

CAR RESALE VALUE PREDICTION

Project **Report** **Format**

1. INTRODUCTION

1.1 Abstract

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature

7.2 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

Name of the Team leader:Harish J

Team ID: PNT2022TMID01300

Roll number: 2019PECIT268

Registration Number: 211419205065

Mobile Number:6379586874

Name of the Team Member 1:ARUNRAJ M

Registration Number:211419205018

Roll number: 2019PECIT237

Mobile Number: 8925402973

MailID:arunrajcr09@gmail.com

Name of the Team Member 2:YUGANDHAR C

Roll number: 2019PECIT226

Registration Number:211419205187

Mobile Number:9345130743

MailID:yugichella@gmail.com

Name of the Team Member3:ARUN T

Roll number:2019PECIT238

Registration Number: 2114205016

Mobile Number: 8056157362

MailID:arunt8523@gmail.com

ABSTRACT

Predicting the price of used cars is one of the significant and interesting areas of analysis. As an increased demand in the second-hand car market, the business for both buyers and sellers has increased. For reliable and accurate prediction it requires expert knowledge about the field because of the price of the cars dependent on many important factors. This paper proposed a supervised machine learning model using KNN (K Nearest Neighbor) regression algorithm to analyze the price of used cars . Through this experiment, the data was examined with different trained and test ratios. As a result, the accuracy of the proposed model is around 85% and is fitted as the optimized model. The predictions are then evaluated and compared in order to find those which provide the best performances. A seemingly easy problem turned out to be indeed very difficult to resolve with high accuracy. All the four methods provided comparable performance. In the future, we intend to use more sophisticated algorithms to make the predictions . Determining whether the listed price of a used car is a challenging task, due to the many factors that drive a used vehicle's price on the market. The focus of this project is developing machine learning models that can accurately predict the price of a used car 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 makes and models across cities in the United States. Our results show that Random Forest model and K-Means clustering with linear regression yield the best results, but are compute heavy. Conventional linear regression also yielded satisfactory results, with the advantage of a significantly lower training time in comparison to the aforementioned methods

1.LITERATURE SURVEY

2.1. EXISTING PROBLEM

Several studies and related works have been done previously to predict used car prices around the world using different methodologies and approaches, with varying results of accuracy from 50% to 90%. In (Pudaruth, 2014) the researcher proposed to predict used car prices in Mauritius, where he applied different machine learning techniques to achieve his results like decision tree, K-nearest neighbours, Multiple Regression and Naïve Bayes algorithms to predict the used cars prices, based on historical data gathered from the newspaper.

Achieved results ranged from accuracy of 60-70 percent, the author suggested using more sophisticated models and algorithms to make the evaluation, with the main weakness off the decision tree and naïve Bayes that it is required to discretize the price and classify it which accrue to more inaccuracies. Moreover, he suggested a larger set of data of data to train the models hence the data gathered was not sufficient.

(Monburinon, et al., 2018) Gathered data from a German e-commerce site that totalled to 304,133 rows and 11 attributes to predict the prices of used car using different techniques and measured their results using Mean Absolute Error (MEA) to compare their results. Same training dataset and testing dataset was given to each model. Highest results achieved was by using gradient boosted regression tree with a MAE of 0.28, and MEA of 0.35 and 0.55 for mean absolute error and multiple linear regression respectively. Authors suggested adjusting the parameters in future works to yield better results, as well as using one hot encoding instead of label encoding for more realistic data interpretations on categorical data.

(Gegic, Isakovic, Keco, Masetic, & Kevric, 2019) from the International Burch University in Sarajevo, used three different machine learning techniques to predict used car prices. Using data scrapped from a local Bosnian website for used cars totalled at 797 car samples after pre-processing, and proposed using these methods: Support Vector

Machine, Random Forest and Artificial Neural network. Results have shown using only one machine learning algorithm achieved results less

than 50%, whereas after combining the algorithms with pre calcification of prices using Random Forest, results with accuracies up to 87.38% was recorded.

(Noor & Jan, 2017) were able to achieve high level of accuracy using Multiple linear regression models to predict the price of cars collected from used cars website in Pakistan called Pak Wheels that totalled to 1699 records after pre-processing, and where able to achieve accuracy of 98%, this was done after reducing the total amount of attributes using variable selection technique to include significant attributes only and to reduce the complexity of the model.

(K.Samruddhi & Kumar, 2020) Proposed using Supervised machine leaning model using K-Nearest Neighbour to predict used car prices from a data set obtained from Kaggle containing 14 different attributes, using this method accuracy reached up to 85% after different values of K as well as Changing the percent of training data to testing data, expectedly when increasing the percent of data that is tested better accuracy results are achieved. The model was also cross validated with 5 and 10 folds by using K fold method.

(Gongqi, Yansong, & Qiang, 2011) proposed using Artificial Neural Network (ANN) through a combined method of BP neural network and nonlinear curve fit and have achieved accurate value prediction with a feasible model.

(Listiani, 2009) used Support Vector Machines to evaluate leased cars prices, results have shown that SVM is far more accurate in large dataset with high dimensional data than Multiple linear regression. Whereas the computation Multiple linear regression can take several minutes and the SVM would take up to a day to compute the results. Multiple linear regression may be simple, but SVM is far more accurate. Moreover, the study includes Samples with up to 178 attributes which is far more than the proposed variable in our study, hence the use of multiple linear regression may be more suitable in our case.

(Kuiper, 2008) Collected data from General Motor of cars that are produced in 2005, where he as well used variable selection technique to include the most relevant attributes in his model to reduce the

complexity of the data. He proposed used Multivariate regression model that would be more suitable for values with numeric format.

In order to predict the price of used cars, researchers (Nabarun Pal, 2018) used a supervised learning method known as Random Forest. Kaggle's dataset was used as a basis for predicting used car prices. In order to determine the price impact of each feature, careful exploratory data analysis was performed. 500 Decision Trees were trained with Random Forests. It is most commonly used for classification, but they turned it into a regression model by transforming the problem into an equivalent regression problem. Using experimental results, it was found that training accuracy was 95.82%, and testing accuracy was 83.63%. By selecting the most correlated features, the model can accurately predict the car price.

In light of the number of works that have been done in this field, another group of researchers (Jian Da Wu, 2017) conducted research on this topic and tried to develop a system that consists of three components: a data acquisition system, a price forecasting algorithm, and a performance analysis. Due to its adaptive learning capability, a conventional artificial neural network (ANN) with a back-propagation network is compared to the proposed ANFIS. In the ANFIS, qualitative fuzzy logic approximation as well as adaptive neural network capabilities are included. Using ANFIS as an expert system in predicting used car prices showed better results in the experiment. Using GUI, the consumer can get accurate and convenient

information about used cars' purchasing prices, and experiments proved that the proposed system could provide accurate and convenient price forecasting.

Hence, from all literature review it is concluded that used cars price prediction is an important topic which is the area of many researchers nowadays. So far, the best achieved accuracy is 83.63% on kaggle's dataset using random forest technique. The researchers have tested multiple regressors and final model is regression model using linear regression.

Method :

The topic such as this can be assessed with mathematical models derived from quantitative data. A multiple variable regression can analyze the data by assessing the role each independent variable plays in determining the dependent variable (in this case, resale value). Significance can also be assessed by observing the p-values for each variable. The use of a statistical model will aide in making a claim on this, and to identify some of the major contributors to resale value in automobiles.

Data Collection :

The data used for this regression will be quantitative in nature. The sources of data are what someone would expect for used car information. Four sources that are used include Kelly Blue Book, Edmunds, a government fuel economy resource, and Car and Driver. Kelly Blue Book and Edmunds will both serve as data sources, with each source providing different aspects of the independent variables used. With the cooperation of these sources, data regarding price of a car-including new and used-with the respective age, mileage, make, condition, miles per gallon, safety ratings, and hybrid technology information will be obtained. These variables will allow for a regression to be run and an equation to be estimated.

Expected Outcomes :

Before I can make predictions regarding the influence each variable will have on resale value, a review of prior research and literature is appropriate. This will allow me to make a more confident prediction as well as confirm which variables are needed to produce a strong equation that explains much of the variations in vehicle depreciation. An expected equation could look like this:

$$\text{Resale Value (DV)} = \text{Intercept} - B3(\text{Age}) - B4(\text{Mileage}) + B1(\text{Make}) + B2(\text{MPG}) + B5(\text{Hybrid Tech})$$

2.2.

