Used Vehicle Price Prediction

Navdeep Singh

College of Engineering and Physical Science, University of Guelph

Guelph, ON N1G 2W1 Canada

***Abstract* - In the car industry, the manufacturer sets the price of a new car, with the government incurring some additional expenditures in the form of taxes. Customers who purchase a new car may rest comfortable that the money they spend will be at market price without a concern for paying too much. However, due to the trend of rising automobile prices and a lack of demand at those rising prices, used car sales are on the rise all over the world. Because of this trend, there is a pressing need for a Used Vehicle Price Prediction system that can accurately analyze a car's worthiness based on a range of factors. The current system comprises a process in which a vendor sets a price seemingly at random, and the buyer has no knowledge of the car or its current value. Vendors have no idea what automobiles are worth or what price it should be listed for. To address this issue, we have devised a solution that will be effective in predicting the price of a used car accurately. A supervised machine learning model has been trained to achieve this task. Supervised learning is a type of machine learning in which machines are trained using well "labelled" training data, and are trained on the basis of that data to predict the output. Supervised learning can be further divided into two types of problems: (a) Regression (b) Classification. The algorithms we have used produce regressions because they produce a continuous value rather than a classified value as an output. Because of this, rather than predicting the price range of a car, it will be possible to anticipate the actual price of a car.**

***Keywords* - Random Forest Regression, Random Search CV (Cost Variance), Extra Trees Regression, Ridge Regression, Lasso Regression, Decision Tree Regressor, Random grid, K Neighbor Regressor, SVR, XGB Regressor**

I. Introduction

Due to the numerous elements that influence the price of a used vehicle on the market, determining if the quoted price of a used car is accurate is a difficult undertaking. The goal of this project is to create machine learning models that can properly forecast the price of a used car based on its attributes to allow buyers to make educated decisions. On a dataset consisting of the sale prices of various brands and models, we will develop and evaluate several learning approaches. We will examine the results of numerous machine learning algorithms, such as Linear Regression, K-NN Regression, Support Vector Regression, XGB Regression, Random Forest Regression, Extra Trees Regression, Ridge Regression, Lasso Regression, and Decision Tree Regressor, and pick the best one based on our evaluations of each algorithm. The car's pricing will

be determined based on several factors. Regression Algorithms are employed because they offer us with a continuous number as an output rather than a categorized value, allowing us to anticipate the specific price of a car rather than its price range. Pre-processed data is simulated into different regression techniques and each model has been evaluated to choose the best one.

II. Requirements

The following list represents the minimum requirements needed to install Enthought Python and associated applications:

- Modern Operating System:
    - Windows 7 or 10
    - Mac OS X 10.11 or higher, 64-bit
    - Linux: RHEL 6/7, 64-bit (almost all libraries also work in Ubuntu)
- x86 64-bit CPU (Intel / AMD architecture)
- PIP 2.7
- Jupyter Notebook
- Chrome
- 4 GB RAM
- 5 GB free disk space

Users will find that computers bought in recent years will meet (and usually exceed) these hardware requirements.

III. Methodology

In this system, there are two basic phases:

1. Training phase: Using the data in the data set, the system is trained to fit a model (line/curve) depending on the algorithm selected.

2. Testing phase: the system is given inputs and its functionality is tested. The precision is tested.

As a result, the data that is used to train or test the model must be accurate. Because the system is intended to detect and estimate the price of a used car, proper algorithms must be employed to complete the two jobs. Different algorithms were compared for their accuracy before being chosen for further use. The best candidate for the job was chosen based on our evaluations of the algorithms.

III.I. Objective

To create an efficient and effective model that estimates the price of a used car based on the inputs of the user. To obtain high precision among different models that has been used in predicting the price. Linear Regression, K-NN Regression, Support Vector Regression, XGB Regression, Random Forest Regression, Extra Trees Regression, Ridge Regression, Lasso Regression, and Decision Tree Regression techniques are used to train the model. The most important evaluation of the different models will be how accurate they are. Different comparison tools are used to contrast among various techniques used in this project. Accuracy scores of different models are used to select the best model.

