

Diamond price prediction algorithm

SDSE Final Report

Group1000: Dohyun Park, Sanggu Kang, Seungeun Hong

INTRODUCTION

Diamond is an exquisite gemstone that has captured fascination for its timeless elegance and enduring beauty. It is formed deep within the Earth's mantle under intense pressure and heat making it a rare, dazzling treasure that symbolizes love, commitment and luxury. Their unique characteristics make up a complex set of factors that influence their pricing in which experts face challenges to make accurate predictions. Certain features include carat weight, cut quality, color grading, clarity. In this project, the team aims to build a price prediction algorithm of diamonds that offers a strategic advantage in navigating the dynamic of the diamond market. Developing the model we expect to suggest a data-driven decision making process for the diamond industry.

PROBLEM DESCRIPTION

Developing an accurate diamond price prediction algorithm is an exquisite work that requires several factors of the stone itself and some external factors. Examples of external factors are market trends, economic conditions. However, turning these terms into a written figure is very complicated and needs a project-scale effort. Therefore, the team decided to focus on the internal aspects of a diamond in building an algorithm that predicts the price of a given data.

Features used in this project are carat, size of a diamond, table, depth, clarity, color and cut. Price is set as the target column in creating our model. Going through the process of data cleansing and modification, we proceed to machine learning where the team suggests 3 different models. These techniques are linear regression, decision tree regression and random forest regression. The goal is to find a model that optimizes the mean squared error, r-squared value. Throughout this report, we analyze the results of these algorithms and provide a thorough explanation behind our decision.

DATA

Data source: <https://www.kaggle.com/datasets/joebeachcapital/diamonds/data>

The original dataset consists of 10 features including various aspects of a diamond. Columns with numerical values are *carat*, *x*, *y*, *z*, *table*, *depth*, *price*. Columns named as *x*, *y*, *z* are values indicating the size of a diamond on a 3D *x*, *y*, *z* axis. *Table* is a value that indicates the facet which can be seen when the stone is viewed face up, the flat top surface. *Depth* is another size feature that shows the stone's height measured from the bottom tip to the flat top surface called the table. *Price* is used as the team's target value for the project. Others with non numerical values are *clarity*, *color*, *cut*. *Clarity* refers to how clear a diamond is. The fewer and less imperfections is considered a better clarity. And it is sorted in 8 levels from 'I1' indicating the worst clarity to 'IF' the best clarity. *Color* refers to the color of a diamond and it is classified into 7 different colors. Letters 'D' to 'F' are considered colorless while 'G' to 'J' represent a very faint color. Hence, in alphabetical order, a diamond is considered to have a higher value. *Cut* refers to how a diamond is shaped. A better cut creates a symmetrical and luminous diamond. It is categorized into 5 levels; 'Fair', 'Good', 'Very Good', 'Premium', 'Ideal'.

The team decided to use all the features of the given dataset for the following reasons. First, the dimension of the dataset is only 10 which is just enough to conduct several machine learning methods. Also, all of the columns are key aspects of a diamond which directs to determining the value of a diamond. Lastly, the team expects to seek meaningful correlation of certain features such as *x*, *y*, *z* that are related to each other.

Before building an algorithm that predicts the price of a given diamond data, the team went through pre-processing to cleanse the dataset. We start off by checking for null values in part 3 and it is shown that there are no null values in the original dataset. In part 4 we modified the dataset by encoding non numerical values, *clarity*, *color*, *cut* into numerical values and kept the categorical aspect of these variables. This enables us to proceed and build a linear regression model. Next, we discover the basic statistical information for each feature in part 6. This action helps us recognize how these statistical figures of each variable greatly vary, in which sets as a guideline to whether the dataset should be standardized or not. All values except *depth* and *table* have a similar scale so the team decided to conduct the model without standardization. In part 7, with a glance of the distribution for all features in every possible pair, the team noticed a meaningful correlation between certain features. Hence, we further processed by creating a correlation heatmap for all features. Image shown under part 8 is the correlation heatmap created in calculation of a correlation matrix. Areas marked as red are interpreted that the two columns have a strong correlation while areas colored as blue indicate that there is weak or no correlation between two columns. With this matrix, the team noticed a strong correlation between the following pairs.

