Project 4 Regression Analysis

Inesh Chakrabarti, Lawrence Liu, Nathan Wei

Introduction

For the first part of this project we will do regression analysis. The dataset we chose to use is one of diamond characteristics. We will conduct regressions to predict the price of a diamond given some features.

Dataset

Let us begin by understanding the dataset. The dataset consists of information about 53940 round-cut diamonds with ten features:

Feature	Description
`carat`	weight of the diamond (0.2–5.01)
`cut`	quality of the cut (Fair, Good, Very Good, Premium, Ideal)
`color`	diamond colour, from J (worst) to D (best)
`clarity`	a measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best))
`x`	length in mm (0–10.74)
`y`	width in mm (0–58.9)
`z`	depth in mm (0–31.8)
`depth`	total depth percentage
`table`	width of top of diamond relative to widest point (43-95)
`price`	price in US dollars ($326-$18,823)

Question 1.1

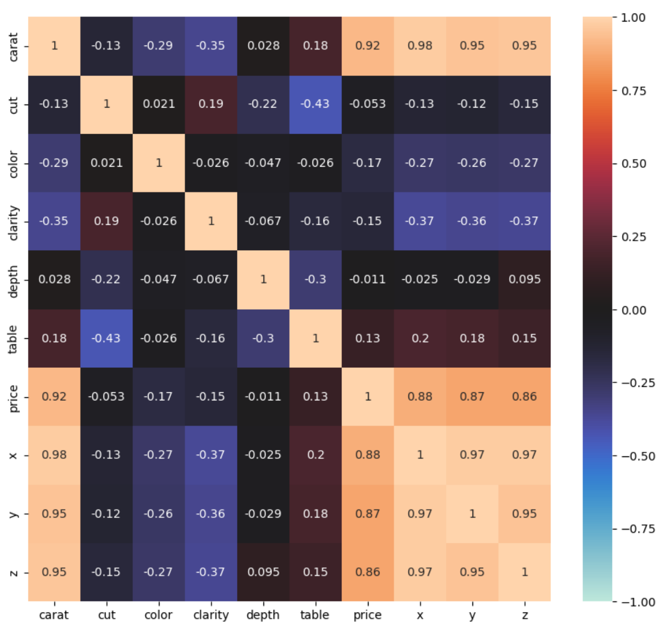


Figure 1: Feature Pearson Correlation Heatmap

We will be using the nine features to predict the `price`. We can begin by computing the Pearson correlation matrix heatmap for these features in the dataset in Figure 1. Note that a pearson correlation coefficient r is defined as such:

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \tag{1}$$

We have assigned quantitative values to the qualitative labels `cut`,`color`,`clarity` as ascending natural numbers based on ideality.

Feature	Correlation
`carat`	0.9215914337868304
`cut`	-0.05349263851362828
`color`	-0.1725093772499559
`clarity`	-0.14680175361025616
`depth`	-0.010647725608533299
`table`	0.12713358133531918
`x`	0.8844357793744166
`y`	0.865421694764742
`z`	0.861250266123968

(a) Price

Feature	Correlation
`carat`	0.7694571626172851
`cut`	0.00542011950342582
`color`	-0.011980043670033661
`clarity`	0.04512538515850012
`depth`	-0.03572374489729493
`table`	0.08458507638109278
`x`	0.7873455524189906
`y`	0.7717301198408058
`z`	0.7655421629234554

(b) Price per Carart

Table 1: Pearson Correlation Coefficients

The values for correlation for `price` is given in Table 1(a). We see that, unsurprisingly, there is a massive collection of high correlation squares at the bottom right. These indicate high Pearson correlation coefficient between `price` and `x`, `y`, `z`. Similarily, there is also a high correlation with `carat`. All of these suggest that the size of the diamond itself is the most significant predictor as to its price.

However, unexpectadly, we had a negative perason coeffciient for the quality of the `cut`, `color`, and `clarity`. This was odd, so we similarily calculated the Pearson correlation values for Price per Carat instead, given in Table 1(b). We observed that the coefficient for `cut` and `clarity` became slightly positive, while color became close to zero. These seemed more in line with our expectations,and confirmed that the rarity of high carat and high clarity in a diamond at the same time was making the corresponding values in Table 1(a) negative.

