State-of-the-Art Diamond Price Predictions using Neural Networks

Charley Yejia Zhang, Sean Oh, Jason Park

***Abstract*— In this paper, we discuss and evaluate models to predict the prices of diamonds given their properties. This is important for diamond retailers to appropriately set prices and for customers to estimate prices for diamonds by knowing just a few features about each stone. Using a Kaggle dataset with diamonds' recorded properties for each example, we show that we can build an extremely successful model using neural networks (NN) to predict a diamond's price given these properties. We describe our method of choosing this model, optimizing results, and discussing implications of our state-of-the-art method compared to the previous best method.**

I. INTRODUCTION AND DATASET

For centuries, civilizations all around the world have coveted diamonds for their aesthetic beauty and captivating scintillation. Regarded as a symbol for wealth, they are priced for the affluent by the diamond industries that dominate the market. While these diamond producers and marketers have been extremely successful in the past, with the onset of the information age, consumers are now able to cross-reference prices of similar diamonds from other companies before making purchasing decisions. Laboratory-assessed quality can be readily determined for diamonds as well, facilitating the categorization of the qualities of a diamond. With these resources comes the task of estimating the price of cut diamonds given their properties.

There are many factors that might affect the price of a diamond, but the most common ones are referred to as the 4 Cs: **carat**, **cut**, **color**, and **clarity**.

Carat: Carat is the mass of the diamond. 1 carat (ct) is equal to 200mg. This is the only quantitative measure of the 4 Cs. Carat is non-linearly related to the price, as shown in Figure I.

Cut: Cut refers to both the shape of the stone and the quality of its scintillation. The cut perfection is classified from "Fair" to "Ideal".

Color: Diamond colors vary from colorless to a light yellow. The more colorless a diamond is, the more expensive it is likely to be. The standard is a classification developed by the Gemelogical Institute of America and is the most used out of all of the color grading schemes; it uses an alphabetical score, "D" being the most colorless and "Z" being a prominent yellow.

Clarity: Diamonds may have internal blemishes and fractures which decrease their transparency, which in turn decreases their value. Clarity is graded on a scale from FL (Flawless) to I3 (Obvious Inclusions) based on the size, nature, position, and quantity of internal blemishes.

There are several other properties that might affect the price of a diamond. While these might not be as popular as the commonly used 4 Cs, they still may measure some factors of the diamond that could affect the quality and consequently the price of the diamonds.

Depth: Depth is measured as the ratio between z and the average width of the top of the diamond.

Table: Table is measured as the width of the top of the diamond at its widest point.

The dataset chosen for this predictive task is pulled from Kaggle [1] and contains 53,940 samples with 10 properties each: carat, cut, color, clarity, depth, table, price, and the x, y, z dimensions. The carat is measured in carats, the cut is measured qualitatively as a grade from "Fair", "Good", "Very Good", "Premium", and "Ideal", the color is measured with grades from "J" being the worst to "D" being the best, the clarity is measured using the standard from "I1", "SI2", "SI1", "VS2", "VS1", "VVS2", "VVS1", and "IF", the depth is measured as the ratio of the diamond's z axis to its average diameter, the table is measured as the width of the diamond at its widest point, the coordinates are measured in millimeters, and the price is measured in US dollars. In Figure 1, we can see the distribution of the data for each of the 10 properties.

