CIS4930 - 16903
Summer 2025

EarthMover

Heavy Learning for Deep Impact

Introduction to Machine Learning

CIS4930 - 16903

Contents

Summer 2025

Contents

Meet the team

Diego Stecca
Team Leader
(El Capitan)

Oversaw project direction, coordinated overall progress, and facilitated problem-solving discussions during team meetings.

Daniel Lipszyc
Vice Team Leader

Supported Diego in managing tasks, maintained meeting agendas, and stepped in to guide technical discussions when needed.

Meynard Guillermo
Meeting Coordinator

Organized and scheduled focused sub-team sessions (2–3 members) to tackle discrete milestones; this approach maintained daily progress and made collaboration efficient and enjoyable.

Danilo Inestroza
Data Storyteller

Transformed our Jupyter Notebook into a clear, narrative-driven report by reorganizing headings, adding concise explanations, and designing illustrative visuals, making complex results accessible to non-technical stakeholders.

Carlos Felipe
Wall Breaker

Led investigative efforts whenever we encountered blockers—debugging lengthy GridSearchCV runs, identifying performance bottlenecks, and sourcing advanced techniques to improve model accuracy.

CIS4930 - 16903

Goal:
Bulldozer Price Prediction

Project Objective:

- Forecast auction sale prices of used bulldozers
- Employ supervised regression models
- Generate reliable price estimates to guide buyers & sellers

Summer 2025

Dataset Overview

Blue Book for Bulldozers dataset

Contains historical auction records alongside detailed equipment specifications

Source

https://www.kaggle.com/datasets/farhanreynaldo/blue-book-for-bulldozer

Number of Records

401,125

Number of Features

53

Data types

datetime, float64, int64, object

```
<class 'pandas.core.frame.DataFrame'>
Index: 401125 entries, 205615 to 400217
Data columns (total 53 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   SalesID                   401125 non-null  int64
 1   SalePrice                 401125 non-null  int64
 2   MachineID                 401125 non-null  int64
 3   ModelID                   401125 non-null  int64
 4   datasource                401125 non-null  int64
 5   auctioneerID              380989 non-null  float64
 6   YearMade                  401125 non-null  int64
 7   MachineHoursCurrentMeter  142765 non-null  float64
 8   UsageBand                 69639 non-null   object
 9   saledate                  401125 non-null  datetime64[ns]
 10  fiModelDesc               401125 non-null  object
 11  fiBaseModel               401125 non-null  object
 12  fiSecondaryDesc           263934 non-null  object
 13  fiModelSeries             56908 non-null   object
 14  fiModelDescriptor         71919 non-null   object
 15  ProductSize               190350 non-null  object
 16  fiProductClassDesc        401125 non-null  object
 17  state                     401125 non-null  object
 18  ProductGroup              401125 non-null  object
 19  ProductGroupDesc          401125 non-null  object
...
 51  Differential_Type         69411 non-null   object
 52  Steering_Controls         69369 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(6), object(44)
memory usage: 165.3+ MB
```

Dataset Overview (cont.)

Feature Relevance

1

Identification & Source

{SalesID, MachineID, ModelID, datasource, auctioneerID}

These features track each sale record, the specific machine, its model family, & the data source, showing systematic differences across auction houses

2

Usage & Condition

{YearMade, MachineHoursCurrentMeter, UsageBand}

Capture the machine's age, total hours of operation, & a binned usage category to model nonlinear depreciation and wear effects on value

3

Machine Specifications & Attachments

{<all other features>}

Detailing the bulldozer's powertrain, chassis, optional equipment, & configuration variants helps explain how design choices & extra attachments affect prices