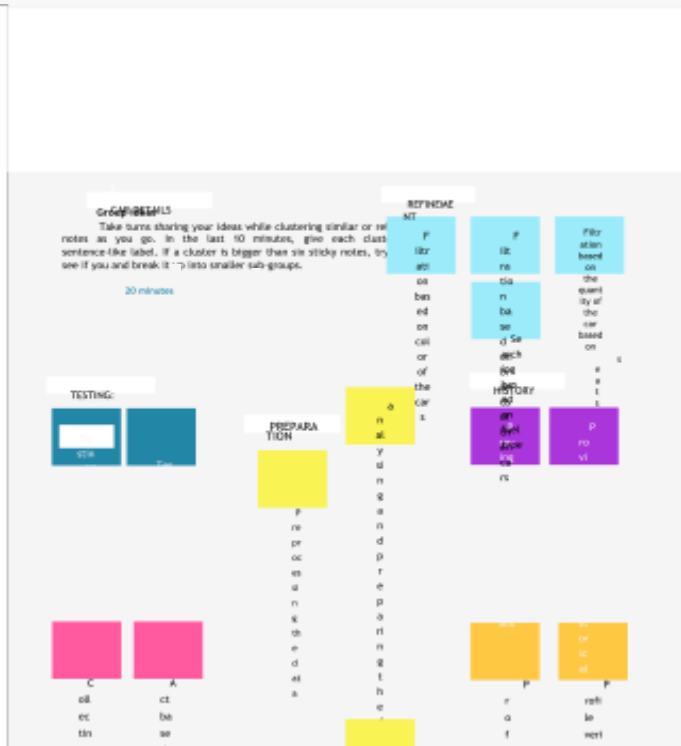
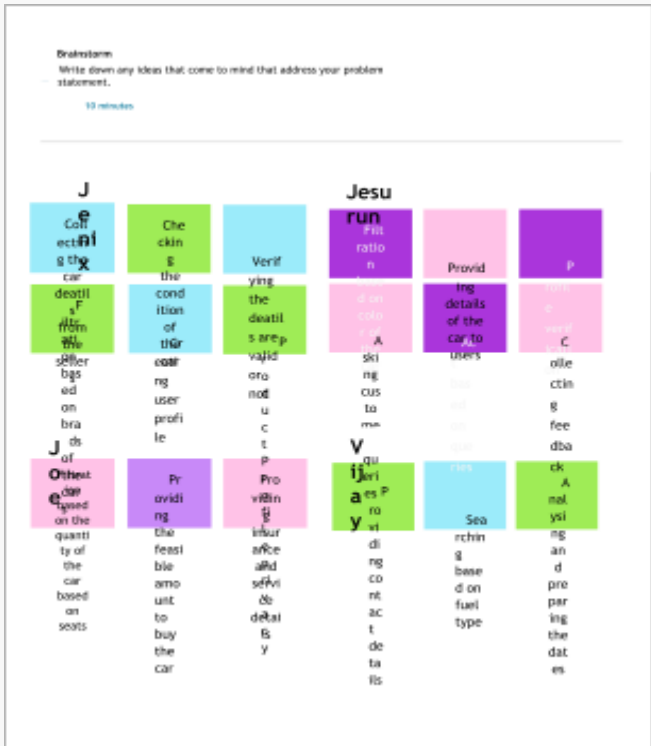
REFERENCES

- [1] Pudaruth, Sameerchand. "Predicting the price of used cars using machine learning techniques." *Int. J. Inf. Comput. Technol* 4, no. 7 (2014): 753-764.
- [2] Monburinon, Nitis, Prajak Chertchom, Thongchai Kaewkiriya, Suwat Rungpheung, Sabir Buya, and Pitchayakit Boonpou. "Prediction of prices for used car by using regression models." In *2018 5th International Conference on Business and Industrial Research (ICBIR)*, pp. 115-119. IEEE, 2018.
- [3] Gegic, Enis, Becir Isakovic, Dino Keco, Zerina Masetic, and Jasmin Kevric. "Car price prediction using machine learning techniques." *TEM Journal* 8, no. 1 (2019): 113.
- [4] Noor, Kanwal, and Sadaqat Jan. "Vehicle price prediction system using machine learning techniques." *International Journal of Computer Applications* 167, no. 9 (2017): 27-31.
- [5] <https://ieeexplore.ieee.org/Xplore/home.jsp>
- [6] <https://www.analyticsvidhya.com/blog/2018/08/k-nearestneighbor-introduction-regression-python/>
- [7] <https://machinelearningmastery.com/k-fold-cross-validation/>

2.
2.1.

IDEATION PHASE EMPATHY MAP





2.3.

PROPOSED SOLUTION

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	I am a customer. I'm trying to buy a second hand car. But I cannot estimate the price of the car. Because I need a trustworthy platform to predict the price of the car. Which makes me feel Frustrated and Confused.
2.	Idea / Solution description	Deciding whether a used car is worth the posted price when you see listings online can be difficult. Several factors, including mileage, make, model, year, etc. can influence the actual worth of a car. From the perspective of a seller, it is also a dilemma to price a used car appropriately [2-3]. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car prices.
3.	Novelty / Uniqueness	As there are so many ongoing experiments that use statistical approaches and some traditional methods to focus on predicting item sales. Most researches have experimented by taking single algorithm to predict sales. In this thesis Machine Learning algorithms such as Simple Linear Regression, Support Vector Regression, Gradient Boosting algorithm, and Random Forest Regression are considered for predict the most effective metrics such as accuracy, mean absolute error, and max error are considered for measuring algorithm efficiency. This method will be very beneficial in the future for advanced item sales forecasting.
4.	Social Impact / Customer Satisfaction	Predicting prices of a used car is a challenging task because of a high number of features and parameters that should be considered to generate accurate results. The first and foremost step is data gathering and pre-processing data. Therefore the results generated are highly accurate.so the customer was satisfied.
5.	Business Model (Revenue Model)	Deciding whether a used car is worth the posted price when you see listings online can be difficult. Several factors, including mileage,

		make, model, year, etc. can influence the actual worth of a car. From the perspective of a seller, it is also a dilemma to price a used car. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car prices.
6.	Scalability of the Solution	We started with understanding the use case of machine learning in the Automotive industry and how machine learning has transformed the driving experience. Moving on, we looked at the various factors that affect the resale value of a used car and performed exploratory data analysis (EDA). Further, we build a Random Forest Regression model to predict the resale value of a used car. We could have also used simpler regression algorithms like Linear Regression and Lasso Regression. Still, we need to make sure there are no outliers in the dataset before implementing them. Pair plots and scatter plots help visualize the outliers

2.4. PROBLEM SOLUTION FIT

Project Title: Car Resale Value Prediction

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID00778

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS Who is your customer? i.e. working parents of 0-5 y.o. kids Dealers who sell used cars and customers who buy them.	6. CUSTOMER CONSTRAINTS CC What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices. Some constraints that the customers face may be worried about the condition of the car, is too expensive and so on.	5. AVAILABLE SOLUTIONS AS Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital note-taking. Customers can be assured about the condition of the car and can be given a test drive, then the customer may be more inclined to buy the car.	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS J&P Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one, explore different sides. We help dealers and customers predict the price of a used car.	9. PROBLEM ROOT CAUSE RC What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations. The root cause may be because the customer doesn't want to make a decision that he might regret later because he is the one using the car.	7. BEHAVIOUR BE What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits, indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace) The Customer may spend some time researching the prices of used cars and may also see if the dealer has any reviews.	
Focus on J&P, fit into BE, understand RC	3. TRIGGERS TR What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news. People around them buying used cars.	10. YOUR SOLUTION SL If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour. Our solution is that we will predict the ideal price for the car based on a lot of research and data and give the customer the most affordable approach.	8. CHANNELS of BEHAVIOUR CH 8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7 8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development. 1. The customer can talk see the car and compare it with other cars and offers 2. The customer can see if the car is really in good condition and if it suits their needs.	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER EM How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure -> confident, in-control - use it in your communication strategy & design. Customers feel frustrated because they don't know how much a used car is worth.			

3. Solution Requirements (Functional & Non-functional)

3.1. Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Profile	User information Bank details
FR-4	Database	Car database Customer database
FR-5	Features and Technology	Performance of the car Fuel capacity, mileage etc.
FR-6	Feedback	Feedback through Form Feedback through Gmail Feedback through LinkedIn

3.2. Non-functional Requirements

Following are the non-functional requirements of the proposed solution.

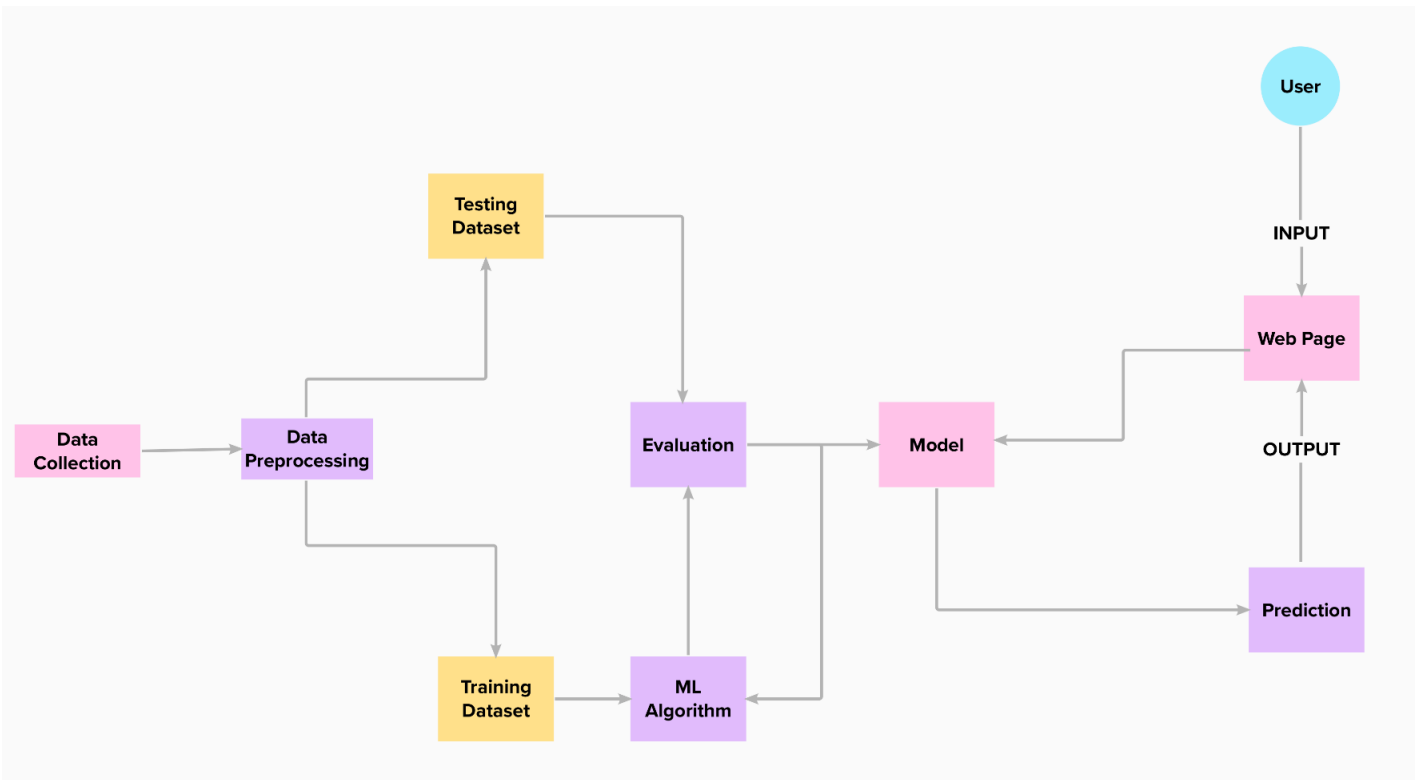
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Great UI (user interface), Quick adaptation of user.
NFR-2	Security	Aware of fraud and scams, Protect your password and account personal details.
NFR-3	Reliability	Rate of occurrence of failure is less, Failure free.
NFR-4	Performance	Perform value and correct prediction value, The landing page must support several users must provide 5 second or less response time
NFR-5	Availability	Uninterrupted services must be available all time except the time of server updation.
NFR-6	Scalability	that can handle any amount of data and perform many computations in a cost-effective and timesaving way to instantly serve millions of users residing at global locations.