Carat and *x*, *carat* and *y*, *carat* and *z*, *carat* and *z*, *x* and *y*, *x* and *z*, *y* and *z*. The result seems reasonable considering that carat, weight of a diamond, is greatly affected by the size of the stone and *x*, *y*, *z* determining the size of it. Through part 3 to 8 the dataset is cleansed into a form which allows for the team to conduct several machine learning techniques.

METHODOLOGIES

In part 1, we imported all the modules that we needed in this project. This includes one of the most commonly used modules such as numpy, pandas. Also we imported seaborn and matplotlib.pyplot to draw the graphs of various columns. To make modification on non numerical columns we imported OneHotEncoder from sklearn.preprocessing. In the machine learning phase, we used train_test_split module from sklearn.model_selection to partition the dataset into a training set and a test set, mean_squared_error and r2_score from sklearn.metrics to compute the values of mean squared error and r squared score to determine the accuracy of our model. Lastly, we imported LinearRegression module from sklearn.linear_model, DecisionTreeRegressor from sklearn.tree and RandomForestRegressor from sklearn.ensemble each to develop an algorithm using methods of linear regression, decision tree regression and random forest regression.

Since the methods we are implementing are all supervised machine learning, the team divided the dataset into target *y* value and others in part 9. Also we partitioned the dataset into a training set and test set with the test set ratio of 0.3.

Our first model is linear regression, part 10 in the code. We chose this method as our first approach because of the linearity noticed in the graph under part 7 of the code. If you take a closer look at the graphs you will notice that there is a linear relationship between certain *x* features and the price column which is our target value *y*. In addition, we can observe a relevant correlation between the price column and some *x* values in the correlation heatmap, shown below part 8 in code.

The second model in our training process is the decision tree regression. The dataset consists of 9 *x* values which is considered as a decent number of variables to implement this methodology. The given dataset is not a subset of discrete categories which makes it impossible to use a classification method. To form an algorithm tree-like structure where each internal node represents a feature and each leaf node contains a numerical value, the team chose decision tree regression. Most importantly this algorithm is valuable in handling non-linear patterns. Hence, we expect this way of approach could compensate for the results obtained within linear regression.

Lastly we used random forest regression as a way to improve the results we gained using decision tree regression. This algorithm is an ensemble learning method that leverages the strength of multiple decision trees that mitigates overfitting, a common issue that a decision tree model faces. Key factors that allow this model to produce more accurate predictions lies in combining the outputs of multiple trees instead of stacking the results in one tree. By doing this the model becomes more resilient to outliers and noise in the dataset. Additionally, the randomization introduced in both data sampling and feature selection is the key in reducing the risk of overfitting which in turn makes this model a powerful tool for predicting continuous outcomes.

RESULTS AND ANALYSIS

In part 10 of our code, we can observe the results for our first model, linear regression. Evaluation measures are mean squared error(MSE) and r-squared on the training set and r-squared on the test set. Mean squared error for this model is 1944007.58, r-squared on training data and on test data is respectively 0.87 and 0.88. As mean squared error denotes the average of squared error for all dataset and the linear line the model produced, the model is considered as a ‘good’ model with a lower figure of MSE. While there is no absolute threshold that determines the performance of a model, the figure we obtained using linear regression seems too high to suggest that this model is a ‘good’ algorithm. Same applies on a r-squared perspective. There is no threshold set in determining whether a model is ‘good’ or ‘bad’. In this case, we get 87% of r-squared value on the training data and 88% on the test data. With this result, we can interpret that our linear regression model explains about 87% variance of the original dataset. The difference between the consequences of r-squared values of the training dataset and the test dataset comes from the random partition of training data and test data. The figure 87% seems to be a valid value to evaluate the model as ‘good’ because it explains most of the variance of the original dataset as well as seeming to be far from overfitting.

To visualize the results of this method the team drew a graph in part 11 of code that shows the original dataset into scattered dots with a linear line we created on top. Blue dots on the graph represent the actual values of the dataset and the red line represents the linear line we created. With the red line centered at the distribution of blue dots, we can intuitively conclude that this is a decent model.