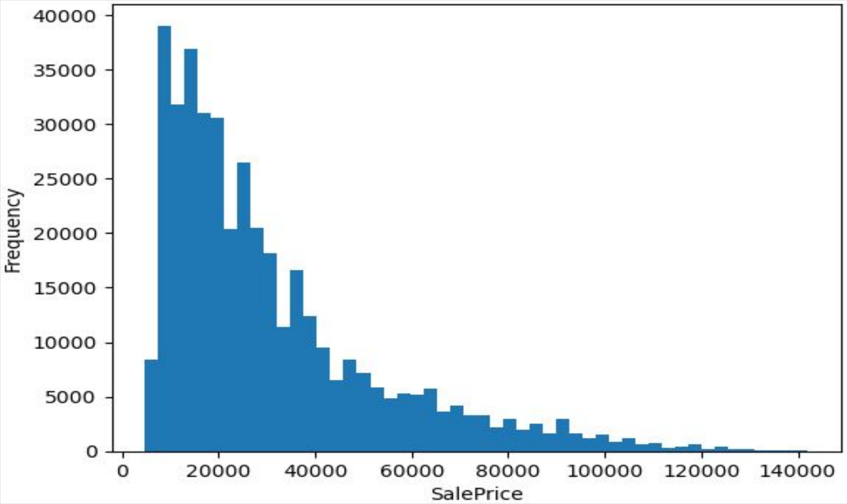
4

Prediction Target

{SalePrice}

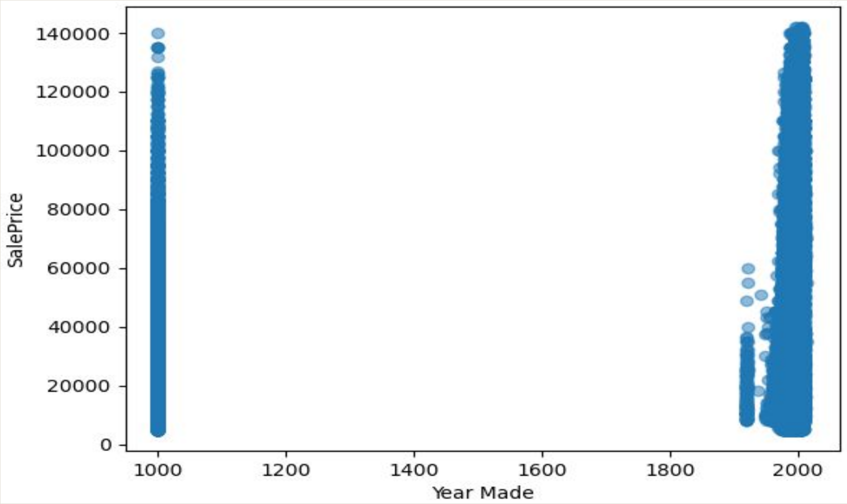
Outcome we aim to predict, with all other features serving as supporting variables

Exploratory Data Analysis (EDA) - Data Visualization



Distribution of SalePrice

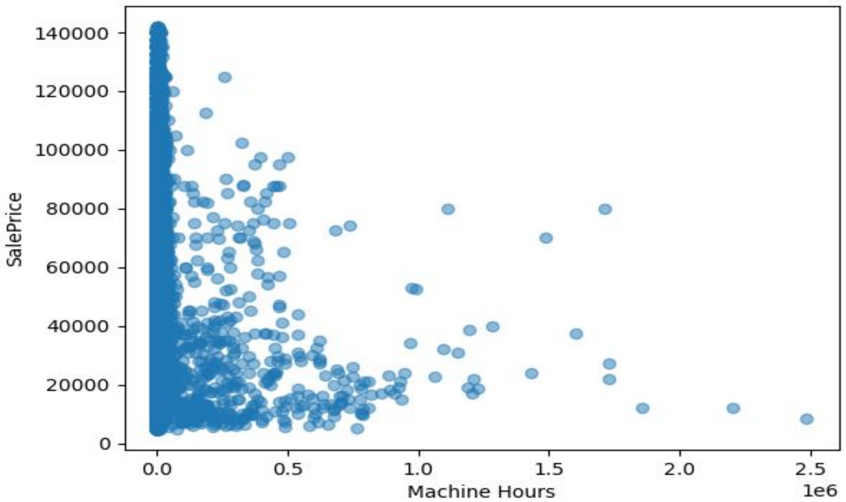
Histogram reveals that low sale prices are most common & steadily declines as prices rise



SalePrice vs YearMade

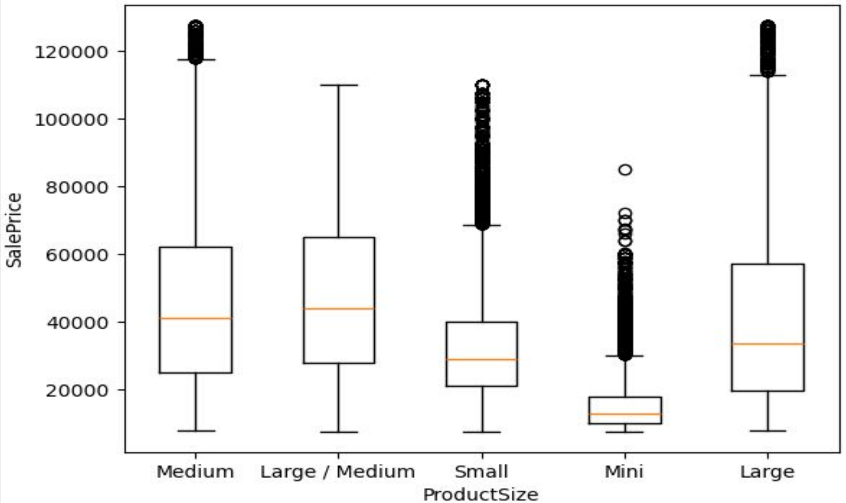
Erroneously records bulldozers as early as the year 1000 despite their 1st appearance in 1929, so we will list all unique YearMade entries to identify any further anomalies

EDA - Data Visualization (cont)



SalePrice vs MachineHoursCurrentMeter

Plot shows that machine usage substantially influences sale price & reveals that the dataset is predominantly comprised of nearly new bulldozers



SalePrice by Top 5 ProductSizes

Sale prices and spread grow with ProductSize: Small/Mini are low and tight, while Large/Large-Medium show the highest medians, widest ranges, and outliers.