Question 1.2

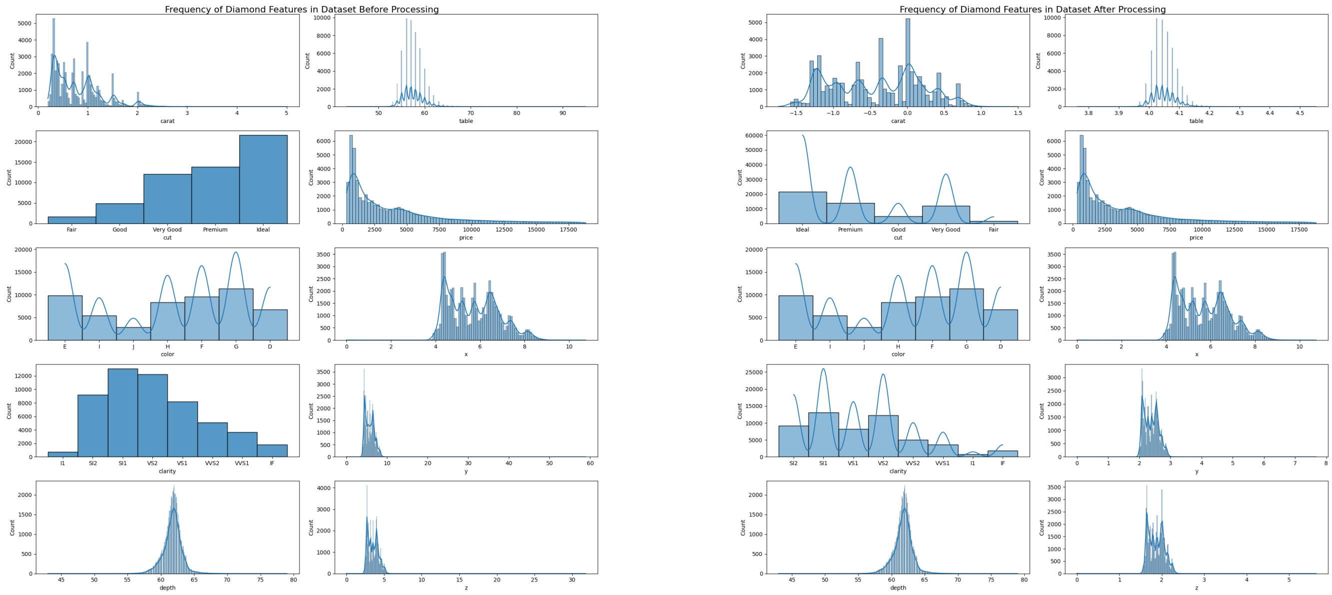
Feature	Skewness
`carat`	1.116645920812613
`depth`	-0.08229402630189467
`table`	0.7968958486695427
`x`	0.3786763426463927
`y`	2.4341667164885554
`z`	1.5224225590685583

(a) Before Processing

Feature	Method	Skewness
`carat`	Box Cox	0.020450070764268666
`depth`	No Change	-0.08229402630189467
`table`	Logrithm	0.5993289324435086
`x`	No Change	0.3786763426463927
`z`	Square Root	0.0139742634447758

(b) After Processing

Table 2: Skewness of Features



(a) Before Processing

(b) After Processing

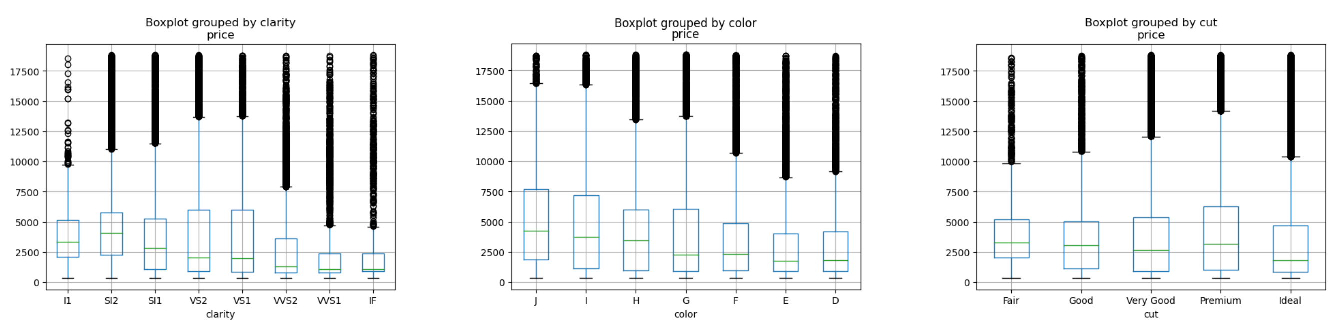
Figure 2: Histogram and KDE for Features

