Random Forest, these trees should be as independent as possible while still being effective predictors themselves. There is a trade-off between creating independent trees while ensuring each tree is a strong predictor. Thus, when building a Random Forest, there are several hyperparameters that must be considered which will affect the model’s performance:

Number of Trees: The number of trees used to construct the ensemble with

Sampling Method: What percentage of the training data is used to construct the bootstrap samples.

Size of Trees: How large are each of the individual trees built.

Number of Variables in Each Split: How many variables are considered at each split to be used as the primary splitter.

To make the trees more varied, Random Forest employs two techniques. Firstly, each tree is built using a bootstrapped sample of the training data. That way, each tree is constructed using a slightly different subset of the training data. Secondly, instead of simply choosing the variable that minimizes the loss in impurity at each split, each split uses the best variable out of a random subset of the variables. Combined, these two techniques result in more variation between the individual trees making up the ensemble, reducing their covariance and the error rate of the model. There is a trade-off, however, as these techniques also reduce the predictive power of the individual trees.

The combination of these hyperparameters which minimizes the mean of squared residuals of the ensemble model for a given dataset must be determined. Different datasets will be affected by these hyperparameters in different ways. To determine the optimal tree for predicting diamond prices, models were built using several versions of each hyperparameter, with the model resulting in the lowest error rate being kept. For each hyperparameter, 3 different values were considered. This resulted in 81 models being assessed (each combination of the above hyperparameters). The following hyperparameter values were considered:

Number of Trees: Models consisting of 200, 500 and 1000 trees were assessed. These represented a small, medium and large number of trees being used to construct a model.

Sampling Method: Models built using bootstrap samples equal to 33%, 66% and 100% of the training data were considered. Again, this represented a small, medium and large proportion of the original training data being used to construct the individual trees. It should be noted that, since bootstrapping builds a sample with replacement, even a bootstrap sample which is the same size as the training data (i.e. a 100% sample) will not exactly match the training data as a bootstrap sample is constructed by sampling from the original training dataset with replacement.

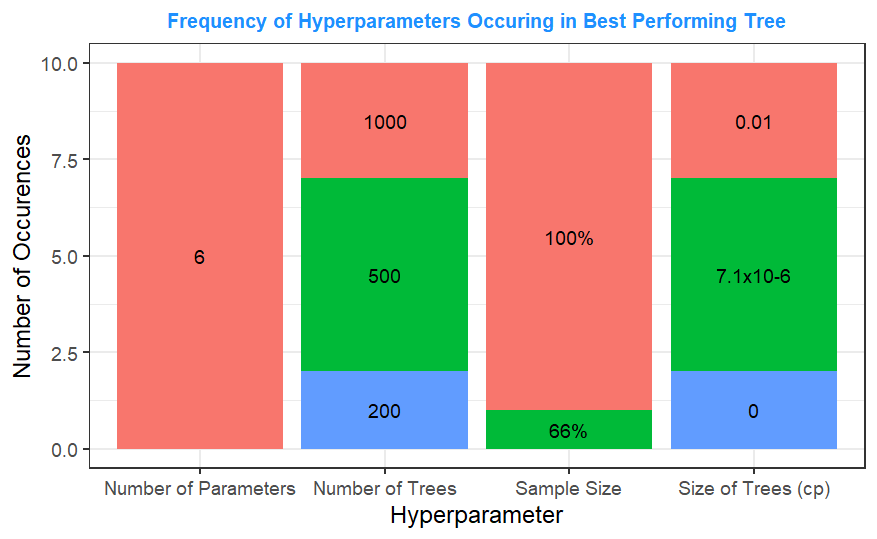
Size of Trees: Complexity parameters of 0 (a maximal tree), 7.1x10-6 (the ideal cut off for the individual Regression Tree constructed in section 7.1) and 0.01 (representing a smaller tree than the previous two) were considered as potential cut offs for the individual trees being built. These represented large, medium and small trees being used in the ensemble respectively.

Number of Variables in Each Split: Models considering 2 variables per split, 3 variables per split (roughly the square root of the number of parameters, the default number of parameters considered using Random Forest) and 6 variables per split were constructed.

To determine the effectiveness of the models, the out of bag cases (the cases that were not used by the bootstrap sample generated for each model) were used as test data. Each model was used to predict the prices of its out of bag cases, and the model which most accurately predicted the prices was considered the best. This ensures that the selected model is the one that most accurately predicts the prices of new data, reducing the likelihood of overfit occurring. To measure the accuracy, the mean of the squared residuals (MSR) was calculated, with the lowest value indicating the most accurate model.

Each of the 81 models were reproduced ten times. Since trees are unstable, this was done to ensure the model being selected consistently outperformed against the other 80 models. Due to the size of the training data, the time taken to build an individual Random Forest was too long for this method to be employed on the entire dataset. As a result, in each of the ten trials, a random sample of 2000 diamonds were taken from the training data and used to build the Random Forests. The MSR of each of the 81 hyperparameter combinations for each of the ten trials can be found in appendix 2 (with the lowest and highest MSR in each trial being highlighted).

It was found that a Random Forest built with 500 trees, 100% of the training data used for the bootstrapped samples, a complexity parameter of 7.1x10-6 and 6 variables considered per split had the highest accuracy compared to other models. The model was the highest performing model for three of the ten trials, with no other model having the best performance more than once. As a result, a Random Forest model using these hyperparameters and the entire training data was built. This model is the one that will be used for the prediction of diamond prices.

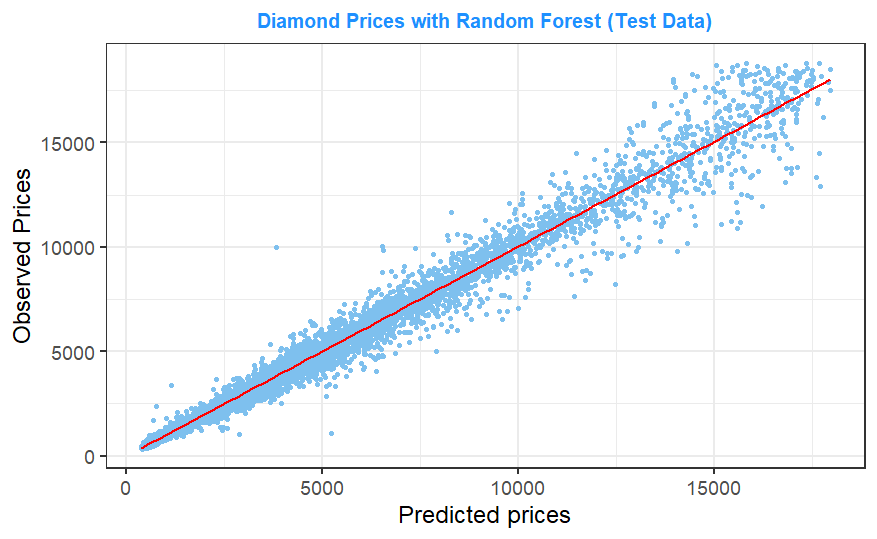
To the right is a graph showing the number of times each value for a hyperparameter appeared in the best Random Forest model built for each of the ten trials. For all ten trials, the most accurate Random Forests were built with 6 variables considered at each split. This indicates that this hyperparameter contributed the most to the model’s performance. For sampling method, using 100% of the sample was optimal in all but one case, where a model using 66% of the sample being the best model in one trial. The number of trees used in the best performing models were 500 in five cases, 1000 in three and 200 in two. For tree sizes, 7.1 x10-6 appeared as the best performing complexity parameter 5 times each, with maximal trees being optimal for 2 models and 0.01 being optimal for 3 models.

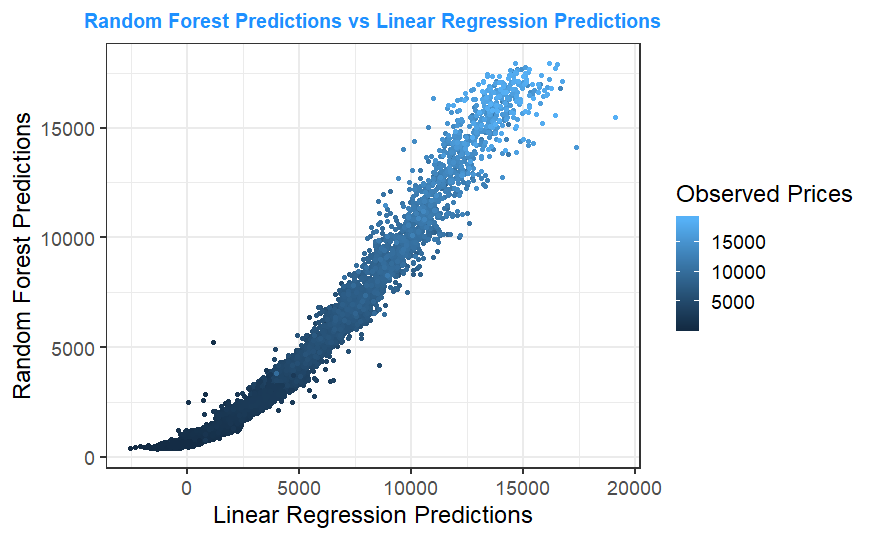
The hyperparameters used in optimal Random Forest model consistently outperformed models built using other hyperparameters, especially for sampling method and number of variables considered at each split. Having said that, hyperparameters do not act in isolation and their impact depends on the other hyperparameters used. The fact that only one combination of hyperparameters consistently outperformed the others is a strong indication that said combination will consistently result in the most effective Random Forest for predicting the price of diamonds.

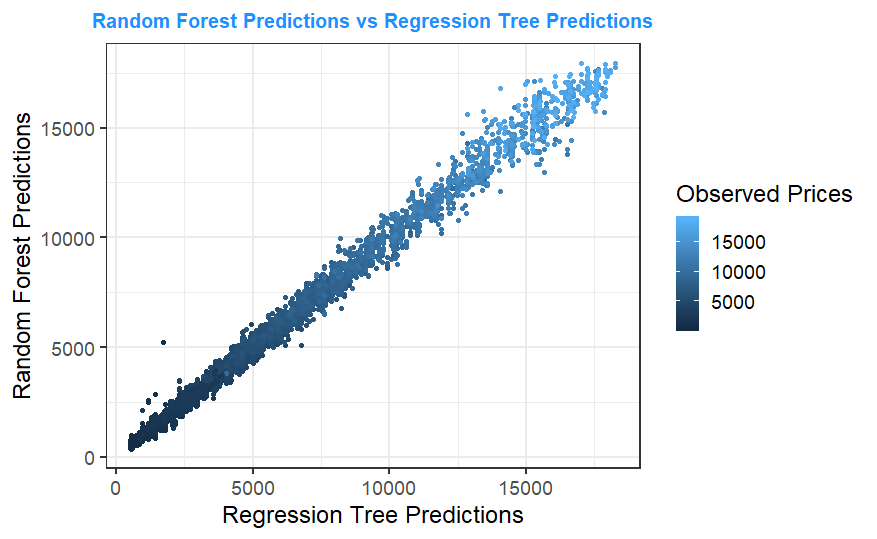
9. Evaluation of the Random Forest Model

Various methods of evaluating the performance of the optimal Random Forest model found in section 8 were conducted.

9.1 Optimal Model Performance

As with the models in section 7, the performance of the Random Forest model was evaluated by using it to predict the prices of the test data diamonds. The predictions were compared to the observed diamond prices (see right). As with the Regression Tree and Linear Regression models, there is a very high correlation between predicted and observed prices, indicating that the model can accurately predict the prices of diamonds that were not used to build the model. The Random Forest outperformed the other two models, with a Pearson’s correlation coefficient of 0.991 and a RMSE of 525 (see appendix 3).

Below are two graphs comparing the predictions made by the Random Forest against the Regression Tree and Linear Regression models. This gives an indication of the closeness in the predictions made by these two models in comparison to those made by the Random Forest. The graphs are coloured by the observed prices of each diamond. A line of best fit was not included since it obscured the visibility of the observed price changes.



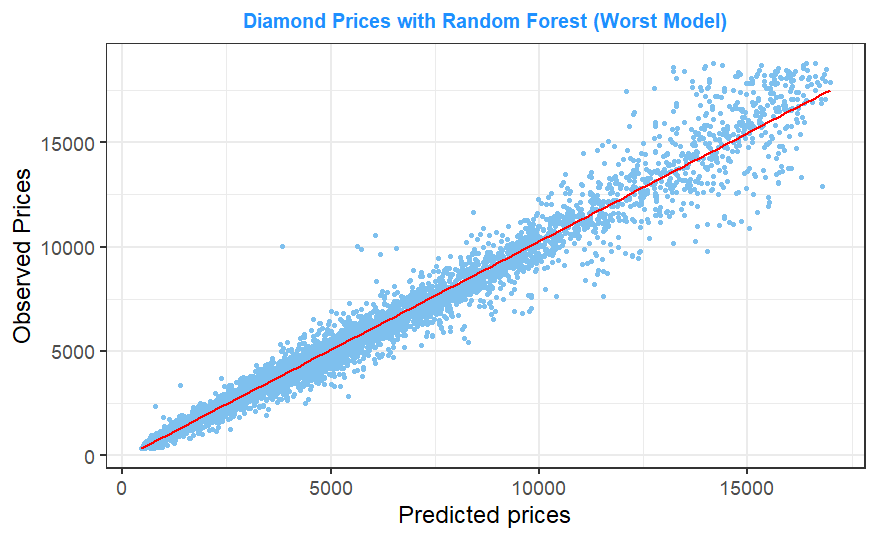
In both cases, the predicted prices align closely to the observed prices, with higher predictions made by each model for diamonds observed as having higher prices. This is to be expected as all three models were able to accurately predict diamond prices. With a Pearson’s correlation coefficient of 0.997 and a RMSE of 299, the predictions made by the Random Forest and Regression Tree models are nearly identical. The predictions made by the Random Forest and Linear Regression models where less similar, with a Pearson’s correlation coefficient of 0.974 and a RMSE of 868. The Linear Regression tended to give slightly higher predictions for lower priced diamonds than the Random Forest, and slightly lower predications for higher priced diamonds. This echoes the relationship between the Linear Regression model’s predictions and the observed diamond prices discussed in section 7.2.

While the Random Forest does outperform compared to the other two models, due to the complexity and time taken to build a Random Forest, a simpler model may be a better choice depending on how the model will be used. As described in section 7.2, the linearity of the relationships between price and other variables allows for a Linear Regression model to predict prices with high accuracy. The relationship between the Random Forest predictions and predictions made by the other two models, especially the Regression Tree model, were very strong. This suggest that either model could act as an effective replacement for the Random Forest model if a less complex or quicker to construct model were needed.

The decision to use a Random Forest depends on the trade-off between accuracy and complexity. If the increase in accuracy of the Random Forest is worth the cost of using a more complex model, then the Random Forest should be used. For this project, a Random Forest was requested, but it is important to note that for this dataset at least, a simpler model can still effectively predict diamond prices, and may be desirable in circumstances where a more straightforward and quicker model is needed.

9.2 Contribution of Hyperparameters

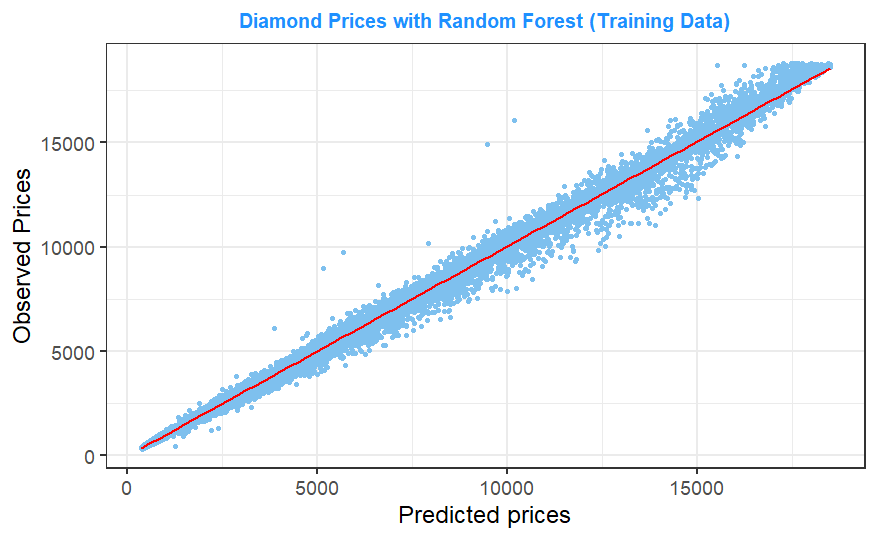
In the ten trials conducted in section 8, the Random Forest built with 200 trees, 66% of the training data used for the bootstrapped samples, a complexity parameter of 7.1x10-6 and 2 variables considered per split had the lowest accuracy for 5 of the 10 trials, with the second worst performing model (being the worst 3 times) differing only by its complexity parameter (which was 0.01).

 To examine the importance of selecting the appropriate hyperparameters, this model was used to predict the prices of the test data diamonds (see right). From examining the graph, a very similar relationship between predicted and observed prices can be seen. With a correlation coefficient of 0.989 and a RMSE of 579, while this Random Forest did underperform compared to the optimal Random Forest model, it still performed slightly better than the Regression Tree model (appendix 3).

Ultimately, while taking the time to determine the best hyperparameters did improve model performance, selecting any sensible hyperparameters can still lead to the building of an effective Random Forest for the prediction of diamond prices. Having said this, the improved performance of a Random Forest compared to a simpler model such as a Regression Tree depends almost entirely on the selection of optimal hyperparameters. Thus, for it to be worthwhile to use a Random Forest, selection of the optimal hyperparameters is necessary, as otherwise a simpler model could be implemented more easily and would make more accurate price predictions.

9.3 Model Performance for Training Data

Overfit occurs when a model so effectively predicts values for the training data that it becomes less accurate for new data. This tends to occur with more complex models. As mentioned in section 8 the model was selected based on its ability to predict the out of bag cases and not its accuracy at predicting the training data. Thus, the chosen model is less likely to be subject to overfit. Nevertheless, a test for overfit was conducted.

To test for overfit in the model, a comparison between the Random Forest’s accuracy in predicting the prices of the training and test data was undertaken. A large difference between the accuracy of the model when predicting with the training data compared to the test data would indicate overfit in the model. As expected, the model is more effective at predicting the prices of diamond used to build the model (see graph on the right). A Pearson’s correlation coefficient of 0.998 was found, indicating near perfect correlation between the observed and predicted prices. The RMSE between predicted and observed prices was 243. However, the accuracy of the model when predicting prices for the training data was not so much greater than the accuracy for the test data so as to indicate the occurrence of overfit.