4.

Project Design Phase-I

5.1.

Data Flow Diagram & User Stories

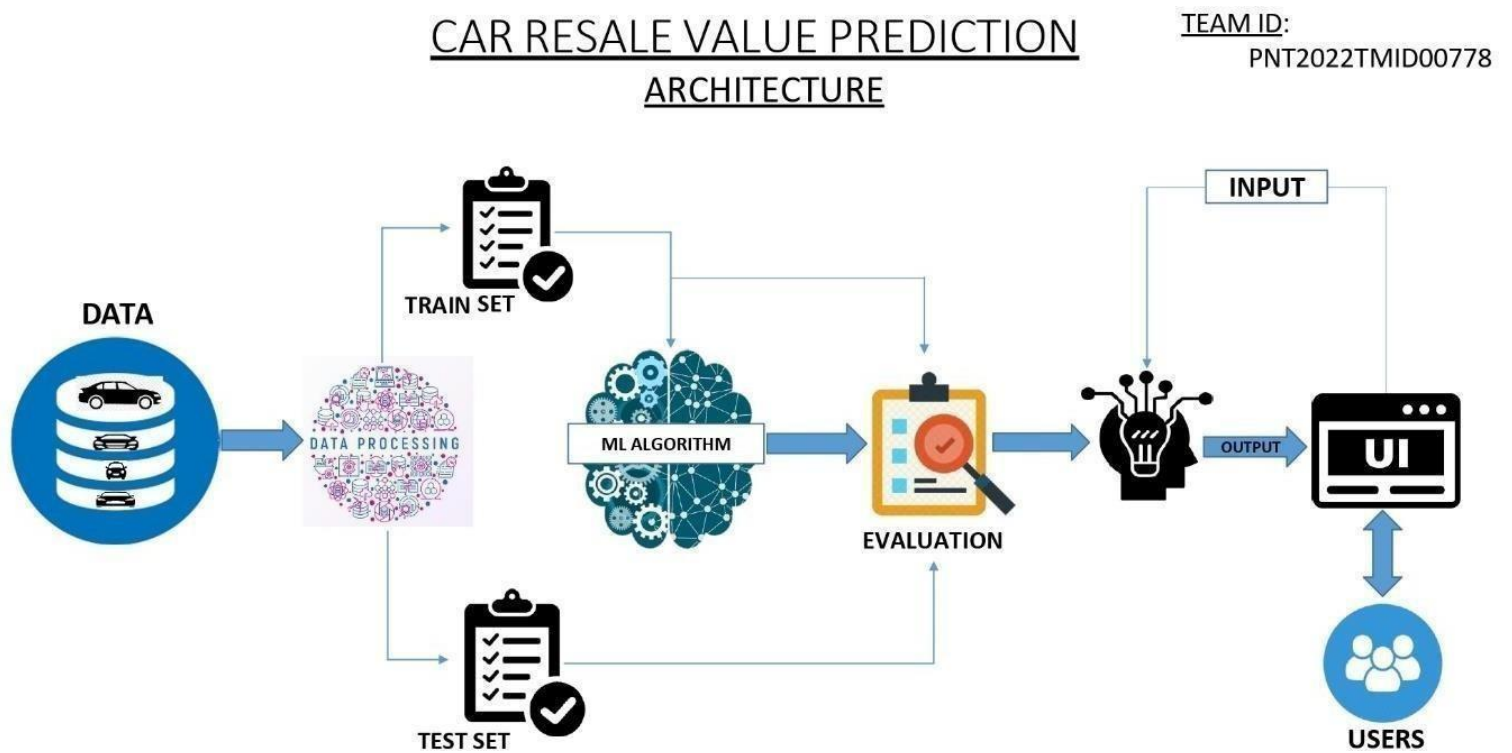


User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Data Entry	USN-1	As a user, I can enter the car details in the application.	I can enter the car details	Medium	Sprint-1
Customer (Mobile user)	Obtain output	USN-2	As a user, I will receive car resale value in the application.	I can receive my car resale value	High	Sprint-1
Customer (Mobile user)	Data Entry	USN-1	As a user, I can enter the car details in the application.	I can enter the car details	Medium	Sprint-1
Customer (Mobile user)	Obtain output	USN-2	As a user, I will receive car resale value in the application.	I can receive my car resale value	High	Sprint-1

5.2.

SOLUTION ARCHITECTURE



5.2. Technology Architecture

Technical Architecture:

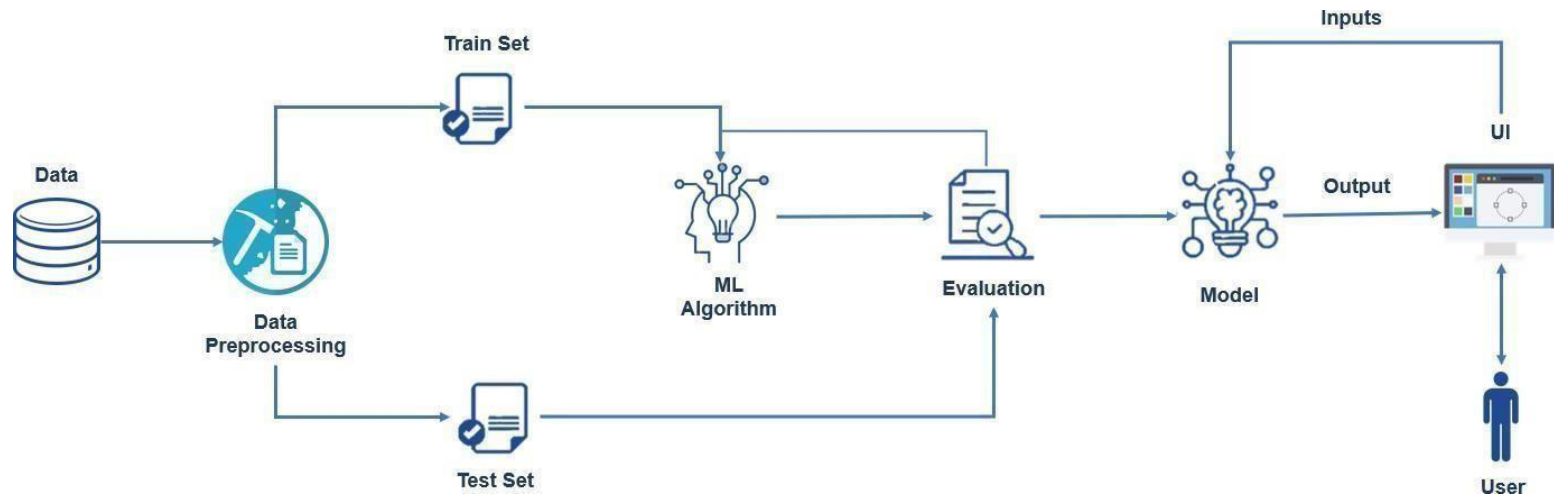




Table-1 : Components & Technologies:

S.N o	Component	Description	Technology
1.	User Interface	How user interacts with application e.g.Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js /React Js etc.
2.	Data Pre - Processing	Checking if the given data has any missing values,duplicate values, outliers and other noises that can affect the performance of the model.	Python, Pandas, Numpy, Matplotlib,Seaborn.
3.	Splitting the DataSet	The data set is split into test and train data for themodel.	Python, Sk - Learn
4.	Predicting the values	After the model is trained using various machine learning algorithms, some code is written to predict the value of a used car.	Python, Sk - Learn
5.	Database	The data is stored in the database.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other StorageService or Local Filesystem
8.	Machine Learning Model	There are various Machine Learning Models that can be used like Linear Regression, Multi-Linear Regression, Decision Tree, Random Forest, SVMetc.	Python, Sk - Learn