Now, we examine the frequency distributions within each feature. We find that some of our numerical features are somewhat skewed—apparent from Table 2(a) and Figure 2(a). We target a skewness less than 0.5, as this would imply that the distribution is somewhat symetric. As such, we try three different methods: square-root, logrithm, and box-cox transformation. The first two are somewhat self-explanatory; box-cox uses a non-zero value of λ and conducts the following transformation:

$$y_\lambda^{'} = \frac{y^\lambda - 1}{\lambda \cdot \bar{g}_y^{\lambda-1}} \tag{2}$$

where $\bar{g}_y$ is defined as the geometric mean of y.
We next try all of these methods, along with no transformatino, and take the minimum to try to minimize skewness in our data. The results are shown in Table 2(b) and Figure 2(b).

Question 1.3



(a) Clarity

(b) Color

(c) Cut

Figure 3: Categorical feature vs price boxplots

Begin by noting that in these boxplots, the features are arranged from left to right in terms of ascending desirablity.

Begin by observing the boxplot for `clarity`. Note that the trend for the outlier cutoffs ($1.5 \times$ IQR) seems to mirror the frequency histogram plot in Figure 2(a). This is somewhat interesting, as this implies that we simply find more expensive diamonds of average clarity, which also have a wider range of prices. Meanwhile, the diamonds with the best clarity seem to have a low cutoff, which could be attributed to the small sample size, but also the fact that these diamonds are also likely to be smaller. This is apparent from Figure 1, where we see that there is a negative coefficient in the case comparing `clarity` with `x`, `y`, and `z`. Note that size of the diamond was established as the best predictor for the price of the diamond from earlier. Meanwhile, it is apparent that the diamond with worst clarity have few that breach the 10000 dollar price mark, as we only see a few individual dots on the boxplot indicating these diamonds.

Now, let us analyze the boxplot for `color`. We see that there is a negative trend for the outlier cutoff as the desirability of the color increases. This To analyze the boxplot for `cut`, we need to also look as the histogram of diamond features in Figure 2(a). Note that we also have increasing numbers of diamonds as we move left to right. We see that the data generally follows our expectation, as the outlier cutoff seems to be higher for each one. However, this changes when the `cut` is Ideal. This makes sense, most of the diamonds are cut ideally, and smaller diamonds are also probably easier to cut well; meanwhile, if a diamond isn't cut very well, but there was some attempt to cut it to some reasonable extent, it was probably very large. This explains why Ideal seems to break the observed trend.

Finally, let us observe the boxplot for `cut`. This plot can be explained with similar explanation to that of `clarity`. In fact,

Question 2.1

Now, before we train our regression models, we begin by splitting our data into training and testing sets. As such, we must now standardize the feature columns. The function used to do this, `scaledTrainTest()`, as well as `scaledTrainTestSplit()` can be found in `utils.py`.

Question 2.2

Next, we begin with feature selection. We note that some of the features may not be useful or may cause overfitting in our models as they do not carry useful useful information about the variable that we are trying to predict. To tell whether this is the case, we use two metrics: Mutual Information (MI) and F score.