While the relationship between diamond price and other variables can be expressed effectively with a linear model, this does not entirely capture the relationships in the data. While simpler models could accurately predict the price of diamonds, the Random Forest model constructed was more accurate. If construction time and model complexity are not a concern, a Random Forest model will result in more accurate estimations of diamond prices. However, if a quicker or simpler model was preferred, a Regression Tree or Linear Regression model could still be used to good effect. Selection of optimal hyperparameters improved the performance of the Random Forest model. Despite this, even a sub-optimal choice of hyperparameters, provided the selection is sensible, can result in an effective model, although the loss in performance means that a simpler model should be used instead of doing this. The accuracy of the Random Forest model when making predictions for the data used to build the model and unseen data was not different enough to indicate the presence of overfit in the model. This is to be expected, as the selection of an optimal Random Forest model was determined based on its ability to accurately predict prices for the out of bag cases, which themselves were not used in the construction of the model.

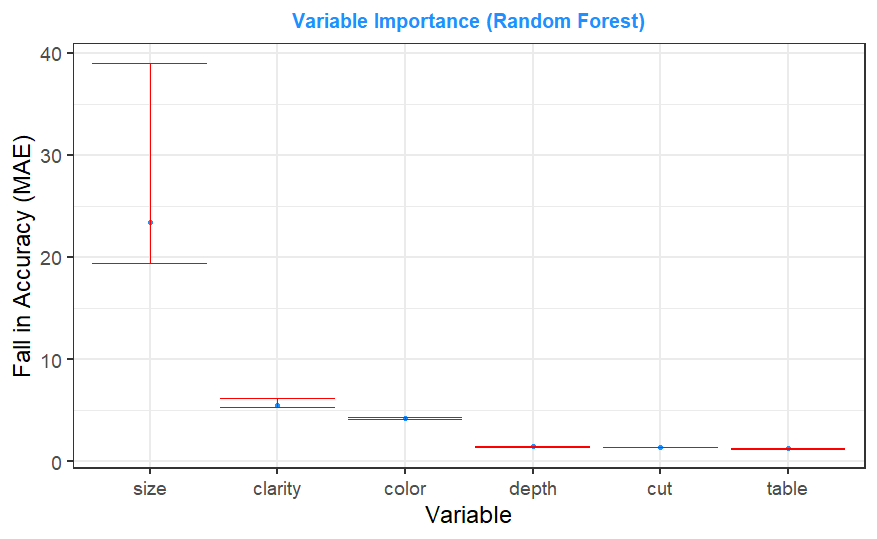
10. Model Interpretation

It is important to understand the underlying methods by which a model is making predictions. Knowing why a model is making the predictions that it is may be a legal requirement, but it also helps to ensure that the assumptions that the model is making are known to and understood by the user. This will allow the user to question and adjust model predictions as needed, as well as explain to others why a certain prediction was made. Below several techniques used to interpret prediction models are applied to the Random Forest model constructed here.

10.1 Variable Importance

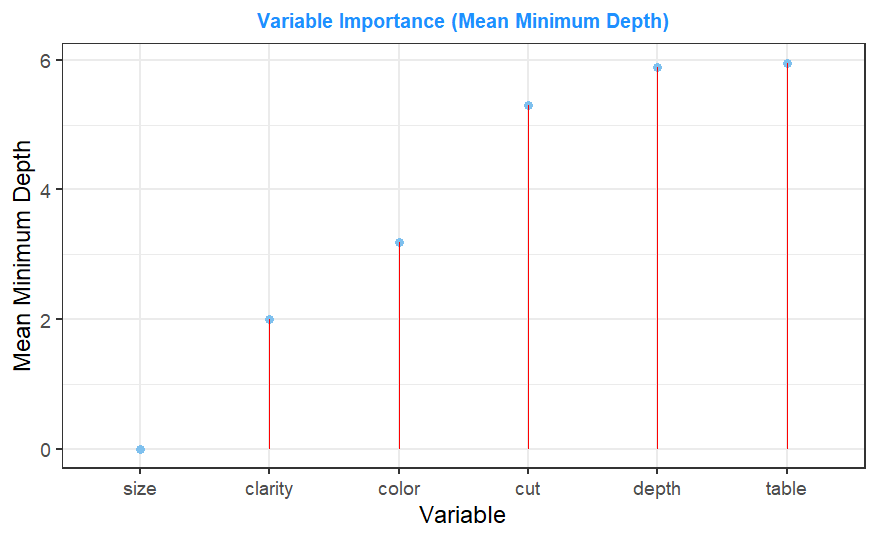
From an initial analysis of the data, its was assumed that size would be important in determining the price of a diamond due to the strong positive relationship between the two variables. Diamond colour and clarity also had strong relationships with diamond price, indicating that they would also contribute noticeably to the prediction of prices. Diamond cut, table % and depth % had much weaker relationships with price, indicating that these variables would contribute less to model performance. This assessment aligned closely to the variable importance measure given by the Regression Tree model (section 7.1).

Regression Trees measure variable importance by impurity improvement, however with Random Forests, variable importance is determined by a variable’s contribution to prediction. Contribution to prediction is determined by shuffling the values of a given variable and using the model to predict the prices for the data containing these shuffled variables. If there is a large fall in accuracy compared to the unshuffled dataset from doing this, the shuffled variable is considered to be important to the model’s performance. In other words, if a variable is important, when this variable is randomised the model’s performance drops noticeably. Accuracy is determined by the mean absolute error between the predicted and observed prices.



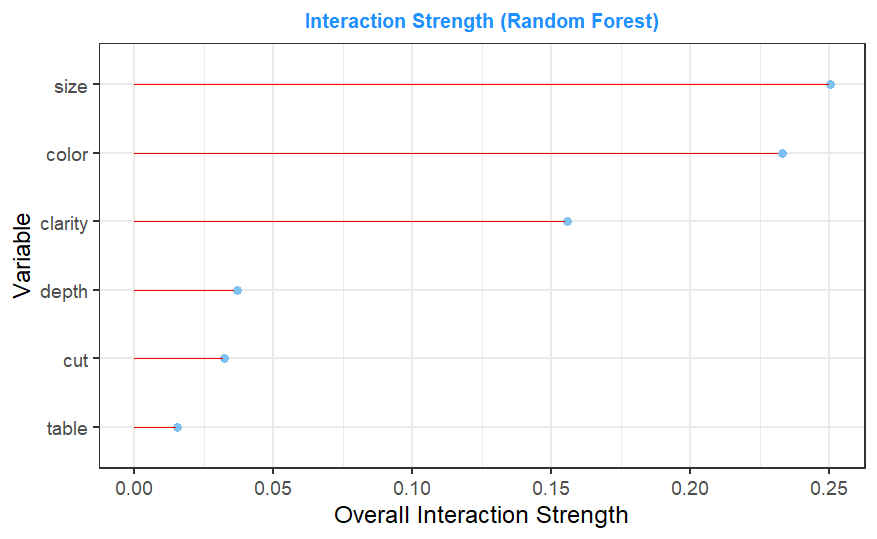
On the right is a graph showing the loss in accuracy resulting from the shuffling of each variable used by the Random Forest model. The process was repeated 10 times to give a measure of uncertainty to the variable importance measures (shown by the red line). Based on contribution to prediction, variable importance in the Random Forest model closely aligns with expectations, with size being the biggest contributor to the model’s performance, clarity and colour contributing noticeably and depth %, cut and table % contributing comparatively little.

As a further verification of the importance of these variables, the mean minimum depth of each variable across the trees was examined. The mean minimum depth of a variable measures the average depth of the first time a variable is used as a split in each tree used to construct the Random Forest. Since tree splits are selected based on whichever split results in the greatest reduction in impurity, the minimum depth of a variable is a good indication of a variable’s importance. A low depth indicates a more important variable.

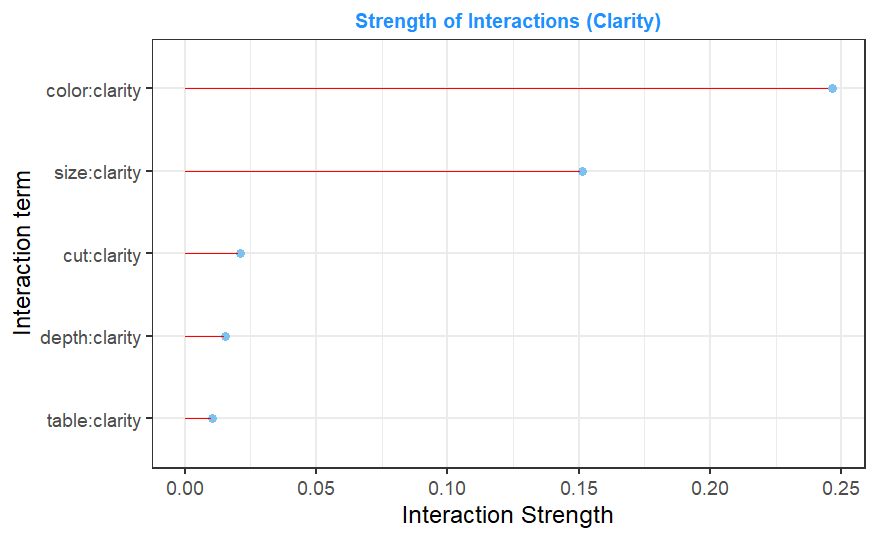
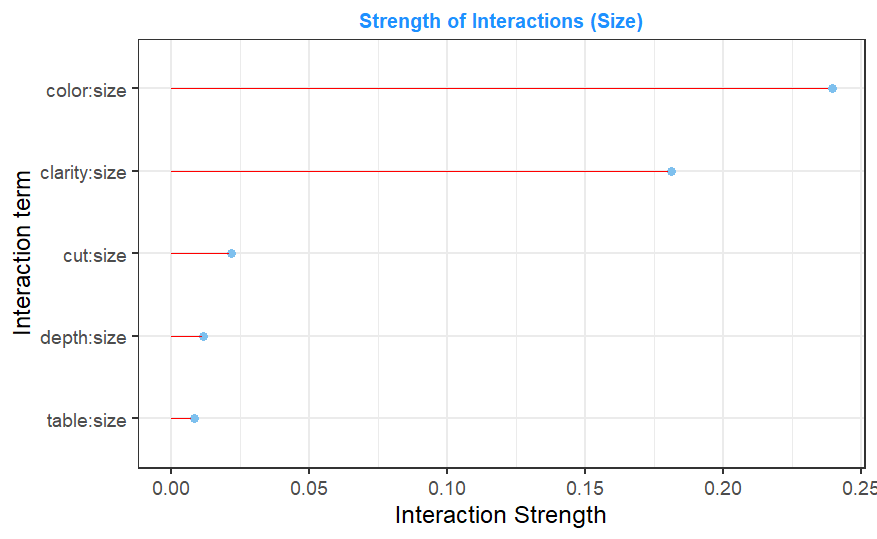
The graph on the right shows the mean minimum depth of each variable. Compared to contribution to prediction, measuring variable importance with mean minimum depth yields almost identical results. Size, clarity and colour remain the most important variables to the prediction of diamond prices while depth % and table % contribute very little. The only difference is clarity, which in terms of contribution to prediction contributed as much to the fall in accuracy of predictions as depth % and table %. Based on mean minimum depth, cut was somewhat more important to the prediction of diamond prices. This aligns more closely to the initial analysis of the dataset, where a smaller relationship could be seen between the cut of a diamond and its price, albeit one that is more prominent than the relationships between a diamond’s price and its depth % or table %.

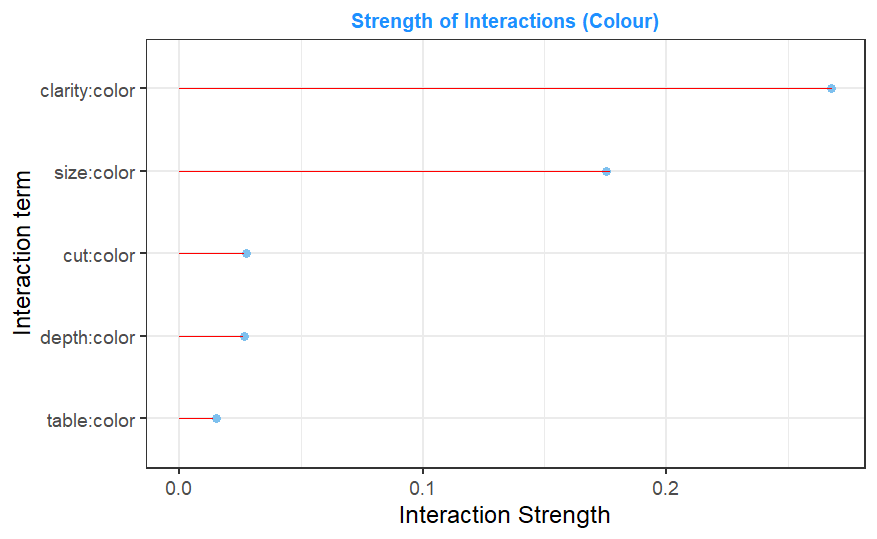
The importance of different variables to the prediction of the price of a diamond with the Random Forest model align closely both with the initial analysis of the dataset (sections 4 and 5) and with the variable importance measure given by the Regression Tree model (section 7.1). As expected, a diamond’s size is the most important factor in determining the price of a diamond. Clarity and colour also contributed to this prediction. Cut contributed somewhat to predictions while depth % and table % contributed very little.

10.2 Interaction Terms

In section 7.2, the interaction terms considered significant to the prediction of the price of a diamond for the Linear Regression model were shown. Random Forests incorporate interaction automatically in the model (provided the trees are deep enough). In the Random Forest model constructed here, individual trees had a very low cut-off (7.1x10-6) resulting in trees with a large enough depth to capture any interaction terms possible with this dataset due to its low number of independent variables.

To determine the importance of interaction terms in the Random Forest model, Friedman’s H-statistic was calculated for each variable. This statistic returns a value between 0 and 1 indicating the proportion of a variable’s variance captured by interaction terms in the model. The graph of this statistic is shown on the right. The variables size, colour and clarity all had larger interaction strengths than depth %, cut and table %. Between 15% and 25% of the variance of size, colour and clarity came from interaction terms. The individual interaction terms relating to size, colour and clarity are graphed below, again using Friedman’s H statistic to determine the importance of each interaction term.



From these three graphs, it is clear that the interactions between size, colour and clarity are the only ones contributing greatly to the prediction of diamond prices by the Random Forest model. This aligns with our expectations outlined in sections 5 and 7.2 that these three interaction terms would be the most important to the prediction of a diamond’s price. The contribution of other interaction terms is minimal, indicating that they are not hugely important to the accuracy of the Random Forest model.

10.3 Contribution of Variables to Prediction

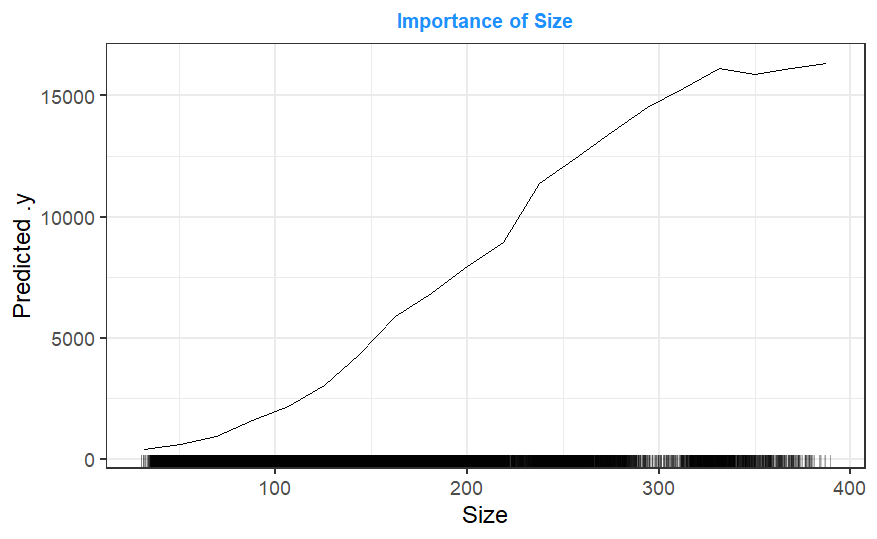
Having determined the importance of each variable to the prediction of the price of a diamond as well as the most important interaction terms, the contribution of these variables and interactions to the predicted price is now examined. A partial dependence plot has been created and assessed for each variable as well as the three most important interaction terms (colour-size, colour-clarity and size-clarity). A partial dependence plot assesses the impact each observation of a variable has on the prediction of a diamond’s price. This is done by creating a dataset of observations (using both the training and test data) for each unique value of the variable. In each dataset, each occurrence of the variable is replaced by the corresponding unique value, and a prediction is made for each new case. The average prediction for each dataset is obtained and these average predictions are plotted against the unique value associated with the dataset. The resulting graph shows how predictions alter for different values of a given variable.

A downside of partial dependence plots is that they are distorted when dependence between variables affects a prediction. Some of the cases made by the artificial datasets used to construct the partial dependency plot will not make sense in the real world. This in turn can somewhat distort the results produced by the partial dependence plot. An example of this occurring can be seen in section 10.3.2.

To plot the interaction terms, colour and clarity were treated as continuous variables. This was only done for the construction of these graphs, and in the Random Forest model itself they remain ordinal. For colour, larger numbers represented the less desirable colours (for example, D, the best colour according to the data providers, was represented by 1, E by 2 and so on). For clarity, larger numbers represented lower clarity ratings (for example, IF is represented by 1, VVS1 by 2 and so on).

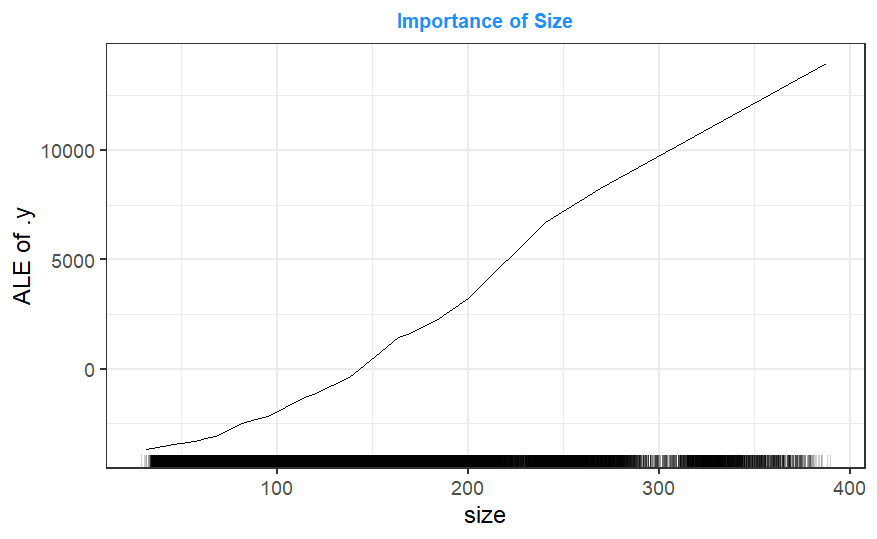
10.3.1 Contribution of Size to Prediction

The graph on the right shows the change in the predicted price of the model for different size values. The lines on the bottom indicate the frequency of the observed size values. As size increases, we see a rise in the predicted price made by the model (all other variables held equal). This matches the relationship we observed between price and size in section 4.2. There does appear to be a flattening of this relationship for larger sizes, though the decrease in frequency of diamonds of this size in the dataset may be the cause of this. Since there are fewer large diamonds, the reliability of this section of the graph can be called into question. It may be the case that, were more large diamonds available in the dataset, the positive relationship would continue. However, without these observations, there is no way to say with much certainty.