Table-2: Application Characteristics:

S.N o	Characteristics	Description	Technology
1.	Open-Source Frameworks	Anaconda Navigator, Jupyter Notebook, Python,Flask.	Python
2.	Security Implementations	Aware of Fraud and Scams, Protection of password and account details.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Whether demand increases gradually or abruptly, scalable web architecture can accommodate any load without compromising the application's integrity.	Microservies, Progressive Web Apps(PWA)
4.	Availability	Availability of application like load balancers,distributed servers etc	IBM Cloud
5.	Performance	Good Performance is expected.	IBM Cloud

5.3. CUSTOMER JOURNEY

1 Phases <small>High-level steps your user needs to accomplish from start to finish</small>	Team ID : PNT2022TMID00778			
	OPEN WEBAPP	ENTER THE CAR FEATURES	PREDICT CAR RESALE VALUE	RESULT
2 Steps <small>Detailed actions your user has to perform</small>	Wants to predict the resale value of the car accurately open the car resale value prediction module	Enter the features of the car Click predict and get result	prediction of car resale price	Display the predicted value
3 Feelings <small>What your user might be thinking and feeling at the moment</small>  	Eager and Happy	happy to find a car resale predicted price	ecstatic	Feeling Good
	Unexcited	SAD	Unhappy	Feeling bad
4 Pain points <small>Problems your user runs into</small>	Not happy with number of features to enter	stressed for entering more features	worried about time taken to see result	worried about accuracy
5 Opportunities <small>Potential improvements or enhancements to the experience</small>	Better and Good design	user friendly	Quicker response	High and Good accuracy

5.

Project Planning Phase

5.1. SPRINT DELIVERY PHASE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Dataset reading and Pre processing	USN-1	Cleaning the dataset and splitting to dependent and independent variables	2	High	Jenix A
Sprint-2	Building the model	USN-2	Choosing the appropriate model for building and saving the model as pickle file	1	High	Jeshuran Ben edict
Sprint-3	Application building	USN-3	Using flask deploying the ML model	2	Medium	Joe Nikesh PJ
Sprint-4	Train the model in IBM	USN-4	Finally train the model on IBM cloud and deploy the application	2	Medium	Arjunan Vijay Manohar

Velocity:

We have a 5-day sprint duration, and the velocity of the team is 15 (points per sprint). The team's average velocity (AV) per iteration unit (storypoints per day)

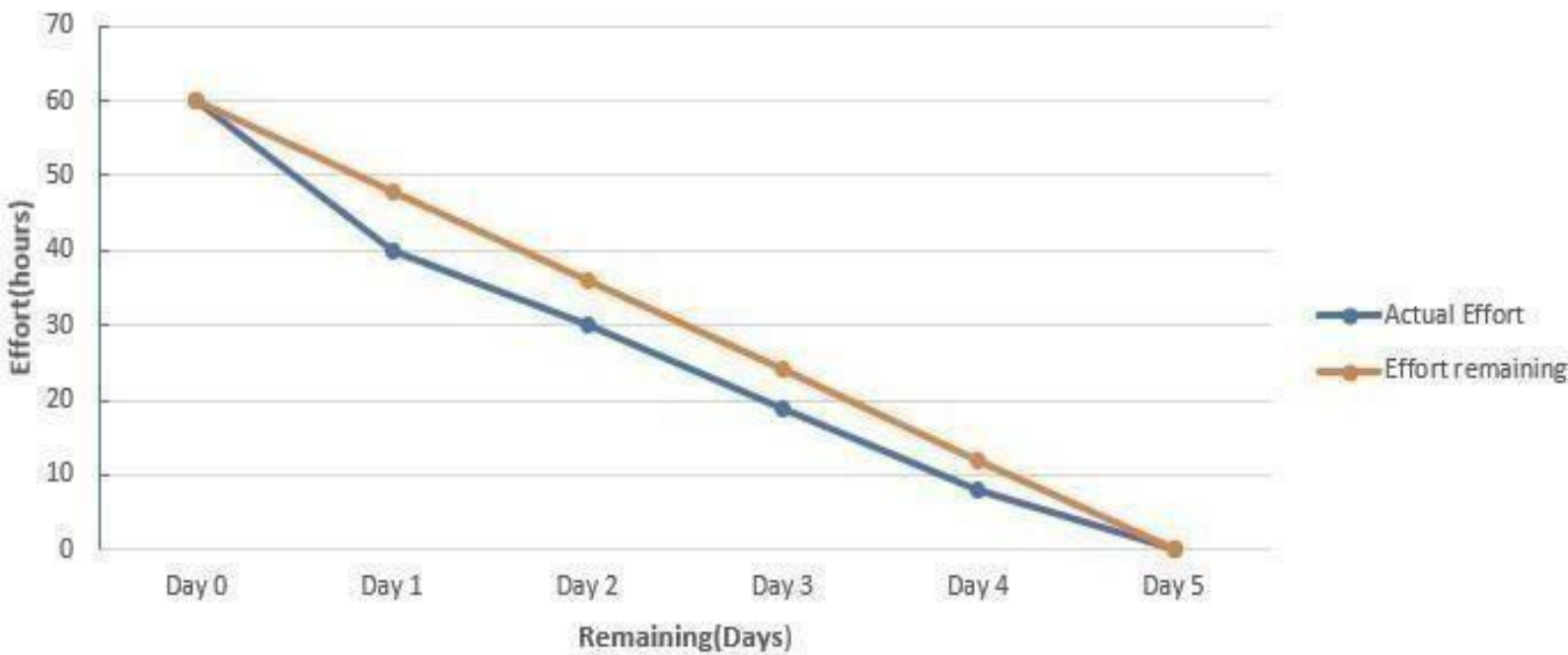
$$\begin{aligned} \text{Actual Velocity} &= \frac{\text{Sprint Duration}}{\text{Velocity}} = \frac{15}{5} \\ &= 3 \end{aligned}$$

Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date(Actual)
Sprint-1	15	5 Days	24 Oct 2022	29 Oct 2022	15	29 Oct 2022
Sprint-2	15	5 Days	31 Oct 2022	05 Nov 2022	15	05 Nov 2022

Goal:60 hours in 5 days

Burndown chart



5.2. MILESTONE AND ACTIVITY LIST

Activity Number	Activity Name	Detailed Activity Description	Assigned To	Status / Comments
1	Preparation Phase	<ul style="list-style-type: none"> Access the resources (courses) in project dashboard Access the guided project workspace Create GitHub account & collaborate with Project Repository in project workspace Set-up the Laptop / Computers based on the prerequisites for each technology track 	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar.	It refers to done the listed activities in the preparation phase and done Prerequisites, Registration, Environment setup
2	Ideation Phase	<ul style="list-style-type: none"> Literature survey on the selected project & Information Gathering Preparation of Empathy Map Canvas to capture the user Pains & Gain Prepare list of problem statements List the ideas by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance 	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	The activities in ideation phase refers to when gathering the idea for project information and picturize in Empathy map
3	Project Design Phase -I			

3.1	Proposed Solution	Preparation of proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	The solution for the project is prepared as a standard document structure from Team members
-----	-------------------	--	---	---

3.2	Problem SolutionFit	Prepared problem is analyzed and make effectivesolutions for the problem	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	
3.3	Solutio n Archite cture	Prepare an architecture for solution	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	Suitable block diagram template used to prepare Solution architecture
4	Project Design Phase -II			
4.1	Require ment Analysi s	Prepare the Functional Requirement and NonFunctional Document	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	Listing of functional and non-functional requirements of project.
4.2	Customer Journey	Preparation of customer journey maps to understand the user interactions & experiences with the application (entry to exit)	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	Customer journey map prepared by suitable template by team members.
4.3	Data Flow Diagrams	Prepare a Data Flow Diagram for Project use level0(Industry Standard)	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	Use suitable data flow diagram rulesand standards to prepare level 0 DFD
4.4	Techno logy Archite cture	Prepare Technology Architecture of the solution	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	

5	Project Planning Phase			
5.1	Milestones & Tasks	Prepare Milestone & Activity List	Jenix A, Jeshuran Benedict, Joe Nikesh P J, Arjunan Vijay Manohar	
5.2	Sprint Schedules	Prepare Sprint Delivery Plan	Jenix A, Jeshuran Benedict, Joe Nikesh P J,	



6	Project Development Phase			
6.1	Coding & Solutioning	Sprint-1 Delivery: Develop the Code, Test and push it to GitHub.	On Progress	
6.2	Acceptance Testing	<ul style="list-style-type: none"> Sprint-2 Delivery: Develop the Code, Test and push it to GitHub. Sprint-3 Delivery: Develop the Code, Test and push it to GitHub. 	On Progress	
6.3	Performance Testing	Sprint-4 Delivery: Develop the Code, Test and push it to GitHub.	On Progress	

Milestone:

When project begins then it is expected that project related activities must be initiated. In project planning, series of milestones must be established. Milestone can be defined as recognizable endpoint of software project activity. At each milestone, report must be generated. Milestone is distinct and logical stage of the project. It is used as signal post for project start and end date, need for external review or input and for checking budget, submission of the deliverable, etc. It simply represents clear sequence of events that are incrementally developed or build until project gets successfully completed. It is generally referred to as task with zero-time duration because they are used to symbolize an achievement or point of time in project. It helps in signifying change or stage in development.