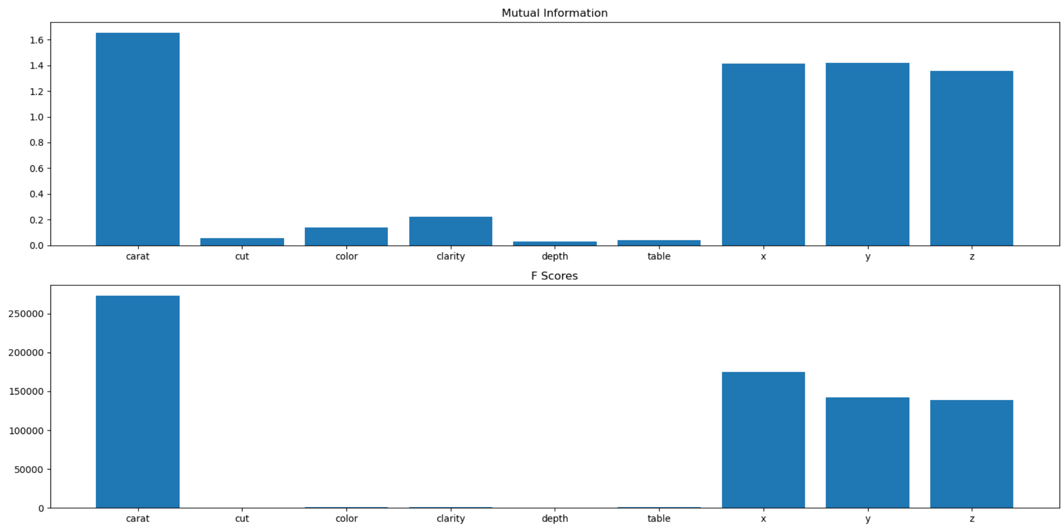


Figure 4: Bar graph of F score and Mutual Information for features

Feature	Mutual Information	F Score
`carat`	1.64589286853222835	273144
`cut`	0.058351547792578895	139
`color`	0.13968295427702282	1465
`clarity`	0.21672877051000627	1079
`depth`	0.030486803506200033	5.074
`table`	0.03818752327438668	802
`x`	1.4058664238563194	174973
`y`	1.4172107199595354	142130
`z`	1.3569258463570115	138947

Table 3: Mutual Information and F Score Values for features

It is clear by observation from Figure 3 and Table 2 that the lowest mutual information is present in `depth` and `table`. It is also important to note that the mutual information in `cut` is also very small. This makes sense as we saw earlier that the Pearson correlation coefficient for `cut` with respect to price per carat was almost zero. Similarily, the Pearson correlation coefficients for depth and table were very close to zero.

From this point, we will be testing the regression models without these three features, and with these three features and comparing the performance to determine the general performance.

Regression Models

Question 3

For this entire section we perform 10-fold cross validation and then measure the average RSME error for the training and validation sets. For the random forest model we also measure "Out-of-Bag Error" and R^2 score.
To explain OOB error, let us first examine how a Random Forest algorithm with bootstrapping works. If we bootstrap, we sample the training data, and then use this small sample of the training data in order to train a single decision tree model. We can then ensemble many such trees, all trained with different

samples of the original data. OOB error is calculated on the training set, where we calculate the average error of the trees' predictions on data that was not included in its bootstrap sample.

R^2 score is normally used as the coefficient of determination—that is—the proportion of the variation in the labels that can be predicted from the features. For random forests, we compute it as:

$$R^2 = 1 - \frac{\text{MSE}}{\text{Var(labels)}}$$

Note that R^2 for non-linear predictors like random forest can in fact be negative.

Linear Regression

Question 4.1

We begin with a simple regression, least square linear regression. Begin by noting the optimization problem:

$$\underset{\theta}{\text{argmin}} \frac{1}{2}||\mathbf{Y} - \theta^T \hat{\mathbf{X}}||^2 \tag{3}$$

Taking the derivative with respect to θ and setting it equal to zero gives us

$$\theta = \left(\mathbf{X}^T\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{Y} \tag{4}$$

Now, we can add regularization terms, beginning with the simple L1 regularization, also known in this case as a Lasso regression. This would make our new optimization problem:

$$\underset{\theta}{\text{argmin}} \frac{1}{2}||\mathbf{Y} - \theta^T \hat{\mathbf{X}}||^2 + \alpha||\theta|| \tag{5}$$

Note that in this case, our learned values for θ would become somewhat sparse as L1 regularization linearly penalizes any non-zero parameters. As such, the gradient for the regularization term does not change until the parameter is zero, which leads to sparsity. However, in this case, as there aren't many terms, we would simple disconsider features that aren't very important.