Predicted Price

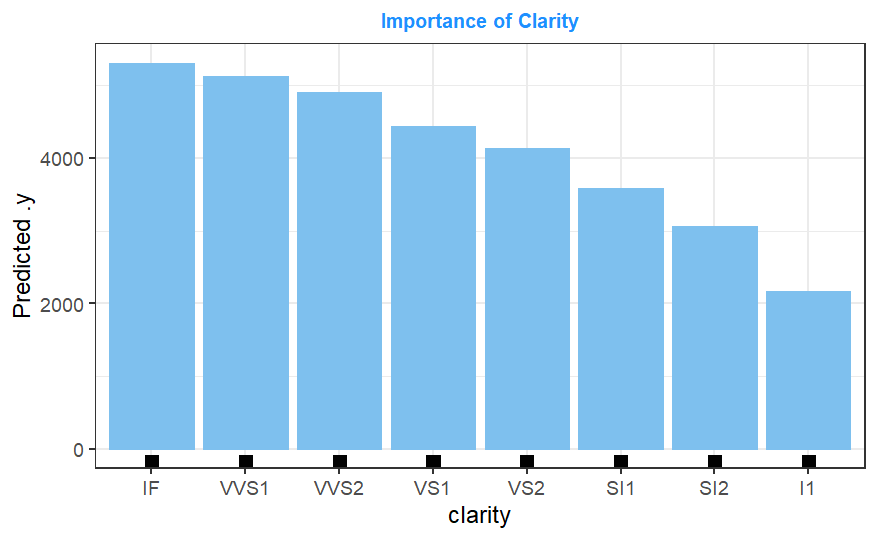
An alternative to the partial dependence plot for measuring variable contribution is the accumulated local event (ALE) plot. Instead of measuring the average prediction for a given variable, ALE plots examine how predictions change within small windows for a given variable. An ALE plot was constructed for the size variable (see right). ALE plot values are scaled by their change in price as opposed to partial dependence plots which are scaled by the predicted price itself. This can be seen through the negative ALE values observed for lower sized diamonds. This does not mean that smaller diamonds resulted in negative prices (as it would in a partial dependence plot) but rather that smaller diamonds result in a lower predicted diamond price. The ALE plot shows a very similar positive relationship between diamond prices and size. This is to be expected, since both methods measure a variable’s contribution to predictions. As a result, this is the only ALE plot constructed in this report.



Predicted Price

10.3.2 Contribution of Clarity to Prediction

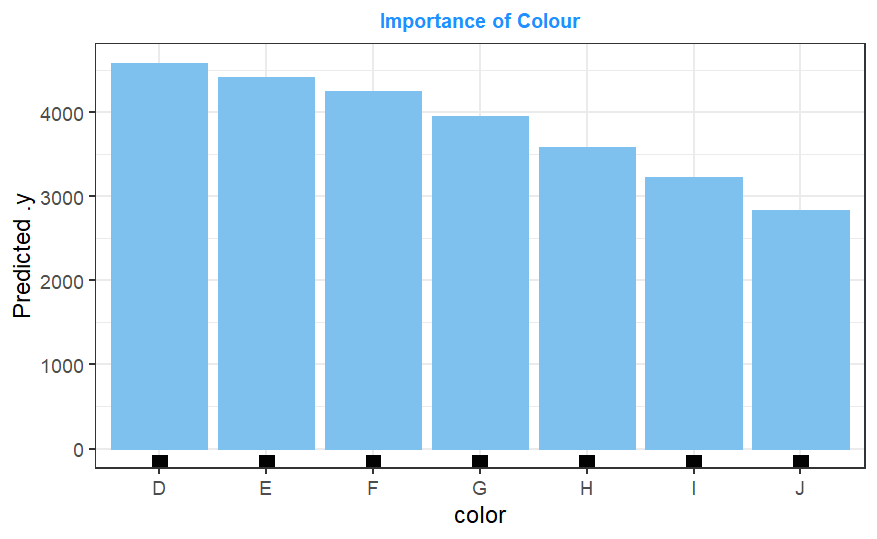
Initially, the effect of clarity on the predictions made by the Random Forest model appear to contradict previous findings. When diamonds had better clarity, all else held equal, they were priced higher by the model. The reason for this is the relationship observed between a diamond’s clarity and its size. As examined in section 4.5, larger diamonds tended to have less clarity. However, this fact, due to how partial dependence plots are constructed, is not captured by the graph on the right. Considered in isolation, having a higher clarity caused predicted diamond prices to rise. However, due to the negative relationship between size and clarity, diamonds with higher clarity tend to be smaller. Since a diamond’s size is an important factor in determining its price (as observed in sections 5, 7.1 and 10.1), the overall effect is that diamonds having higher clarity tended to be priced lower (as seen in section 4.5), despite better clarity itself causing a diamond to be priced higher, due to their smaller size.



Predicted Price

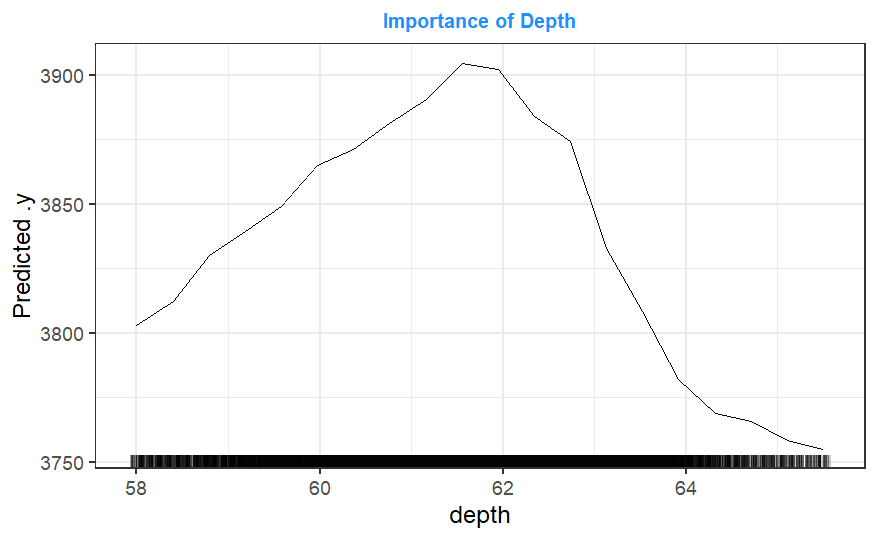
10.3.3 Contribution of Colour to Prediction

As with clarity, the contribution of a diamond’s colour to the Random Forest’s prediction of price appears to contradict what was observed in section 4.1.3. Once again, this is a result of the fact that the partial dependence plot does not capture the fact that diamonds with better ranked colours also tended to be smaller. In isolation, having a better colour does result in a higher predicted price for a diamond. However, due to the negative relationship between a diamond’s size and colour (observed in section 4.4), the reality is that diamonds with better colours tend to be smaller. Due to the relationship between size and price, the result is that the observed diamonds with better colours had lower prices due to also being smaller than those with worse colours. However, in isolation, a better colour will result in a diamond being priced higher, as expressed by the partial dependence plot.



Predicted Price

10.3.4 Contribution of Depth % to Prediction

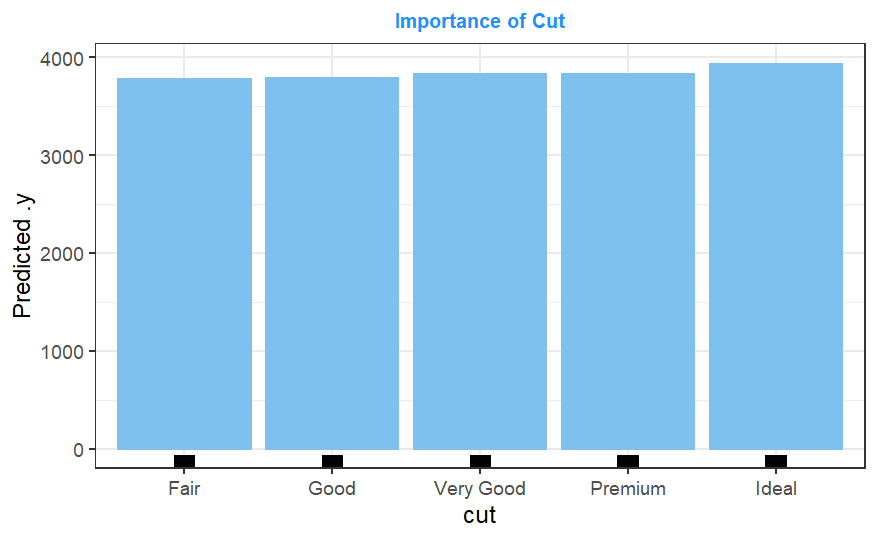


Predicted Price

The bell curve found in section 4.1.5 has been effectively captured by the Random Forest model. Higher and lower depth % values resulted in lower prices. However, it is important to look at the scale of the y-axis here. The variation in prices as a result of different depth % values was only around 150. In other words, while this bell curve effect was captured by the model, its contribution to the prediction of a diamond’s price was minimal. The difference between the prices of diamonds with ideal depth % values and those with less ideal values is relatively small. This would explain the low variable importance of depth % observed in sections 7.1 and 10.1.

10.3.5 Contribution of Cut to Prediction

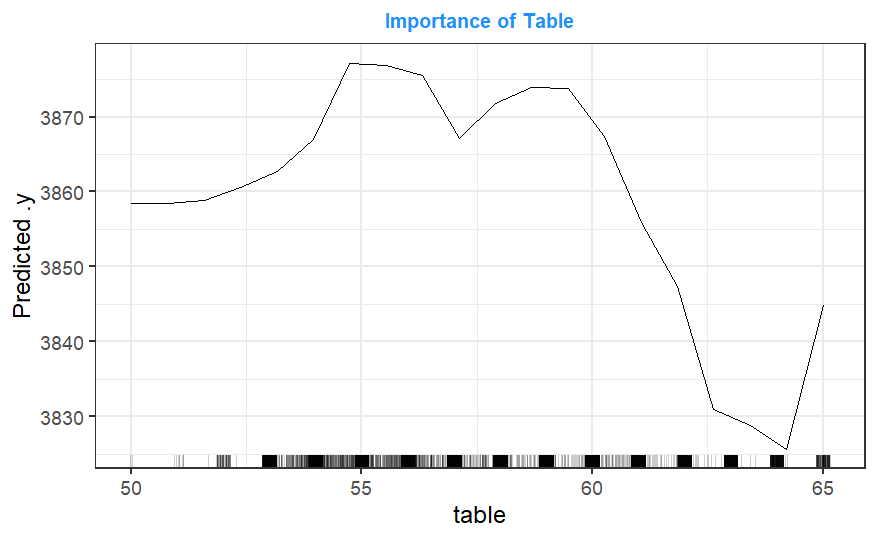
The change in predicted price for a change in cut is very small. There is a very slight increase as the cut quality of a diamond rises, but other than that it appears that different diamond cuts do not influence the price prediction of the Random Forest model. This aligns with the preliminary analysis of the data (section 5), the variable importance measure of the Regression Tree model (section 7.1) and the variable importance measure for the Random Forest model (section 10.1) which all found a diamond’s cut to contribute little to the prediction of its price. Thus, diamonds with different cut quality, all else held equal, did not tend of be priced differently to each other by the Random Forest model.



Predicted Price

10.3.6 Contribution of Table % to Prediction

As with depth %, the Random Forest model effectively captured the rough bell curved relationship between table % and diamond price (discussed in section 4.1.6). Furthermore, the variation in predicted price from changes in table % was very low (being less than 60), highlighting the findings in section 10.1 which indicate that table % contributed very little to the prediction of the price of a diamond by the Random Forest model.

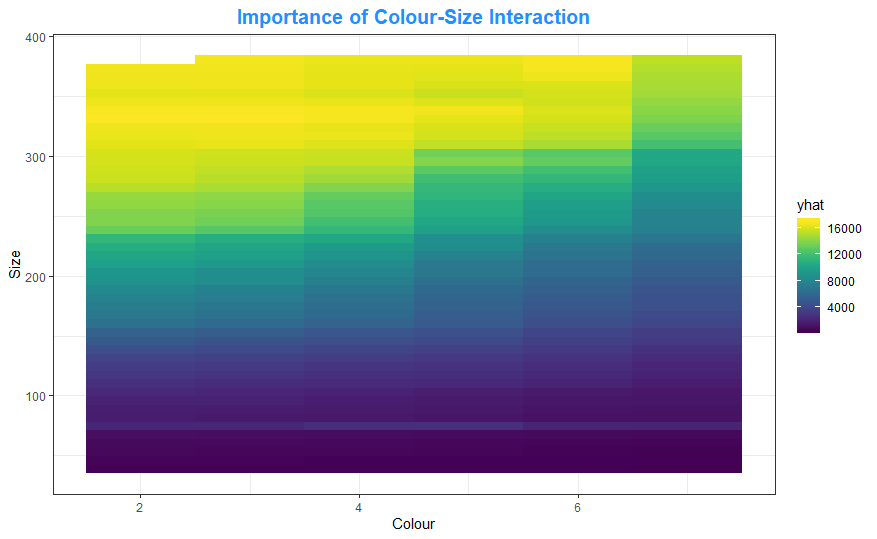


Predicted Price

From looking at the distribution of table % values, the fact that many of these values were whole numbers (discussed in section 3.5) can be seen. As a result, the values between these observations for the predicted prices are unreliable, as they are not based on actual model predictions. Rather, they are simply used to maintain a continuous line for the graph. For example, the relationship between table % and price shown for table % values around 50 was based on very few observations. As a result, the predictions made by the Random Forest model for diamonds with these table % values will be unreliable, and it may be the case that diamonds with lower table % are priced differently.

10.3.7 Contribution of Colour-Size Interaction to Prediction

Examining the colour-size interaction partial dependence plot, it can be seen that, as a diamond’s size increases, the impact of this increase on the predicted price of a diamond depends on the diamond’s colour. For lower sizes, the effect of colour appears to be almost irrelevant on a diamond’s price, indicating that a diamond’s colour is unimportant when the diamond is small. However, when pricing larger diamonds, the impact of a diamond’s colour became more important, with diamonds with better colours being priced higher than those with worse colours when size is large. This is most evident for diamonds with a size between 200 and 300, where the change in price as colour changes is most prominent. This aligns with the potential reason for a colour-size interaction term identified in section 5, with colours being less relevant to the pricing of smaller diamonds due to colour being more difficult to see when a diamond is small. Irrespective of whether this is the reason for this interaction occurring, the importance of colour on a diamond’s price depends on the diamond’s size when making predictions with the Random Forest model.

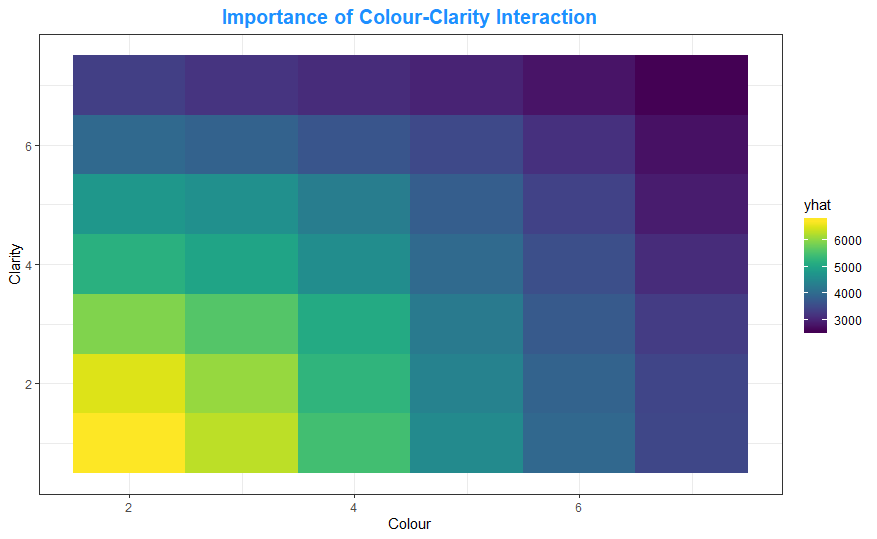


Price

It should be noted that this plot says nothing of the negative correlation between a diamond’s size and its colour (as observed in section 4.4). This is because the partial dependence plot examines how the model treats diamonds of different sizes and colours, saying nothing about whether such diamonds are likely to be observed. How common large diamonds with better colours are is not important to the predictions made by the model, simply that it can effectively predict the prices such diamonds should be when they occur.

10.3.8 Contribution of Colour-Clarity Interaction to Prediction

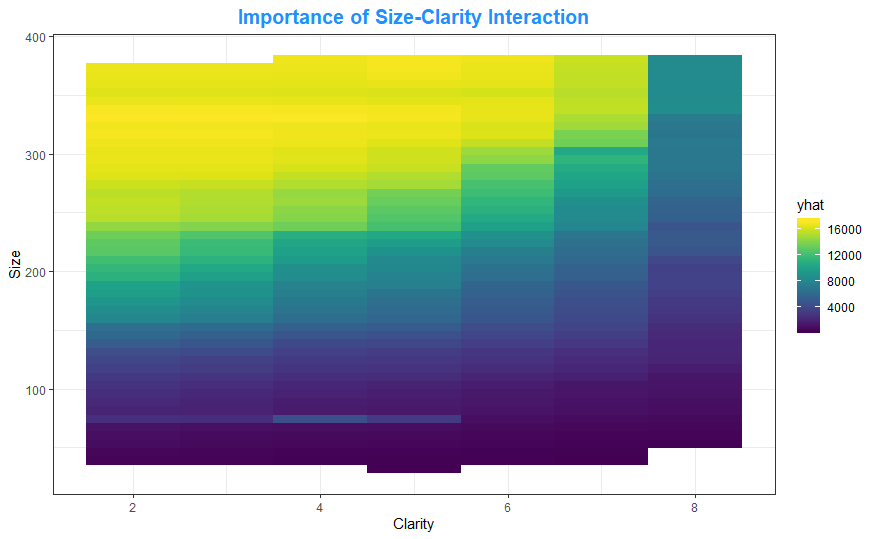
From the partial dependence plot on the right, the effect of a diamond’s clarity on the contribution of its colour to the prediction of price is clear. There is a consistent relationship throughout the graph, with diamonds of higher clarity but with a worse colour receiving lower price predictions than those with better colours, while diamonds with better colours but lower clarity having lower predicted prices than those with higher clarity. From this graph, it is clear that the predictions made by the Random Forest model for diamond prices depends on the colour-clarity interaction.