5.3.

REPORTS FROM JIRA



6. CODING & SOLUTIONING

6.1. FEATURE 1

Importing Required Libraries

In []:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Reading the Dataset

In []:

```
df = pd.read_csv('autos.csv', parse_dates=['dateCrawled', 'dateCreated', 'lastSeen'], e
```

Cleaning the Dataset

In []:

```
df.columns
```

Out[3]:

```
Index(['dateCrawled', 'name', 'seller', 'offerType', 'price', 'abtest',
      'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model',
      'kilometer', 'monthOfRegistration', 'fuelType', 'brand',
      'notRepairedDamage', 'dateCreated', 'nrOfPictures', 'postalCode',
      'lastSeen'],
      dtype='object')
```

In []:

```
# Rearranging the Columns
df = df[['dateCrawled', 'name', 'seller', 'offerType', 'abtest',
      'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model',
      'kilometer', 'monthOfRegistration', 'fuelType', 'brand',
      'notRepairedDamage', 'dateCreated', 'nrOfPictures', 'postalCode',
      'lastSeen', 'price']]
```

In []:

```
# Dropping the Unwanted Columns
df.drop(columns= ['seller', 'offerType', 'nrOfPictures'], inplace = True)
```

In []:

```
df.drop(columns= ['dateCrawled', 'dateCreated', 'lastSeen'], inplace = True)
```

Missing Values

In []:

```
# Checking for Missing Values
df.isna().sum()
```

Out[6]:

```
name                0
abtest              0
vehicleType         37869
yearOfRegistration  0
gearbox             20209
powerPS             0
model               20484
kilometer           0
monthOfRegistration  0
fuelType            33386
brand               0
notRepairedDamage   72060
postalCode          0
price               0
dtype: int64
```

In []:

```
# Removing Missing Values
df['vehicleType'].fillna(df['vehicleType'].mode()[0], inplace = True)
df['gearbox'].fillna(df['gearbox'].mode()[0], inplace = True)
df['model'].fillna(df['model'].mode()[0], inplace = True)
df['fuelType'].fillna(df['fuelType'].mode()[0], inplace = True)
df['notRepairedDamage'].fillna(df['notRepairedDamage'].mode()[0], inplace = True)
```

In []:

```
df.isna().sum()
```

Out[8]:

```
name                0
abtest              0
vehicleType         0
yearOfRegistration  0
gearbox             0
powerPS             0
model               0
kilometer           0
monthOfRegistration  0
fuelType            0
brand               0
notRepairedDamage   0
postalCode          0
price               0
dtype: int64
```

Duplicate Values

```
In [ ]:
```

```
# Checking for Duplicates
df.duplicated().sum()
```

Out[9]:

4703

```
In [ ]:
```

```
# Removing Duplicates
df = df.drop_duplicates()
```

```
In [ ]:
```

```
df.duplicated().sum()
```

Out[11]:

0

Label Encoding

```
In [ ]:
```

```
df.info()
```

<class 'pandas.core.frame.DataFrame'>

Int64Index: 366825 entries, 0 to 371527

Data columns (total 14 columns):

#	Column	Non-Null Count	Dtype
0	name	366825 non-null	object
1	abtest	366825 non-null	object
2	vehicleType	366825 non-null	object
3	yearOfRegistration	366825 non-null	int64
4	gearbox	366825 non-null	object
5	powerPS	366825 non-null	int64
6	model	366825 non-null	object
7	kilometer	366825 non-null	int64
8	monthOfRegistration	366825 non-null	int64
9	fuelType	366825 non-null	object
10	brand	366825 non-null	object
11	notRepairedDamage	366825 non-null	object
12	postalCode	366825 non-null	int64
13	price	366825 non-null	

int64 dtypes: int64(6), object(8)

memory usage: 42.0+ MB

In []:

```
from sklearn.preprocessing import  
LabelEncoder le = LabelEncoder()  
df['name'] = le.fit_transform(df['name'])  
df['abtest'] = le.fit_transform(df['abtest'])  
df['vehicleType'] =  
le.fit_transform(df['vehicleType']) df['gearbox'] =  
le.fit_transform(df['gearbox']) df['model'] =  
le.fit_transform(df['model']) df['fuelType'] =  
le.fit_transform(df['fuelType']) df['brand'] =  
le.fit_transform(df['brand'])  
df['notRepairedDamage'] = df['notRepairedDamage'].replace({'nein' : 0, 'ja' : 1})
```

```
2  vehicleType      366825 non-null int64  
3  yearOfRegistration  366825 non-null int64  
4  gearbox          366825 non-null int64  
5  powerPS          366825 non-null int64  
6  model            366825 non-null int64  
7  kilometer        366825 non-null int64  
8  monthOfRegistration 366825 non-null int64  
9  fuelType         366825 non-null int64  
10 brand            366825 non-null int64  
11 notRepairedDamage 366825 non-null int64  
12 postalCode        366825 non-null int64  
13 price            366825 non-null  
int64 dtypes: int64(14)  
memory usage: 42.0 MB
```

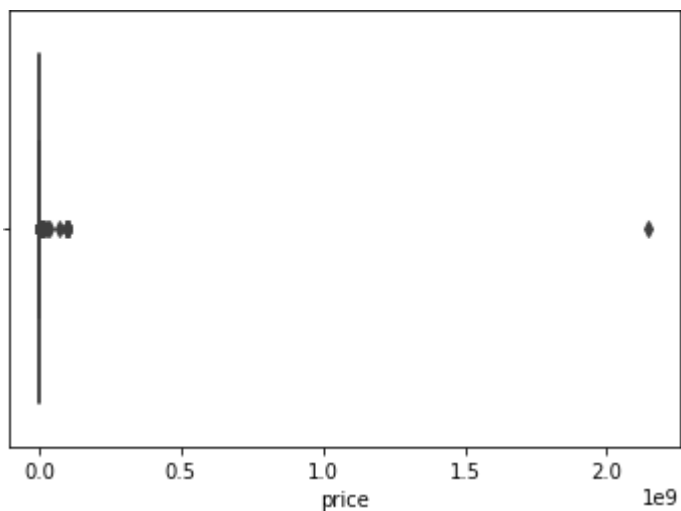
Identifying and Handling Outliers

In []:

```
# Checking for outliers in 'price' column
```


In []:

```
sns.boxplot(x = df['price'])  
sns.boxplot(xlabel='price')
```



In []:

```
a = df['price'].quantile(q=[0.75,0.25])  
a
```

Out[17]:

```
0.75    7150.0  
0.25    1150.0  
Name: price, dtype: float64
```

In []:

```
IQR = a.iloc[0] -  
a.iloc[1] IQR
```

Out[18]

:

6000.0

```
In [ ]:  
]:
```

```
upper =  
a.iloc[0]+(1.5*IQR) lower  
= a.iloc[0]-(1.5*IQR)  
upper
```

Out[20]:

16150.0

In []:

```
lower
```

Out[21]:

-1850.0

In []:

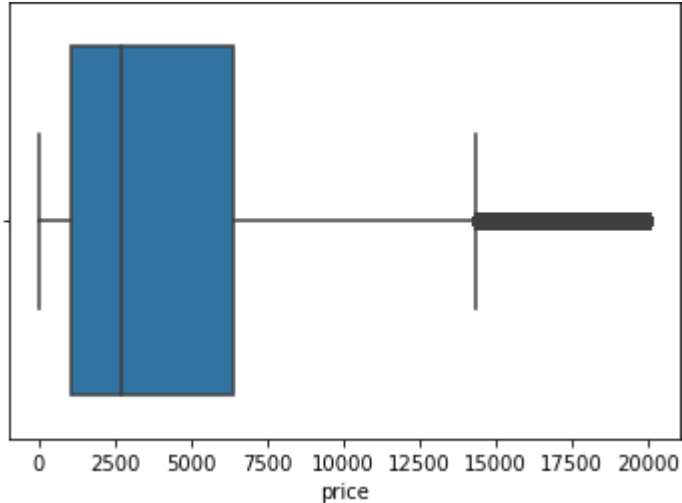
```
# Dropping outliers in price  
a = df[df['price'] > 20000].index  
df.drop(a, inplace = True)
```

In []:

```
sns.boxplot(x = df['price'])
```

Out[23]:

<AxesSubplot:xlabel='price'>

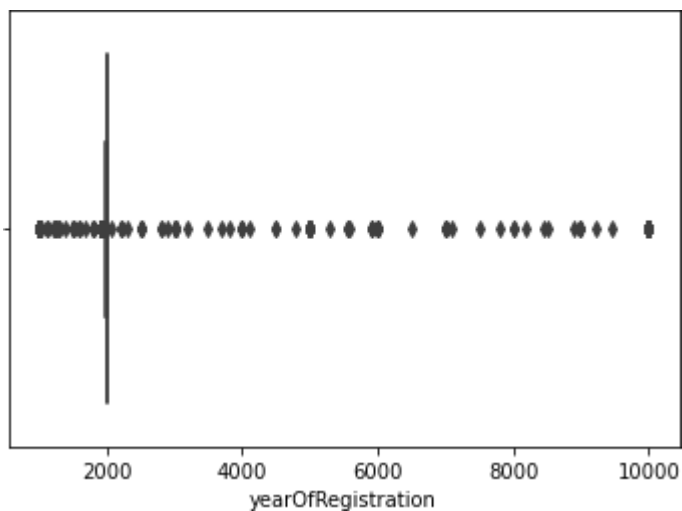


```
# Checking for outliers in 'yearOfRegistration' column
```