With L2 regularization, also known as a Ridge regression, we would have:

$$\underset{\theta}{\text{argmin}} \frac{1}{2}||\mathbf{Y} - \theta^T \hat{\mathbf{X}}||^2 + \lambda||\theta||^2 \tag{6}$$

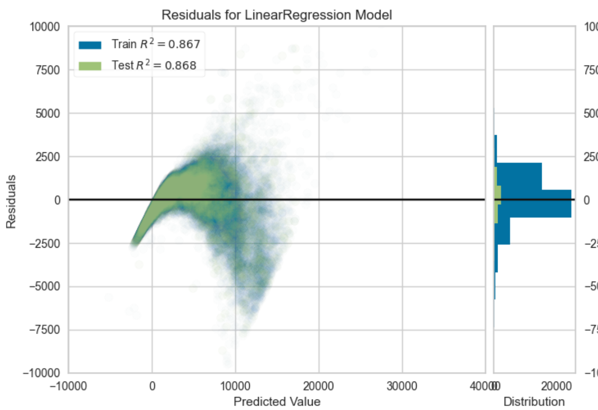
Note in this case, we wouldn't have parameters that are zero; rather, we would have extremely small parameters. This is because the gradient for L2 regularization decreases rapidly as the parameter itself becomes smaller and smaller due to the parabolic nature of the term.

Question 4.2 and 4.3

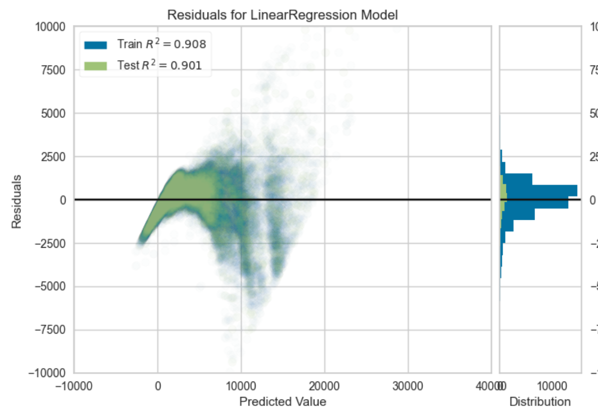
Model	Description	Train RSME	Test RSME
1	Features processed	1434.7215231192772	1443.41152861054185
2	Features unprocessed	1216.495767837737	1217.7987618616985
3	Features processed using log(`price`)	963.6508209107394	963.0840609301274
4	Features unprocessed using log(`price`)	1215.167267164955	1292.934288205552

Table 4: Ordinary Least Squares Regression Model

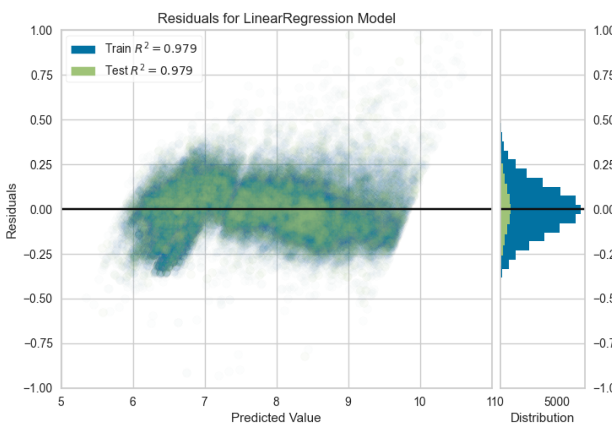
Beginning with ordinary least squares regression, I first varied the number of features available to the model based on mutual information and quickly found that keeping all the features led to the best regression model. This makes sense due to the nation of a linear regression—if a feature isn't very relevant, its associated parameter will be very small—and so for Ridge and Lasso, we also kept all of the features.



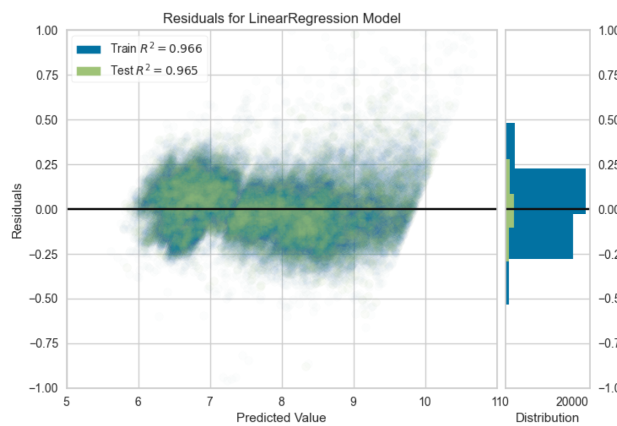
(a) Model 1 Residuals



(b) Model 2 Residuals



(c) Model 3 *log* Residual



(d) Model 4 *log* Residuals

Figure 5: Residual Plots

We then tested the linear regression model with processed data (deskewed as in Question 1.2) and unprocessed data. We noted that the model without processed features seemed to perform much better as shown in Table 4 and visualized in Figure 5. Note that Model 2's residual histogram is distributed far more sysmtetrically than Model 1, implying that the linear regression fit better in this case. However, we see in both Figures 5(a) and 5(b) that Models 1 and 2 predict many of the prices to be negative.