Data Cleaning

Step 1

Drop columns of that have at least 85% null values.

Step 2

Scale the numerical values.

Step 3

Converting object-typed features into categories

Step 4

Fill in numerical missing values with median

Step 5

Convert categories into numerical values

Goal

To ensure data is accurate, complete, and consistent for reliable analysis

```
<class 'pandas.core.frame.DataFrame'>
Index: 401125 entries, 205615 to 400217
Data columns (total 42 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   SalesID                   401125 non-null  int64
 1   SalePrice                 401125 non-null  int64
 2   MachineID                 401125 non-null  int64
 3   ModelID                   401125 non-null  int64
 4   datasource                401125 non-null  int64
 5   auctioneerID              380989 non-null  float64
 6   YearMade                  401125 non-null  int64
 7   MachineHoursCurrentMeter  142765 non-null  float64
 8   UsageBand                 69639 non-null   object
 9   saledate                  401125 non-null  datetime64[ns]
 10  fiModelDesc               401125 non-null  object
 11  fiBaseModel               401125 non-null  object
 12  fiSecondaryDesc           263934 non-null  object
 13  fiModelDescriptor         71919 non-null   object
 14  ProductSize               190350 non-null  object
 15  fiProductClassDesc        401125 non-null  object
 16  state                     401125 non-null  object
 17  ProductGroup              401125 non-null  object
 18  ProductGroupDesc          401125 non-null  object
 19  Drive_System              104361 non-null  object
...
 40  Differential_Type         69411 non-null   object
 41  Steering_Controls         69369 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(6), object(33)
memory usage: 131.6+ MB
```

Models **WITHOUT** GridSearchCV

Linear Regression:

Normal, Lasso, Elastic Net

For linear regression we included the base model as well as one with Lasso and one with Elastic Net regularization.

Decision Tree

This model was chosen for its interpretability and ability to capture non-linear patterns. It performed significantly better than linear regression, as it could model more complex relationships in the data. However, it also showed signs of overfitting.