Price

10.3.9 Contribution of Size-Clarity Interaction to Prediction

The effect of the size-clarity interaction term on the Random Forest model’s prediction of prices closely mirrors that of the colour-size interaction. For smaller diamonds, the effect of a diamond’s clarity on its price was minimal while for larger diamonds, having less obvious inclusions resulted in the diamond being priced higher by the model. The most notable difference between the colour-size and size-clarity interaction is the fact that diamonds with the lowest clarity rating saw very little change in price as size increased. This indicates that diamonds of this clarity, regardless of their size, tend to be noticeably less desirable.

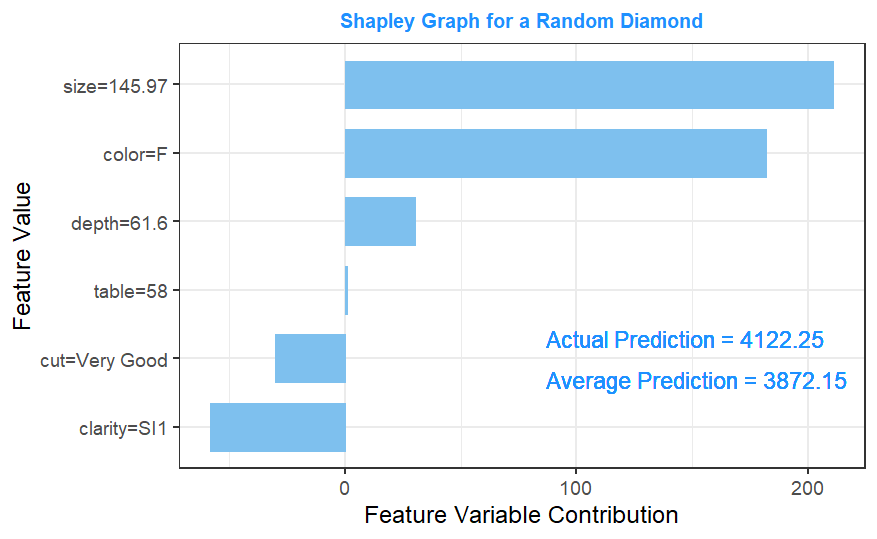


Price

Like with colour, this interaction may be due to the difficulty in distinguishing the clarity of a smaller diamond (as discussed in section 5). Whatever the case may be, the effect of a diamond’s clarity on the predicted price depends on the size of the diamond, with clarity contributing more to the difference in diamond prices when diamonds are larger. As mentioned in section 10.3.7, the negative relationship between clarity and size observed in section 4.5 does not have any bearing on the partial dependence plot produced, which is to be expected.

10.4 Shapley Values

Having assessed the Random Forest model from a global perspective, here we examine the predicted price of an individual diamond case by the model. The Shapley values are values calculated for each variable for an individual case (a single diamond in the dataset). The Shapley value of a variable is determined by measuring a variables contribution to the prediction made by a model. Since the contribution of a variable depends on the order at which the variable is removed from a prediction, each Shapley value is the average of many different calculated values, where the order the variable is removed from the case is random at each measurement. The result is a measurement indicating what effect having the value for a variable the case had contributed to the difference between the prediction made for the case by the model and the average price prediction made by the model.

As an example, the Shapley values were calculated for a random diamond with the results graphed on the right. The model’s predicted price for this diamond was above its average prediction. From the Shapley graph, we can infer that the size of the diamond and its colour contributed the most to this higher prediction, indicating the diamond has a more valuable colour and a larger size. The clarity of the diamond resulted in a negative contribution to the prediction, indicating that this diamond’s clarity is less desirable, resulting in a lower predicted price.

Shapley values can be used in conjunction with the Random Forest model when making predictions in the future. When predicting the price of a diamond, calculating the Shapley values as above will allow users to explain precisely why diamond was priced as it was. This allows for a better understanding of what the model is doing when predicting the price of a new diamond. Should pricing need to be explained to a third-party, or a specific diamond’s pricing need evaluating, producing a Shapley value plot is a good method by which to understand exactly why the model predicted the price that it did for the diamond in question.

Having investigated and interpreted how the Random Forest makes predictions of diamond prices, several inferences have been made about how the model works. How the model predicts the price of a diamond closely aligns with expectations about how the model would work based on the initial exploration of the dataset and the behaviour of the Linear Regression and Regression Tree models constructed in section 7. When predicting diamond prices using the Random Forest model, size was found to be the variable of greatest importance, with colour and clarity also being important to the prediction of prices. A diamond’s cut quality was found to contribute somewhat to predictions made while table % and depth % contributed very little. This aligns with expectations developed from the initial exploration of the dataset (section 5) and the variables of importance to the Regression Tree model built (section 7.1). The partial dependence plots explored how these variables contributed to the predictions made. The size-colour, size-clarity and colour-clarity terms were the interactions contributing the most to the predictions made by the Random Forest model, aligning with expectations developed from the initial exploration of the dataset as well as the interactions of importance to the Linear Regression model built in section 7.2. Partial dependence plots for these interaction terms were also created, showing exactly how these interactions influenced the price predictions made with the Random Forest model. A Shapley values plot was generated for a random case in the dataset, showing how the individual values associated with the case affected the price prediction made for it. As mentioned previously, such a plot could be produced to accompany any predictions made if users which to produce a more detailed explanation of why an individual diamond is priced the way it is.

11 Conclusion and Limitations of the model

This report details the construction, evaluation and interpretation of a Random Forest model used to predict the price of a diamond based on six recorded variables. The dataset was cleaned to ensure it was appropriate for the construction of the model, and an initial analysis was undertaken to identify any relationships existing within the dataset. Two additional models, a Regression Tree and a Linear Regression model, were constructed to further explore the dataset as well as provide a basis for comparing the performance of the Random Forest model built. The construction of the Random Forest model is also described in this report, identifying how the hyperparameters used in the model were optimized for the diamond dataset.

After this, an evaluation of the model built was conducted. The model was able to accurately predict the prices of diamonds from the test dataset, indicating the model will be effective in predicting diamond prices for new data. The model performed better than the Regression Tree model, Linear Regression model and a Random Forest model with suboptimal hyperparameters, however it should be noted that these models still performed well, with the Pearson’s correlation coefficient between each of these model’s predicted prices for the test dataset and the observed prices being over 0.95. There was not a large difference between the Random Forest model’s performance for the training and test dataset, indicating that overfit is not an issue with the model.

In terms of model interpretation, the size parameter was found to be noticeably more important to the Random Forest model’s prediction of prices than the other variables considered. A diamond’s colour and clarity were also found to be important to the model’s predictions, with cut quality contributing somewhat while depth % and table % contributed very little. The interactions between size and colour, size and clarity, and colour and clarity were found to noticeably influence the prediction of a diamond’s price by the Random Forest model, while the other interaction terms contributing almost nothing to predictions made. The nature of the effect of each variable as well as the three most important interaction terms on the predicted prices were explored through the construction of partial dependence plots. A Shapley value plot was constructed for a random diamond showing how the value the diamond had for each recorded variable contributed to the predicted price made by the Random Forest model for said diamond.

In the sample, there were no diamonds with the clarity classification of FL, I2 or I3. Thus, it was not possible to develop a model which can accurately predict prices for diamonds of these kinds, and it is not recommended to use this model to attempt to predict the prices of said diamonds.

Furthermore, as with all models, values falling outside the ranges of the data present will not have been considered by this model. For example, the largest diamond size in the dataset was 388 mm3 while the smallest was 32 mm3. The model has not been built with diamonds outside of this range in mind. It is entirely possible that the relationship between size and price could be different outside of this range, and the model would not capture this change. This could lead to inaccurate price predictions for diamonds of this type. As a result, users should be aware of this fact when making predictions with this model. Extrapolation is the act of making estimations assuming the observed trends will continue and is generally best avoided for these reasons. That being said, due to the size of the dataset used to build this model, assuming the data accurately represents the population of diamonds, diamonds of this type should be very rare.

It should be noted that a model makes a prediction based on learned data. While the models assessed in this report were able to accurately predict diamond prices, even for data that had not been used to build the models, the outputs of these models are by no means a hard truth. There will almost certainly be real world cases where a diamond’s price does not match what this model predicts, likely due to a factor not considered by this model. For example, suppose a diamond’s price was high due to having some historical significance. By inputting the size, colour and other variables used to make predictions by these models, there is no way the model would be able to infer this historical significance, and thus the model would almost certainly undervalue the diamond in question. When this is the case, be aware that, while effective, the model does not have expert knowledge on diamond pricing. If a clear reason exists that is unrelated to the variables recorded in this dataset for a diamond’s price to be higher or lower, this will need to be accounted for manually. Calculating the Shapley values for cases where the predicted price seems inaccurate can grant insight into why the model gave the prediction it did for the case in question.

Topic of Choice: Extremely Randomized Trees

A Random Forest model was developed for the diamond dataset in order to predict the price of diamonds. However, Random Forests are one of many ensemble techniques that can be used for supervised learning. Here, the extremely randomized trees (or Extra Trees) ensemble technique proposed by Geurts, Ernst and Wehenkel (2006) is examined. This technique is similar to the Random Forest technique but with several key distinctions which are described below. After that, an Extra Trees model is built for the diamond data and compared to the Random Forest model created previously.

Random Forests and Extra Trees are two ensemble models that use randomization to reduce the variance of the trees the model is made from. Trees are high variance models and unstable, contributing to higher error rates (Geurts, Ernst and Wehenkel, 2006, Pg.4). By combining multiple trees with low covariance, techniques like Random Forest and Extra Trees reduce this variance, creating better performing models.

The main difference between Random Forests and Extra Trees is their method of selecting tree splits. Random Forests choose each split from a random subset of the observed variables, selecting the variable and split that result in the largest drop in impurity from the subset of variables. This results in more diverse trees being created, increasing the independence of these trees with the objective of improving the predictive performance of the ensemble. Extra Trees take this idea of randomness further, building trees where each split is selected completely at random. This creates trees that are even more diverse. However, each individual tree is itself a worse predictive model. Thus, an Extra Trees model will have a lower variance but a higher bias, both of which contribute to the error of the ensemble model generated. Ideally, the independence of individual trees created reduces the variance by enough to offset the increase in bias resulting from the individual trees being poorer models themselves (Geurts, Ernst and Wehenkel, 2006, Pg.6).

As well as this, Random Forests bootstrap the training data used to construct each tree, resulting in each tree being built from a slightly different subset of the data. This helps to further reduce the similarity of each tree, reducing variance. The cost of this is that each tree is created using less of the full training sample (a bootstrap sample that is the same size as the data it is created from will leave approximately 36% of cases unused). As a result, Random Forests suffer an increase in bias to reduce variance. Since Extra Trees have higher bias and lower variance, reducing the bias is more important. Thus, the full training sample is used when constructing trees (Geurts, Ernst and Wehenkel, 2006, Pg.6).

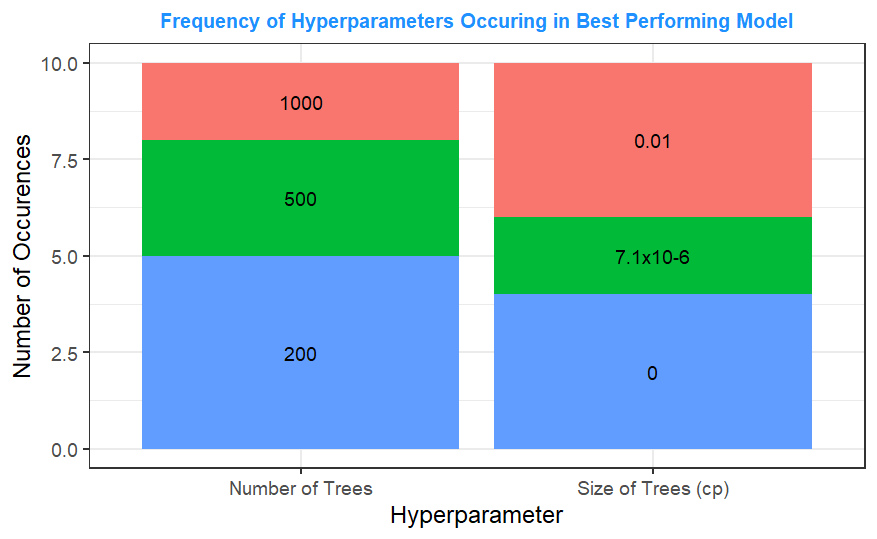
Geurts, Ernst and Wehenkel examined 24 datasets using several models including Random Forests and Extra Trees and found the predictive performance of these two models to be similar for the most part (2006). However, there are two major differences in the performance of these two techniques. Firstly, while Extra Trees tend to be larger than Random Forests, they can be constructed more quickly (Geurts, Ernst and Wehenkel, 2006, Pg.9). One of the downsides of Random Forests is the time taken to create a model. When speed is a concern, such as with real time data, Extra Trees can produce a result more quickly than a Random Forest. Secondly, for datasets with few relevant predictors and many irrelevant ones (or noisy data), Extra Trees generally outperform Random Forests. With noisy data, the trees making up both methods tend to overfit the data. However, due to the low covariance between the trees built using the Extra Trees method, these overfits tend to occur in differing ways. As a result, when combined in an ensemble, the effect of overfit was lessened (Geurts, Ernst and Wehenkel, 2006, Pg.20). This serves to make Extra Trees more accurate for noisy datasets.

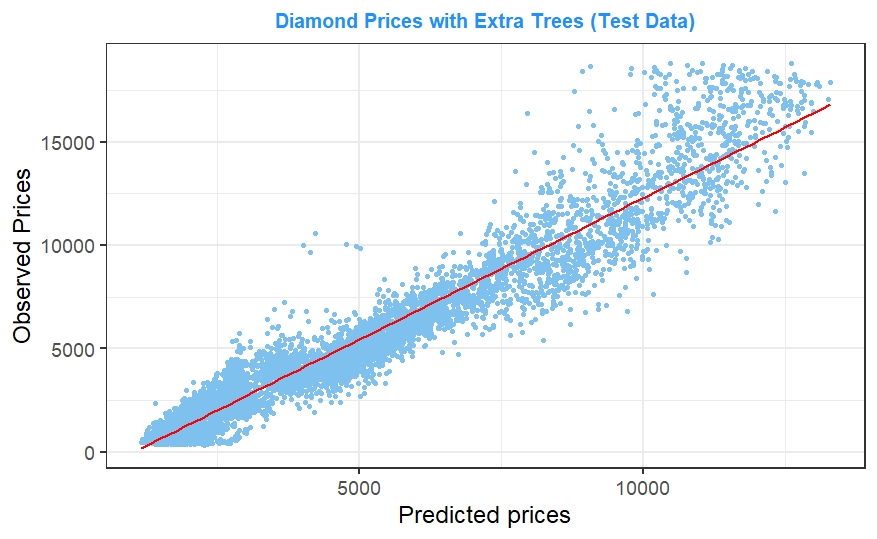
The diamond dataset does not contain a large amount of noise variables, consisting of only 6 variables. Of these 6, 3 contributed strongly to the prediction of prices for the Random Forest model built (section 10.1). Based on Guerts, Ernst and Wehenkel’s findings, we would expect the Extra Trees model to perform similarly to the Random Forest model constructed previously. However, due to the size of the dataset, one of the downsides of the Random Forest model is the time taken to construct a model. If the Extra Trees model performs similarly to the Random Forest model while taken significantly less time to create, it would likely be the preferred choice for predicting the price of diamonds.

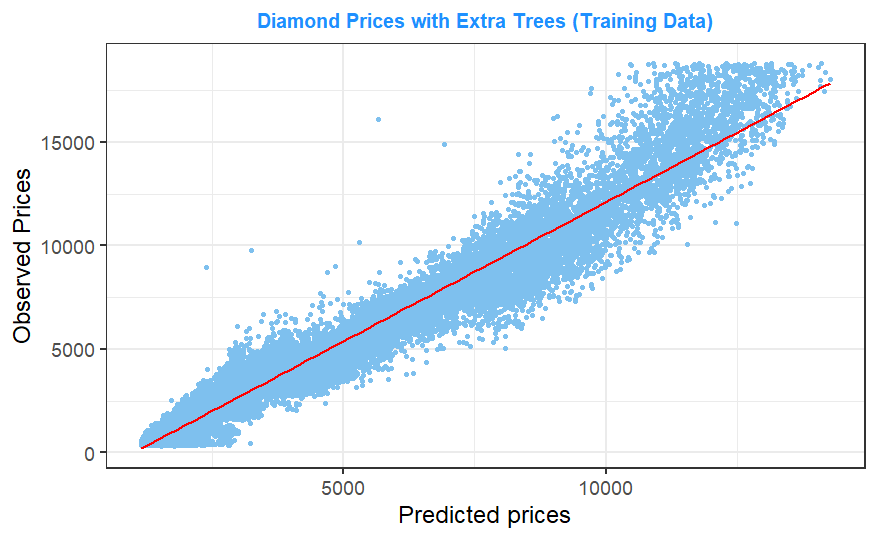
To determine the optimal Extra Trees model, several hyperparameters were examined as in section 8. Since the number of variables examined at each split had to be one and the entire set of training data was used to construct the model, the only hyperparameters to be considered were the number of trees built and the size of these trees. Models consisting of 200, 500 and 1000 trees with complexity parameters of 0 (a maximal tree), 7.1x10-6 (the ideal cut off for the individual Regression Tree constructed in section 7.1) and 0.01 (representing a smaller tree than the previous two) were assessed, resulting in nine models being considered. Once again, these models were built ten times due to the instability of trees, to ensure the model being selected consistently outperformed the other models.

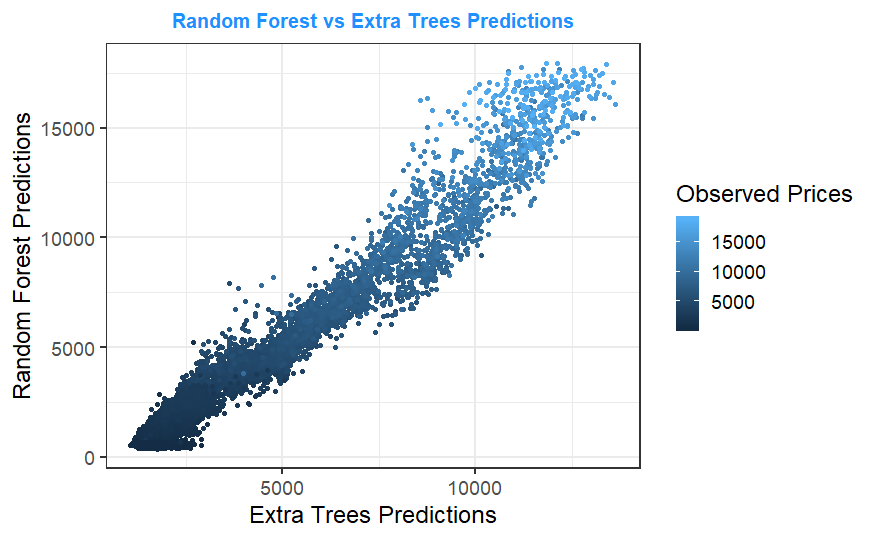
Since Extra Trees do not result in any out of bag cases and due to the time taken to construct each model, each of these nine models across each of the ten trials was constructed using 2000 random cases from the training data. Each model was then used to predict prices for the remaining 39,946 training cases. Model performance was then determined based on which model gave predictions that resulted in the lowest RMSE between the predicted and observed values. This allowed for the best model to be selected based on its prediction of data that was not used in the model’s construction, while still saving the test data for comparing the Extra Tree model’s performance to that of the Random Forest model. The performance of each of these models over the ten trials is available in appendix 4.