As such, we decided to test the model by taking the natural logrithm of `price` before fitting the regression. Note in Table 4 that with unprocessed Features, this model performed about the same, but with processed features the RSME decreased very significantly. We also see that our new *log* residual plots seem to follow a linear trend, which makes sense as this implies that the error is increasing as the price of the diamond itself increases. We also note that unlike earlier, Model 3 has normally distributed residuals while Model 4 does not. This is interesting, because this is the opposite of Models 1 and 2; that is, this time, the model with deskewed data has the more normal distribution of residuals.

Next, we test the Lasso Model by iterating through alpha values 10^k such that $k \in \mathbb{Z}$ and $-4 \geq k \leq 1$. We also try both the log(`price`) model, and try both with and without deskewing. As seen in Table 5, we see that the best performming model is the same as for ordinary least squares regression, as well as the general trends across the models. The only difference in this case that we see is that leaving the features unprocessed while also taking the log(`price`) leads to the error blowing up. We note that this issue persists for all future models referred to in this report as well.

Model	Description	Best Alpha	Best Train RSME	Best Test RSME
`1`	Features processed	1	1434.8446180295655	1443.270394653097
`2`	Features unprocessed	1	1216.5716991188874	1217.1322328759766
`3`	Features processed using log(`price`)	0.001	953.1890102312589	952.4417958988445
`4`	Features unprocessed using log(`price`)	10	4270.277347211739	4270.276711329372

Table 5: Lasso Regression Model

Next we test the Ridge model with all the same parameters are the Lasso model, except we also test whether or not feature standardization is palying a role in improving the model performance. Note the description column has been left out as it is the same as the previous two models. We see in Table 6 that contrary to our expectations, scaling/standardizing the features did not seem to play a very big roll in improving the model. This was surprising, given how a linear regression is usually sensitive to changes in magnitude (making scaling beneficial).

Model	Standardized	Best Alpha	Best Train RSME	Best Test RSME
`1`	Yes	10	1434.8513489310335	1442.9008267138393
`1`	No	10	1435.8556914721148	1442.1066793353307
`2`	Yes	10	1216.6969576343665	1217.9346926613257
`2`	No	1	1216.4978665798574	1217.7934651327528
`3`	Yes	0.001	964.0857752044615	963.5016367407248
`3`	No	0.0001	963.4272568650325	962.8249050664665
`4`	Yes	1	1215.16766015031	56321.703744924875
`4`	No	1	1215.1676667481079	56321.78526878937

Table 6: Ridge Regression Model

Ultimately, we find that our best performing model was Lasso Model `3` with an alpha value of 0.001, as this gave us the lowest RSME of all our linear regression models.

Question 4.4

p-values for different features tell us the chance that a certain parameter is zero— that is, a parameter corresponding to a certain feature is zero. This means that if a p-value for a feature is very small, the feature is also probably not very important to the linear model.

Polynomial Regression

Question 5.1

After creating an array of the polynomial features, we find, unsurprisingly, that `carat` is involved in all of them. However, we never use a higher power of `carat`—rather, we multiply by different powers of `clarity`,

Feature	Mutual Information
carat $\times$ clarity	1.8159676557242
carat $\times$ clarity^2	1.8052014657565039
carat $\times$ color^2	1.7327044648084868
carat $\times$ clarity^4	1.7102713903040332
carat $\times$ color^{2}clarity^2	1.6728060460257277

Table 7: Most Salient Polynomial Features

and in one case by color^2. This is interesting, as earlier it seemed as if all of the cateogrical features were either not very important, or we made some mistake in assigning ascending natural numbers as values for them. However, we see here that the product is somewhat valuable. This confirms our findings when we analyzed Figure 3 in Question 1.3; we find that keeping `carat` constant, higher values for `clarity` and `color` do in fact increase the price of the diamond. This confirms that the misleading trends were simply due to the rarity of larger diamonds as we have better `clarity` and `color`. Thus, although `caraat` is by far the most important singular feature, we find that when used in conjunction, `clarity` and `color` can be interprretted to have some meaning as well.