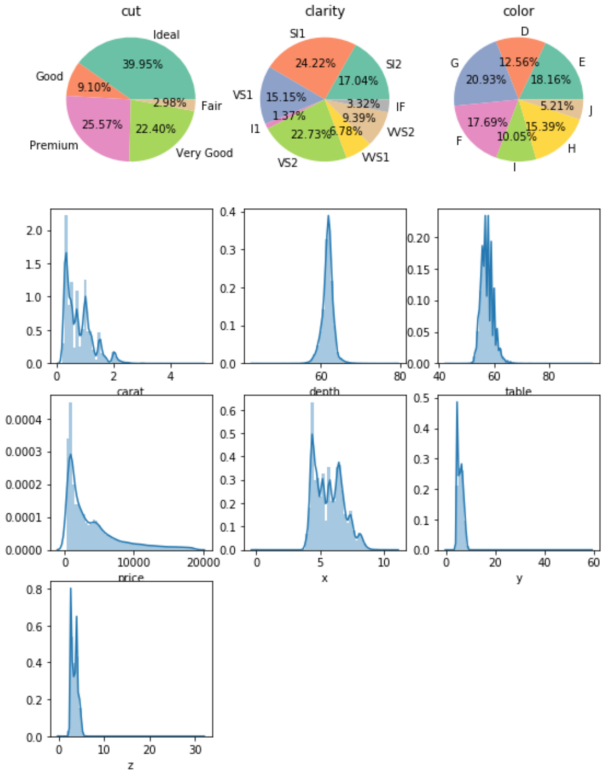

Fig. 1. Distributions of data on different properties

The price distribution is skewed right, so we are able to take the logarithm of the prices to obtain a more normal result for a better dataset, shown in Figure 2.

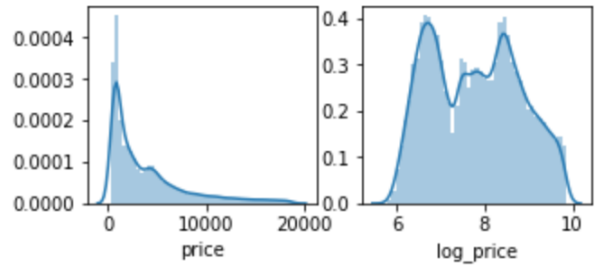

Fig. 2. Taking the logarithm of the prices

We can explore the data by taking a heatmap of the correlations between each of the potential features (Figure 3).

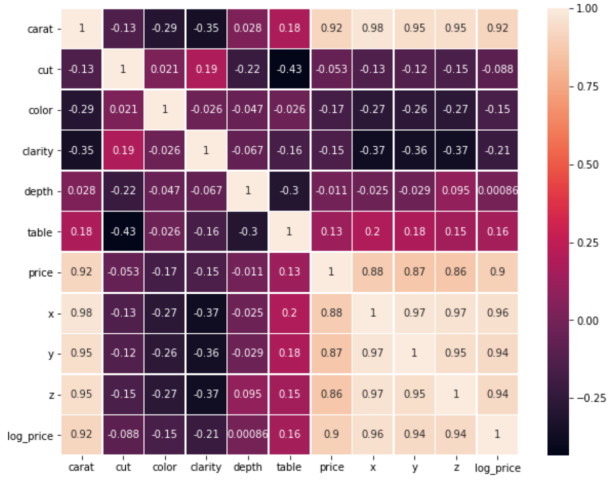

Fig. 3. Heatmap of the correlations between each of the features

One thing that's noticeable from the heatmap is that x, y, z, and carat are highly correlated. This means that the diamonds mostly share the same proportions in terms of dimensions and mass. We can verify this by plotting the x, y, z, and carat data, as the data is highly linear in respect to these properties (Figure 4). As dimensions increase, the mass of the diamond should increase.

It's worthwhile to identify that price is strongly linearly correlated with the carat of a diamond, while not perfectly correlated. We know now that we can use carat as a strong predictor for the price of a diamond.

The heatmap does not reveal everything about the data, however. Other than carat, the size, and the price of the diamond, there does not seem to be any linear correlation, but if we plot the data visually while labeling the categorical properties, we can see there is some sort of pattern (Figure 5).

While it's a little difficult to tell if there is some sort of relationship between the color of a diamond and its mass

