VEHICLE RESALE VALUE

PREDICTION

Using Random Forest Regressor

Developed by: Manikanta Thomurothu, Ganji Teja Naidu,

Pulavarthi Sindhusha

**Introduction:**

**Overview:**

The focus of this project is developing machine learning models that can accurately predict the price of a used vehicle based on its features, in order to make informed purchases. We implement and evaluate various learning methods on a dataset consisting of the sale prices of different vehicles and models. Our results show that Random Forest model regression yield the best results and Decision Tree Regression yield moderate results. Conventional multi linear regression also yielded satisfactory results, with the advantage of a significantly lower training time in comparison to the other methods.

**Purpose:**

Deciding whether a used vehicle is worth the posted price when you see listings online can be difficult. Several factors, including mileage, model, year, etc. can influence the actual worth of a vehicle. From the perspective of a seller, it is also a dilemma to price a used vehicle appropriately. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used vehicle prices.

**Literature Survey:**

# **Existing Problem:**

The prices of new vehicles in the industry is fixed by the manufacturer with some additional costs incurred by the Government in the form of taxes. So, customers buying a new vehicle can be assured of the money they invest to be worthy. But due to the increased price of new vehicles and the incapability of customers to buy new vehicles due to the lack of funds, used vehicle sales are on a global increase. There is a need for a used vehicle price prediction system to effectively determine the worthiness of the vehicle using a variety of features. Even though there are websites that offers this service, their prediction method may not be the best. Besides, different models and systems may contribute on predicting power for a used vehicle’s actual market value. It is important to know their actual market value while both buying and selling.

**Proposed Solution:**

The proposed solution for this is Random Forest Regression (because the output is continuous, we apply regression.

Regression is continuous output based on independent variables.

**Theoretical Analysis:**

**Hardware/Software Designing:**

● Jupyter Notebook Environment

● Spyder Ide

● Machine Learning Algorithms

● Python (pandas, NumPy, matplotlib, seaborn, sklearn)

● HTML

● Flask

We developed this vehicle resale value by using the Python language which is an interpreted and high-level programming language and using the Machine Learning algorithms. for coding we used the Jupyter Notebook environment of the Anaconda distributions and the Spyder, it is an integrated scientific programming in the python language. For creating a user interface for the prediction, we used the Flask. It is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions, and a scripting language to create a webpage is HTML by creating the templates to use in the functions of the Flask and HTML

**Experimental Investigation:**

**Source of Data:**

Kaggle is an online community for descriptive analysis and predictive modelling. It collects variety of research fields dataset from data analytic practitioners. Data scientists compete to build the best model for both descriptive and predictive analytic. It however allows individual to access their dataset in order create models and also work with other data scientist to solve various real-world analytics problems. The input dataset used in developing this model has been downloaded from Kaggle.

https://www.kaggle.com/orgesleka/used-cars-database

**Structure of Dataset:**

The dataset contains design characteristics of the used vehicles. This is nicely organized using common format and a standardized set of associate features of used vehicles. Structure of Dataset The dataset contains 20 columns representing the different features, 371528 samples exist. The 20 columns.

**Data Pre-processing:**

Coming to the step of data pre-processing we use four libraries pandas, NumPy, matplotlib, seaborn. Load the given dataset and check for any null values in dataset. Later The aim was to assess skewness of each variable and detecting outliers. The outliers were eliminated by using IQR (Inter-Quartile-Range).

**Correlation Analysis:**

A correlation analysis was employed in this study to examine if explanatory variables share the same linear relationship with the outcome variable in order to detect duplications of variables in the dataset. Among other things, highly correlations between variables were observed in the dataset. The Pearson correlation coefficient r, takes a range of values between +1 to -1. A value of 0 indicates that there of no relationship between the two variables. A value less than zero indicate a negative relationship and a value greater than zero connotes a positive association: that is as one unit of variable increases, so does the value of the other variable.

**Partition of Data:**

The dataset was partitioned into two parts for training and testing purpose: 70% of the entire dataset for training the selected models and 30% for testing purpose. Most importantly, the respective training and validation dataset were randomly sampled to circumvent sampling biasness. Particularly, 10-fold cross validation is a technique to evaluate a predictive model by partitioning the original dataset into a training set to train the model, and a validation/test dataset to assess it.