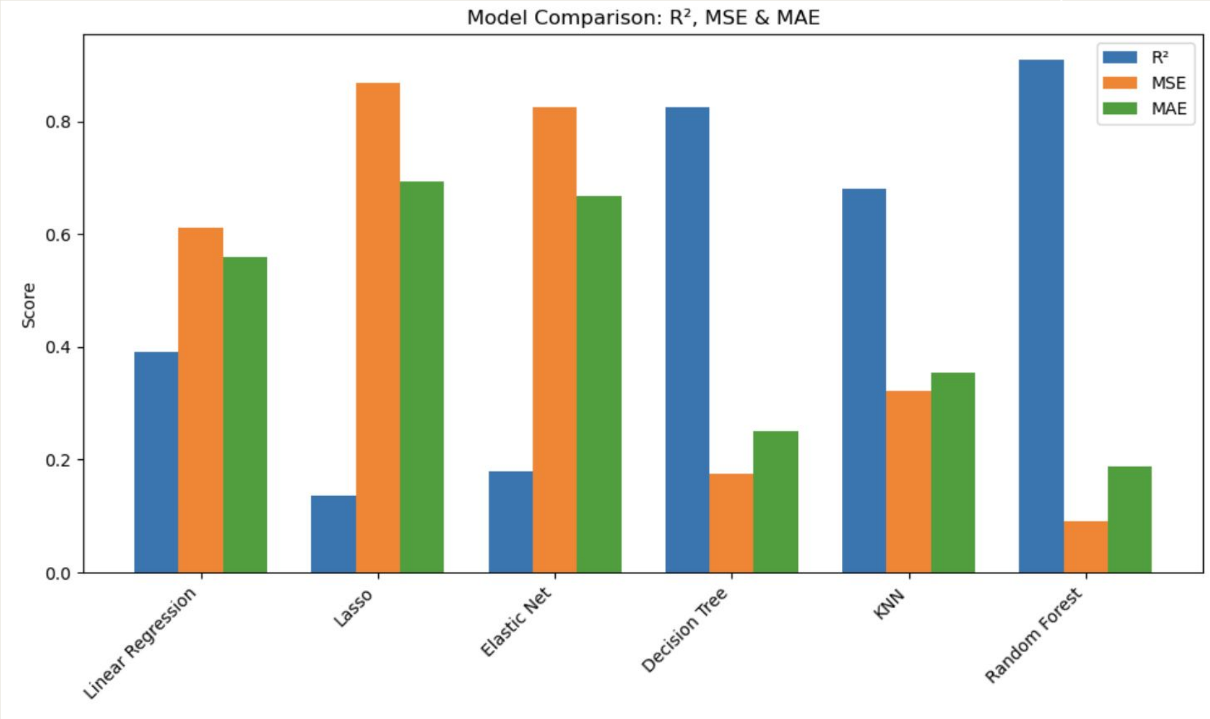
KNN

KNN was used to assess performance based on proximity-based decision-making. It doesn't make assumptions about the data distribution and can perform well in capturing local patterns.

Random Forest

We chose to include this alongside the decision tree to contrast their results and demonstrate how the ensemble method performs better. The difference isn't massive but as the random forest is intended to reduce overfitting, it did so on our data, Improving the R2 score and decreasing error.