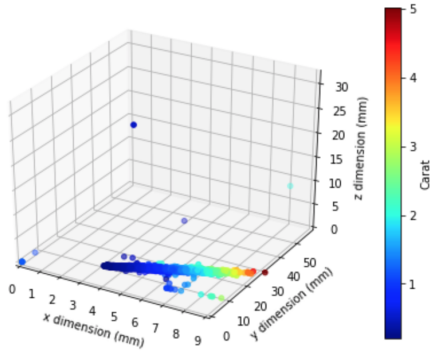

Fig. 4. Correlation of x, y, z, and mass

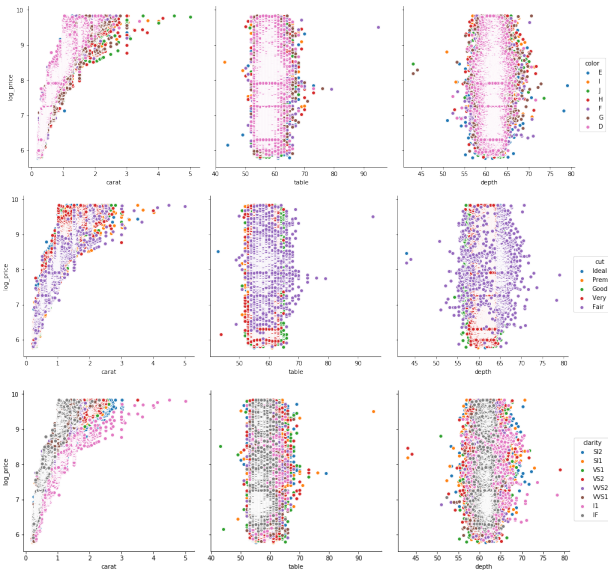

Fig. 5. Correlation of categorical features and other features with price

and its price, there is a little bit of structure that suggests that higher priced diamonds with less mass tend to be of higher color grade in the first plot in Figure 5. The plots also suggest that higher priced diamonds with less mass tend to be of better quality cut, and very strongly so for higher clarity. Interestingly, diamonds with high quality cut and clarity are tightly distributed around around a table size of 56mm and a depth of 61mm, while lower quality cut and clarity have much more variant distributions for table and depth size. As a takeaway however, in reference to our predictor, we can visually conclude that there is some positive relationship between cut, clarity, and color with price, and we can base a predictor based off of these features.

II. PREDICTIVE TASK

Our goal was to predict the price of diamonds using features such as carat, clarity, color, cut, depth, table length, x, y, and z axis lengths in millimeters. The depth, table length, x, y, and z axis lengths were given as numerical data. Thus, we normalized the data by subtracting the mean from each data point and dividing by the standard deviation.

Likewise, we log transformed the price data in order to reduce the right skew in distribution. The carat, clarity, color, and cut were given as categorical data so we converted it to a feature vector by mapping them to a sequential integer range with zero mean. For example, the cut feature of a diamond is represented as Fair, Good, Very Good, Premium, or Ideal where Fair is the worst while Ideal is the best. Thus, Fair would be represented as negative two while Ideal would be represented as positive two. We also attempted using a one-hot encoding of each categorical property. Furthermore, we calculated volume using x, y, z lengths as an additional feature. We also tried using the logarithm of the carat instead of the carat as the data is slightly skewed right. As for the models, since we are predicting prices, we needed to use regressors over classifiers. Thus, we decided to use several regression models such as random forest regression, linear ridge regression, support vector regression, and artificial neural networks. To evaluate our prediction, we needed to split the data into training, validation, and test sets. We randomly shuffled the data, using 60% of the data for our training set and 20% for each of the validation and test sets. We trained our model on the training set and used it to generate a list of predictions for the validation sets. To evaluate our performance, we calculated the root mean square error and coefficient of determination between the predictions vector and the actual prices vector. The coefficient of determination (R2 Score) is used as our similarity metric and is defined below:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

where:

$$SS_{res} = \sum_i (y_i - f_i)^2$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$

We initially attempted to use mean square error but due to the magnitude of the variation of the prices, root mean square error was the better option. We used the performance on the validation set to tweak our parameters and used the test set to emulate the performance of our model on unseen data. Because of the complexity of artificial neural networks and them being universal function approximators, we expected this model to perform better than the others. Thus, we decided to use simple linear ridge regression and linear support vector regression models as baselines to compare to the neural network. Furthermore, to evaluate our neural network, we used the mean square error as our loss function and we calculated and graphed the loss and R2 score over the validation and training set.