In []:

In []:

```
sns.boxplot(x = df['yearOfRegistration'])
plt.xlabel('yearOfRegistration')
```



In []:

```
a = df['yearOfRegistration'].quantile(q=[0.75,0.25]) a
```

Out[26]:

```
0.75    2008.0
0.25    1999.0
Name: yearOfRegistration, dtype: float64
```

In []:

```
IQR = a.iloc[0] - a.iloc[1] IQR
```

Out[27]

:

9.0

In []:

```
upper = a.iloc[0] + (1.5 * IQR)
lower = a.iloc[0] - (1.5 * IQR)
```

```
In [ ]:
```

```
upper  
lower
```

```
2021.5
```

```
In [ ]:
```

```
lower
```

```
Out[30]:
```

```
1994.5
```

```
In [ ]:
```

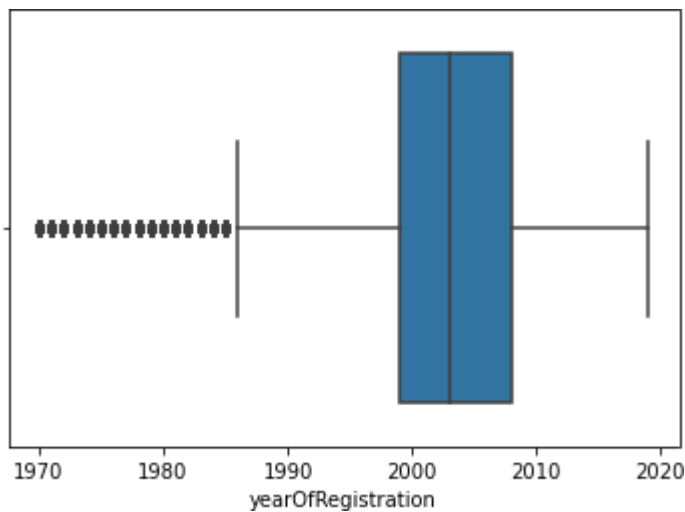
```
# Dropping outliers in yearOfRegistration  
a = df[df['yearOfRegistration'] >  
2019].index df.drop(a, inplace = True)  
a = df[df['yearOfRegistration'] <  
1970].index df.drop(a, inplace = True)
```

```
In [ ]:
```

```
sns.boxplot(x = df['yearOfRegistration'])
```

```
Out[32]:
```

```
<AxesSubplot:xlabel='yearOfRegistration'>
```

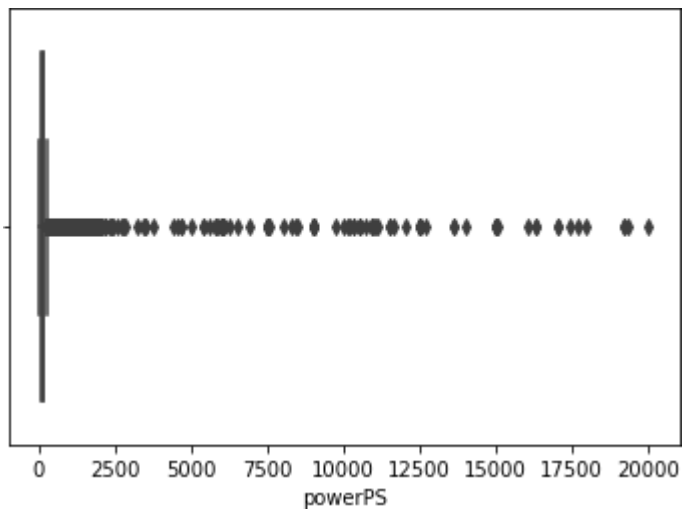


```
In [ ]:
```

```
# Checking for outliers in 'powerPS' column
```

In []:

```
sns.boxplot(x = df['powerPS'])  
sns.boxplot(xlabel='powerPS')
```



In []:

```
a = df['powerPS'].quantile(q=[0.75,0.25])  
a
```

Out[35]:

```
0.75    141.0  
0.25     69.0  
Name: powerPS, dtype: float64
```

In []:

```
IQR = a.iloc[0] - a.iloc[1]  
IQR
```

Out[36]

:

72.0

In []:

```
upper = a.iloc[0]+(1.5*IQR)  
lower = a.iloc[0]-(1.5*IQR)
```

```
In [ ]:  
1:
```

```
upper
```

```
249.0
```

```
In [ ]:
```

```
lower
```

```
Out[39]:
```

```
33.0
```

```
In [ ]:
```

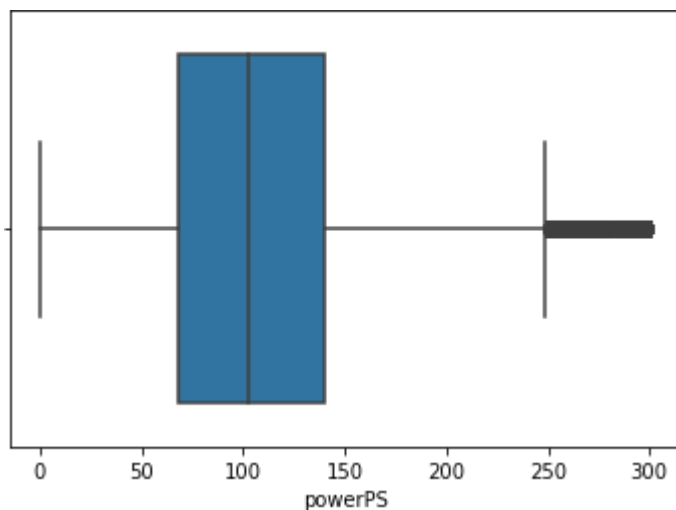
```
# Dropping outliers in powerPS  
a = df[df['powerPS'] > 300].index  
df.drop(a, inplace = True)
```

```
In [ ]:
```

```
sns.boxplot(x = df['powerPS'])
```

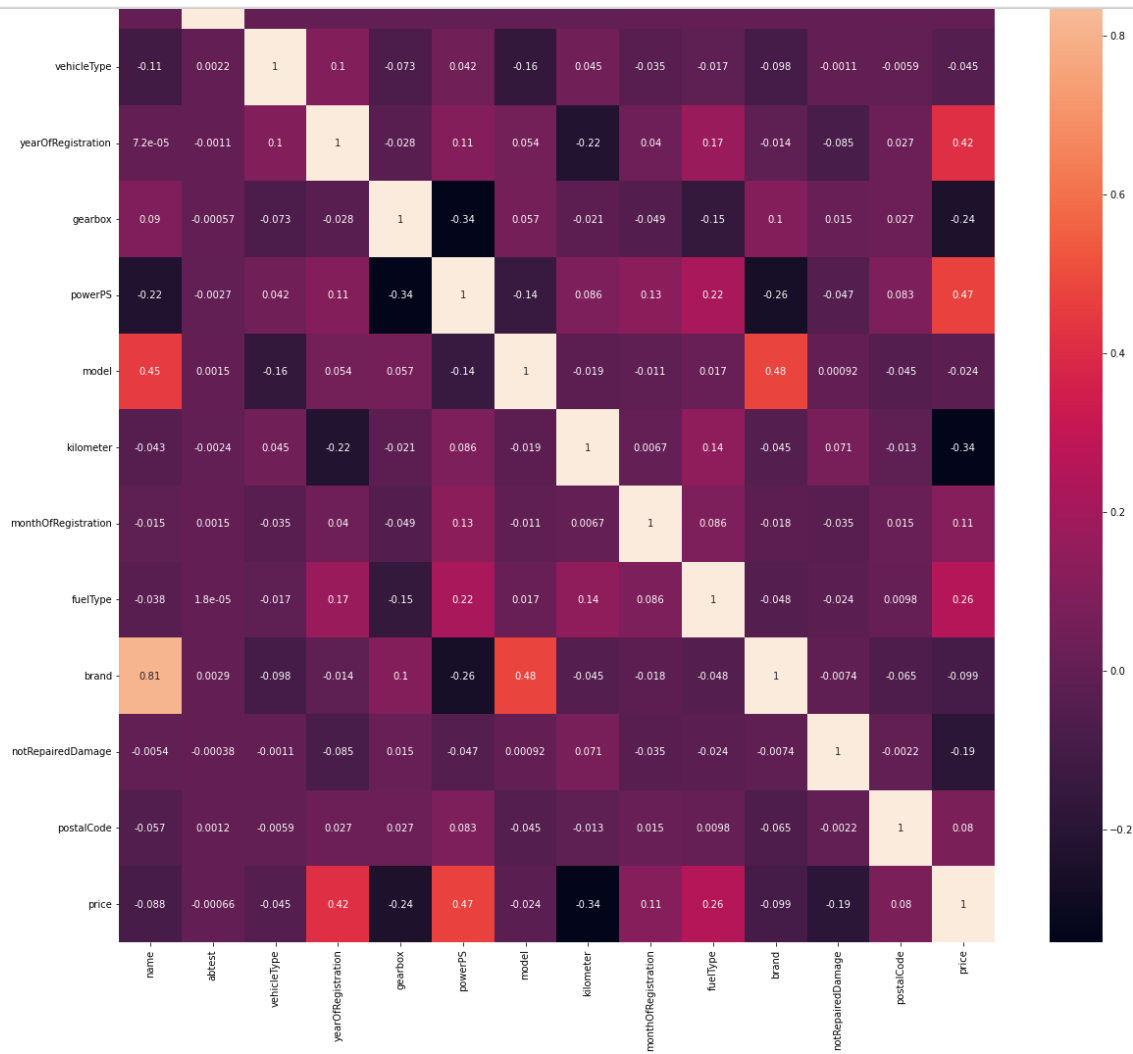
```
Out[41]:
```

```
<AxesSubplot:xlabel='powerPS'>
```