Evaluation Metrics **WITHOUT** GridSearchCV



Linear Regression Model

R^2 Score: 0.392

Mean Squared Error (MSE): 0.612

Mean Absolute Error (MAE): 0.559

Decision Tree Regression Model

R^2 Score: 0.826

Mean Squared Error (MSE): 0.175

Mean Absolute Error (MAE): 0.251

Lasso

R^2 Score: 0.136

Mean Squared Error (MSE): 0.869

Mean Absolute Error (MAE): 0.693

K-Nearest Neighbors

R^2 Score: 0.680

Mean Squared Error (MSE): 0.322

Mean Absolute Error (MAE): 0.354

Elastic Net

R^2 Score: 0.180

Mean Squared Error (MSE): 0.825

Mean Absolute Error (MAE): 0.667

Random Forest Regressor

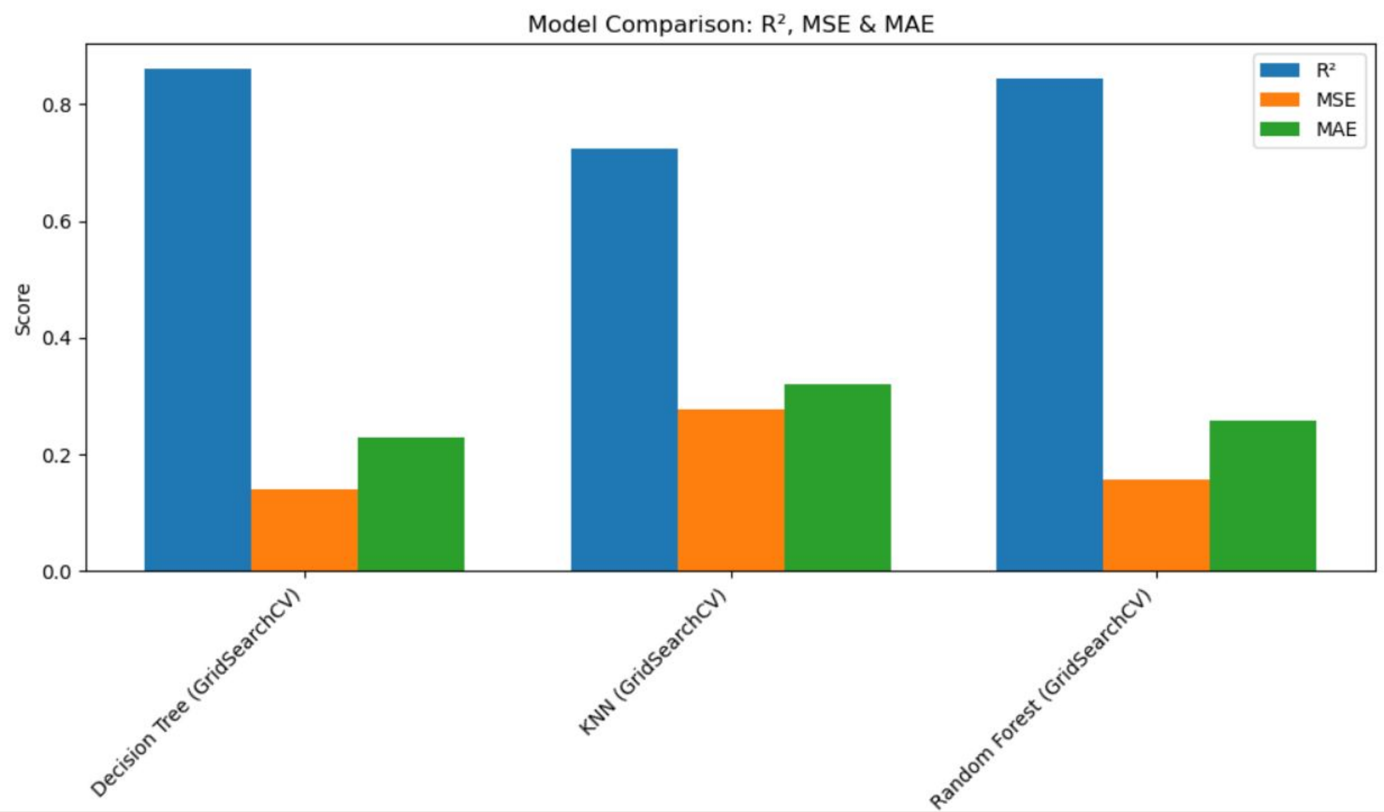
R^2 Score: 0.909

Mean Squared Error (MSE): 0.092

Mean Absolute Error (MAE): 0.188

Evaluation Metrics **WITH** GridSearchCV

Rankings



R^2 Score: **(Higher is better)**

1st: Decision Tree (0.861)

2nd: Random Forest (0.845)

3rd: KNN (0.724)

MSE: **(Lower is better)**

1st: Decision Tree (0.140)

2nd: Random Forest (0.156)

3rd: KNN (0.277)

MAE: **(Lower is better)**

1st: Decision Tree (0.230)

2nd: Random Forest (0.256)

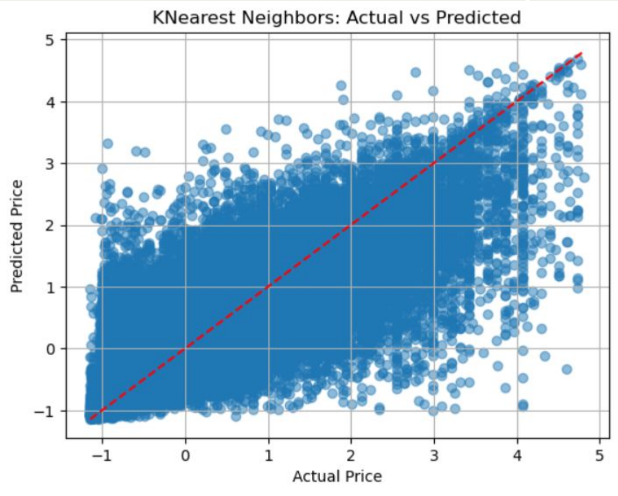
3rd: KNN (0.321)

Actual vs Predicted Price Scatter Plot

The *actual vs predicted price* scatter plot shows the model's predicted values versus the ground truth values.

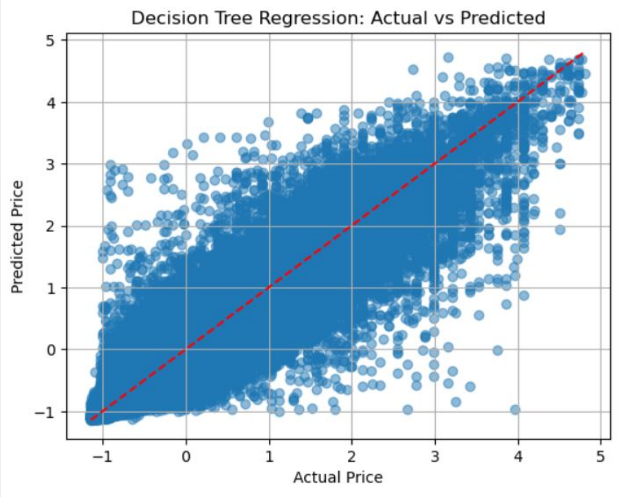
The optimal plot would be one with its points as close to the red line (y = x) .

The more optimal the plot is the higher its corresponding R^2 score is.



Random Forest (Below):

This model actually has a much lower spread than the plot for Decision Tree. However, it veers significantly below the red line. This results in a R^2 score that is a close second to Decision Tree (0.845 vs 0.861).

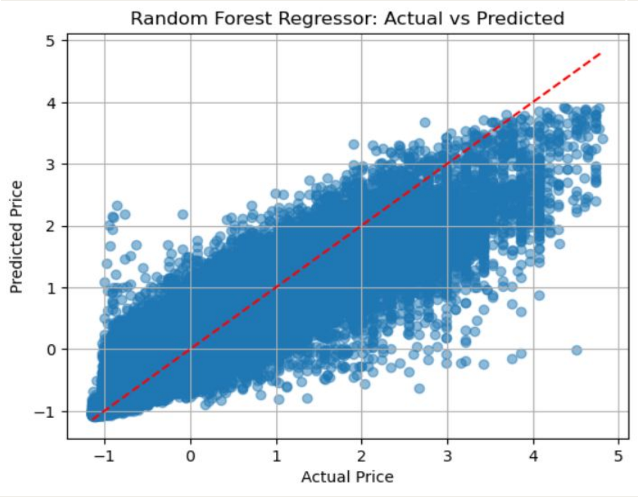


Decision Tree (Left):

This model has the most optimal graph of all three. Its points are all centered around the red line without veering to one side. As a result it has the highest R^2 score.