III.II. Proposed System

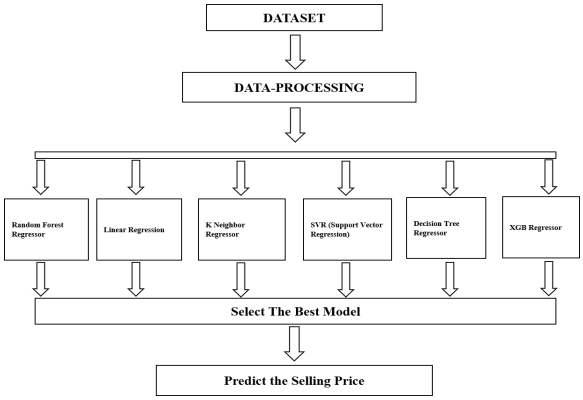


Fig 1. Model

The method begins by gathering the dataset, as illustrated in the diagram above. Data Preprocessing is the next phase, which involves data cleansing, data reduction, and data transformation.

Prices will then be predicted using a variety of machine learning algorithms. Linear Regression, Ridge Regression, Random Forest Regression, Decision Tree Regression, Lasso Regression, K Neighbor Regressor, SVR (Support Vector Regression), XGB Regressor are among the algorithms used. The model with the most accurate price prediction will be chosen as our conclusive model. Following the selection of the best model, the user is presented with a forecasted price based on their inputs. Model selection is based on *root mean square deviation.* The lesser the RMSE, the more accurate the model.

III.III. Dataset

vehicle_dataset.csv dataset comprises data from www.cardekho.com's used automobile listings. CarDekho.com is India's leading vehicle search engine, assisting users in finding the ideal car for them. This dataset provided the most easily available and appropriate data for our testing and training purposes. Expert evaluations, complete specs and prices, comparisons, as well as videos and photographs of all automobile brands and models available in India are all available on its website and app. The organization has partnerships with a number of automakers, over 4000 automobile dealers, and a number of financial institutions. This dataset has been downloaded from https://www.kaggle.com/c/vehicleprice.

This information can be utilized for a variety of applications, including price prediction, which demonstrates the application of linear regression in Machine Learning.

The following are the columns in the given dataset:

1. Car_Name - the name of the car
2. Year - car's manufacturing year
3. Selling_Price - selling price of a car in INR
4. Present_Price - present price of a car in INR
5. Kms_Driven - kilometers driven
6. Fuel_Type - type of fuel - [Petrol/ Diesel/ CNG]
7. Seller_Type - seller type - [Private/ Dealership]
8. Transmission - transmission type - [Automatic/ Manual]
9. Owner - owner - [first/second/third]

All prices provided from the kaggle dataset are in Indian Rupees. In this project, prices are converted into Canadian Dollar. The conversion rate used to convert the price from INR to CAD was 0.017. For example, if the selling price of a car is 875,000 INR then it will be equivalent to 14,875 CAD.

III.IV. Data Preprocessing

Data preprocessing is the procedure for preparing raw data for use in a machine learning model. It's the first and most important stage in building a machine learning model.

It is not always the case that we come across clean and prepared data when working on a machine learning project. Before doing any data-related activity, it is necessary to clean the data and format it. As a result, we use a data preprocessing task for this. The steps are as follows:

Getting the dataset

1. Importing libraries
2. Importing datasets
3. Finding Missing Data
4. Encoding Categorical Data
5. Splitting dataset into training and test sets
6. Feature scaling

III.IV.I. Getting the Dataset

The initial requirement for creating a machine learning model is a dataset, as machine learning models are built based on data. The data is made available from www.kaggle.com.[1]

III.IV.II. Libraries

We need to import several predefined Python libraries in order to execute data preparation with Python. These libraries are used for a variety of tasks. For data preprocessing, we will utilize the following libraries:

- pandas
- numpy
- matplotlib.pyplot
- seaborn
- ExtraTreesRegressor
- train_test_split
- RandomForestRegressor
- RandomizedSearchCV
- metrics
- LinearRegression
- KNeighborsRegressor
- SVR
- DecisionTreeRegressor
- XGBRegressor

III.IV.III. Independent and Dependent Variables

Next, the dataset has been split into dependent variables and independent variables. The independent variables will be the remaining feature set (year, model, etc.) as they are independent from the selling price, and the dependent variable for our purposes will be the selling price of the car as it is dependent on the features of the car.

III.IV.IV. Finding Missing Data

The next phase in the data preparation process is to deal with missing data in the dataset. If some data in our dataset is missing, it could create a significant challenge for our machine learning model so handling missing values in the dataset is required. There are mainly two ways to handle missing data, which are:

- By deleting a particular row
- By calculating the mean.

