CAR RESALE VALUE PREDICTION

Submitted by

SANJANA SURESH - 312319104142
ROSE MARY CHITTILAPPILLY - 312319104131
SHARANYA N - 312319104155
VIJAY ADHIRA C - 312319104185

PROJECT ID: PNT2022TMID00170

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

TABLE OF CONTENT

1. INTRODUCTION

1.1 Project Overview

This system “Car Resale Value Prediction” aims to build a regression model to predict used cars' resale value based on multiple aspects, including vehicle mileage, year of manufacturing, fuel consumption, fuel type,brand, repairs made, gear type and model. This model can benefit sellers, buyers, and car manufacturers in the used cars market. Upon completion, it can output a relatively accurate price prediction based on the information that user’s input. Various regression methods, including linear regression, polynomial regression, support vector regression, decision tree regression, and random forest regression, were applied in the research to achieve the highest accuracy. This system was implemented as a web application where the user enters the details of the car to get an estimation of the car’s resale value, using Sklearn's supervised machine-learning techniques integrated with the Spark-Sklearn library.

1.2 Purpose

Car resale value prediction helps the user to predict the resale value of the car depending upon various features like kilometers driven, fuel type, etc. The purpose of this system is of commercial interest to sellers to be able to predict the resale value of cars with better accuracy. The most essential elements for forecast are brand and model, period use of vehicle, mileage of vehicle, gear type and fuel type utilized in the vehicle. The user enters the details of the car into the form given and accordingly the car resale value is predicted.

2. LITERATURE SURVEY

2.1 Existing problem

The used car price prediction problem has a certain value because different studies show that the market of used cars is destined for continuous growth in the short term. In fact, leasing cars is now a common practice through which it is possible to get hold of a car by paying a fixed sum for an agreed number of months rather than buying it. Once the leasing period is over, it is possible to buy the car by paying the residual value, i.e. at the expected resale price. It is therefore in the interest of vendors to be able to predict this value with a certain degree of accuracy since if this value is initially underestimated, the installment will be higher for the customer which will most likely opt for another dealership. It is therefore clear that the price prediction of used cars has a high commercial value, especially in developed countries where the economy of leasing has a certain volume.

This problem, however, is not easy to solve as the car's value depends on many factors including a year of registration, manufacturer, model, mileage, horsepower, origin, and several other specific information such as type of fuel and braking system, condition of bodywork and interiors, interior materials, safety index, type of change (manual, assisted, automatic, semi-automatic), number of doors, number of previous owners, if it was previously owned by a private individual or by a company and the prestige of the manufacturer.

Unfortunately, only a tiny part of this information is available, and therefore it is very important to relate the results obtained in terms of accuracy to the features available for the analysis. Moreover, not all the previously listed features have the same importance, some are more so than others, and therefore is essential to identify the most important ones, on which to perform the analysis.Since some attributes of the dataset aren't relevant to our analysis, they have been discarded; so, as mentioned above, this fact must be taken into account when conclusions on the accuracy are drawn.

2.2 References

1. **Used car price prediction**, by Praful Rane, Deep Pandya, Dhawal Kotak in 2021. The authors have implemented and evaluated the dataset and compared the performance of various machine learning algorithms like Linear Regression, Ridge Regression, Lasso Regression, Elastic Net, Decision Tree Regressor and choose the best out of it. Depending on various parameters they determined the price of the car.
   ADVANTAGES: Determine the accurate price used car price prediction.
   DISADVANTAGES: To improve the accuracy of the model.

2. **Used Cars Price prediction and Valuation using Data Mining Techniques**, by Abdulla AlShared (2021). The author used Random Forest Regressor, Logistic Regression to predict the used car price
   ADVANTAGES: Pre-processing and transformation, Random Forest Regressor came out on top with 95% accuracy followed by Bagging Regressor with 88%.
   DISADVANTAGES: Only a few data have been used.

3. **Used Car Price Prediction using K- Nearest Neighbor Based Model**, by K.Samruddhi , Dr. R.Ashok Kumar
   (2020). The authors Used Cars data set was taken and data processing was done to filter the data and to remove some unnecessary data. The model was trained with the processed data using the KNN algorithm to predict the sales of used cars with higher accuracy.
   ADVANTAGES: The K nearest Neighbor algorithm and we got accuracy 85% where the accuracy of linear regression is 71%
   DISADVANTAGES: The optimization of the model with improved accuracy was not obtained

4. **Used Cars Price Prediction using Supervised learning techniques** by Mukkesh Ganesh,PattabiramanVenkatasubbu in2019. ANOVA, Lasso Regression, Regression Tree, Tukey's Test we will create, train and test the effectiveness of our statistical models.
   ADVANTAGES: The prediction error rate of all the models was well under the accepted 5% of error.
   DISADVANTAGES: The mean error of the regression tree model was found to be more than the mean error rate of the multiple regression and lasso regression models.