The optimal model was then constructed using the full training data. The optimal Extra Trees model was constructed using 200 maximal trees (or trees with a complexity parameter of 0). As indicated by the literature (Geurts, Ernst and Wehenkel, 2006, Pg.9), the Extra Trees model used larger trees than the Random Forest model of the same data, which had a complexity parameter of 7.1x10-6. Thus, the individual trees used in the Random Forest model were smaller.

To the right is a graph showing the number of times each value for a hyperparameter appeared in the best Extra Trees model built for each of the ten trials. In half of the trials, 200 trees were used to construct the optimal Extra Trees model, with 500 trees resulting in the best model three times and 1000 trees for two of the optimal models. In terms of tree size, a maximal tree was optimal four times, with trees with a complexity parameter cut-off at 7.1x10-6 being optimal twice and trees with a complexity parameter cut-off of 0.01 being optimal four times. Compared to the Random Forest model, the hyperparameter values determined to be optimal for the Extra Trees model did not consistently outperform the other options. That being said, the Extra Trees model using 200 maximal trees was the highest performing model for four of the ten trials, indicating that, when combined, these hyperparameters resulted in an Extra Trees model that consistently outperformed the other hyperparameter combinations considered.

To the right is a graph showing the predicted diamond prices produced by the Extra Trees model for the test data compared to the observed prices. The graph shows that the Extra Trees model’s predictions are not as accurate as those of the other models examined in this report (see Section 7 and 9). The RMSE of 1386 act as further evidence that this model underperforms when compared not only to the optimal Random Forest, but all the models built for the prediction of diamond prices (appendix 3). The Pearson’s correlation coefficient of 0.970 between observed and predicted prices was higher than that of the Linear Regression model. This is likely a result of the Linear Regression model only capturing linear relationships between variables and prices, leading to a greater correlation between the observed diamond prices and those predicted by the Extra Trees model, despite the difference between the values being higher.

To examine whether this worse performance was a result of overfit, the Extra Trees model’s performance for predicting the diamond prices of the training data was examined. The graph on the right shows this relationship. The Extra Trees model’s performance for the training data does not appear to be much greater than its performance on the test data. The Pearson’s correlation coefficient between the predicted and observed prices was 0.979 and the RMSE was 1272. This indicates that overfit in this model was not a major issue, and that, for the diamond dataset, an Extra Trees model is simply less effective at predicting prices when compared to other models.

Comparing the predictions of the Extra Trees model to those of the optimal Random Forest model built in section 8, it becomes clear that the two models differ in their predictions of diamond prices (see right). While the Pearson’s correlation coefficient between the two models’ predictions is high, at 0.978, a RMSE of 1227 indicates a comparatively large difference between the prices predicted by these two models. Despite being highly correlated, the difference between the predicted values is large enough to indicate that the Extra Trees model’s predictions are quite different to those made by the Random Forest model.

In their paper, Geurts, Ernst and Wehenkel found Extra Trees models to either outperform or perform similarly to Random Forest models (2006). However, for the diamond dataset, the Extra Trees model underperformed compared to the Random Forest model. On top of this, the predictions made by the two models were quite different. This indicates that Extra Trees do not always either outperform or perform similarly to Random Forests. Such a situation that was not explored in this paper. However, they also noted that Extra Trees models, by selecting random splits in their trees, increased the bias of their trees to reduce covariance (2006, Pg.6). The underperformance of the Extra Trees model built for the diamond data indicates that this trade-off was ultimately worse for this dataset, as the reduction in error rate from reducing covariance did not appear to offset the increase in error rate from the greater bias.

Further evidence of this comes from the optimal hyperparameters used in the Random Forest model. The optimal Random Forest model considered 6 of the 7 potential independent variables at each split. The Random Forest models which considered more variables per split outperformed those that considered less. Considering more variables per split reduces the bias of the individual trees in a Random Forest but increases their covariance. Thus, reducing bias and increasing covariance resulted in a better performing Random Forest model. As a result, a model like Extra Trees, which effectively only considered 1 variable per split, was not likely to outperform the Random Forest model.

In terms of time taken to construct the models, the Extra Trees model took significantly less time to construct. The time taken to construct each model was recorded using the “Sys.time()” functionin the R programming language. The Random Forest model for the entire dataset took 90 minutes to construct, while the Extra Trees model took just over 3 minutes. This aligns with Geurts, Ernst and Wehenkel’s findings. Had the Extra Trees model performed as well as the Random Forest, an argument could be made for using it due to the greatly reduced time taken to build the model. However, there was no indication by the data providers this would be a concern. Furthermore, both the Linear Regression and Regression Tree models took approximately 4 seconds to construct. On top of this, both models outperformed compared to the Extra Trees model when it came to predicting prices for the test cases. Thus, if the Random Forest model was found to be unsuitable due to time taken to construct it, a Linear Regression or Regression Tree model would still be preferable to an Extra Trees model, both in terms of accuracy and time taken to be built.

The performance of Extra Trees model seemed to improve when datasets contain a large number of noise variables, which is not the case for the diamond dataset. Geurts, Ernst and Wehenkel indicated an Extra Trees model would have a similar performance to a Random Forest model even if this was not the case (2006). Compared to the other models used to predict diamond prices, the Extra Trees model underperformed. This indicates that, without the correct conditions, Extra Trees models can underperform compared to Random Forests, and even more basic models. The reduction in time taken to create an Extra Trees model indicated by Geurts, Ernst and Wehenkel (2006, Pg.9) was observed for the diamond dataset. However, compared to simpler models such as a Regression Tree or Linear Regression model, both of which had a stronger performance than the Extra Trees model for this dataset, the Extra Trees model was still slower. If time taken to produce a model was a concern, either of these simpler models would be preferable for the diamond dataset.

For another dataset where Extra Trees performed as well as Random Forest and time taken to build a model was important, the Extra Trees model may be a more desirable chouce. However, there was no indication given that diamond prices would need to be predicted quickly, and the Extra Trees model underperformed compared both to the Random Forest model and other simpler models for this dataset. As a result, for the diamond data at least, there does not appear to be a situation where an Extra Trees model is preferable to a Random Forest or a simpler model.

Bibliography

- Geurts, P., Ernst, D. and Wehenkel, L. (2006). *Extremely randomized trees*. Machine Learning, 63(1), pp.3-42.

Appendices

Appendix 1: Near Zero Variance Test Values

|  |  |  |
| --- | --- | --- |
| Variables | Frequency Ratio | % of Unique Values |
| Carat | 5.8:5 | 0.506 |
| Cut | 7.8:5 | 0.009 |
| Colour | 5.8:5 | 0.013 |
| Clarity | 1.4:5 | 0.015 |
| Depth % | 5.2:5 | 0.341 |
| Table % | 5.1:5 | 0.236 |
| Price | 5.2:5 | 2.151 |
| Length | 5.1:5 | 1.025 |
| Width | 5:5 | 1.020 |
| Depth | 5.2:5 | 0.694 |

Appendix 2: Performance of Random Forest for each Combination of Hyperparameters: Mean of Squared Residuals

*\*The lowest MSR values for each trial are highlighted in green and the highest in red*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hyperparameter Combination | MSR Trial 1 | MSR Trial 2 | MSR Trial 3 | MSR Trial 4 | MSR Trial 5 | MSR Trial 6 | MSR Trial 7 | MSR Trial 8 | MSR Trial 9 | MSR Trial 10 |
| mtry= 2, ntree= 200, cp= 0, boot size= 660 | 1309711 | 1215890 | 1480802 | 1310238 | 1419636 | 1244057 | 1225625 | 1524176 | 1518264 | 1379843 |
| mtry= 3, ntree= 200, cp= 0, boot size= 660 | 888916 | 859539 | 1099033 | 936905 | 1018910 | 869616 | 847069 | 1069224 | 1030699 | 902304 |
| mtry= 6, ntree= 200, cp= 0, boot size= 660 | 693788 | 685114 | 839925 | 709458 | 769352 | 655458 | 635014 | 807483 | 818399 | 657816 |
| mtry= 2, ntree= 500, cp= 0, boot size= 660 | 1308965 | 1263498 | 1480560 | 1328034 | 1435698 | 1300700 | 1260131 | 1505945 | 1467109 | 1375745 |
| mtry= 3, ntree= 500, cp= 0, boot size= 660 | 888516 | 838265 | 1085589 | 957746 | 1007830 | 858222 | 859223 | 1032800 | 1039569 | 895577 |
| mtry= 6, ntree= 500, cp= 0, boot size= 660 | 677648 | 670611 | 849387 | 696449 | 775021 | 662491 | 638890 | 783489 | 803263 | 664639 |
| mtry= 2, ntree= 1000, cp= 0, boot size= 660 | 1323089 | 1196990 | 1460773 | 1397494 | 1406984 | 1257353 | 1266168 | 1478385 | 1489958 | 1353420 |
| mtry= 3, ntree= 1000, cp= 0, boot size= 660 | 887751 | 834131 | 1065274 | 949027 | 1008545 | 865906 | 843351 | 1042719 | 1039462 | 890035 |
| mtry= 6, ntree= 1000, cp= 0, boot size= 660 | 677778 | 675247 | 841683 | 707987 | 779579 | 668920 | 623379 | 777754 | 798579 | 650365 |
| mtry= 2, ntree= 200, cp= 7.1x10-6, boot size= 660 | 1360609 | 1226081 | 1521269 | 1428204 | 1400327 | 1337726 | 1265119 | 1449766 | 1563355 | 1386960 |
| mtry= 3, ntree= 200, cp= 7.1x10-6, boot size= 660 | 887001 | 846043 | 1120374 | 948733 | 998982 | 892235 | 869212 | 1038924 | 1033277 | 909783 |
| mtry= 6, ntree= 200, cp= 7.1x10-6, boot size= 660 | 706678 | 686223 | 853073 | 720730 | 798458 | 659967 | 626872 | 797791 | 794236 | 660569 |
| mtry= 2, ntree= 500, cp= 7.1x10-6, boot size= 660 | 1248822 | 1200820 | 1439103 | 1401552 | 1421458 | 1248682 | 1238598 | 1497541 | 1529376 | 1316747 |
| mtry= 3, ntree= 500, cp= 7.1x10-6, boot size= 660 | 881506 | 854577 | 1080707 | 954923 | 1011213 | 851021 | 848849 | 1048650 | 1062802 | 899222 |
| mtry= 6, ntree= 500, cp= 7.1x10-6, boot size= 660 | 677347 | 668765 | 839331 | 704384 | 781041 | 661742 | 642009 | 776456 | 791007 | 665446 |
| mtry= 2, ntree= 1000, cp= 7.1x10-6, boot size= 660 | 1300444 | 1225984 | 1492753 | 1346445 | 1424420 | 1244349 | 1264927 | 1511160 | 1478557 | 1338723 |
| mtry= 3, ntree= 1000, cp= 7.1x10-6, boot size= 660 | 889217 | 856255 | 1092513 | 934747 | 1007297 | 878683 | 848773 | 1049304 | 1036710 | 888072 |
| mtry= 6, ntree= 1000, cp= 7.1x10-6, boot size= 660 | 671281 | 674920 | 835132 | 705197 | 779423 | 667562 | 627402 | 780845 | 802196 | 665064 |
| mtry= 2, ntree= 200, cp= 0.01, boot size= 660 | 1308529 | 1212108 | 1514193 | 1383183 | 1445232 | 1253863 | 1324113 | 1528649 | 1496605 | 1354760 |
| mtry= 3, ntree= 200, cp= 0.01, boot size= 660 | 942292 | 897061 | 1056447 | 942654 | 1046887 | 869313 | 845321 | 1024117 | 1081805 | 933614 |
| mtry= 6, ntree= 200, cp= 0.01, boot size= 660 | 687536 | 678591 | 838455 | 700371 | 801018 | 672253 | 645086 | 788192 | 802412 | 665983 |
| mtry= 2, ntree= 500, cp= 0.01, boot size= 660 | 1330869 | 1212412 | 1515568 | 1364312 | 1434554 | 1257286 | 1265870 | 1468178 | 1512135 | 1390649 |
| mtry= 3, ntree= 500, cp= 0.01, boot size= 660 | 897135 | 839478 | 1054662 | 916384 | 1004703 | 868070 | 860616 | 1050951 | 1084731 | 907301 |
| mtry= 6, ntree= 500, cp= 0.01, boot size= 660 | 670498 | 663013 | 839087 | 713337 | 783894 | 665635 | 637055 | 778761 | 793597 | 663984 |
| mtry= 2, ntree= 1000, cp= 0.01, boot size= 660 | 1301967 | 1199625 | 1481081 | 1357112 | 1428913 | 1223585 | 1279766 | 1488452 | 1529559 | 1327615 |
| mtry= 3, ntree= 1000, cp= 0.01, boot size= 660 | 888491 | 860620 | 1088216 | 936517 | 999231 | 860600 | 849182 | 1055254 | 1057260 | 905549 |
| mtry= 6, ntree= 1000, cp= 0.01, boot size= 660 | 674011 | 675535 | 840945 | 710870 | 777135 | 663492 | 628404 | 780618 | 791186 | 652130 |
| mtry= 2, ntree= 200, cp= 0, boot size= 1320 | 1034953 | 935324 | 1223197 | 1068667 | 1118565 | 1009803 | 982024 | 1259154 | 1178720 | 1128708 |
| mtry= 3, ntree= 200, cp= 0, boot size= 1320 | 680558 | 682456 | 879002 | 771354 | 829859 | 698143 | 646472 | 811137 | 862589 | 722268 |
| mtry= 6, ntree= 200, cp= 0, boot size= 1320 | 565202 | 590381 | 766776 | 647534 | 683618 | 611160 | 511518 | 648376 | 719992 | 538396 |
| mtry= 2, ntree= 500, cp= 0, boot size= 1320 | 1012382 | 948412 | 1235333 | 1067208 | 1145000 | 998997 | 959494 | 1187964 | 1247586 | 1049492 |
| mtry= 3, ntree= 500, cp= 0, boot size= 1320 | 690555 | 690003 | 879573 | 745318 | 830585 | 718291 | 618640 | 833800 | 861413 | 701290 |
| mtry= 6, ntree= 500, cp= 0, boot size= 1320 | 568791 | 586263 | 766097 | 630344 | 666586 | 598531 | 513046 | 650953 | 697096 | 541483 |
| mtry= 2, ntree= 1000, cp= 0, boot size= 1320 | 1020576 | 963162 | 1213522 | 1048166 | 1131769 | 977831 | 944743 | 1194904 | 1228801 | 1016107 |
| mtry= 3, ntree= 1000, cp= 0, boot size= 1320 | 690851 | 685967 | 889813 | 757411 | 832092 | 716998 | 631352 | 810199 | 850969 | 695442 |
| mtry= 6, ntree= 1000, cp= 0, boot size= 1320 | 571736 | 581487 | 763995 | 629919 | 667135 | 596711 | 510276 | 646674 | 705053 | 536898 |
| mtry= 2, ntree= 200, cp= 7.1x10-6, boot size= 1320 | 1078076 | 1002506 | 1212651 | 1100282 | 1164870 | 1023617 | 947457 | 1153657 | 1234271 | 1030109 |
| mtry= 3, ntree= 200, cp= 7.1x10-6, boot size= 1320 | 699323 | 700541 | 868480 | 760498 | 856273 | 704212 | 630746 | 843086 | 884327 | 736505 |
| mtry= 6, ntree= 200, cp= 7.1x10-6, boot size= 1320 | 573022 | 584522 | 779234 | 637996 | 678759 | 598288 | 516064 | 643053 | 706513 | 532919 |
| mtry= 2, ntree= 500, cp= 7.1x10-6, boot size= 1320 | 1001636 | 963791 | 1216780 | 1111900 | 1134215 | 1014706 | 979079 | 1139109 | 1211129 | 1050822 |
| mtry= 3, ntree= 500, cp= 7.1x10-6, boot size= 1320 | 711121 | 690452 | 884633 | 759575 | 824420 | 716822 | 642645 | 807139 | 873778 | 685670 |
| mtry= 6, ntree= 500, cp= 7.1x10-6, boot size= 1320 | 567465 | 583126 | 764719 | 636633 | 675043 | 597920 | 521677 | 647444 | 689996 | 535906 |
| mtry= 2, ntree= 1000, cp= 7.1x10-6, boot size= 1320 | 1017067 | 942941 | 1205728 | 1063175 | 1133397 | 1000162 | 945279 | 1185657 | 1229465 | 1032982 |
| mtry= 3, ntree= 1000, cp= 7.1x10-6, boot size= 1320 | 704722 | 683059 | 878154 | 754527 | 826149 | 705198 | 631155 | 809570 | 854585 | 705345 |
| mtry= 6, ntree= 1000, cp= 7.1x10-6, boot size= 1320 | 565569 | 575291 | 760563 | 632013 | 666642 | 597702 | 513419 | 649068 | 698573 | 534856 |
| mtry= 2, ntree= 200, cp= 0.01, boot size= 1320 | 1067293 | 963287 | 1302750 | 1138085 | 1127648 | 995840 | 978538 | 1203449 | 1190818 | 1073581 |
| mtry= 3, ntree= 200, cp= 0.01, boot size= 1320 | 683435 | 677129 | 890755 | 753063 | 844717 | 731148 | 621260 | 841771 | 853678 | 715128 |
| mtry= 6, ntree= 200, cp= 0.01, boot size= 1320 | 572328 | 579772 | 764803 | 632338 | 657977 | 596258 | 533592 | 654619 | 702247 | 535649 |
| mtry= 2, ntree= 500, cp= 0.01, boot size= 1320 | 1014175 | 993911 | 1244499 | 1092922 | 1125875 | 969131 | 947423 | 1175145 | 1182953 | 1032353 |
| mtry= 3, ntree= 500, cp= 0.01, boot size= 1320 | 693294 | 671662 | 896308 | 748809 | 831192 | 709547 | 636236 | 828784 | 850516 | 708467 |
| mtry= 6, ntree= 500, cp= 0.01, boot size= 1320 | 557340 | 589754 | 766743 | 624144 | 674072 | 594256 | 514705 | 637826 | 700156 | 537420 |
| mtry= 2, ntree= 1000, cp= 0.01, boot size= 1320 | 1007519 | 939726 | 1189788 | 1064120 | 1146219 | 993816 | 955802 | 1188208 | 1238756 | 1046662 |
| mtry= 3, ntree= 1000, cp= 0.01, boot size= 1320 | 689957 | 678381 | 881253 | 749195 | 831808 | 705573 | 624313 | 818922 | 856919 | 695520 |
| mtry= 6, ntree= 1000, cp= 0.01, boot size= 1320 | 567010 | 581375 | 766898 | 623481 | 670650 | 596978 | 511273 | 645892 | 699850 | 530974 |
| mtry= 2, ntree= 200, cp= 0, boot size= 2000 | 951232 | 901180 | 1147771 | 944602 | 1103612 | 946995 | 875576 | 1096593 | 1148558 | 999167 |
| mtry= 3, ntree= 200, cp= 0, boot size= 2000 | 620071 | 654873 | 854221 | 700374 | 797137 | 680951 | 584221 | 747315 | 809031 | 635701 |
| mtry= 6, ntree= 200, cp= 0, boot size= 2000 | 560839 | 571638 | 770892 | 624325 | 657132 | 596415 | 491200 | 617678 | 679577 | 491578 |
| mtry= 2, ntree= 500, cp= 0, boot size= 2000 | 897333 | 870306 | 1114339 | 991947 | 1034964 | 921541 | 856649 | 1053540 | 1123790 | 918486 |
| mtry= 3, ntree= 500, cp= 0, boot size= 2000 | 609438 | 633440 | 828602 | 683923 | 774445 | 667473 | 573314 | 756098 | 804279 | 628463 |
| mtry= 6, ntree= 500, cp= 0, boot size= 2000 | 541103 | 564439 | 763781 | 612525 | 648631 | 587201 | 494294 | 612253 | 684140 | 491514 |
| mtry= 2, ntree= 1000, cp= 0, boot size= 2000 | 880986 | 853629 | 1090616 | 964496 | 1037419 | 905033 | 841221 | 1053012 | 1090704 | 937019 |
| mtry= 3, ntree= 1000, cp= 0, boot size= 2000 | 626260 | 635794 | 810823 | 700746 | 767657 | 663639 | 566816 | 725850 | 791201 | 621740 |
| mtry= 6, ntree= 1000, cp= 0, boot size= 2000 | 544127 | 554468 | 775038 | 613405 | 646879 | 586627 | 491983 | 609922 | 677957 | 493902 |
| mtry= 2, ntree= 200, cp= 7.1x10-6, boot size= 2000 | 938133 | 883257 | 1110096 | 966897 | 1110642 | 920989 | 836838 | 1041187 | 1161682 | 1041355 |
| mtry= 3, ntree= 200, cp= 7.1x10-6, boot size= 2000 | 634801 | 647421 | 814059 | 695459 | 748302 | 671741 | 581529 | 739170 | 815496 | 631382 |
| mtry= 6, ntree= 200, cp= 7.1x10-6, boot size= 2000 | 552790 | 570232 | 763145 | 621506 | 652020 | 596155 | 502944 | 621435 | 692646 | 498868 |
| mtry= 2, ntree= 500, cp= 7.1x10-6, boot size= 2000 | 871238 | 884644 | 1081375 | 976810 | 1043442 | 888241 | 817997 | 1092103 | 1110012 | 967507 |
| mtry= 3, ntree= 500, cp= 7.1x10-6, boot size= 2000 | 618072 | 636222 | 801027 | 696468 | 764959 | 664067 | 568317 | 726735 | 789039 | 606359 |
| mtry= 6, ntree= 500, cp= 7.1x10-6, boot size= 2000 | 545313 | 564044 | 772039 | 607098 | 648540 | 584644 | 499701 | 609119 | 683597 | 488040 |
| mtry= 2, ntree= 1000, cp= 7.1x10-6, boot size= 2000 | 893391 | 863209 | 1086768 | 933321 | 1036593 | 920156 | 838226 | 1084128 | 1078925 | 936668 |
| mtry= 3, ntree= 1000, cp= 7.1x10-6, boot size= 2000 | 629931 | 628260 | 816781 | 688826 | 764961 | 663304 | 561074 | 740954 | 787562 | 613790 |
| mtry= 6, ntree= 1000, cp= 7.1x10-6, boot size= 2000 | 548700 | 559051 | 762926 | 623215 | 640548 | 590637 | 492631 | 611680 | 676211 | 490155 |
| mtry= 2, ntree= 200, cp= 0.01, boot size= 2000 | 952911 | 903707 | 1114046 | 967338 | 1070685 | 963651 | 884703 | 1091396 | 1109727 | 940639 |
| mtry= 3, ntree= 200, cp= 0.01, boot size= 2000 | 660350 | 638931 | 852316 | 709782 | 803744 | 643538 | 575767 | 748871 | 846455 | 621335 |
| mtry= 6, ntree= 200, cp= 0.01, boot size= 2000 | 559245 | 574684 | 781186 | 628134 | 639519 | 592591 | 499701 | 620836 | 677855 | 504909 |
| mtry= 2, ntree= 500, cp= 0.01, boot size= 2000 | 869776 | 858119 | 1105120 | 990604 | 1042900 | 908189 | 829230 | 1061971 | 1112662 | 932073 |
| mtry= 3, ntree= 500, cp= 0.01, boot size= 2000 | 621293 | 620858 | 820676 | 688197 | 761602 | 664793 | 561601 | 731169 | 794757 | 625858 |
| mtry= 6, ntree= 500, cp= 0.01, boot size= 2000 | 552692 | 552905 | 763064 | 613304 | 645841 | 589641 | 493615 | 615287 | 682805 | 492863 |
| mtry= 2, ntree= 1000, cp= 0.01, boot size= 2000 | 909611 | 866900 | 1094373 | 952094 | 1029475 | 897918 | 842619 | 1048836 | 1096972 | 945323 |
| mtry= 3, ntree= 1000, cp= 0.01, boot size= 2000 | 622932 | 626881 | 803996 | 690636 | 766933 | 645775 | 554453 | 731298 | 791557 | 619798 |
| mtry= 6, ntree= 1000, cp= 0.01, boot size= 2000 | 544327 | 557148 | 770235 | 613107 | 646510 | 588575 | 491631 | 608966 | 678724 | 498000 |