The ‘By deleting a particular row’ technique has been used to handle the missing data. The picture below illustrates that there are no missing values present in the dataset.

```
print(df['Transmission'].unique())
print(df['Owner'].unique())
print(df['Seller_Type'].unique())
print(df['Fuel_Type'].unique())

# checking any missing or null values and taking care of it.
df.isnull().sum()
# Note there are no null values
```

```
['Manual' 'Automatic']
[0 1 3]
['Dealer' 'Individual']
['Petrol' 'Diesel' 'CNG']
```

Out[9]:

Year	0
Selling_Price (CAD)	0
Present_Price (CAD)	0
Kms_Driven	0
Fuel_Type	0
Seller_Type	0
Transmission	0
Owner	0

dtype: int64

Fig 2. Missing values

Because there is no missing data, this problem can be ignored and not dealt with.

III.IV.V. Encoding Categorical Data

Categorical data is data which can be one of a limited number of fixed values. In our dataset there are two categorical variables: Transmission, and Fuel Type. Machine learning models work based on mathematics and numbers: if our dataset has a categorical variable, then it may create trouble while building the model. Therefore it is necessary to encode these categorical variables into numbers. The categorical transmission and fuel type is converted numerics as follows:

- Transmission: [Automatic, Manual] to [1,2]
- Fuel type: [Petrol, Diesel, CNG] to [1,2,3]

III.IV.VI. Splitting Dataset into Training and Test Sets

During the preprocessing necessary for machine learning, one must divide the dataset into a training set and a test set. This is an important step in data preprocessing because it allows developers to both train the model using the training set and estimate the performance and accuracy of the model using the test set.

To split the data set into training and testing, we have to select the splitting size. For this dataset, we choose 0.3 which means 30% of the data will be used as a testing set and the rest 70% will be used as a training set. This value was chosen because of the size of the dataset. Being a limited size with $n > 1000$, we were forced to use a splitting size higher than if we had access to more data. In the future, providing more data was made available, this value could be reduced to improve the training size (and therefore accuracy) of our models.

III.IV.VII. Feature Scaling

In machine learning, feature scaling is the final phase of data preprocessing. It is a strategy for standardizing the dataset's independent variables in a defined range. In feature scaling, we place all of our variables in the same range and scale so that none of them dominate the others. For scaling selling price and present price, we convert price on a scale of 0 - 100.

III.IV.VIII. Additional Features

Additional features are commonly engineered for data that machine learning models will be trained/ tested on. These features can provide better data for the models to be taught with, and can positively affect the results of their testing. One surrogate feature has been derived from existing features to improve the accuracy of the model. The difference between manufacturing year and current year has been computed to create a new feature called Old_Year. This feature keeps track of the number of years the car has been driven.

To analyze the correlation and causation aspect of the model's features, we have plotted a pairwise plot between each other. Pairwise plots show that the

feature Car_Name has no impact on the model. In other words, Car_Name has no correlation with target selling_price. For that reason, we dropped the column Car_Name from the dataset. The heat map shown in the jupyter notebook solidifies the conclusion about Car_Name and other features. It depicts that the Car_Name feature is irrelevant to the model whereas other features have some degree of positive/negative correlation with selling_price. With the help of ExtraTreesRegressor , we can see the importance of the features.

We can see that the present_price feature is most important in the model followed by fuel_type and seller_type.

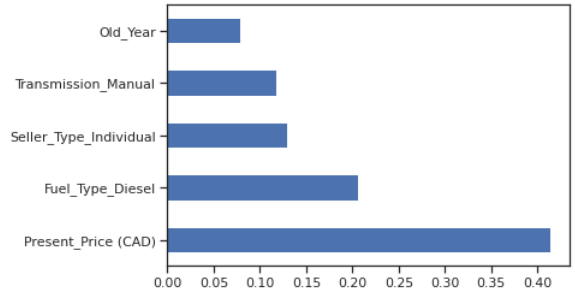


Fig 4. ExtraTreesRegression Feature Importance

III.V. Regressors

Supervised learning is a variety of machine learning in which machines are taught with well-labeled training data in order to predict output using input data similar to the training data. Because regression algorithms generate a continuous value rather than a categorized value as an output, prediction models are trained to use various regressors. As a result, rather than anticipating a car's price range, the regressors will estimate the car's exact price.