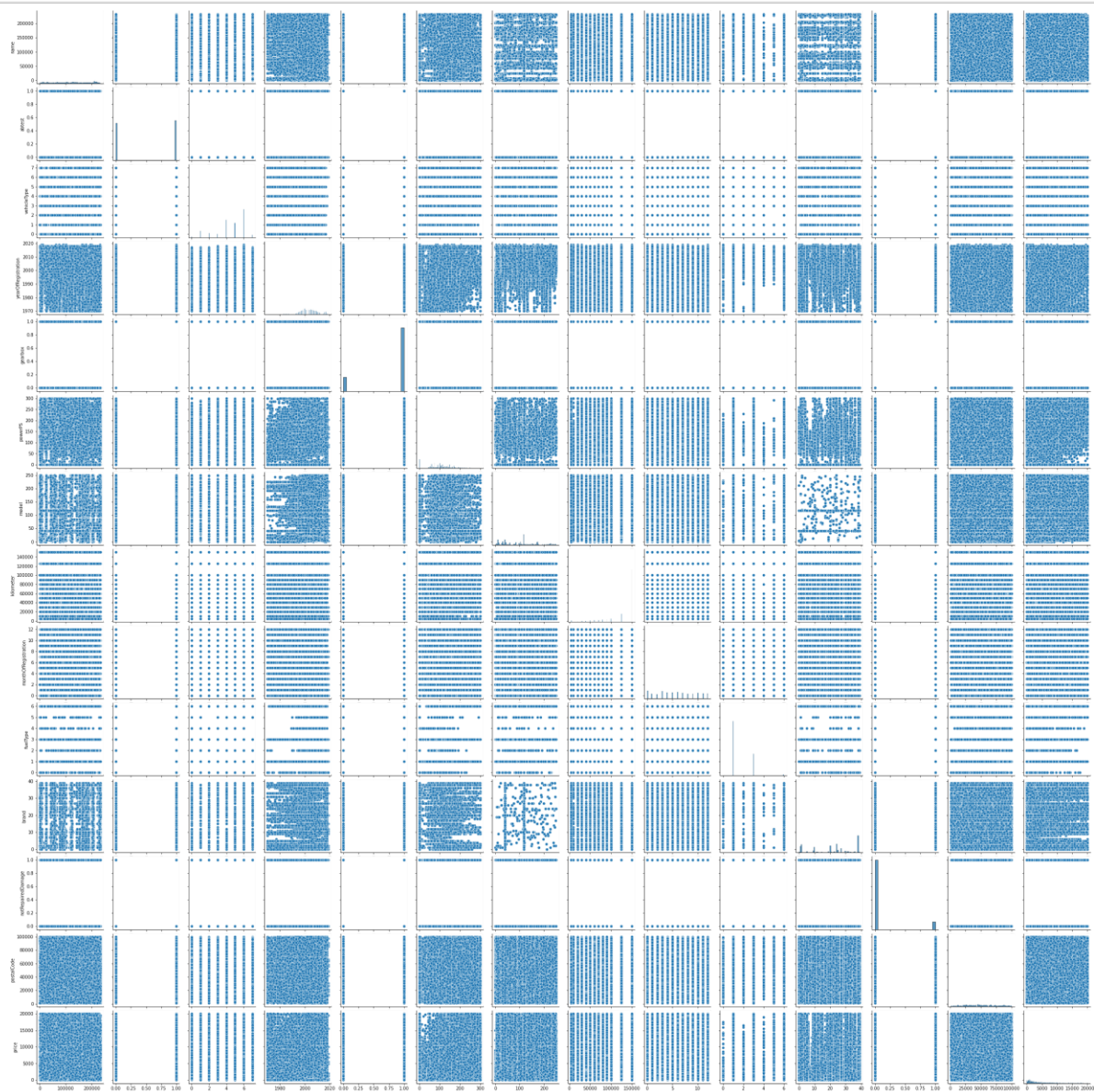


Visualization

```
plt.figure(figsize=(20,20))
sns.heatmap(df.corr(), annot = True)
plt.show()
```



```
sns.pairplot(df)
plt.show()
```



Descriptive Statistics


```
In [ ]:  
1:
```

```
df.nunique()
```

Out[80]:

```
name                218805  
abtest              2  
vehicleType         8  
yearOfRegistration  50  
gearbox             2  
powerPS             299  
model              250  
kilometer           13  
monthOfRegistration 13  
fuelType            7  
brand              40  
notRepairedDamage   2  
postalCode          8140  
price              3708  
dtype: int64
```

In []:

```
df.describe()
```

Out[42]:

	name	abtest	vehicleType	yearOfRegistration	gearbox	
count	344985.000000	344985.000000	344985.000000	344985.000000	344985.000000	34498
mean	117904.585912	0.518202	4.565471	2003.229619	0.819265	10
std	67510.545516	0.499669	1.661815	6.980320	0.384799	5
min	0.000000	0.000000	0.000000	1970.000000	0.000000	
25%	60989.000000	0.000000	4.000000	1999.000000	1.000000	6
50%	119794.000000	1.000000	5.000000	2003.000000	1.000000	10
75%	175396.000000	1.000000	6.000000	2008.000000	1.000000	14
max	233530.000000	1.000000	7.000000	2019.000000	1.000000	30



```
In [ ]:  
df.skew()
```

Out[43]:

name	-0.022347
abtest	-0.072858
vehicleType	-0.917651
yearOfRegistration	-0.360852
gearbox	-1.659392
powerPS	0.189407
model	0.395804
kilometer	-1.737954
monthOfRegistration	0.082692
fuelType	1.542590
brand	-0.172770
notRepairedDamage	2.622955
postalCode	0.075437
price	1.461433
dtype:	float64

In []:

```
df.kurt()
```

Out[44]:

name	-1.200546
abtest	-1.994703
vehicleType	-0.028758
yearOfRegistration	1.432725
gearbox	0.753586
powerPS	0.085471
model	-0.883618
kilometer	1.984077
monthOfRegistration	-1.147400
fuelType	2.400634
brand	-1.310623
notRepairedDamage	4.879922
postalCode	-0.962817
price	1.547299
dtype:	float64

Splitting the Data

In []:

```
# Splitting x and y variables  
x = df.drop(columns = 'price')  
y = df['price']
```

```
# Splitting into test and train
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

In []:

```
# Linear Regression
```

In []:

```
from sklearn.linear_model import
LinearRegression lr = LinearRegression()
lr.fit(x_train, y_train)
```

Out[48]:

LinearRegression()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In []:

```
# Lasso Regression
```

In []:

```
from sklearn.linear_model import Lasso
lasso = Lasso(alpha=0.01, normalize=True)
lasso.fit(x_train, y_train)
```

Out[72]:

Lasso(alpha=0.01, normalize=True)

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In []:

```
# Ridge Regression
```

```
from sklearn.linear_model import Ridge
ridge = Ridge(alpha=0.01, normalize=True)
ridge.fit(x_train, y_train)
```

the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In []:

```
# Decision Tree
```

In []:

```
from sklearn.tree import DecisionTreeRegressor
DT = DecisionTreeRegressor()
DT.fit(x_train, y_train)
```

Out[54]:

DecisionTreeRegressor()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In []:

```
# KNN
```

In []:

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor()
knn.fit(x_train, y_train)
```

Out[56]:

KNeighborsRegressor()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In []:

```
# Random Forest
```

```
In [ ]:  
1:
```

```
from sklearn.ensemble import RandomForestRegressor  
RF = RandomForestRegressor()  
RF.fit(x_train, y_train)
```

the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Checking the Metrics of the models

In []:

```
# Linear Regression  
lr.score(x_test, y_test)
```

Out[59]:

```
0.504616429315972  
2
```

In []:

```
from sklearn.metrics import mean_squared_error  
np.sqrt(mean_squared_error(y_test, lr.predict(x_test)))
```

Out[60]:

```
3124.83924381600  
1
```

In []:

```
# Lasso Regression  
lasso.score(x_test, y_test)
```

Out[61]:

```
0.504627901784278  
7
```

In []:

```
np.sqrt(mean_squared_error(y_test, lasso.predict(x_test)))
```

Out[62]:

```
3124.80305990834  
8
```

```
# Ridge Regression  
ridge.score(x_test, y_test)
```

In []:

```
np.sqrt(mean_squared_error(y_test,ridge.predict(x_test)))
```

Out[64]:

```
3124.849380685733  
6
```

In []:

```
# K Nearest Neighbour  
knn.score(x_test, y_test)
```

Out[65]:

```
0.360426465617484  
7
```

In []:

```
np.sqrt(mean_squared_error(y_test,knn.predict(x_test)))
```

Out[66]:

```
3550.60305731533  
2
```

In []:

```
# Decision Tree  
DT.score(x_test, y_test)
```

Out[67]:

```
0.735189145898358  
9
```

In []:

```
np.sqrt(mean_squared_error(y_test,DT.predict(x_test)))
```

Out[68]:

```
2284.676799722256  
4
```

In []:

```
# Random Forest  
RF.score(x_test, y_test)
```

In [

Out[69]:

0.862194104305205

4

In []:

```
np.sqrt(mean_squared_error(y_test, RF.predict(x_test)))
```

Out[70]:

1648.1274003735057

Saving the Model

In []:

```
import pickle  
pickle.dump(RF, open('Car Resale Value Prediction.pkl', 'wb'))
```


7.