**Feature Scaling:**

Feature scaling is a technique to standardize the independent feature present in the data in a fixed range. It performed to handle highly varying units. If feature scaling is not done the machine Learning Algorithms tends to weigh greater values are larger and smaller values are lower. For Example, if and algorithm is not using feature scaling method the algorithm thought that 3000meter are greater than 3km but it not true this might Lead to give wrong predictions. So, we use Feature scaling to bring back all the values to same magnitude.

**Algorithm Implementation:**

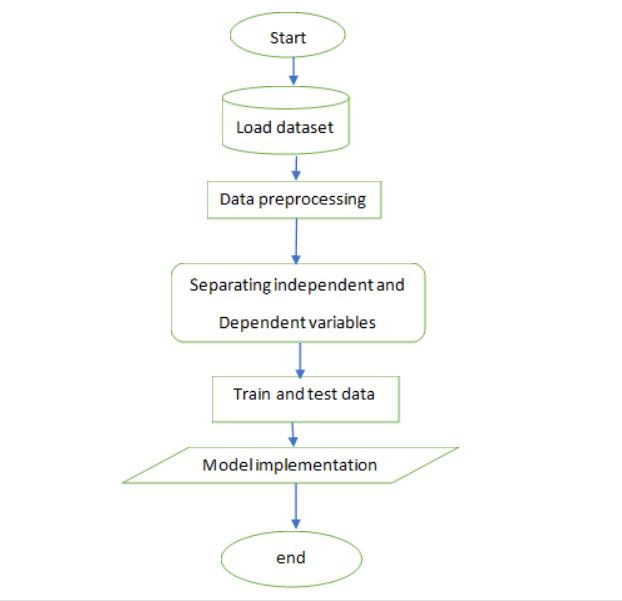
As the output is continuous so we are going to use Regression algorithm we use multilinear regression algorithm and Decision Tree Regression and finally Random forest Regression Algorithm of all these we got the accuracy more for Random Forest Regression so we finalize the Random Forest Regression for this project. Random Forest (RF) is an ensemble machine learning proposed by Leo Bierman that combines predictors tree for predictive or classification task based on independent random samples of observations. In order to grow a tree, multiple random samples are drawn with replacement from the training dataset. This connotes that each tree would be grown with its version of the training dataset. Moreover, a subset of the explanatory variables is also randomly selected at each node during the learning process.

**Evaluation metrics:**

The prediction error is defined as the difference between its actual outcome value and its predicted outcome value. In this study, MSE (Mean Square Error) is used.

This is computed by taking the differences between the target and the actual algorithm outputs, squaring them and averaging over all classes and internal validation samples.

**Flow Chart:**



**Advantages and Disadvantages:**

**Linear Regression:**

Linear regression is one of the most common algorithms for the regression task. In its simplest form, it attempts to fit a straight hyperplane to your dataset (i.e. a straight line when you only have 2 variables). As you might guess, it works well when there are linear relationships between the variables in your dataset.

In practice, simple linear regression is often outclassed by its regularized counterparts (LASSO, Ridge, and Elastic-Net). Regularization is a technique for penalizing large coefficients in order to avoid overfitting, and the strength of the penalty should be tuned.

* **Strengths:** Linear regression is straightforward to understand and explain, and can be regularized to avoid overfitting. In addition, linear models can be updated easily with new data using stochastic gradient descent.
* **Weaknesses:** Linear regression performs poorly when there are non-linear relationships. They are not naturally flexible enough to capture more complex patterns, and adding the right interaction terms or polynomials can be tricky and time-consuming.

**Multi Linear Regression:**

Multiple regression is used to examine the relationship between several independent variables and a dependent variable. While multiple regression models allow you to analyze the relative influences of these independent, or predictor, variables on the dependent, or criterion, variable, these often-complex data sets can lead to false conclusions if they aren't analyzed properly.

* **Strengths:** There are two main advantages to analyzing data using a multiple regression model. The first is the ability to determine the relative influence of one or more predictor variables to the criterion value. The second advantage is the ability to identify outliers, or anomalies.
* **Weaknesses:** Any disadvantage of using a multiple regression model usually comes down to the data being used.

**Regression Tree (Ensembles):**

Regression trees (a.k.a. decision trees) learn in a hierarchical fashion by repeatedly splitting your dataset into separate branches that maximizethe information gain of each split. This branching structure allows regression trees to naturally learn non-linear relationships.