Following is a brief description of each regressor used in our proposed system of predicting price.

III.V.I. Linear Regression

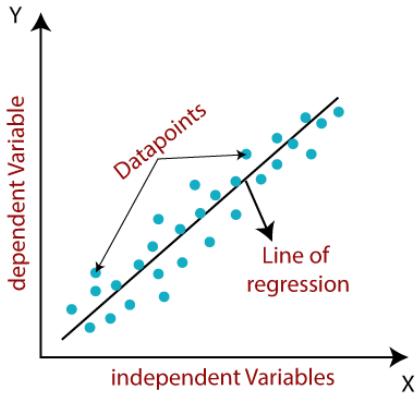


Fig 3. Linear Regression

One of the most basic and widely used Machine Learning methods is linear regression. Linear regression is a statistical technique for performing predictive analysis. The linear regression algorithm reveals a linear relationship between a dependent (y) variable and one or more independent (y) variables, thus the name. Because linear regression reveals a linear relationship, it determines how the value of the dependent variable changes as the values of the independent variables change.[2]

III.V.II. Random Forest Regression

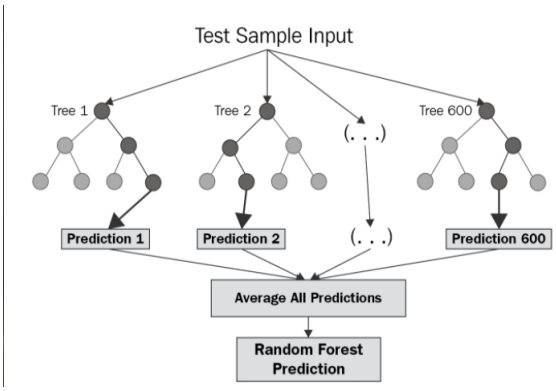


Fig 4. Random Forest Regression

A Random Forest is an ensemble technique that uses several decision trees and a technique, called "Bootstrap and Aggregation" or "bagging", to solve both regression and classification problems. Instead of depending on individual decision trees, the algorithm aggregates numerous decision trees to determine the final prediction.

As a fundamental learning model, Random Forest uses several decision trees. Row and feature sampling are done at random from the dataset, resulting in sample datasets for each model. This section is known as Bootstrap Aggregation.[3]

III.V.III. Decision Tree Regressor

In the shape of a tree structure, a decision tree constructs regression or classification models. It incrementally cuts down a dataset into smaller and smaller sections while developing an associated decision tree. A tree with decision nodes and leaf nodes is the end result of this regressor.

III.V.IV. Ridge Regression

Ridge regression is one of the varieties of linear regression that introduces a small amount of bias to improve long-term predictions.

Ridge regression, or L2 regularization, is a model regularization technique that reduces the model's complexity.[4]

The cost function is changed in this method by including a penalty term. The penalty of Ridge Regression is the degree of bias introduced into the model. We may determine it by multiplying the squared weight of each individual feature by the lambda.

In ridge regression, the cost function will be written as:

$$\sum_{i=1}^{M}(y_i - y'_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}\beta_j{}^2$$

Fig 5. Ridge Regression Cost Function

III.V.V. Lasso Regression

Least Absolute Shrinkage and Selection Operator is abbreviated as "LASSO." Lasso regression is a type of regularization. For a more accurate forecast, it is preferred over regression approaches. Shrinkage is used in this model. Data values are shrunk towards a central point known as the mean in shrinkage. Simple, sparse models are encouraged by the lasso approach (i.e. models with fewer parameters). This type of regression is ideal for models with a lot of multicollinearity or when you wish to automate elements of the model selection process, such as variable selection and parameter removal.[5] The equation for the cost function of Lasso regression will be:

$$\sum_{i=1}^{M}(y_i - y'_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}|\beta_j|$$

Fig 6. Lasso Regression Cost Function

III.V.VI. Support Vector Regression

Support Vector Regression (SVR) is a regression technique that uses the same principles as SVM. On the basis of a training sample, the aim of regression is to identify a function that approximates mapping from an input domain to real numbers. So let's take a closer look at how SVR truly works.