*MSR = Mean of the Squared Residuals*

*mtry = Number of variables in each split*

*ntree = Number of trees*

*cp = Size of Trees*

*boot size = Sampling method*

Appendix 3: Pearson’s correlation coefficient and RMSE of optimal Random Forest, sub-optimal Random Forest, Regression Tree, Linear Regression and Extra Trees.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Extra Trees | | Linear Regression | | Regression Tree | | RandomoForest (worst hyperparameters) | | RandomoForest (Best hyperparameters) | |
| r | **RMSE** | **r** | **RMSE** | **r** | **RMSE** | **r** | **RMSE** | **r** | **RMSE** |
| 0.979 | 1272 | 0.967 | 985 | 0.988 | 593 | 0.989 | 579 | 0.991 | 525 |

*r = Pearson’s Correlation Coefficient*

*RMSE = Root-Mean-Square Error*

Appendix 4: Performance of Extra Trees for each Combination of Hyperparameters: Root-Mean-Square Error

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hyperparameter Combination | RMSE Trial 1 | RMSE Trial 2 | RMSE Trial 3 | RMSE Trial 4 | RMSE Trial 5 | RMSE Trial 6 | RMSE Trial 7 | RMSE Trial 8 | RMSE Trial 9 | RMSE Trial 10 |
| 200 trees, 0 CP | 1555 | 1562 | 1562 | 1487 | 1585 | 1534 | 1599 | 1515 | 1552 | 1584 |
| 500 trees, 0 CP | 1574 | 1620 | 1559 | 1589 | 1565 | 1570 | 1564 | 1557 | 1574 | 1558 |
| 1000 trees, 0 CP | 1583 | 1558 | 1580 | 1565 | 1551 | 1591 | 1577 | 1531 | 1589 | 1562 |
| 200 trees, 7.1x10-6 CP | 1615 | 1562 | 1573 | 1597 | 1553 | 1583 | 1590 | 1549 | 1643 | 1575 |
| 500 trees, 7.1x10-6 CP | 1548 | 1565 | 1562 | 1577 | 1581 | 1542 | 1552 | 1573 | 1569 | 1582 |
| 1000 trees, 7.1x10-6 CP | 1566 | 1570 | 1563 | 1575 | 1590 | 1561 | 1595 | 1573 | 1594 | 1553 |
| 200 trees, 0.01 CP | 1626 | 1589 | 1611 | 1583 | 1612 | 1553 | 1532 | 1578 | 1626 | 1642 |
| 500 trees, 0.01 CP | 1551 | 1561 | 1544 | 1615 | 1527 | 1603 | 1591 | 1560 | 1580 | 1570 |
| 1000 trees, 0.01 CP | 1583 | 1553 | 1560 | 1536 | 1550 | 1575 | 1560 | 1584 | 1558 | 1604 |

*RMSE = Root-Mean-Square Error*

Lowest RMSE for each trial has been highlighted

Appendix 5: R Script used for the analysis

#Set working directory

setwd("C:/Users/pie93/Desktop/Data Analytics/Assignment")

#Load environment

load('myEnvironment.RData')

theme\_set(theme\_bw(base\_size=18))

rmse = function(m, o){

sqrt(mean((m - o)^2))

}

#Libraries Used

library(VIM)

library(caret)

library(janitor)

library(rpart) #Classification Tree

library(outliers)

library(ggrepel)

library(randomForest)

library(randomForestExplainer)

library(e1071)

library(iml)

library(devtools)

library(pdp)

#Save Environment

save.image(file='myEnvironment.RData')

#Import Dataset

diamond<-read.csv("diamonddata.csv",header = TRUE)

#Exploration of Data

#Ensure Variables are Stored Correctly

summary(diamond)

str(diamond)

#Check for Duplicate Entries

chdup<-duplicated(diamond)

table(chdup)

#No duplicate entries found

#Table %

length(which(diamond$table%%1 == 0))

#Check for Missing Data

oaggr<- aggr(diamond)

summary(oaggr)

#No missing data

#Check for 0s in carat weight

which(diamond$carat < 0.000001)

which(diamond$depth < 0.000001)

which(diamond$table < 0.000001)

#Check for 0s in size dimensions

which(diamond$x < 0.000001)

which(diamond$y < 0.000001)

length(which(diamond$z < 0.000001))

tabyl(which(diamond$x < 0.000001), which(diamond$y < 0.000001), which(diamond$z < 0.000001))

diamond[which(diamond$z < 0.000001),]

#Remove values

diamond <- diamond[-which(diamond$z < 0.000001),]

#Check for near zero variance in any variables

x<-nearZeroVar(diamond,saveMetrics=TRUE)

x

#No variables with a large proportion of identical values

#Missing Classifications

levels(diamond$clarity)

#Store data with outliers in case

diamondWithOutliers <- diamond

#Outliers

summary(diamond$carat)

#Carat

id1 <- boxplot.stats(diamond$carat, coef = 2)

diamond <- diamond[-which(diamond$carat > id1$stats[5]),]

# No. of outliers: 250

#None below lower cutoff

#Depth

id1 <- boxplot.stats(diamond$depth, coef = 2)

diamond <- diamond[-which(diamond$depth > id1$stats[5]),]

diamond <- diamond[-which(diamond$depth < id1$stats[1]),]

# No. of outliers: 1130

#None below lower cutoff

#Table

id1 <- boxplot.stats(diamond$table, coef = 2)

diamond <- diamond[-which(diamond$table > id1$stats[5]),]

diamond <- diamond[-which(diamond$table < id1$stats[1]),]

# No. of outliers: 84

#X

id1 <- boxplot.stats(diamond$x, coef = 2)

# No. of outliers: 2

#None left

#Y

id1 <- boxplot.stats(diamond$y, coef = 2)

# No. of outliers: 3

diamond <- diamond[-which(diamond$y > id1$stats[5]),]

#None below lower cutoff

#Z

id1 <- boxplot.stats(diamond$z, coef = 1.5)

# No. of outliers: 29

diamond <- diamond[-which(diamond$z > id1$stats[5]),]

diamond <- diamond[-which(diamond$z < id1$stats[1]),]

#Examine Relationships

#Correlation between depth and depth %

#Depth vs Depth %

ggplot(data = diamond[-which(diamond$z > 30),], mapping = aes(x = depth, y = z)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Depth (mm)", x="Diamond Depth (%)") + ggtitle("Diamond Depth Versus Depth Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond$depth, diamond$z)

#Pearson correlation coefficient

#Suggests weak positive relationship, not enough to remove from study

#Depth % vs Depth/Length

ggplot(data = diamond[-which(diamond$z > 30),], mapping = aes(x = depth, y = z/x)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Depth/Diamond Length", x="Diamond Depth (%)") + ggtitle("Diamond Depth/Diamond Length Versus Depth Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond[-which(diamond$x < 0.000001),]$depth, diamond[-which(diamond$x < 0.000001),]$z/diamond[-which(diamond$x < 0.000001),]$x)

#Depth % vs Depth/Width

ggplot(data = diamond[-which(diamond$z > 30),], mapping = aes(x = depth, y = z/y)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Depth/Diaomnd Width", x="Diamond Depth (%)") + ggtitle("Diamond Depth/Diamond Width Versus Depth Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond$depth, diamond$z/diamond$y)

#Correlation between width and table %

#Width vs Table %

temp <- diamond[-which(diamond$y > 30),]

ggplot(data = temp[which(temp$table < 90),], mapping = aes(x = table, y = y)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Width (mm)", x="Diamond Table (%)") + ggtitle("Diamond Width Versus Table Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond$table, diamond$y)

#Pearson correlation coefficient

#Suggests weak positive relationship, not enough to remove from study

#Table % vs Width/Length

ggplot(data = diamond[-which(diamond$y > 30),], mapping = aes(x = table, y = y/x)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Width/Diamond Length ", x="Diamond Table (%)") + ggtitle("Diamond Width/Diamond Length Versus Table Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond[-which(diamond$x < 0.000001),]$table, diamond[-which(diamond$x < 0.000001),]$y/diamond[-which(diamond$x < 0.000001),]$x)

#Table % vs Width/Depth

ggplot(data = diamond, mapping = aes(x = table, y = y/z)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond Width/Diamond Depth", x="Diamond Table (%)") + ggtitle("Diamond Width/Diamond Depth Versus Table Percentage") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond[-which(diamond$z == 0),]$table, diamond[-which(diamond$z == 0),]$y/diamond[-which(diamond$z == 0),]$z)

#Price vs Carat

ggplot(data = diamond, mapping = aes(x = carat, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Carat weight") + ggtitle("Diamond Price Versus Carat Weight")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#With line of best fit

ggplot(data = diamond, mapping = aes(x = carat, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Carat weight") + ggtitle("Diamond Price Versus Carat Weight")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm')

cor(diamond$price, diamond$carat)

#Investigate the weird vertical lines at intervals 1, 1.5 and 2 (may be large no at these specific weights, but why is distribution so wide)

#Price vs Cut

ggplot(data = diamond, mapping = aes(cut, price, fill = cut)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Cut", y="Price of Diamond") + ggtitle("Diamond Price by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("Ideal", "Premium", "Very Good", "Good", "Fair"))

#Doesnt seem to be much correlation between cut and price, if anything slightly negative relationship

#Price vs Color

ggplot(data = diamond, mapping = aes(color, price, fill = color)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Colour", y="Price of Diamond") + ggtitle("Diamond Price by Colour")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#These seem to be negatively correlated with price in terms of what is supposedly best

#Price vs clarity

ggplot(data = diamond, mapping = aes(clarity, price, fill = clarity)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Clarity", y="Price of Diamond") + ggtitle("Diamond Price by Clarity")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

#General tend towards a negative relationship; as the clarity increases, the prices decreased.

#Price vs Depth

depthPlot <- ggplot(data = diamond, mapping = aes(x = depth, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond Depth Percentage") + ggtitle("Diamond Price Versus Depth Percentage")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

depthPlot + stat\_function(colour = "red", size = 0.9, fun = function(x) dnorm(x, mean = mean(diamond$depth), sd = sd(diamond$depth\*2)) \* max(diamond$price) \* 6)

cor(diamond$price, diamond$depth)

#Vast majority of values between 55 and 70 percent mark, says almost nothing about price

#Is there a case for removing this?

#Price vs table

tablePlot <- ggplot(data = diamond, mapping = aes(x = table, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond Table (%)") + ggtitle("Diamond Price Versus Table Percentage")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

tablePlot + stat\_function(colour = "red", size = 0.9, fun = function(x) dnorm(x, mean = mean(diamond$table), sd = sd(diamond$table\*1.5)) \* max(diamond$price) \* 9)

#Similar to depth, with one outlier at well over 90% depth, large lines occuring at regular intervals (percentages recorded to whole number?)

#Huge number of values at fixed intervals; were some diamonds rounded to nearest % when being meausured?

#Price vs x

ggplot(data = diamond, mapping = aes(x = x, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond length(mm)") + ggtitle("Diamond Price Versus Length")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Entries for 0 - how does a diamond have 0 length? Nothing between 0 and roughly 3.3

#Clear exponential growth in price as length increases

cor(diamond$price, diamond$x)

#Cor captures linear relationship, not necessarily exponential relationship

#Price vs y

ggplot(data = diamond, mapping = aes(x = y, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond width(mm)") + ggtitle("Diamond Price Versus Width")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Entries for 0 - how does a diamond have 0 width? Nothing between 0 and roughly 4/5

#Clear exponential growth in price as width increases

#One outlier with width near 60mm - shoudl this be removed?

cor(diamond$price, diamond$y)

#Cor captures linear relationship, not necessarily exponential relationship

#Price vs z

ggplot(data = diamond, mapping = aes(x = z, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond depth(mm)") + ggtitle("Diamond Price Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Entries for 0 - how does a diamond have 0 width? Nothing between 0 and roughly 4/5

#Clear exponential growth in price as depth increases

#One outlier with depth near 30mm - should this be removed?

cor(diamond$price, diamond$z)

#Cor captures linear relationship, not necessarily exponential relationship

#Does carat peak on price due to negative correlation with another important variable?

