I am analyzing the diamond dataset from Kaggle. My goal is to figure out which linear regressor model can predict a diamond's price the best given the features in the dataset. First, I need to see which features are helpful for predicting the price and remove outliers and incorrect data. After that, I can use cross validation to test the accuracy of various models and use metrics such as mean absolute error, mean root squared error and r2 to compare them. I use the pandas, NumPy, matplotlib and sklearn python libraries.

The diamond dataset has 53940 rows and 11 columns. A lot of context is given on the Kaggle page. The columns include "Unnamed: 0", an index counter, "carat", the weight of the diamond, "cut", the quality of the cut, "color", the color of the diamond, "clarity", how clear the diamond is, "x", "y", and "z", which refers to the length, width, and depth of the diamond respectively, "depth", a percentage calculated by dividing depth by the mean of length and width, "table", the width of the top of the diamond relative to the widest point' and "price", the price of the diamond. The cut, color, clarity are all categorical variables, and the rest are numerical. The Kaggle page describes the hierarchy of the possible values each of those variables can have. This makes it simple to change these categorical columns to numerical using label encoding. I do this by mapping each value of those variables to a number where the worst value is 0 and the better values have higher numbers. After that, I create a correlation coefficient matrix of the edited dataframe. The columns x, y, z, and carat all have high (above .85) correlation with price and with each other. Excluding the index column, cut, color, clarity, depth, and table all have low (below |.175|) correlation with price. I use scatterplots to explore the relationship between depth and price, and table and price. Both plots show that diamonds with the same depth or table could have prices that range from near $0 to above $17500. This is true for most of the common values depth and table could have. There is clearly little to no

correlation there. The categorical columns I encoded have discrete values so I don't think a scatterplot would be a good visualization to use. I decide to create visualizations showing the inter quartile range (IQR) of each possible value of the categorical variable as well as the IQR for the middle 50% of carat, x, y, and z values. The IQR for the middle 50% of the carat, x, y, and z values never exceed $2418. All but two of all the possible values for the categorical variables have IQRs above $2418. Besides 40% of the diamonds having an "Ideal" cut, all the other categorical values represent between 3% and 25% of the dataset. The main point is to show that the categorical variables have a huge spread, making it a terrible predictor for price. After that, I display scatterplots of carat, x, y, and z vs price to look for outliers and to show the relationship between them. 20 rows either have an x, y, or z value of 0. These are clearly incorrect, and they get deleted. The carat and x columns have no huge significant outliers. Y and Z have some values above 30 and below 2 that are outliers, and those rows get removed as well. The depth column is based on the x, y, and z columns. This makes it easy to tell which rows have seemingly incorrect data. All of the rows I deleted have a clear mismatch between z/mean(x,y) and the depth shown in the dataframe. Then, I decide to create a new feature called volume, which is the product of x, y, and z. These columns have a huge correlation with each other and refer to the size of the diamond. I also show that the correlation coefficients of y and z slightly increased after removing the few outliers. Carat and volume have a .92 r2 score with price, higher than all the other columns.

Carat and volume are the only features I use to build my linear regressor models. I create a train subset and a test subset using the train_test_split method from sklearn. I looked through the sklearn library and found 10 linear models I can compare using the diamond dataset. These 10 models are regressors, need no customer parameters, unique from each other, and they don't

crash my program. The Ridge and Lasso regressors were too similar to OLS linear regression so I kept OLS linear regression and did not use Ridge or Lasso. The 10 regressor models I chose were Linear, Decision Tree, Huber, K-Nearest Neighbors, Passive Aggressive, Random Forest, Gradient Boosting, AdaBoost, Bagging, and Extra Trees. The last 5 models are ensemble models. I use a loop to perform cross validation with 10 folds on all of the models and to get metrics such as root mean squared error (RMSE), mean absolute error (MAE), and r2. I put all of this data into a dataframe and visualize the data using various bar charts. The top three of each metric are all ensemble models. The worst performing model is the Passive Aggressive model. This makes sense since the model is made for input data that comes in sequentially and the model gets updated step by step. In this case, the entire dataset is used at once. Gradient Boosting is clearly the best linear regressor model as since it has the highest r2 score, lowest RMSE, and lowest MAE.