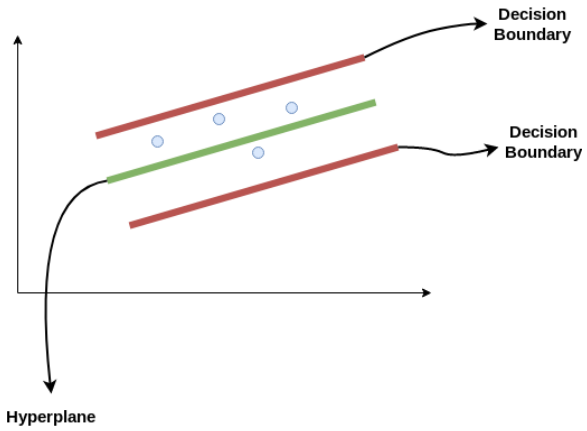


Fig 7. SVR

Consider the decision boundary to be the two red lines, and the hyperplane to be the green line. When we move forward with SVR, our goal is to essentially consider the points that are within the decision boundary line. The hyperplane with the most point and furthest distance from the decision boundaries is the best fit line for us.[6]

III.V.VII. XGB Regressor

Extreme Gradient Boosting is a type of ensemble machine learning that can be used to solve classification and regression predictive modeling issues.

Decision tree models are used to create ensembles. To repair prediction mistakes caused by past models, trees are introduced to the ensemble one at a time and fitted. The boosting model is a sort of ensemble machine learning model.

Models are fitted using a gradient descent optimization approach and any arbitrary differentiable loss function. Gradient boosting gets its name from the fact that the loss gradient is minimized when the model is fitted, much like a neural network.[7]

III.V.VIII. K-NN Regression

KNN regression is a non-parametric method that approximates the relationship between independent variables and continuous outcomes by averaging data in the same neighborhood in an understandable manner.

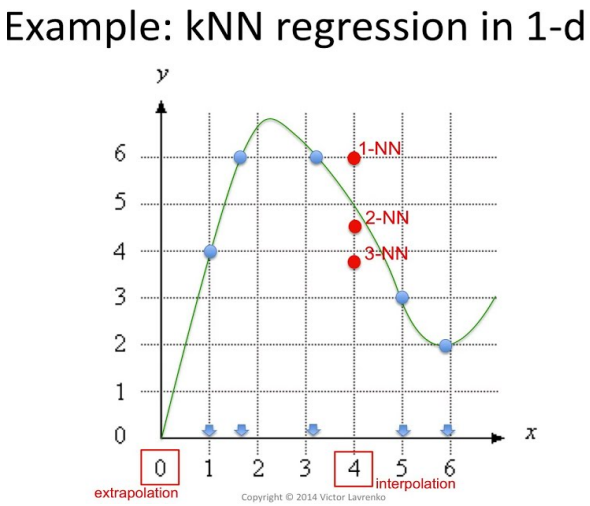


Fig. 8. KNN Regression

The analyst must set the size of the neighborhood, or it can be picked using cross-validation to find the size that minimizes the mean-squared error.

IV. Evaluation

Improving an existing machine learning model can be accomplished in three ways:

- Feature engineering and additional (high-quality) data
- Customize the algorithm's hyperparameters.
- Various algorithms should be attempted.

To improve the used price prediction model, we have used high quality data and feature engineering. We have tried different machine learning algorithms to find the best regressor for our model. Summaries of each algorithm’s results are provided.

IV.I. Linear Regression

The first algorithm we attempted to solve this problem with was linear regression. Linear regression performed poorly: maintaining scores of only 89.26% and 85.18% for the training and testing sets, respectively. The root mean square error (RMSE) produced by a linear regression model was 3.58, significantly higher than other algorithms'.

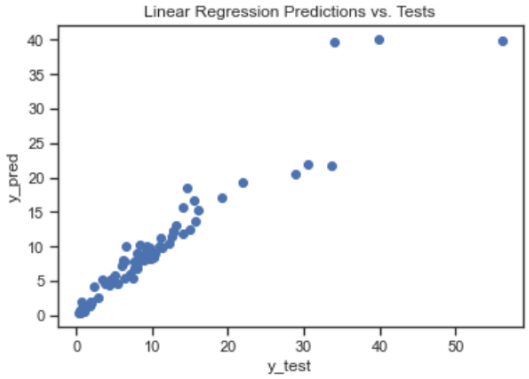


Fig 9. Actual vs Predicted Selling_Price of Linear Regression

IV.II. XGB Regression

The next algorithm we modeled was the XGB Regression. This regression performed extremely well; after training, the model scored 100.00% and 91.22% on the training and testing sets, respectively. The produced RMSE was much lower than the previous linear regression's error mark, at 2.75.