KNN (Above):

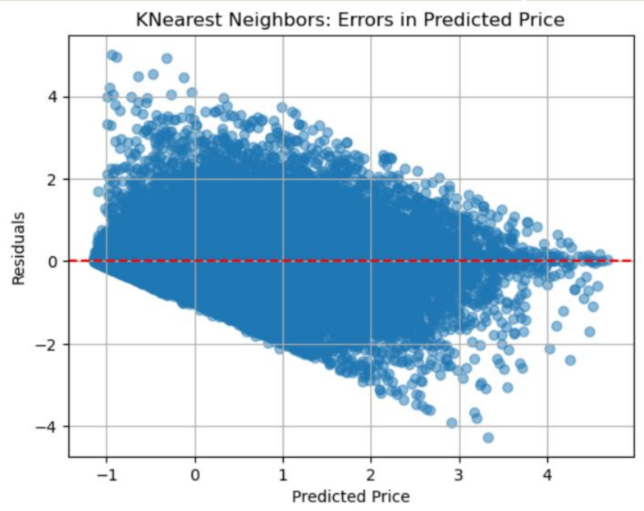
This model's points are very spread out and and veer below the red line. As a result it has the lowest R^2 score.



Errors in Predicted Price Scatter Plot

The *errors in predicted price* scatter plot shows the residuals or the difference between the predicted and ground truth value.

The optimal plot would be one with its points as close and symmetrical to the red line (y = x).



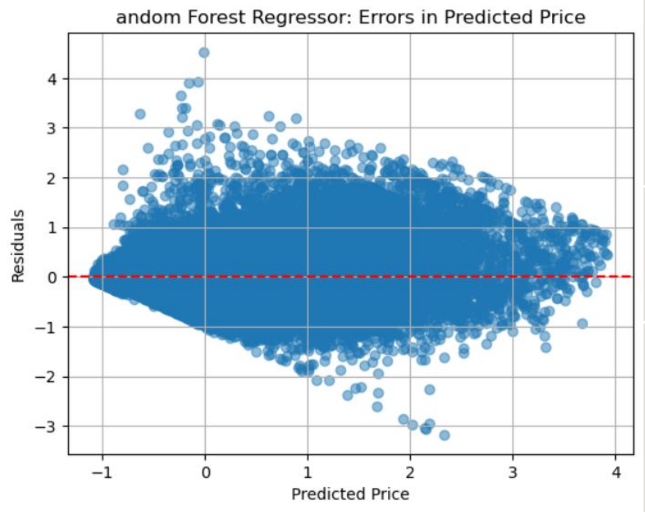
Random Forest (Below):

This had the lowest spread and most symmetrical to the red line. However, it becomes skewed on larger price residuals as seen before on the previous plots.

Decision Tree (Left) & KKN (Above):

Both have similar spread in errors with the only difference being that Decision Tree's errors are a bit closer to the red line.

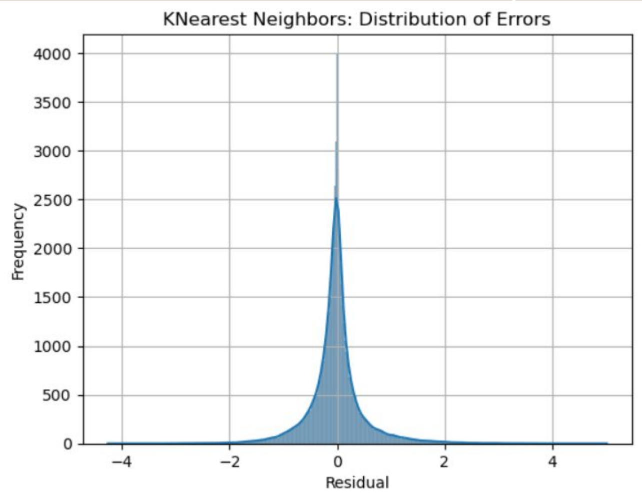
This means they had similar errors when predicting the price of the same test values.