III. MODEL

A. Proposed Models

The first baseline model that we used was linear ridge regression. However, we noticed that there was high training performance but it was unable to generalize to our training and validation data. We also tried using a linear support vector regressor. This too, had poor performance on the validation and testing set while overfitting on the training data. Thus, we increased the regularization constant from 10 to 1000 which seemed to help it generalize to a R2 score of -0.13. However, this was not sufficient and we knew we could improve it using a different model. Thus, we decided that perhaps a more complex non-linear model would provide better results and thus we experimented with using neural networks as a regressor for diamond prices. In the end, the non linear behavior of neural networks resulted in a much better test performance and thus, we decided a neural network would be the best model for our price prediction problem.

B. Final Model

After deciding on using a neural network model over models like ridge regression and linear SVR, our goal was to find which data preprocessing, feature selection, network architectures resulted in the best performance. A more detailed explanation on empirical performances and implications of the results of these tests can be found in the 'V. Results' section. We first decided on which features to include and the representations of each of the features. From R2 score results, we concluded that including all features in their raw numerical form yielded the best performance. As for network architecture, we observed that a 2 layer NN with 200,100 hidden nodes per layer was optimal. We trained our network using our training set (60% of dataset) and observed testing performance on the validation set (20% of dataset) to gauge how well how model generalizes to unseen data and have a detection system for over-fitting. We also had early stoppage if our validation error increased for more than 2 epochs. We then trained the network for 700 epochs with batches of 100. We used the Adams optimizer with an initial learning rate of 0.004 with a decay of 0.25 after every 50 epochs. Our model was optimized by reducing the mean square error:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2$$

After, we trained the network using these optimal design choices and hyper-parameters, we experimented with data processing to nudge out extra performance. After normalizing all of our numerical data, we found a performance boost of around 0.2% which is significant at such high scores. This boosted our R2 score from 0.977 to around 0.980. After, we noticed in the data that the curve between the spatial features (x,y,z,carat) and price has approximately a positive exponential relationship. Furthermore, the distribution of the prices seem to be right skewed. Based on these observations, we decided to train our network on logged prices and had it predict log prices. We ran performance evaluations on both raw prices (by just taking the exponential of the predicted prices when testing for R2) and logged prices. This pushed our model over the previous state-of-the-art performance toward an R2 score of .99241 (log prices) and 0.98111 (raw prices).

Overall, our validation checking within the model has allowed it to generalize extremely well and experienced no problems with over-fitting as training, validation, and test scores were all very similar. Along the way, we ran into the problem with larger models converging and training slowly but fixed this by realizing that our model does not require that level of complexity and reducing the number of layers and nodes per layer.

IV. LITERATURE

Our dataset came from one of the many datasets available on the Kaggle competition website, and users used the data to predict prices and cuts. We also found other literature regarding the same problem that we tackled on this paper, specifically, predicting the price of diamonds based on various features using artificial neural networks.[2] However, they used a different dataset such as the Rapaport price list, International Gemological Institute (IGI) and European Gemological Laboratory (EGL) reports, and other Internet published data. Furthermore, they used certification as one additional feature besides the cut, clarity, color, and carat features that we used in our model. They stated that certification of a diamond implied an increase in price. In addition to artificial neural networks, they also used models such as Classification and Regression Trees (CART) and Chi-Square Automatic Interaction Detection (CHAID). For the neural network, they were able to achieve a test performance of 96.5% with 15 hidden layers and 300 parameters. Likewise, the CHAID test performance was 84.8% and the CART test performance was 85.6% which were both less accurate than the neural network as expected.[2] However, since they are using different datasets and different evaluation metrics, a direct comparison of our results is difficult to make. Nevertheless, the conclusion they reached were similar to ours in that the neural network was able to achieve better performance than other models on the prediction of the price of diamonds. Furthermore, we compared our results to Kaggle competitors who used various models such as gradient boosting regressors which achieved a log based test performance R2 Score of 98.9 and random forest regressors achieving 98.8.[3][4] Since our neural network log based test performance was 99.24, our model seemed to be as successful as other top models currently being used for this task.

V. RESULTS

In this section, we will discuss the varying degrees of success we observed choosing different features and model architectures as well as their performances on the diamond dataset. We will then discuss our results, compare it to alternative solutions, and analyze the different parameters involved to gain insights on their effect on performance.