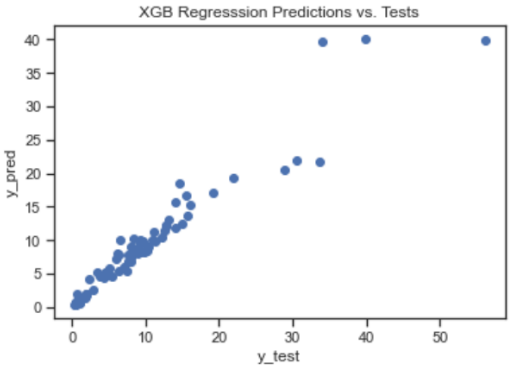


Fig 10. Actual vs Predicted Selling_Price of XGB Regression

IV.III. Decision Tree Regression

Decision Tree Regression was the next algorithm we tested. This regression also performed well, scoring 100.00% and 91.22% on the training and testing sets, respectively. The produced RMSE was similar to XGB Regression's RMSE at 2.69.

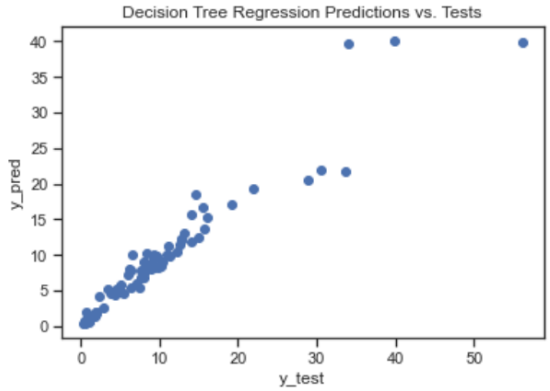


Fig 11. Actual vs Predicted Selling_Price of Decision Tree Regression

IV.IV. K Neighbors Regression

K Neighbors Regression was the next algorithm to model. The training and testing scores of this regression were 91.51% and 83.53, respectively. These values are significantly lower than the XGB and Decision Tree Regressions and closer to Linear

Regression. The RMSE was also similar to Linear Regression's value at 3.79.

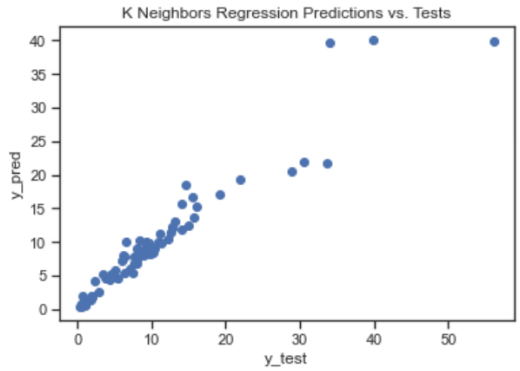


Fig 12. Actual vs Predicted Selling_Price of K Neighbors Regression

IV.V. Support Vector Regression

Support Vector Regression was not an effective regression technique for this use-case. It sponsored a training score of 52.29% and a testing score of 45.67%, by far the lowest of all of our algorithms. The RMSE followed suit, at the highest overall of 6.85.

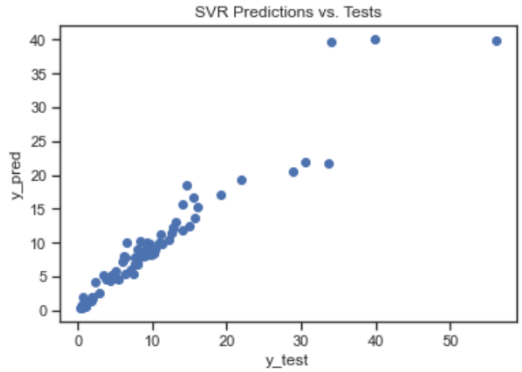


Fig 13. Actual vs Predicted Selling_Price of Support Vector Regression

IV.VI. Lasso Regression

It proved to be a complicated process to create a proper Lasso Regression. After iterating through different lambda values and plotting their coefficients of determination, we were able to find the best fit, and train our model accordingly.