TESTING

8.1. USER ACCEPTANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3

Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8.2. Model Performance Test

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No	Parameter	Values	Screenshot
1.	Metrics	Regression Model: MAE - 1232.3528089560773, MSE - 5063778.951720876, RMSE - 2250.2841935455344, R2 score - 0.87528399134989	<pre>from sklearn.metrics import r2_score r2_score(y_test, RF.predict(x_test)) 0.8752839913498982 from sklearn.metrics import mean_squared_error mean_squared_error(y_test, RF.predict(x_test)) 5063778.951720876 from sklearn.metrics import mean_squared_error np.sqrt(mean_squared_error(y_test, RF.predict(x_test))) 2250.2841935455344 from sklearn.metrics import mean_absolute_error mean_absolute_error(y_test, RF.predict(x_test)) 1232.3528089560773</pre>
2.	Tune the Model	Validation Method - train_test_split	<pre># Splitting x and y variables x = df.drop(columns = 'price') y = df['price'] # Splitting into test and train from sklearn.model_selection import train_test_split x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)</pre>

8. RESULTS

9.1 Performance Metrics

```
from sklearn.metrics import r2_score  
r2_score(y_test, RF.predict(x_test))
```

0.8752839913498982

```
from sklearn.metrics import mean_squared_error  
mean_squared_error(y_test, RF.predict(x_test))
```

5063778.951720876

```
from sklearn.metrics import mean_squared_error  
np.sqrt(mean_squared_error(y_test, RF.predict(x_test)))
```

<IPython.core.display.Javascript object>

2250.2841935455344

```
from sklearn.metrics import mean_absolute_error  
mean_absolute_error(y_test, RF.predict(x_test))
```

1232.3528089560773

9. ADVANTAGES & DISADVANTAGES

Advantages:

- Variants usually don't matter in the used car market. If you search well, you can get a top-spec less driven car in the used car market at a price which you would have otherwise paid for a lower variant in case of buying a new car.
- If you buy a car from a brand authorised dealership, you get a warranty on the repair.
- If we are buying a used car that was launched a year ago, you can save up to 20% on its original cost.

Disadvantages:

- Some cars may be lemons. They look fine on the outside but can land in huge repair costs while you use them.
- Be a very informed customer and check each and every possible detail before buying.

10. CONCLUSION

The increased prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a global increase. Therefore, there is an urgent need for a Used Car Price Prediction system which effectively determines the worthiness of the car using a variety of features. The proposed system will help to determine the accurate price of used car price prediction.

11. FUTURE SCOPE

In future this machine learning model may bind with various websites which can provide real time data for price prediction. Also we may add large historical data of car price which can help to improve accuracy of the machine learning model. We can build an android app as a user interface for interacting with users. For better performance, we plan to judiciously design deep learning network structures, use adaptive learning rates and train on clusters of data rather than the whole dataset.

12. APPENDIX

Source

Code: app.py

```
from flask import Flask, render_template, request
import numpy as np
import pickle
import requests

API_KEY = "g1tklKaS5mB-mBztAhCCv8gMemWgixeSSsD4qW0Smqr9" token_response =
requests.post('https://iam.cloud.ibm.com/identity/token',
              data={"apikey": API_KEY,
                    "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

app = Flask(__name__)

def ValuePredictor(to_predict_list):
    to_predict =
    np.array(to_predict_list)
    loaded_model = pickle.load(open("Car Resale Value Prediction.pkl", "rb"))
    result = loaded_model.predict([to_predict])
    return result[0]

@app.route('/')
def index():
    return render_template("Untitled.html")

@app.route('/login', methods=['POST'])#binds to an url
def login():
    if request.method == 'POST':
        to_predict_list = request.form.to_dict()
        to_predict_list = list(to_predict_list.values())
        to_predict_list = list(map(int, to_predict_list))
        result = ValuePredictor(to_predict_list)

        # payload_scoring = {"input_data": [{"fields":
        [['abtest','vehicleType','yearOfRegistration','gearbox','powerPS',
        'kilometer','monthOfRegistration','fuelType','brand','notRepairedDamage']], "values":
        to_predict_list}}

        # response_scoring = requests.post('https://us-
        south.ml.cloud.ibm.com/ml/v4/deployments/d90bd0ad-9588-4a46-a97
```



```

1c0e56c56ae0/predictions?version=2022-11-12', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
    # print("Scoring response")

    # print(response_scoring.json())

    return render_template("Untitled.html",y = "The Resale Value of the Car is : $"
+ str(round(result)))

@app.route('/admin')#binds to an
url def admin():
    return "Hey Admin How are you?"

if __name__ == '__main__':
    app.run(debug=
    True)

```

index.html

```

<html>
<head>
<style>
    body{
        background-image: url({{
            url_for('static',filename='images/images.jpeg')}});
        background-repeat:no-repeat;
        background-attachment:fixed
        ;

        background-size:100%}

        h1 {padding-top:30px;}

        label{
            width:200px;
            display:inline-block;
        }

        div{
            padding-top:20px

```

;

padding-left:50px

;

}

select{

width:150p

x

}

</style>

</head>

<body text='white'>

<h1><center>Car Resale Value Prediction</center></h1>

<div>

<form action = "/login" method= "POST" >

<p><label for = "abtest">Choose AB Test</label>

<select name ="ab">

<option hidden></option>

<option Value = "0">Control</option>

<option Value = "1">Test</option>

</select></p>

<p><label for = "vehicle">Choose Vehicle Type</label>

<select name ="vehicle">

<option hidden></option>

<option Value = "0">Andere </option>

<option Value = "1">Bus </option>

<option Value = "2">Cabrio </option>

<option Value = "3">Coupe</option>

<option Value = "4">Kleinwagen</option>

<option Value = "5">Kombi</option>

<option Value = "6">Limousine</option>

<option Value = "7">Suv</option>

</select></p>

<p><label>Year Of Registration</label>

<input type= "number" name = "year" /></p>

<label for = "gear">Choose Gear Box</label>

<select name ="gear">

<option hidden></option>

<option Value = "0">Automatic</option>

<option Value = "1">Manuel</option>

</select>

<p><label> Number of Power PS </label>

<input type= "number" name = "power" /></p>

<p><label> Number of Kilometers Driven</label>

<input type= "number" name = "kilometer" /></p>

<p><label> Month Of Registration </label>

<input type= "number" name = "month" /></p>

<p><label for = "fuel">Choose Fuel Type</label>

<select name ="fuel">

<option hidden></option>

<option Value = "0">Andere</option>

<option Value = "1">Benzin</option>

<option Value = "2">CNG</option>

<option Value = "3">Diesel</option>

<option Value = "4">Elektro</option>

<option Value = "5">Hybrid</option>

<option Value = "6">LPG</option>

</select></p>

<p><label for = "brand">Choose the Brand of the Car</label>

<select name ="brand">

<option hidden></option>

<option Value = "0">Alfa_romeo</option>
<option Value = "1">Audi</option>
<option Value = "2">Bmw</option>
<option Value = "3">Chevrolet</option>
<option Value = "4">Chrysler</option>
<option Value = "5">Citroen</option>
<option Value = "6">Dacia</option>
<option Value = "7">Daewoo</option>
<option Value = "8">Daihatsu</option>
<option Value = "9">Fiat</option>
<option Value = "10">Ford</option>
<option Value = "11">Honda</option>
<option Value = "12">Hyundai</option>
<option Value = "13">Jaguar</option>
<option Value = "14">Jeep</option>
<option Value = "15">Kia</option>
<option Value = "16">Lada</option>
<option Value = "17">Lancia</option>
<option Value = "18">Land_rover</option>
<option Value = "19">Mazda</option>
<option Value = "20">Mercedes_benz</option>
<option Value = "21">Mini</option>
<option Value = "22">Mitsubishi</option>
<option Value = "23">Nissan</option>
<option Value = "24">Opel</option>
<option Value = "25">Peugeot</option>
<option Value = "26">Porsche</option>
<option Value = "27">Renault</option>
<option Value = "28">Rover</option>
<option Value = "29">Saab</option>
<option Value = "30">Seat</option>
<option Value = "31">Skoda</option>

```
<option Value = "32">Smart</option>
<option Value = "33">Sonstige_autos</option>
<option Value = "34">Subaru</option>
<option Value = "35">Suzuki</option>
<option Value = "36">Toyota</option>
<option Value = "37">Trabant</option>
<option Value = "38">Volkswagen</option>
<option Value = "39">Volvo</option>
</select></p>
```

```
<label for = "damage">Are the Damages Repaired</label>
<select name ="damage">
<option hidden></option>
<option Value = "0">Yes</option>
<option Value = "1">No</option>
</select>
```

```
<p><input type="submit" value = "Submit" /></p>
</form>
<h4>{{y}}</h4></div>
```

```
</body>
```

```
</html>
```

GitHub :

<https://github.com/IBM-EPBL/IBM-Project-7011-1658845046>

Project Demo Link:

<https://github.com/IBM-EPBL/IBM-Project-7011-1658845046>

[/blob/main/Project%20Development%20Phase/Car%20Resale%20 Value%20Prediction.mp4](#)