A. Varying Features

We chose our final features by comparing performances of different features through a vanilla neural network. From the data analysis section, we showed that there is a very strong positive correlation between {x, y, z, carat} and {price}. Other features showed weak to virtually no correlation with price. Although there is no direct correlation that can be seen in data, other features could have information embedded in them that will help predict price with higher accuracy so we included them in. To gauge the importance of features and figure out what to include in our final model, we tried the following feature vectors with their performances on a 2 hidden layer neural network with fully connected layers of 100 and 100 nodes, respectively (trained until convergence via cross-validation):

Feature Performances	
Features	Performance (R2 Score)
{cu(1h),co(1h),cl(1h),d,t}	.5102
{ca,x,y,z}	.7946
{ca,d,t,x,y,z}	.8112
{cu(1h),co(1h),cl(1h),ca,d,t,x*y*z}	.9587
{cu(1h),co(1h),cl(1h),ca,d,t,x,y,z}	.9608
{cu(num),co(num),cl(num),ca,d,t,x*y*z}	.9762
{cu(num),co(num),cl(num),ca,d,t,x,y,z}	.9775

Note: All entires above that are not one-hot (1h) values are just raw numerical (num) representatives of given data.
Also - {cu:cut, co:color, cl:clarity, ca:carat, d:depth, t:table}

From above, a feature vector with just {carat,x,y,z} generates a decent predictor, however, the rest of the features are essential for adding details to improve predictions. We also notice that changing {cut, color, clarity} from one-hot representations to numerical ones improved performance. As a result, our final model will include all column metadata as features in each of their numerical forms.

It is clear that features that work well for our task are features with high correlations with the price. When the features that have the weakest correlation with price are chosen, we observe the worst results (top entry in the table above). Also, the improvement in performance from using numerical instead of one-hot is probably caused by each label having an intrinsic score. Take cut, for an example. When cut is represented with one-hot next to all other cut labels, it loses the intrinsic information about the diamond's quality which is definitely correlated with price. However, with a numeric representation, this quality can be expressed effectively, thus making this feature format more advantageous.

B. Varying NN Models

With the feature vector decided, we next chose the best NN architecture by training different networks and evaluating their performances. The table below presents our experiments:

Different NN Architecture Performances	
Layer Descriptions	Performance (R2 Score)
{100,100,50}	-0.897
{100}	.9608
{50,100}	.9762
{100,100}	.9762
{200,100}	.9764

Note: These describe the number of nodes per hidden layer. For an example, a NN described as 50,40 has all input nodes feeding into the first hidden layer of 50 nodes which then feeds into the second hidden layer of 40 nodes which feeds into the 1 output layer.

Interestingly, we found that deeper networks did not bode well for convergence and performance. We found the sweet spot in terms of network complexity to be 2 hidden layers. The 1 layer network we tried probably was not complex enough to represent more complicated hidden patterns inside data which showed in its lower R2 score. So, from empirical observations, we deduced that a 2 layer network was the most appropriate in terms of convergence time and model complexity to find meaningful patterns within our dataset without too much risk of over-fitting. Regarding 2 layer architectures, we found that have more nodes in the hidden layer added performance and reducing the nodes of the second layer improved training. This makes sense because the first layer is in charge of extractor all lower level features in the dataset which requires more nodes to represent. We implemented the best performing 2 layer model – the {200,100} network – as the architecture of our final model .

C. Best Model Performance

Combining the best performing features and NN architectures from above with various data-processing techniques described in 'Methods', we were able to achieve a state-of-art R2 score on this diamond dataset. Below, you can find the final performances as well as the training loss, R2 score of the training set, validation loss, and R2 score of validation set for every epoch during training.

Final Model Performance	
Description	R2 Score
Final Test Performance (Log Prices)	.99241
Final Test Performance (Raw Prices)	.98111
Final Validation Performance	.98170
Final Training Performance	.98259

Note: model was trained across 700 epochs with a batch size of 100. Initial learning rate was 0.004. Used Adams optimizer with a learning rate decay of 0.25 after every 50 epochs until convergence.