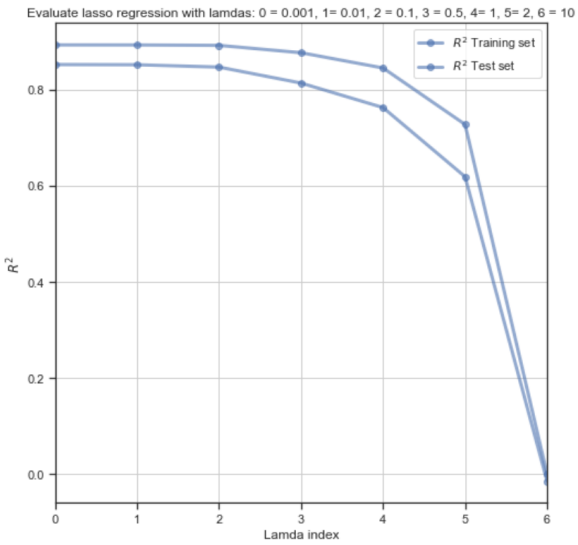


Fig 14. Lasso Regression Lambda R^2 Values

With the optimal alpha value chosen, the Lasso Regression was able to score similarly to the Linear Regression: training and test scores of 89.17 and 84.65, respectively. The RMSE of this regression was 3.64, again similar to the Linear Regression.

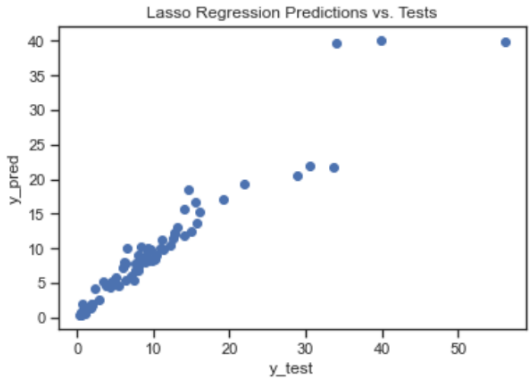
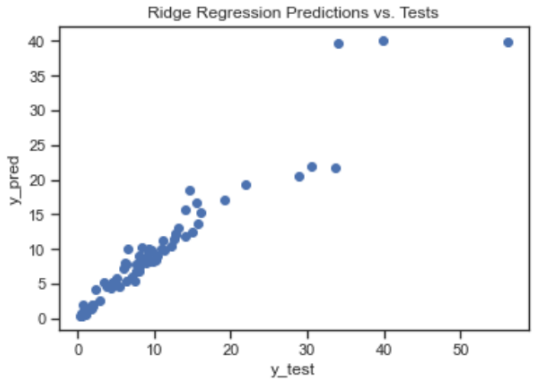


Fig 15. Actual vs Predicted Selling_Price of Lasso Regression

IV.VII. Ridge Regression

In order to create a Ridge Regression appropriate for our methods, we needed to select an alpha penalty parameter to reduce the model's complexity while in exchange introducing bias. Fitting the model with normalization provided $\alpha = 0.1077$. This value provided our regression with a training score of 88.51%, a test score of 82.59%, and a RMSE of 3.88, an unimpressive score compared to XGB and Decision Tree Regressions.



IV.VIII. Random Forest Regression

Random Forest regression has been selected to implement the used vehicle price prediction model. Random Forest Regression has a RMSE value of 2.79. To reduce the RMSE, we have tuned the hyperparameters of the algorithm.

There are two ways to find the optimal hyperparameters:

1. Grid Search
2. RandomizedSearchCV.

The main difference is that in grid search, we define the combinations and train the model, whereas in RandomizedSearchCV, the model chooses the combinations at random. Both are extremely successful methods for fine-tuning parameters that improve model generalization. Random search is the best parameter search technique when there are fewer dimensions.

We have used the random grid to search for the optimum hyperparameters by running a randomized search on hyperparameters in order to improve the model. RandomizedSearchCV implements a "fit" and a "score" method. It also implements "score_samples", "predict", "predict_proba", "decision_function", "transform" and "inverse_transform" if they are implemented in the estimator used. Following hyperparameters have been varied to find the optimal solution for the algorithm:

- Number of trees in random forest
- Number of features to consider at every split
- Maximum number of levels in tree
- Minimum number of samples required to split a node
- Minimum number of samples required at each leaf node

With n_iter = 20, cv = 5, verbose = 2, random_state = 42, n_jobs = 1, scoring='neg_mean_squared_error' and using three fold cross validation, RandomSearchCV reduced the RMSE value to 2.73. Below is the scatter plot between actual and predicted selling price of a used vehicle.