5. **Predicting the Price of Used Cars** using Machine Learning Techniques by SameerchandPudaruth in 2014. The authors implemented and evaluated the data using Multiple Linear regression analysis, K-nearest neighbors, naive bayes and decision trees have been used to make predictions of used cars.
   ADVANTAGES: Accuracy dangled between 60-70% for different combinations of parameters.
   DISADVANTAGES: Only a few numbers of records that have been used.

2.3 Problem Statement Definition

The main aim of this project is to predict the price of used cars using various Machine Learning (ML) models. This can enable the customers to make decisions based on different inputs or factors namely. It is easy for any company to price their new cars based on the manufacturing and marketing cost it involves. But when it comes to a used car it is quite difficult to define a price because it is influenced by various parameters like car brand, manufactured year etc. The goal of our system is to predict the best price for a used car based on the previous data related to sold cars using machine learning.

- Brand or Type of the car one prefers (Example: Ford, Hyundai)
- Model of the car namely (Example: Ford Figo, Hyundai Creta)
- Location (Example: Delhi, Chennai, Mumbai)
- Year of manufacturing (Example: 2020, 2021)
- Type of fuel (Example: Petrol, Diesel)
- Price range or Budget
- Type of transmission which the customer prefers (Example: Automatic or Manual)
- Mileage

The project Car Price Prediction deals with providing the solution to these problems. Through this project, we will get to know which of the factors are significant and tell us how they affect the car’s worth in the market

PROBLEM STATEMENT CREATION

GENERAL FORMAT:

"___ is a _____who needs a way to____so that _____"

PROBLEM STATEMENT 1:

"The customer is a potential buyer of a used car who needs a way to find a reliable source for quotation as he needs to avoid being deceived by the seller".

Five W's for problem statement 1:

QUESTION	DESCRIPTION
WHO does the problem affect?	**The problem affects the customer buying the car**
WHAT are the issues and boundaries of the problem?	**The issue of this problem is that they want a reliable source for quotation, and the problem boundaries include the customer, geographic area in which the car is sold, the dealer, etc.**
WHEN does the issue occur?	**The issue occurs when the customer looks to purchase a car for resale.**
WHERE is the issue occurring?	**The issue occurs at the side of the customer who wants to find a reliable price.**
WHY is it important that we fix the problem	**It allows the customer to feel satisfied with the price he is paying for the car.**

PROBLEM STATEMENT 2:

“The customer is a potential buyer of a used car who needs a way to independently gain access to a quotation as he wants to avoid needing to rely on a large number of people to request it.”

Five W’s for problem statement 2:

QUESTION	DESCRIPTION
WHO does the problem affect?	**The customer, while looking to purchase a car for resale and in search of quotations.**
WHAT are the issues and boundaries of the problem?	**The issue of this problem is that the customer does not want to rely on a large number of people for assessing the quotation, and instead wants to be able to independently access a quotation.**
WHEN does the issue occur?	**It occurs while the customer is trying to find a quotation for a resale car.**
WHERE is the issue occurring?	**At the customer end when he is looking for a quotation.**
WHY is it important that we fix the problem	**It allows the customer to be independent while looking up the quotation for a car for resale.**

PROBLEM STATEMENT 3:

The car dealer is a potential seller of a used car who needs a way to provide precise quotation as he needs to avoid undercharging or overcharging the customer.

Five W's for problem statement 3:

QUESTION	DESCRIPTION
WHO does the problem affect?	**The problem affects the dealer who is selling the car that is up for resale.**
WHAT are the issues and boundaries of the problem?	**The issue is that the dealer wants to quote a correct price so that he does not undercharge the customer and endure a loss, or overcharge the customer and lose his reputation.**
WHEN does the issue occur?	**The issue occurs when the dealer provides the customer with a quotation for the car up for resale.**
WHERE is the issue occurring?	**The issue occurs on the dealer's side, when he wants to provide a quotation for a used car.**
WHY is it important that we fix the problem	**This will allow the dealers to feel confident about their quotation, without worrying if they are undercharging or overcharging the customer.**

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

Edit this template
Right-click to unlock

Empathy Map Canvas

Gain insight and understanding on solving customer problems.

1

Build empathy and keep your focus on the user by putting yourself in their shoes.

Share your feedback

3.2 Ideation & Brainstorming

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we Give an accurate quotation prediction for car resale value?

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

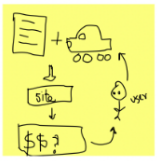
2

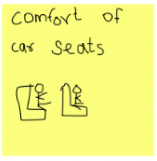
Brainstorm

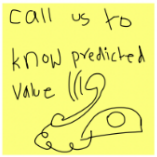
Write down any ideas that come to mind that address your problem statement.