Distribution of Errors Histogram Plot

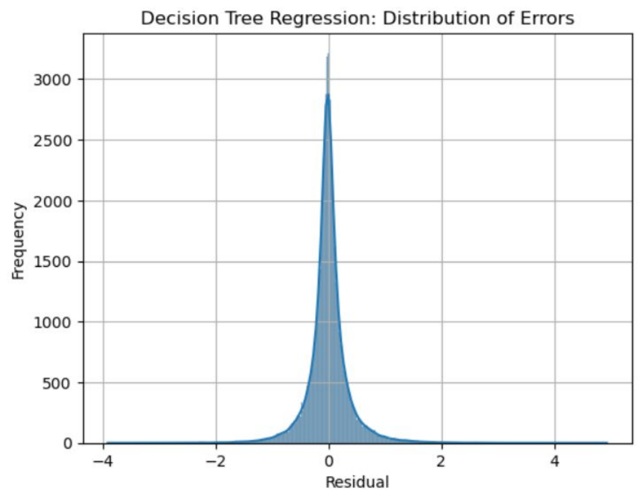
The *distribution of errors* histogram plot shows the frequency and spread of prediction errors.

The optimal plot is one that closely resembles the normal distribution centered at zero.



Random Tree (Below):

Distribution least like the normal distribution because it is heavily skewed to the left.

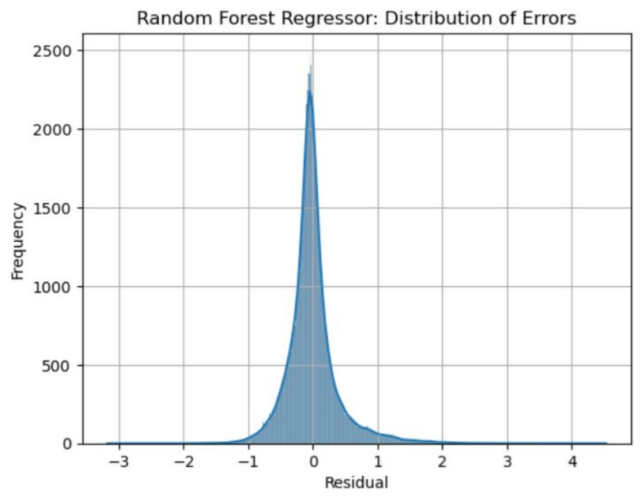


Decision Tree (Left):

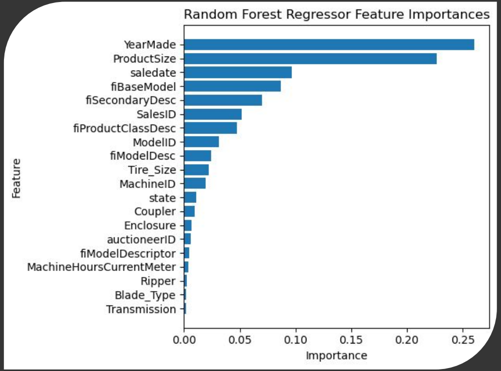
Has the distribution closest to a normal distribution and centered at zero.

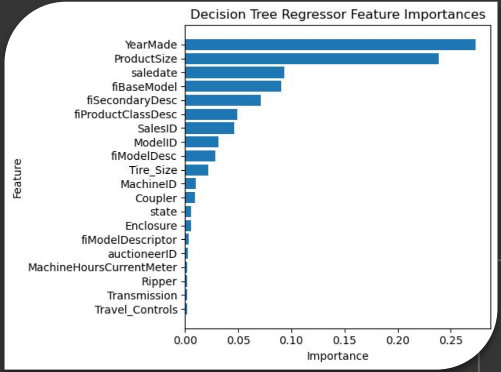
KNN (Above):

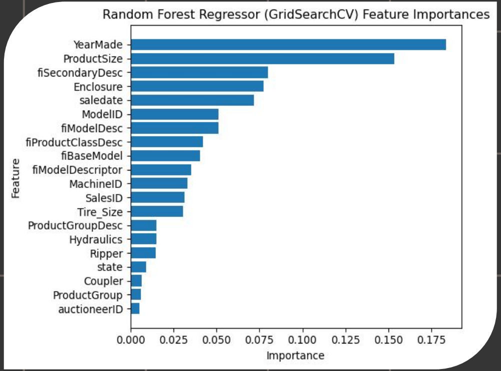
Also has a distribution that is pretty close to the normal distribution. However, it is slightly skewed to the left.