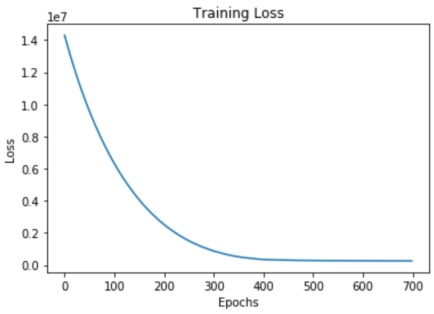

Fig. 6. Training Set Loss

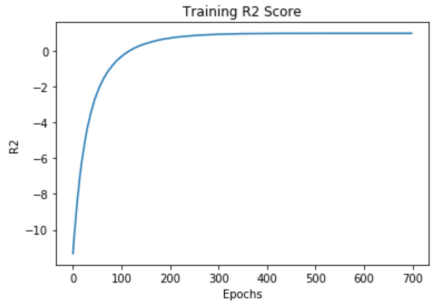

Fig. 7. Training Set R2 scores

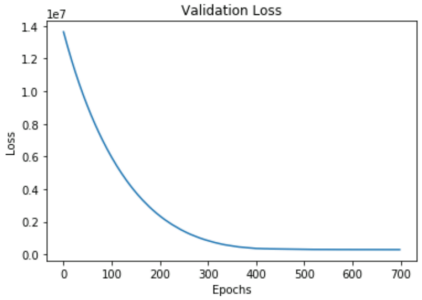

Fig. 8. Validation loss during training

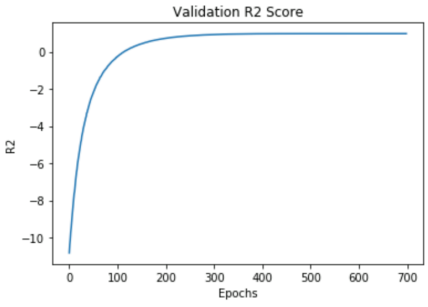

Fig. 9. Validation R2 scores during training

D. Discussion

Compared with the best results on Kaggle and the internet regarding this dataset [3], our model resulted in a superior R2 score on testing data it has never seen. The previous state-of-the-art model used a gradient booster ensemble to

Performance Comparison	
Model (Using Log Prices)	Test R2 Score
Our Model	.99241
Best Model on Kaggle	.98879
Linear SVR Baseline	-0.13001
Linear Ridge Regression Baseline	-0.14371

Fig. 10. Performance Comparison

predict diamond prices. Gradient boosters are extremely effective predictors; their use has grown tremendously and can be seen in countless winning Kaggle competition models. The fact that our neural network out-performed this robust model underlines the strengths that these networks have to learn patterns in data, generalize to unseen data, and create effective mappings. Neural networks are universal function approximaters and their power is shown in our results. The parameters of our model are the weights connecting layers within the NN and the biases for each node. Each set of weights and biases serve as a loose interpretation of how important a certain feature or abstraction is to the final price prediction. Negatives weights and biases might indicate that a certain properties brings down a price while positives weights and biases indicate the opposite.

REFERENCES

[1] Diamonds. Diamonds — Kaggle, Unknown, 25 May 2017, www.kaggle.com/shivam2503/diamonds.

[2] Cardoso, Margarida G. M. S., and Luis Chambel. A Valuation Model for Cut Diamonds. International Transactions in Operational Research, Blackwell Publishing, 7 July 2005, onlinelibrary.wiley.com/doi/10.1111/j.1475-3995.2005.00516.x/pdf.

[3] "Diamonds are hard" — Kaggle, Joseph Obarzanek, Novemeber 2017, https://www.kaggle.com/doodmanbro/diamonds-are-hard/notebook

[4] Greedy Function Approximation: A Gradient Boosting Machine. Author(s): Jerome H. Friedman. Source: The Annals of Statistics, Vol. 29, No. 5 (Oct., 2001), pp. 1189-1232. Published by: Institute of Mathematical Statistics. Stable URL: http://www.jstor.org/stable/2699986.