In part 12, we can observe the results for the decision tree regressor method. Mean squared error is 396588.61, r-squared on training data and on test data is 0.98 and 0.97, respectively. MSE for this model is roughly 20% of the first model indicating that the decision tree regressor is a better model in a MSE

evaluation perspective. R-squared on the training data and on the test data supports this interpretation. With the value of 98% on the training dataset and 97% on the test dataset, this model explains almost the whole variance of the original dataset. With MSE holding a less value and r-squared greater value than the ones of a linear regression we might conclude that decision tree regressor is a better model. However, the value of r-squared almost reaching 100% waives us from making final decisions. While it is considered that a model is 'good' as r-squared increases, it is important to ensure that a model is not overfitted. But in this case where r-squared is at great value, it seems that the model is overfitting. Therefore, the team decided to conduct a random forest regression, which is known to mitigate the overfitting issue of a decision tree regression.

In part 13, we can observe the results for the random forest regression. Mean squared error is 293108.52, r-squared on training data and on test data is 1.00 and 0.98, respectively. MSE for this model is roughly 15% of the first model and roughly 70% of the second model. This calculation indicates that the random forest regression model is a 'better' model than the linear regression and decision tree regression model in terms of MSE evaluation measures. Same applies on a r-squared perspective. At a glance it might seem that the last model exceeds performance of the previous models since it has less value of MSE and greater value of r-squared, which is universally known as a condition of a 'good' model. Yet it is doubtful to conclude that the model is 'better' than the previous models since the r-squared on the training data is 100%. When a model obtains a r-squared value of 100%, we can confidently conclude that the model is overfitted. Therefore this last model is greatly overfitted.

Comparing the results of these 3 models, the team balanced out the evaluation measures in context to conclude which model is considered 'better' than others. Decision tree regression model and random forest regression model is conceived as overfitting which is a critical error to say that these models are 'good'. This in turn makes the linear regression model the only candidate in deciding whether it is a 'good' model or not. Although the MSE of the linear regression model is greater than other models, we could improve the model by standardizing the columns in which the mean of its value greatly varies from other variables.

CHALLENGES

There are several challenges the team has dealt with and that could be improved in future approaches.

First, as mentioned at the last paragraph of the ‘results and analysis’ part, we could have obtained lower MSE in the linear regression model which would have supported our final decision of choosing the first model as the best algorithm.

Secondly, seen in the correlation heatmap in part 8 of the code, there is a strong correlation between the columns of x , y and z . Initially, the team decided to proceed without any modification to adjust the relation these variables share because the original dataset only had 9 features. By reducing the dimension, the team thought that it would result in building an inaccurate model. After comparing the results of 3 different models, we could conclude that the evaluation measures could be improved by conducting a principal component analysis(PCA) of the highly correlated variables. In future projects, we could conduct a PCA on columns x , y and z and group these features into a new column which represents the size of a diamond and merge it into the given dataset. With the action of grouping the highly correlated features into one, we expect to obtain a better linear regression model.

Furthermore, the result of r-squared measures for decision tree regression model and the random forest regression model seems off in a theoretical view. As a random forest regression model is a ‘forest’ of decision trees, it is expected to have more accurate results and mitigate the issues that a decision tree regression model has. However the r-squared measure on the training data has increased from 98% to 100% in the second model to the last model. This indicates that the random forest model is even more overfitted compared to the decision tree model. In future attempts, this could be solved by partitioning the dataset into a different ratio or setting various maximum depths for the decision tree regression.

Lastly, analyzing the graphs on the distribution of the data, we could notice a linear relation on values x , y , z and the price column. This could be interpreted as reasonable yet could have been achieved rather on an intuitive approach. In future attempts, we expect to make progress by adding external factors such as market trends, economic trends and couple other internal factors of a diamond.

CONCLUSION

In conclusion, the development and implementation of a diamond price prediction algorithm represents a pivotal stride towards navigating the complexities of a diamond market. With advanced machine learning techniques and making the best of the internal factors of a diamond, the team hopes this project suggests a guide in price prediction and to the market flow. However, it is crucial to acknowledge the challenges and make improvements in future attempts by using updated datasets or even adding external factors.