Feature Importance







Regular Random Forest & Decision Tree

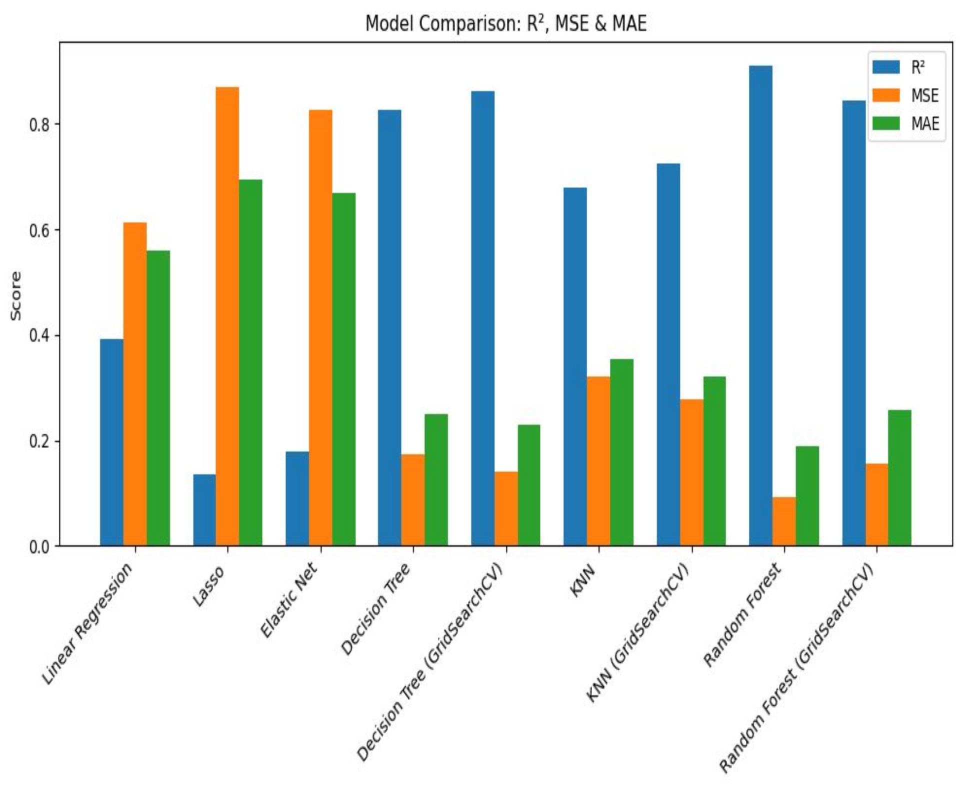
The regular random forest and decision tree did best and second best respectively. The features that were the most important to training the models were the same except a few lower ranked features. The most important features between the two being Year Made, Product Size, and Sale Date.

Random Forest (GridSearchCV)

Coming in third best, the Random Forest with GridSearchCV differed slightly with Year Made, Product Size, and Secondary Description (Sub-Model)

Analysis: Model Performance Summary (Ranked by R^2 Score)

	R-Squared	MSE	MAE
Random Forest Regressor	0.909	0.092	0.188
Decision Tree Regression (GridSearchCV)	0.861	0.140	0.230
Random Forest Regressor (GridSearchCV)	0.845	0.156	0.257
Decision Tree Regression	0.826	0.175	0.251
K-Nearest Neighbors (GridSearchCV)	0.724	0.277	0.321
K-Nearest Neighbors	0.680	0.322	0.354
Linear Regression	0.392	0.612	0.559
Elastic Net	0.180	0.825	0.667
Lasso	0.136	0.869	0.693



Analysis

Key Takeaways

1

Random Forest Regressor Top Performer

Random Forest (default) achieved the highest R^2 and lowest errors, indicating strong predictive power

2

Hyperparameter Tuning Boosts Decision Tree & KNN Performance

GridSearchCV noticeably improved Decision Tree and KNN results, demonstrating the value of hyperparameter tuning

3

Linear Models Underperform on Complex Data

Linear models (Lasso, Elastic Net) underperformed on this dataset, suggesting limited linear signal or need for stronger regularization

Reflection

Opportunities for Further Improvement

1

Tree-Based Tuning

Use RandomizedSearchCV or Bayesian optimization to expand and optimize tree hyperparameters like max_depth, min_samples_split, and splitting criteria

2

Random Forest Expansion

Broaden n_estimators, max_features, and bootstrap sampling, using out-of-bag scores for validation

3

KNN Feature Selection

Apply wrapper methods after filter steps to remove irrelevant features and sharpen distance metrics

4

Advanced Ensembles

Beyond Random Forests, investigating gradient-boosted trees or stacking ensembles could capture complementary strengths of multiple base learners

5

Neural Networks

Experimenting with deep learning models might uncover nonlinear patterns that tree ensembles miss

Reflection

Project Limitations & Challenges

Time and Compute Constraints:

Conducting an exhaustive grid search for the Random Forest over a wide hyperparameter space required over 12 hours of runtime—highlighting the balance between exploration depth and available computational resources.

Thank you