#Carat vs clarity

ggplot(data = diamond, mapping = aes(clarity, carat, fill = clarity)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Clarity", y="Carat Weight") + ggtitle("Diamond Carat by Clarity")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

#Very strong negative relationship is evident, alligning with price (In fact this may be what causes neg trend of clarity)

#Carat vs x

ggplot(data = diamond, mapping = aes(x = x, y = carat)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Carat Weight", x="Diamond length(mm)") + ggtitle("Carat of Diamond Versus Length")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Very clear exponential growth, again matching price

#Carat vs y

ggplot(data = diamond, mapping = aes(x = y, y = carat)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Carat Weight", x="Diamond width(mm)") + ggtitle("Carat of Diamond Versus Width")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Very clear exponential growth, again matching price

#Carat vs z

ggplot(data = diamond, mapping = aes(x = z, y = carat)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Carat Weight", x="Diamond depth(mm)") + ggtitle("Carat of Diamond Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Very clear exponential growth, again matching price, however there is several more outliers present

#Relationship between x, y and z

#X vs Y

ggplot(data = diamond, mapping = aes(x = x, y = y)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond width(mm)", x="Diamond length(mm)") + ggtitle("Diamond Length Versus Width")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + geom\_smooth(method='lm', color = "red")

cor(diamond$x, diamond$y)

#Extremely high correlation, good grounds for combining them

#X vs Z

ggplot(data = diamond, mapping = aes(x = x, y = z)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond depth(mm)", x="Diamond length(mm)") + ggtitle("Diamond Length Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(diamond$x, diamond$z)

#Extremely high correlation, good grounds for combining them

#Y VS Z

ggplot(data = diamond, mapping = aes(x = z, y = y)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond width(mm)", x="Diamond depth(mm)") + ggtitle("Diamond Depth Versus Width")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(diamond$z, diamond$y)

#Extremely high correlation, good grounds for combining them

#Derived variable: size

diamond$size <- diamond$x \* diamond$y \* diamond$z

diamondWithAllVariables <- diamond

diamond<- diamond[,-c(9,10,11)]

#Price vs Size

ggplot(data = diamond, mapping = aes(x = size, y = price)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Price of Diamond", x="Diamond size (mm^3)") + ggtitle("Diamond Price Versus Size")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

cor(diamond[,c(2,6,7,9)])

#Size vs Carat

ggplot(data = diamond, mapping = aes(x = size, y = carat)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Carat Weight of Diamond", x="Diamond size (mm^3)") + ggtitle("Carat Weight Versus Size")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(diamond$size, diamond$carat)

var(diamond$size)

var(diamond$carat)

#Remove carat weight

diamond <- diamond[,-2]

#Cut: Why is it uncorrelated to price

#VS Colour

table(diamond$cut, diamond$color)

#VS Clarity

table(diamond$cut, diamond$clarity)

#VS Depth

ggplot(data = diamond, mapping = aes(cut, depth, fill = cut)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Cut", y="Diamond Depth") + ggtitle("Diamond Depth by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("Fair","Good","Very Good","Premium", "Ideal"))

#Negative relationship

#VS Table

ggplot(data = diamond, mapping = aes(cut, table, fill = cut)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Cut", y="Diamond Table") + ggtitle("Diamond Table by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("Fair","Good","Very Good","Premium", "Ideal"))

#VS Size

ggplot(data = diamond, mapping = aes(cut, size, fill = cut)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Cut", y="Diamond Size") + ggtitle("Diamond Size by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("Fair","Good","Very Good","Premium", "Ideal"))

#Color

#VS Clarity

table(diamond$color, diamond$clarity)

#VS Depth

ggplot(data = diamond, mapping = aes(color, depth, fill = color)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Depth", y="Price of Diamond") + ggtitle("Diamond Price by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#VS Table

ggplot(data = diamond, mapping = aes(color, table, fill = color)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Table", y="Price of Diamond") + ggtitle("Diamond Price by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#VS Size

ggplot(data = diamond, mapping = aes(color, size, fill = color)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Colour", y="Size of Diamond") + ggtitle("Diamond Size by Colour")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Positive relationship between colour and size

#Clarity

#VS Depth

ggplot(data = diamond, mapping = aes(clarity, depth, fill = clarity)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Clarity", y="Diamond Depth") + ggtitle("Diamond Depth by Clarity")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

#Variation of clarity tended to decrease for higher clarities

#VS Table

ggplot(data = diamond, mapping = aes(clarity, table, fill = clarity)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Table", y="Price of Diamond") + ggtitle("Diamond Price by Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

#VS Size

ggplot(data = diamond, mapping = aes(clarity, size, fill = clarity)) + geom\_boxplot() + stat\_boxplot(geom="errorbar") +labs(x="Diamond Clarity", y="Diamond Size") + ggtitle("Diamond Size by Clarity")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

#Diamonds with better clarity tended to be smaller

#Depth

#VS Table

ggplot(data = diamond, mapping = aes(x = depth, y = table)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond depth(mm)", x="Diamond length(mm)") + ggtitle("Diamond Length Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

#VS Size

ggplot(data = diamond, mapping = aes(x = depth, y = size)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond depth(mm)", x="Diamond length(mm)") + ggtitle("Diamond Length Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

#Table

#VS Size

ggplot(data = diamond, mapping = aes(x = table, y = size)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Diamond depth(mm)", x="Diamond length(mm)") + ggtitle("Diamond Length Versus Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

#CreateDataPartition, removing ID

set.seed(5)

?createDataPartition

inTrain<-createDataPartition(y=diamond[,-1]$price, p=0.8,list=FALSE)

diamondTrain<-diamond[inTrain,-1]

diamondTest<-diamond[-inTrain,-1]

prop.table(table(diamond$color))

prop.table(table(diamondTrain$color))

prop.table(table(diamondTest$color))

#Analysis using trees

start\_time <- Sys.time()

fitTree = rpart(price ~.,data=diamondTrain, cp=0, method = "anova")

end\_time <- Sys.time()

timeRegTree <- end\_time - start\_time

fitTree = rpart(price ~.,data=diamondTrain, cp=0, method = "anova")

#Determine cutoff

cpt<-fitTree$cptable # where fit is the name of the object created by rpart.

i.min<-which.min(cpt[,4]) #Min xerror

i.se <-which.min (abs(cpt[,4]-(cpt[i.min,4]+cpt[i.min,5]))) #xerror closest to min xerror+its SE

cp.best<-cpt[i.se,1] #Best cp = cp of xerror equal to xerrormin - SE

fitTree1<-prune.rpart(fitTree,cp.best, method = "anova")

printcp(fitTree1)

#Variable Importance

#Size

fitTree1$variable.importance[1]/sum(fitTree1$variable.importance)

fitTree1$variable.importance[2]/sum(fitTree1$variable.importance)

fitTree1$variable.importance[3]/sum(fitTree1$variable.importance)

fitTree1$variable.importance[4]/sum(fitTree1$variable.importance)

fitTree1$variable.importance[5]/sum(fitTree1$variable.importance)

fitTree1$variable.importance[6]/sum(fitTree1$variable.importance)

#Model Effectiveness

treePred <- predict(fitTree1, diamondTest)

treeValues <- data.frame(diamondTest$price)

treeValues$observed <- treeValues$diamondTest.price

treeValues$predicted <- treePred

ggplot(data = treeValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Regression Tree")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(treeValues$observed, treeValues$predicted)

rmse(treeValues$predicted, treeValues$observed)

#Linear Regression Model

start\_time <- Sys.time()

linreg <- glm(price~.^2, family = gaussian, diamondTrain)

end\_time <- Sys.time()

timeLinReg <- end\_time - start\_time

linreg <- glm(price~.^2, family = gaussian, diamondTrain)

summary(linreg)

#Refine (remove all with >5% P-value)

formula <-price~cut + color + clarity + cut \* color + cut \* clarity + cut \* depth + cut \* table + cut \* size + color \* clarity + color \* depth + color \* table + color \* size + clarity \* table + clarity \* size + table \* size

linreg2 <- glm(formula, family = gaussian, diamondTrain)

summary(linreg2)

#Model Effectiveness

lregPred <- predict(linreg2, diamondTest)

lregValues <- data.frame(diamondTest$price)

lregValues$observed <- lregValues$diamondTest.price

lregValues$predicted <- lregPred

ggplot(data = lregValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Linear Regression")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(lregValues$observed, lregValues$predicted)

rmse(lregValues$observed, lregValues$predicted)

#Performance so high due to linearity of relationships between price and othe rvariables of importance

#Test their contribution to the prediciton of the interaction terms

#price~cut + color + clarity + cut \* color + cut \* clarity + cut \* depth + cut \* table + cut \* size + color \* clarity + color \* depth + color \* table + color \* size + clarity \* table + clarity \* size + table \* size'

formula2 <-price~cut + color + clarity + cut \* color + cut \* clarity + cut \* depth + cut \* table + cut \* size + color \* clarity + color \* depth + color \* table + color \* size + clarity \* table + clarity \* size

linregTemp <- glm(formula2, family = gaussian, diamondTrain)

#Model Effectiveness

lregPred2 <- predict(linregTemp, diamondTest)

lregValues2 <- data.frame(diamondTest$price)

lregValues2$observed <- lregValues2$diamondTest.price

lregValues2$predicted <- lregPred2

rmse(lregValues2$observed, lregValues2$predicted)

#Tuning random forest

#Build 10 different random samples of 2000 from the training data, create RF of each hyperparameter combination from these and determine best performance using OoB cases

subDiamondTrain <- diamondTrain[sample(nrow(diamondTrain), 2000), ]

grid <- expand.grid(mtry = c(2,sqrt(7),6), ntree = c(200, 500, 1000), cp = c(0.000, cp.best, 0.01), boot.size = c(0.33 \* nrow(subDiamondTrain), 0.66 \* nrow(subDiamondTrain), 1 \* nrow(subDiamondTrain)))

msr <- matrix(0L, nrow = 81, ncol = 10)

colnames(msr) <- c("MSR1","MSR2","MSR3","MSR4","MSR5","MSR6","MSR7","MSR8","MSR9", "MSR10")

varExp <- matrix(0L, nrow = 81, ncol = 10)

colnames(varExp) <- c("varExp1","varExp2","varExp3","varExp4","varExp5","varExp6","varExp7","varExp8","varExp9", "varExp10")

best <- data.frame( bestMSR = numeric(10), bestVarExp = numeric(10))

for(j in 1:10)

{

subDiamondTrain <- diamondTrain[sample(nrow(diamondTrain), 2000), ]

perf <- grid

perf$msr <- 0

perf$varExp <- 0

for (i in 1:nrow(perf))

{

fitTemp <- randomForest(price~.,data=subDiamondTrain, mtry = perf$mtry[i], ntree=perf$ntree[i], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = perf$cp[i]), sampsize = perf$boot.size[i])

perf$msr[i] <- fitTemp$mse[fitTemp$ntree]

perf$varExp[i] <- fitTemp$rsq[fitTemp$ntree]

msr[i,j] <- fitTemp$mse[fitTemp$ntree]

varExp[i,j] <- fitTemp$rsq[fitTemp$ntree]

}

#Determine best hyperparameters

best$bestMSR[j] <- which(perf$msr == min(perf$msr))

best$bestVarExp[j] <-which(perf$varExp == max(perf$varExp))

}

write.csv(msr, "C:/Users/pie93/Desktop/Data Analytics/Assignment/msrData.csv")

write.csv(grid, "C:/Users/pie93/Desktop/Data Analytics/Assignment/grid.csv")

write.table(varExp, "C:/Users/pie93/Desktop/Data Analytics/Assignment/varExpData.txt", sep="\t")

write.table(best, "C:/Users/pie93/Desktop/Data Analytics/Assignment/bestData.txt", sep="\t")

best <- read.table("C:/Users/pie93/Desktop/Data Analytics/Assignment/bestData.txt", sep="\t")

msr <- read.table("C:/Users/pie93/Desktop/Data Analytics/Assignment/msrData.txt", sep="\t")

for(i in 1:10)

{

print(grid[which(msr[,i] == max(msr[,i])),])

}

#Occurance in best performing model

bestPerfVar <- data.frame(NumberOfVariables = c("2","3","6"), AverageMSE = c(mean(unlist(msr[which(grid$mtry == 2),], use.names = F)),mean(unlist(msr[which(grid$mtry == grid$mtry[2]),], use.names = F)),mean(unlist(msr[which(grid$mtry == 6),], use.names = F))), VarMSE = c(sd(unlist(msr[which(grid$mtry == 2),], use.names = F)),sd(unlist(msr[which(grid$mtry == grid$mtry[2]),], use.names = F)),sd(unlist(msr[which(grid$mtry == 6),], use.names = F))))

bestPerfTreeNo <- data.frame(NumberOfTrees = c(200, 500, 1000), AverageMSE = c(mean(unlist(msr[which(grid$ntree == 200),], use.names = F)),mean(unlist(msr[which(grid$ntree == 500),], use.names = F)),mean(unlist(msr[which(grid$ntree == 1000),], use.names = F))), VarMSE = c(sd(unlist(msr[which(grid$ntree == 200),], use.names = F)),sd(unlist(msr[which(grid$ntree == 500),], use.names = F)),sd(unlist(msr[which(grid$ntree == 1000),], use.names = F))))

bestPerfSampSize <- data.frame(SampleSize = c("33%", "66%", "100%"), AverageMSE = c(mean(unlist(msr[which(grid$boot.size == 660),], use.names = F)),mean(unlist(msr[which(grid$boot.size == 1320),], use.names = F)),mean(unlist(msr[which(grid$boot.size == 2000),], use.names = F))), VarMSE = c(sd(unlist(msr[which(grid$boot.size == 660),], use.names = F)),sd(unlist(msr[which(grid$boot.size == 1320),], use.names = F)),sd(unlist(msr[which(grid$boot.size == 2000),], use.names = F))))

bestPerfTreeSize <- data.frame(TreeSize = c("0","7.1x10-6 ","0.01"), AverageMSE = c(mean(unlist(msr[which(grid$cp == grid$cp[1]),], use.names = F)),mean(unlist(msr[which(grid$cp == grid$cp[10]),], use.names = F)),mean(unlist(msr[which(grid$cp == grid$cp[81]),], use.names = F))), VarMSE = c(sd(unlist(msr[which(grid$cp == grid$cp[1]),], use.names = F)),sd(unlist(msr[which(grid$cp == grid$cp[10]),], use.names = F)),sd(unlist(msr[which(grid$cp == grid$cp[81]),], use.names = F))))

ggplot(bestPerfVar, aes(x = NumberOfVariables, y = AverageMSE)) + geom\_point(col = "Dodgerblue") +

geom\_errorbar(aes(ymax = AverageMSE + 1.96 \* VarMSE, ymin = AverageMSE - 1.96 \* VarMSE), col = "red") + labs(x="Mtry", y="MSE") + ggtitle("Average Performance of Random Forest Models with Different Mtry") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

ggplot(bestPerfTreeNo, aes(x = NumberOfTrees, y = AverageMSE)) + geom\_point(col = "Dodgerblue") +

geom\_errorbar(aes(ymax = AverageMSE + 1.96 \* VarMSE, ymin = AverageMSE - 1.96 \* VarMSE), col = "red") + labs(x="Number of Trees Used", y="MSE") + ggtitle("Average Performance of Random Forest Models for Different Number of Trees") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

ggplot(bestPerfSampSize, aes(x = SampleSize, y = AverageMSE)) + geom\_point(col = "Dodgerblue") +

geom\_errorbar(aes(ymax = AverageMSE + 1.96 \* VarMSE, ymin = AverageMSE - 1.96 \* VarMSE), col = "red") + labs(x="Bootstrap Sample (%)", y="MSE") + ggtitle("Average Performance of Random Forest Models for Different Bootstrap Samples") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

ggplot(bestPerfTreeSize, aes(x = TreeSize, y = AverageMSE)) + geom\_point(col = "Dodgerblue") +

geom\_errorbar(aes(ymax = AverageMSE + 1.96 \* VarMSE, ymin = AverageMSE - 1.96 \* VarMSE), col = "red") + labs(x="Size of Trees (CP)", y="MSE") + ggtitle("Average Performance of Random Forest Models with Tree Sizes") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

xAxis <- c("Number of Parameters","Number of Trees","Sample Size","Size of Trees (cp)")

condition <- rep(c("Small" , "Medium" , "Large") , 4)

value <- c(0,5,9,2,0,3,0,5,10,2,1,3)

data <- data.frame(xAxis,condition,value)

labelMaker <- c("", "","","200","500","1000","","66%","100%","0","7.1x10-6","0.01")

ggplot(data, aes(fill=condition, y=value, x=xAxis)) + geom\_bar(position="stack", stat="identity")+ theme(legend.position = "none")+ labs(x="Hyperparameter", y="Number of Occurences") + ggtitle("Frequency of Hyperparameters Occuring in Best Performing Tree") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + geom\_text(label=labelMaker, size = 5, position = position\_stack(vjust = 0.5))+ annotate("text", x=1, y=5, label= "6", size = 5)

#Ideal model:

start\_time <- Sys.time()

fitIdeal<-randomForest(price~.,data=diamondTrain, mtry = grid$mtry[69], ntree=grid$ntree[69], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = grid$cp[69]), sampsize = nrow(diamondTrain))

end\_time <- Sys.time()

timeExtraRF <- end\_time - start\_time

fitIdeal<-randomForest(price~.,data=diamondTrain, mtry = grid$mtry[69], ntree=grid$ntree[69], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = grid$cp[69]), sampsize = nrow(diamondTrain))

fitWorst<-randomForest(price~.,data=diamondTrain, mtry = grid$mtry[10], ntree=grid$ntree[10], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = grid$cp[10]), sampsize = 27684)

fitTerrible <- randomForest(price~.,data=diamondTrain, mtry = grid$mtry[10], ntree=1, importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = grid$cp[10]), sampsize = 27684)

#Evaluate performance

#Predictive power of the model

#Training data

predRFTrain=predict(fitIdeal,diamondTrain)

RFValuesTrain <- data.frame(diamondTrain$price)

RFValuesTrain$observed <- RFValuesTrain$diamondTrain.price

RFValuesTrain$predicted <- predRFTrain

ggplot(data = RFValuesTrain, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Random Forest (Training Data)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(RFValuesTrain$predicted, RFValuesTrain$observed)

rmse(RFValuesTrain$predicted, RFValuesTrain$observed)

#Test data

predRF=predict(fitIdeal,diamondTest)

RFValues <- data.frame(diamondTest$price)

RFValues$observed <- RFValues$diamondTest.price

RFValues$predicted <- predRF

ggplot(data = RFValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Random Forest (Test Data)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(RFValues$predicted, RFValues$observed)

rmse(RFValues$predicted, RFValues$observed)

#Worst Fit

predRFW=predict(fitWorst,diamondTest)

RFWValues <- data.frame(diamondTest$price)

RFWValues$observed <- RFWValues$diamondTest.price

RFWValues$predicted <- predRFW

ggplot(data = RFWValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Random Forest (Worst Model)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(RFWValues$predicted, RFWValues$observed)

rmse(RFWValues$predicted, RFWValues$observed)

#One tree

predRFW=predict(fitTerrible,diamondTest)

RFTValues <- data.frame(diamondTest$price)