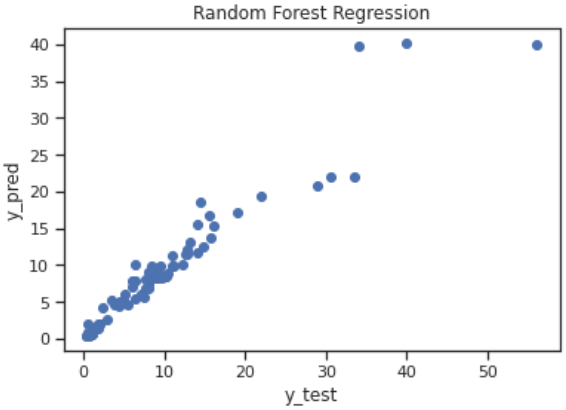


Fig 9. Actual vs Predicted Selling_Price

A predicted against actual plot depicts the model's influence and compares it to the null model. The points should be near to the fitting line, with small confidence bands, for a good fit. Points farthest from the mean, on the left or right of the plot, have the largest leverage and can successfully drag the fitted line toward the point. Both types of points might have a negative impact on the fit. The points are near to the best fit line in this case, showing that the model is good-fit.

Algorithm	**RMSE**
Multiple Linear Regression	3.578945471473329
XGB Regression	2.7544471912845525
Decision Tree Regression	2.6854986317440015
K Neighbors Regression	3.7929453235832855
SVR	6.8526600978832315
Lasso Regression	3.6427885157313478
Ridge Regression	3.879489322674894
Random Forest Regression	2.7871548361727476

Table 1. RMSE

V. Future Scope

In the future, our machine learning model might be linked to a variety of websites that provide real-time data for price forecasting. We could also include previous car price's data to help improve the machine learning model's accuracy. We could create an Android app to serve as the user interface for engaging with the user for optimal accessibility. We intend to optimize performance by carefully designing deep learning network topologies, employing adaptive learning rates, and training on data clusters rather than the entire dataset. Just like any other vehicle price prediction in the market, this prediction model could be used for commercial use as a web app service.

VI. Conclusion

Because of rising new automobile prices and buyers' inability to afford them, used car sales are on the rise all around the world. As a result, there is a pressing need for a Used Vehicle Price Prediction system that accurately analyses the car's worthiness based on a range of factors. The proposed technique will assist in determining the precise price of a secondhand car. Linear Regression, K-NN Regression, Support Vector Regression, XGB Regression, Random Forest Regression, Extra Trees Regression, Ridge Regression, Lasso Regression, and Decision Tree Regressor are the machine learning methods that are compared in this project. Based on our analysis, the Random Forest Regression model has the lowest RMSE and the best accuracy. Hence, vehicle price prediction models can be generated using Random Forest Regression. Once the model is trained, it can be used in any web or mobile application.

References

[1] N. Birla, “Vehicle dataset,” *Kaggle*, 24-Oct-2020. [Online]. Available: https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho. [Accessed: 09-Dec-2021].

[2] Aiken, L. S., West, S. G., & Pitts, S. C. (2003). Multiple linear regression. *Handbook of psychology*, 481-507.

[3] Cootes, T. F., Ionita, M. C., Lindner, C., & Sauer, P. (2012, October). Robust and accurate shape model fitting using random forest regression voting. In the European *Conference on Computer Vision* (pp. 278-291). Springer, Berlin, Heidelberg.

[4] McDonald, G. C. (2009). Ridge regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, *1*(1), 93-100.

[5] Ranstam, J., & Cook, J. A. (2018). LASSO regression. *Journal of British Surgery*, *105*(10), 1348-1348.

[6] Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. Statistics and computing, 14(3), 199-222.

[7] Torres-Barrán, A., Alonso, Á., & Dorronsoro, J. R. (2019). Regression tree ensembles for wind energy and solar radiation prediction. *Neurocomputing*, *326*, 151-160.