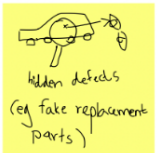
10 minutes

Person 1 - Sanjana Suresh

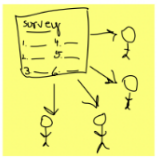








Create a Resale Value Mobile app



scan car through camera and automatically predicts price

Person 2- Rose Mary Chittiappily

Quality of the parts	Analysis of existing database of documentation	Enter License Plate number and get all details of car Regarding previous maintainances
Cost estimation for fixing damages in an old car	Allow user to give feedback and use that to improve sytem	Filter options based on buyer preference
Scan a picture and assess the hygienic maintenance of car interiors	Check accident history in police records	Check bank statements of customers and view car loan data

Person 3 - Sharanya N

Hologram interaction with car for sale	Ask goernment car loan providing representative	Chatting between buyer and seller
Condition of the vehicle	Compare with other listings online	Check for any rusting
Smell of car interior	Request dry cleaning before purchase	Obtain Data sets from Data repositories like kaggle

Person 4 - Vijay Adhira C

Depreciation	Brand	Kilometers Covered
Number of historical owners	what similar cars in the same condition are worth	usage of algorithms to predict resale value
In-Demand	Cross check results from two different algorithms	Manufacturing year

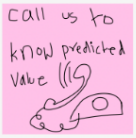
3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Group 1 Ideas for Interaction with customer



Chatting between buyer and seller

Create a Resale Value Mobile app

Hologram interaction with car for sale

Group 2 Feature ideas

scan car through camera and automatically predicts price

Enter License Plate number and get all details of car Regarding previous maintainances

Request dry cleaning before purchase

Cost estimation for fixing damages in an old car

Scan a picture and assess the hygienic maintenance of car interiors

Filter options based on buyer preference

Group 3 Ideas for factors to take into consideration in price calculation

Depreciation

Brand

Manufacturing year

Kilometers Covered

Number of historical owners

In-Demand

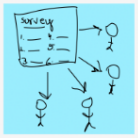
Group 4 Sources of raw Data for analysis

what similar cars in the same condition are worth

Obtain Data sets from Data repositories like kaggle

Check accident history in police records

Analysis of existing database of documentation



Check bank statements of people and view car loan data

Group 5 Ideas to verify our claims and improve accuracy

Compare with other listings online

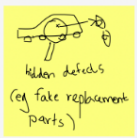
Cross check results from two different algorithms

Allow user to give feedback and use that to improve system

usage of algorithms to predict resale value

Ask goernment car loan providing representative

Group 6 Ideas for focus areas of inspection in the car PHYSICAL INSPECTION



Smell of car interior

Check for any rusting

Quality of the parts

Condition of the vehicle

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To develop an efficient and effective model which predicts the price of a used car according to the user's inputs. To achieve good accuracy. To develop a User Interface(UI) which is user-friendly and takes input from the user and predicts the resale value of a car.
2.	Idea / Solution description	We will be trying various regression algorithms and choose the most efficient algorithm that gives the result with the best accuracy. Then it will be integrated to the web-based application where the user is able to upload his data and obtain the estimated car resale value. The process for testing each algorithm is as follows: 1. Training phase: The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly. 2. Testing phase: the system is provided with the inputs and is tested for its working. The accuracy is checked. Therefore, the data that is used to train the model or test it, has to be appropriate.
3.	Novelty / Uniqueness	Multiple factors are considered while deciding the resale value of the car, and this data is used to improve the accuracy of the algorithm. The focus of the website design puts emphasis on the ease of usage and ensures that the users have a smooth experience without any hassles or complications.
4.	Social Impact / Customer Satisfaction	With the rise in automobile ownership, the used the automobile market is ripe for growth. The healthy development of the used car market requires an accurate used car pricing evaluation. Since the proposed system can be real-time and user friendly in terms of its handling, it is an overall unique proposal idea that gives overall customer satisfaction.
5.	Business Model (Revenue Model)	Advertisement columns on both sides of the website. People are required to log in to use the price estimator. Using the estimator more than 3 times a month requires additional charges.
6.	Scalability of the Solution	Even with a large volume of site traffic, we can still respond to queries of all the customers, as the process is automated and requires no human interaction. The datasets used will also increase over the years, as we gain additional data, improving the overall accuracy of the system.

3.4 Problem Solution fit

Problem-Solution fit canvas 2.0

Car Resale Value prediction

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

1. Parents buying second hand car for their child.
2. Young people just starting out and want a used car.
3. Adults who don't want to spend on a new car
5. Retirees looking to downsize
6. Companies buying cars for a fleet.