RFTValues$observed <- RFTValues$diamondTest.price

RFTValues$predicted <- predRFW

ggplot(data = RFWValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Random Forest (Worst Model)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(RFWValues$predicted, RFWValues$observed)

rmse(RFWValues$predicted, RFWValues$observed)

#Compare Models

#Random Forest vs Regression Tree

RFAgainstRT <- data.frame(diamondTest$price)

RFAgainstRT$RandomForest <- predRF

RFAgainstRT$RegressionTree <- treePred

ggplot(data = RFAgainstRT, mapping = aes(x = RegressionTree, y = RandomForest, color = diamondTest.price)) + geom\_point(stat="identity") + labs(y="Random Forest Predictions", x="Regression Tree Predictions") + ggtitle("Random Forest Predictions vs Regression Tree Predictions")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(color = "Observed Prices")

cor(RFAgainstRT$RandomForest, RFAgainstRT$RegressionTree)

rmse(RFAgainstRT$RandomForest, RFAgainstRT$RegressionTree)

#Random Forest vs Linear Regression

RFAgainstLR <- data.frame(diamondTest$price)

RFAgainstLR$RandomForest <- predRF

RFAgainstLR$LinearRegression <- lregPred

ggplot(data = RFAgainstLR, mapping = aes(x = LinearRegression, y = RandomForest, color = diamondTest.price)) + geom\_point(stat="identity") + labs(y="Random Forest Predictions", x="Linear Regression Predictions") + ggtitle("Random Forest Predictions vs Linear Regression Predictions")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(color = "Observed Prices")

cor(RFAgainstLR$RandomForest, RFAgainstLR$LinearRegression)

rmse(RFAgainstLR$RandomForest, RFAgainstLR$LinearRegression)

#Random Forest vs Worst Tree

RFAgainstWT <- data.frame(diamondTest$price)

RFAgainstWT$RandomForest <- predRF

RFAgainstWT$WorstTree <- predRFW

ggplot(data = RFAgainstWT, mapping = aes(x = WorstTree, y = RandomForest, color = diamondTest.price)) + geom\_point(stat="identity") + labs(y="Random Forest Predictions", x="Random Forest (Worst Model) Predictions") + ggtitle("Random Forest Optimal Parameters vs Worst Parameters Predictions")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(color = "Observed Prices")

cor(RFAgainstWT$RandomForest, RFAgainstWT$WorstTree)

rmse(RFAgainstWT$RandomForest, RFAgainstWT$WorstTree)

#Variable Importance to prediction

importance(fitIdeal)

varImpPlot(fitIdeal)

#Error rate of trees

errorByTrees <- data.frame(Error = fitIdeal$mse, numberOfTrees = seq(1, 500, by=1))

ggplot(data = errorByTrees, mapping = aes(x = numberOfTrees, y = Error)) + geom\_line(col = "Dodgerblue") + labs(x="Number of Trees", y="Error") + ggtitle("Error rate over trees") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Model Interpretation

diamondData <- diamond[,-1]

X <- diamondData[which(names(diamondData) != "price")]

predictor = Predictor$new(fitIdeal, data = X, y = diamondData$price)

#Variable Importance

#Measured by contribution to prediction (Measure reduction in prediction accuracy using MSE (or MAE))

imp2 = FeatureImp$new(predictor, loss = "mae", n.repetitions = 10)

plot(imp2)

imp.dat = imp2$results

ggplot(imp.dat, aes(x = feature, y = importance)) + geom\_point(col = "Dodgerblue") + scale\_x\_discrete(limits=c("size","clarity","color","depth", "cut", "table")) +

geom\_errorbar(aes(ymax = importance.95, ymin = importance.05), col = "red") + labs(x="Variable", y="Fall in Accuracy (MAE)") + ggtitle("Variable Importance (Random Forest)") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Friedman H statistic

interact = Interaction$new(predictor)

interactGraph <- ggplot(data = interact$results, mapping = aes(x = .interaction, y = .feature)) + geom\_point(stat="identity", col = "skyblue2", size = 3) + geom\_segment(aes(x = 0, y = 6, xend = 0.25, yend = 6), col = "red") + labs(y="Variable", x="Overall Interaction Strength") + ggtitle("Interaction Strength (Random Forest)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ scale\_y\_discrete(limits=c("table","cut","depth","clarity", "color", "size"))

interactGraph <- interactGraph + geom\_segment(aes(x = 0, y = 5, xend = 0.2316, yend = 5), col = "red")

interactGraph <- interactGraph + geom\_segment(aes(x = 0, y = 4, xend = 0.155, yend = 4), col = "red")

interactGraph <- interactGraph + geom\_segment(aes(x = 0, y = 3, xend = 0.036, yend = 3), col = "red")

interactGraph <- interactGraph + geom\_segment(aes(x = 0, y = 2, xend = 0.032, yend = 2), col = "red")

interactGraph <- interactGraph + geom\_segment(aes(x = 0, y = 1, xend = 0.015, yend = 1), col = "red")

interactGraph

#Consider specific variables

#Size

interactSize = Interaction$new(predictor, feature = "size")

interactGraphSize <- ggplot(data = interactSize$results, mapping = aes(x = .interaction, y = .feature)) + geom\_point(stat="identity", col = "skyblue2", size = 3) + geom\_segment(aes(x = 0, y = 5, xend = 0.238, yend = 5), col = "red") + labs(y="Interaction term", x="Interaction Strength") + ggtitle("Strength of Interactions (Size)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ scale\_y\_discrete(limits=c("table:size","depth:size","cut:size","clarity:size", "color:size"))

interactGraphSize <- interactGraphSize + geom\_segment(aes(x = 0, y = 4, xend = 0.18, yend = 4), col = "red")

interactGraphSize <- interactGraphSize + geom\_segment(aes(x = 0, y = 3, xend = 0.021, yend = 3), col = "red")

interactGraphSize <- interactGraphSize + geom\_segment(aes(x = 0, y = 2, xend = 0.011, yend = 2), col = "red")

interactGraphSize <- interactGraphSize + geom\_segment(aes(x = 0, y = 1, xend = 0.007, yend = 1), col = "red")

interactGraphSize

#Colour

interactColor = Interaction$new(predictor, feature = "color")

interactGraphColor <- ggplot(data = interactColor$results, mapping = aes(x = .interaction, y = .feature)) + geom\_point(stat="identity", col = "skyblue2", size = 3) + geom\_segment(aes(x = 0, y = 5, xend = 0.268, yend = 5), col = "red") + labs(y="Interaction term", x="Interaction Strength") + ggtitle("Strength of Interactions (Colour)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ scale\_y\_discrete(limits=c("table:color","depth:color","cut:color","size:color", "clarity:color"))

interactGraphColor <- interactGraphColor + geom\_segment(aes(x = 0, y = 4, xend = 0.177, yend = 4), col = "red")

interactGraphColor <- interactGraphColor + geom\_segment(aes(x = 0, y = 3, xend = 0.0268, yend = 3), col = "red")

interactGraphColor <- interactGraphColor + geom\_segment(aes(x = 0, y = 2, xend = 0.0263, yend = 2), col = "red")

interactGraphColor <- interactGraphColor + geom\_segment(aes(x = 0, y = 1, xend = 0.014, yend = 1), col = "red")

interactGraphColor

#Clarity

interactClarity = Interaction$new(predictor, feature = "clarity")

interactGraphClarity <- ggplot(data = interactClarity$results, mapping = aes(x = .interaction, y = .feature)) + geom\_point(stat="identity", col = "skyblue2", size = 3) + geom\_segment(aes(x = 0, y = 5, xend = 0.245, yend = 5), col = "red") + labs(y="Interaction term", x="Interaction Strength") + ggtitle("Strength of Interactions (Clarity)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ scale\_y\_discrete(limits=c("table:clarity","depth:clarity","cut:clarity","size:clarity", "color:clarity"))

interactGraphClarity <- interactGraphClarity + geom\_segment(aes(x = 0, y = 4, xend = 0.1505, yend = 4), col = "red")

interactGraphClarity <- interactGraphClarity + geom\_segment(aes(x = 0, y = 3, xend = 0.0205, yend = 3), col = "red")

interactGraphClarity <- interactGraphClarity + geom\_segment(aes(x = 0, y = 2, xend = 0.015, yend = 2), col = "red")

interactGraphClarity <- interactGraphClarity + geom\_segment(aes(x = 0, y = 1, xend = 0.01, yend = 1), col = "red")

interactGraphClarity

#Partial Dependence plot

pdpSize = FeatureEffect$new(predictor, feature = "size", method = "pdp")

#Compare to ale plot to see if theres a major difference between their methods

aleSize = FeatureEffect$new(predictor, feature = "size", method = "ale")

pdpColour = FeatureEffect$new(predictor, feature = "color", method = "pdp")

pdpClarity = FeatureEffect$new(predictor, feature = "clarity", method = "pdp")

pdpCut = FeatureEffect$new(predictor, feature = "cut", method = "pdp")

pdpDepth = FeatureEffect$new(predictor, feature = "depth", method = "pdp")

pdpTable = FeatureEffect$new(predictor, feature = "table", method = "pdp")

pdpColourSize = FeatureEffects$new(predictor, features = c("color","size"), method = "pdp")

pdpColourClarity = FeatureEffects$new(predictor, features = c("color","clarity"), method = "pdp")

pdpClaritySize = FeatureEffects$new(predictor, features = c("clarity","size"), method = "pdp")

#Plots

aleSize$plot() + ggtitle("Importance of Size")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpSize$plot() + ggtitle("Importance of Size")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpColour$plot() + ggtitle("Importance of Colour")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_col(col = "skyblue2",fill = "skyblue2")

pdpClarity$plot() + ggtitle("Importance of Clarity")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))+ geom\_col(col = "skyblue2",fill = "skyblue2")

pdpCut$plot() + ggtitle("Importance of Cut")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_col(col = "skyblue2",fill = "skyblue2")+ scale\_x\_discrete(limits=c("Fair","Good","Very Good","Premium", "Ideal"))

pdpDepth$plot() + ggtitle("Importance of Depth")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpTable$plot() + ggtitle("Importance of Table")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpColourSize$plot() + ggtitle("Importance of Colour-Size Interaction")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpColourClarity$plot() + ggtitle("Importance of Colour-Clarity Interaction")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

pdpClaritySize$plot() + ggtitle("Importance of Size-Clarity Interaction")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

#Plot the interactions

diamondTrain2 <- diamondTrain

#Convert color and clarity to continuous

diamondTrain2$color <- sapply(diamondTrain2$color, as.numeric)

diamondTrain2$clarity<- factor(diamondTrain2$clarity, levels = c("IF","VVS1","VVS2","VS1", "VS2", "SI1", "SI2", "I1"))

diamondTrain2$clarity <- sapply(diamondTrain2$clarity, as.numeric)

fitIdeal2 <- randomForest(price~.,data=diamondTrain2, mtry = grid$mtry[69], ntree=grid$ntree[69], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = grid$cp[69]), sampsize = nrow(diamondTrain2))

colourSize <- partial(fitIdeal2, pred.var = c("color","size"), plot = TRUE, plot.engine = "ggplot2", chull = TRUE, rug = TRUE)

colourSize <- colourSize + ggtitle("Importance of Colour-Size Interaction")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + labs(x="Colour", y="Size", fill = "Price")

colourClarity <- partial(fitIdeal2, pred.var = c("color","clarity"), plot = TRUE, plot.engine = "ggplot2", chull = TRUE, rug = TRUE)

colourClarity <- colourClarity + labs(x="Colour", y="Clarity", fill = "Price")+ ggtitle("Importance of Colour-Clarity Interaction")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))

claritySize <- partial(fitIdeal2, pred.var = c("clarity","size"), plot = TRUE, plot.engine = "ggplot2", chull = TRUE, rug = TRUE)+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(x="Clarity", y="Size", fill = "Price")

claritySize <- claritySize + labs(x="Clarity", y="Size", fill = "Price")+ ggtitle("Importance of Size-Clarity Interaction")

tableSize <- partial(fitIdeal2, pred.var = c("table","size"), plot = TRUE, plot.engine = "ggplot2", chull = TRUE, rug = TRUE)+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(x="Table %", y="Size", fill = "Price")

#Shapley Values: Could use this to explain the reasoning of a model for predicting new prices (very useful for new predictions)

shapley <- Shapley$new(predictor, x.interest = X[6278,])

shapleyLabel1 <- paste("Actual Prediction =", round(shapley$y.hat.interest,2))

shapleyLabel2 <- paste("Average Prediction =", round(shapley$y.hat.average,2))

shapley$results$feature.value[6] <- "size=145.97"

ggplot(shapley$results, aes(x = feature.value, y = phi))+ geom\_col(col = "skyblue2",fill = "skyblue2", width = 0.7) + coord\_flip()+ labs(y="Feature Variable Contribution", x="Feature Value") + ggtitle("Shapley Graph for a Random Diamond")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + scale\_x\_discrete(limits=c("clarity=SI1","cut=Very Good","table=58","depth=61.6","color=F", "size=145.97")) + geom\_text(label=shapleyLabel1, x=2.3, y=147, size = 6, col = "dodgerblue") + geom\_text(label=shapleyLabel2, x=1.7, y=152, size = 6, color = "dodgerblue")

#Random Forest Explainer

#variable Importance Measures - compare to contribution to fit

importance\_frame <- measure\_importance(fitIdeal)

plot\_multi\_way\_importance(importance\_frame) + ggtitle("Variable Importance (Mean Minimum Depth)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(y="Number of Times Variable was a Root", x="Mean Minimum Depth")

meanMinDepth <- ggplot(data = importance\_frame, mapping = aes(x = variable, y = mean\_min\_depth)) + geom\_point(stat="identity", col = "skyblue2", size = 3) + geom\_segment(aes(x = 2, y = 0, xend = 2, yend = 2), col = "red") + labs(y="Mean Minimum Depth", x="Variable") + ggtitle("Variable Importance (Mean Minimum Depth)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ scale\_x\_discrete(limits=c("size","clarity","color","cut","depth", "table"))

meanMinDepth <- meanMinDepth + geom\_segment(aes(x = 3, y = 0, xend = 3, yend = 3.182), col = "red")

meanMinDepth <- meanMinDepth + geom\_segment(aes(x = 4, y = 0, xend = 4, yend = 5.298), col = "red")

meanMinDepth <- meanMinDepth + geom\_segment(aes(x = 5, y = 0, xend = 5, yend = 5.898), col = "red")

meanMinDepth <- meanMinDepth + geom\_segment(aes(x = 6, y = 0, xend = 6, yend = 5.958), col = "red")

meanMinDepth

#Extra Tree

#Determine complexity parameter

gridExtra <- expand.grid(ntree = c(200, 500, 1000), cp = c(0.000, cp.best, 0.01))

samp <- sample(nrow(diamondTrain), 2000)

subDiamondTrain <- diamondTrain[samp, ]

subDiamondTrainTest <-diamondTrain[-samp, ]

rmseExtra <- matrix(0L, nrow = 9, ncol = 10)

colnames(rmseExtra) <- c("RMSE1","RMSE2","RMSE3","RMSE4","RMSE5","RMSE6","RMSE7","RMSE8","RMSE9", "RMSE10")

bestExtra <- data.frame( bestRMSE = numeric(10))

for(j in 1:10)

{

for (i in 1:9)

{

fitExtraTemp <- randomForest(price~.,data=diamondTrain, mtry = 1, ntree=gridExtra$ntree[i], importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = gridExtra$cp[i]), sampsize = 2000, replace = FALSE)

predET=predict(fitExtraTemp,subDiamondTrainTest)

ETValues <- data.frame(subDiamondTrainTest$price)

ETValues$observed <- ETValues$subDiamondTrainTest.price

ETValues$predicted <- predET

rmseExtra[i,j] <- rmse(ETValues$predicted, ETValues$observed)

}

#Determine best hyperparameters

bestExtra$bestRMSE[j] <- which(rmseExtra[,j] == min(rmseExtra[,j]))

}

write.csv(rmseExtra, "C:/Users/pie93/Desktop/Data Analytics/Assignment/rmseDataExtra.csv")

write.table(bestExtra, "C:/Users/pie93/Desktop/Data Analytics/Assignment/bestDataExtra.txt", sep="\t")

xAxis <- c("Number of Trees","Size of Trees (cp)")

condition <- rep(c("Small" , "Medium" , "Large") , 2)

value <- c(5,2,2,4,3,4)

data <- data.frame(xAxis,condition,value)

labelMaker <- c("200","500","1000","0","7.1x10-6","0.01")

ggplot(data, aes(fill=condition, y=value, x=xAxis)) + geom\_bar(position="stack", stat="identity")+ theme(legend.position = "none")+ labs(x="Hyperparameter", y="Number of Occurences") + ggtitle("Frequency of Hyperparameters Occuring in Best Performing Model") + theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5)) + geom\_text(label=labelMaker, size = 5, position = position\_stack(vjust = 0.5))

#Best Extra Trees

start\_time <- Sys.time()

fitExtra <- randomForest(price~.,data=diamondTrain, mtry = 1, ntree=200, importance=TRUE, na.action=na.omit, control= rpart.control(split="Gini", cp = 0), sampsize = nrow(diamondTrain), replace = FALSE)

end\_time <- Sys.time()

timeExtra <- end\_time - start\_time

#Extra Trees Performance

predET=predict(fitExtra,diamondTest)

ETValues <- data.frame(diamondTest$price)

ETValues$observed <- ETValues$diamondTest.price

ETValues$predicted <- predET

ggplot(data = ETValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Extra Trees (Test Data)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(ETValues$predicted, ETValues$observed)

rmse(ETValues$predicted, ETValues$observed)

#Random Forest vs Extra Trees

RFAgainstET <- data.frame(diamondTest$price)

RFAgainstET$RandomForest <- predRF

RFAgainstET$ExtraTree <- predET

ggplot(data = RFAgainstET, mapping = aes(x = ExtraTree, y = RandomForest, color = diamondTest.price)) + geom\_point(stat="identity") + labs(y="Random Forest Predictions", x="Extra Trees Predictions") + ggtitle("Random Forest vs Extra Trees Predictions")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ labs(color = "Observed Prices")

cor(RFAgainstET$RandomForest, RFAgainstET$ExtraTree)

rmse(RFAgainstET$RandomForest, RFAgainstET$ExtraTree)

#Extra Trees - Training Data

predET=predict(fitExtra,diamondTrain)

ETValues <- data.frame(diamondTrain$price)

ETValues$observed <- ETValues$diamondTrain.price

ETValues$predicted <- predET

ggplot(data = ETValues, mapping = aes(x = predicted, y = observed)) + geom\_point(stat="identity", col = "skyblue2") + labs(y="Observed Prices", x="Predicted prices") + ggtitle("Diamond Prices with Extra Trees (Training Data)")+ theme(plot.title = element\_text(color = "Dodgerblue", size = 15, face = "bold", hjust = 0.5))+ geom\_smooth(method='lm', color = "red")

cor(ETValues$predicted, ETValues$observed)

rmse(ETValues$predicted, ETValues$observed)