Ensemble methods, such as Random Forests (RF) and Gradient Boosted Trees (GBM), combine predictions from many individual trees. We won't go into their underlying mechanics here, but in practice, RF's often perform very well out-of-the-box while GBM's are harder to tune but tend to have higher performance ceilings.

* **Strengths:** Decision trees can learn non-linear relationships, and are fairly robust to outliers. Ensembles perform very well in practice, winning many classical (i.e. non-deep-learning) machine learning competitions.
* **Weaknesses:** Unconstrained, individual trees are prone to overfitting because they can keep branching until they memorize the training data. However, this can be alleviated by using ensembles.

**Random Forest:**

Random forests are made up of many decision trees, and there is no correlation between different decision trees. When we perform the classification task, the new input sample enters, and each decision tree in the forest is judged and classified separately. Each decision tree will get its own classification result, and which classification result of the decision tree Most, then random forest will use this result as the final result.

* **Strengths:** It can come out with very high dimensional (features) data, and no need to reduce dimension, no need to make feature selection. It can judge the importance of the feature. Can judge the interaction between different features. Not easy to overfit. Training speed is faster, easy to make parallel method. It is relatively simple to implement. For unbalanced data sets, it balances the error. If a large part of the features is lost, accuracy can still be maintained.

* **Weaknesses:** Random forests have been shown to fit over certain noisy classification or regression problems. For data with different values, attributes with more values will have a greater impact on random forests, so the attribute weights generated by random forests on such data are not credible.

**Result:**

Since we have to predict the price of a vehicle which is a continuous value, we have used regression technique to solve this problem. We have applied various regression algorithms as follows:

|  |  |
| --- | --- |
| ALGORITHM | ACCURACY |
| Multi Linear Regression | 71.3% |
| Decision Tree Regression | 80.1% |
| Random Forest Regression | 86.8% |

**Conclusion:**

The main aim of this project is to build an accurate machine learning model which predicts the price of a used vehicle based on its features. The model is built using regression models. Out of all the three regression models used, we can conclude that Random Forest Regressor gives greater accuracy (86.8%).

**Future Scope:**

Random forest (RF) is an ensemble classification approach that has proved its high accuracy and superiority. With one common goal in mind, RF has recently received considerable attention from the research community to further boost its performance. Research was invested in boosting the performance of RF. One of the earliest to be reported is by [Latinne et al. (2001)](https://www.tandfonline.com/doi/full/10.1080/21642583.2014.956265). In Bioinformatics and Computational Biology, [Boulesteix et al. (2012)](https://www.tandfonline.com/doi/full/10.1080/21642583.2014.956265) amalgamated 10 years of RF development. Practical aspects of RF including selection of parameters, available RF implementations, important pitfalls, and biases of RF and its variable importance measures were covered. In future, the application of RF will be extended to many other fields that can ease the work force in particular fields.

**Bibliography:**

<https://towardsdatascience.com/predicting-used-car-prices-with-machine-learning-techniques-8a9d8313952>

**HTML CODE:**

<!DOCTYPE html>

<html >

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

<meta charset="UTF-8">

<title>ML API</title>

<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>

<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>

<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>

<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="{{ url\_for('static', filename='css/style.css') }}">

</head>

<style >

body

{

background-image:url('https://mk0analyticsindf35n9.kinstacdn.com/wp-content/uploads/2019/12/AIM\_Predict\_Resale\_Price\_APP-scaled.jpg');

background-repeat: no-repeat;

background-attachment: fixed;

background-position: center;

}

</style>

<body>

<div class="login">

<!-- Main Input For Receiving Query to our ML -->

<form action="{{ url\_for('y\_predict')}}"method="post">

<input type="text" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;"

name="VehType" placeholder="Vehicle Type" required="required" />

<input type="number" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="YOR" placeholder="Year of Registration" required="required" />

<input type="text" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="Gear" placeholder="Gearbox" required="required" />

<input type="number" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="powerPS" placeholder="powerPS" required="required" />

<input type="number" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="km" placeholder="Kilometers Travelled" required="required" />

<input type="number" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="MOR" placeholder="Month Of Registration" required="required" />

<input type="text" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="FT" placeholder="FuelType" required="required" />

<input type="text" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="brand" placeholder="Brand" required="required" />

<input type="text" STYLE="color: #000000; font-family: Verdana; font-weight: bold; font-size: 12px; background-color: #FFFFFF;" name="R/D" placeholder="NotRepaired/Damaged" required="required" />