Question 5.2

In order to test which degree of polynomial was best, we tested integer values between 2 and 6 along with different alpha values for the Ridge regularization. We quickly notice that the higher degree polynomials are almost definitely overfitting, with extremely low training error and extremely high validation error. This makes sense, as higher order polynomials are less generalized in shape, and tend to try to fit the data more "perfectly"—in other words, it's simply a more complicated model. We also see that the model seems to perform better than the basic linear regression, but worse than the log(`price`) linear regression model. This suggests that the relationship between price and the features has some relationship that is more logrithmic, or one that becomes flatter over time. As such, we cannot properly explain this relationship with just a polynomial regression. Therefore, we once again took the log(`price`) in this section and had better results as shown in Table .

Neural Network

Question 6.1

We operated under the assumption that deeper networks would be better than wider networks, but we also tested this idea with two seperately tuned connected neural networks. Their architectures are shown in Figure , while their loss curves and perofrmance are in Figure and Table. The logic applied towards this tuning of the neural network hyperparameters was quite simple—in the cases of overfitting, we increased alpha, a regularization parameter, applied an inverse scaling learning rate, and applied early stopping. We also increase regularization in the first model by using half the batch size.

Question 6.2

The peformance is significantly better than the linear regression, although the time to train for even a singular model is almost as long as the entire grid search for linear or Ridge regression. The difference in all of the models used can be found in Table where we have compiled the best models of each type.

Question 6.3

We did not use an activation function for the output from the last fully connected layer. Initially, we were planning on simply taking the output from the fully connected layer as we were just interested in the raw numebrs predicted by the model; however, we realized that it is an obvious assumption that the price must always be positive and as such we decided to try RELU. However, it imperically performed worse, so we simply did not use an activation function.

Question 6.4

If the depth of the network is very high, the obvious risk is of course overfitting. This is simply because we are increasing the number of neurons, so the network becomes more likely to memorize the training data. At this point, the model would not be able to generalize as well. There is also a higher risk of vanishing or exploding gradients, as they are carried over fro one layer onto the next layer. We also face the issue of internal covariate shift, where every hidden layer's input distribution has to change every time a parameter updates in the previous layer leading to extremely slow training that is not possible without good parameter initializations.

Random Forest

Quesiton 7.1

Maximum number of features has a regulariation effect on the model. This is simply because we are considering less features, and disregarding the features that aren't as useful. As such, we have a simpler model and thus setting this maximum number of features for the tree to split on allows for regularization. However, eliminating too many featuers could be harmful to the model as we could simply be eliminating features that are valuable. The number of trees refers to the number of seperate tree models created that are going to be ensembled. The more models we ensemble while also using bootstrapping allows for more regualrization. However, at the same time, we must balance having enough data for the individual trees to actually train on. Lastly, the depth of each tree obviously has a regularization effect on the model, as we are effectively reducing the complexity and "number of neurons" of the model. We can also try to change the depth by adjusting the minimum number of samples needed in order to split into multiple leaves.

Quesiton 7.2

Random forests are essentially combining small piecewise linear decision boundaries at each of its nodes into a larger non-linear structure. Thus, our final decision boundary is somewhat more complex, and can in fact be non-linear. Logically, we see the decision tree itself can overfit any dataset y setting harsher and harsher if-statements, meaning it can perfectly model/overfit any training set.

Questino 7.3

Question 7.4

To explain OOB error, let us first examine how a Random Forest algorithm with bootstrapping works. If we bootstrap, we sample the training data, and then use this small sample of the training data in order to train a single decision tree model. We can then ensemble many such trees, all trained with different samples of the original data. OOB error is calculated on the training set, where we calculate the average error of the trees' predictions on data that was not included in its bootstrap sample.

R^2 score is normally used as the coefficient of determination—that is—the proportion of the variation in the labels that can be predicted from the features. For random forests, we compute it as:

$$R^2 = 1 - \frac{\text{MSE}}{\text{Var(labels)}}$$

Note that R^2 for non-linear predictors like random forest can in fact be negative.

LightGBM, CatBoost, and Bayesian Optimization

Question 8.1

Question 8.2

We conducted a

Question 8.3

Project 4 Twitter Data

Inesh Chakrabarti, Lawrence Liu, Nathan Wei