<button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>

</form>

<br>

<br>

{{ prediction\_text }}

</div>

</body>

</html>

**CSS CODE:**

@import url(https://fonts.googleapis.com/css?family=Open+Sans);

.btn {

display: inline-block; \*display: inline; \*zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-size: 13px; line-height: 18px; color: #333333; text-align: center;text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75); vertical-align: middle; background-color: #f5f5f5; background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6)); background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6); background-image: -o-linear-gradient(top, #ffffff, #e6e6e6); background-image: linear-gradient(top, #ffffff, #e6e6e6); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff, endColorstr=#e6e6e6, GradientType=0); border-color: #e6e6e6 #e6e6e6 #e6e6e6; border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -webkit-border-radius: 4px; -moz-border-radius: 4px; border-radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); cursor: pointer; \*margin-left: .3em;

}

.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }

.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; }

.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear; transition: background-position 0.1s linear; }

.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }

.btn-primary.active { color: rgba(255, 255, 255, 0.75); }

.btn-primary

{

background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de, #4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4, GradientType=0); border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }

.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }

.btn-block { width: 100%; display:block; }

\* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box; }

html { width: 100%; height:100%; overflow:hidden; }

body {

width: 100%;

height:100%;

font-family: 'Open Sans', sans-serif;

background: #092756;

color: #fff;

font-size: 18px;

text-align:center;

letter-spacing:1.2px;

background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(top, rgba(57,173,219,.25) 0%, rgba(42,60,87,.4) 100%), -moz-linear-gradient(-45deg, #670d10 0%, #092756 100%);

background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -webkit-linear-gradient(-45deg, #670d10 0%,#092756 100%);

background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -o-linear-gradient(-45deg, #670d10 0%,#092756 100%);

background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -ms-linear-gradient(-45deg, #670d10 0%,#092756 100%);

background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to bottom, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), linear-gradient(135deg, #670d10 0%,#092756 100%);

filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );

}

.login {

position: absolute;

top: 40%;

left: 50%;

margin: -150px 0 0 -150px;

width:400px;

height:400px;

}

.login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }

input {

width: 100%;

margin-bottom: 10px;

background: rgba(0,0,0,0.3);

border: none;

outline: none;

padding: 10px;

font-size: 13px;

color: #fff;

text-shadow: 1px 1px 1px rgba(0,0,0,0.3);

border: 1px solid rgba(0,0,0,0.3);

border-radius: 4px;

box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);

-webkit-transition: box-shadow .5s ease;

-moz-transition: box-shadow .5s ease;

-o-transition: box-shadow .5s ease;

-ms-transition: box-shadow .5s ease;

transition: box-shadow .5s ease;

}

input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2);

}

**app.py:**

import numpy as np

from flask import Flask, request, jsonify, render\_template

import pickle

from joblib import load

from sklearn.preprocessing import StandardScaler

app = Flask(\_\_name\_\_)

# model = load('rfg.save')

# load the model

model = load(open('vehicleresaleprice.pkl', 'rb'))

# load the scaler

scaler = load(open('scaler.pkl', 'rb'))

trans=load('transform')

trans1=load('transform1')

trans2=load('transform2')

trans3=load('transform3')

trans4=load('transform4')

@app.route('/')

def home():

return render\_template('index.html')

@app.route('/y\_predict',methods=['POST'])

def y\_predict():

'''

For rendering results on HTML GUI

'''

x\_test = [[x for x in request.form.values()]]

print(x\_test)

x\_test=trans.transform(x\_test)

x\_test=x\_test[:,1:]

x\_test=trans1.transform(x\_test)

x\_test=trans2.transform(x\_test)

x\_test=x\_test[:,1:]

x\_test=trans3.transform(x\_test)

x\_test=x\_test[:,1:]

x\_test=trans4.transform(x\_test)

x\_test=scaler.transform(x\_test)

print(x\_test)

prediction = model.predict(x\_test)

print(prediction)

output=prediction[0]

return render\_template('index.html', prediction\_text='price = {0:.3f}'.format(output))

'''@app.route('/predict\_api',methods=['POST'])

def predict\_api():

#For direct API calls trought request

data = request.get\_json(force=True)

prediction = model.y\_predict([np.array(list(data.values()))])

output = prediction[0]

return jsonify(output)'''

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