6. CUSTOMER CONSTRAINTS CC

1. Initial verifications can be done only online
2. Customer must provide accurate data for correct price prediction.
3. Overlooking certain defects/other details while entering car information can result in incorrect predictions

5. AVAILABLE SOLUTIONS AS

1. Asking a car dealer
2. Asking brokers
3. Existing websites
4. Information from friends, relatives and acquaintances which may not be accurate.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

1. Understand the changing demands of the user
2. Make sure information entered by the seller are true
3. Creation of a website to provide for an easy to use interface where user can find price predictions of car

9. PROBLEM ROOT CAUSE RC

1. There is a need for reliable sources to estimate the price.
2. Sellers may leave out a few details that could affect the price.
3. The feeling of dependency on a large number of people to assess and provide the price may be unsettling, and the user feels a want to be able to independently find a price estimation.

7. BEHAVIOUR BE

1. Customers seek online listings of similar cars
2. Customers seek advice from a large range of people
3. Need to sell their car for space, money or to get a replacement
4. Customer will send all the documents and proof of ownership and of the condition of the car

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Customers who leased cars and want to purchase the car for its resale value and hence need price prediction.

4. EMOTIONS: BEFORE / AFTER EM

If the buyer or seller feels like they could find a better deal some other way, customers loose trust in the product

10. YOUR SOLUTION SL

1. A Car price predictor that is implemented using a regression technique.
2. A flask based Website that hosts the price predictor and allows the user to have a comfortable and easy experience .

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

The users will estimate resale value of a car with ease using the website

8.2 OFFLINE

1. Ask other dealers
2. Ask friends and family for opinions
3. Look at other listings

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

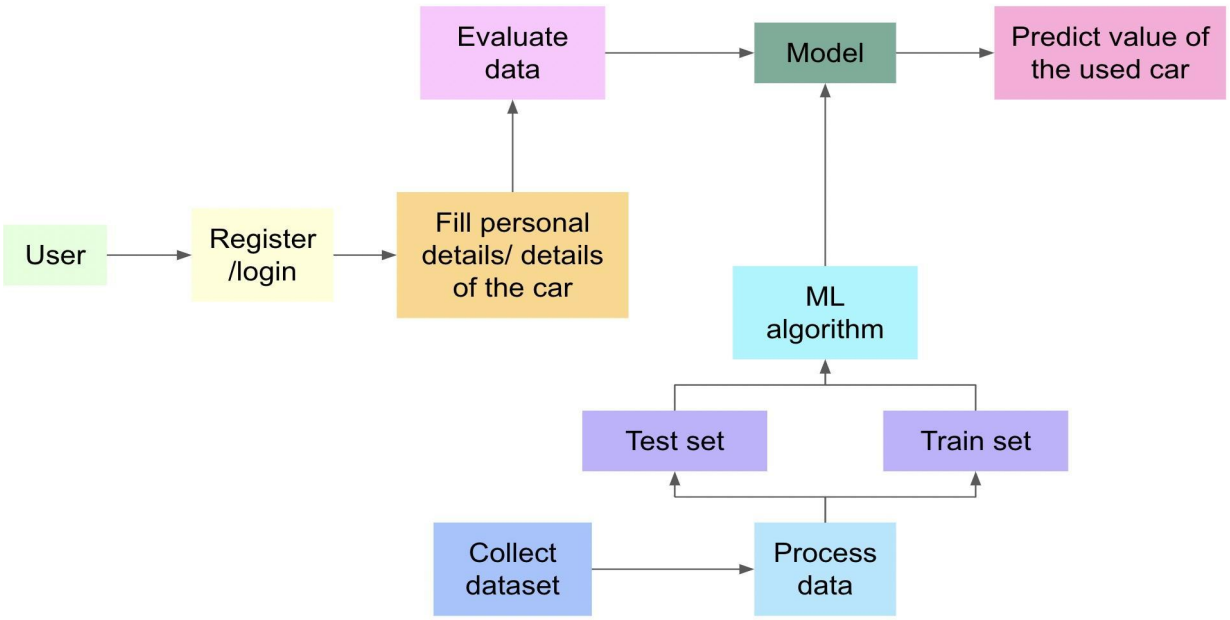
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Phone Number Registration through Email Registration through Google Account
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Profile	Personal Details Interest in buying or selling
FR-4	Information about the vehicle	Registration details Brand, Model Insurance and loan Price Condition
FR-5	Value Prediction	Analyze the brand, model, condition and estimate a value
FR-6	Make Recommendations	Make recommendations based on the needs of the customer (budget, model, fuel, manual/automatic)

4.2 Non-Functional requirements

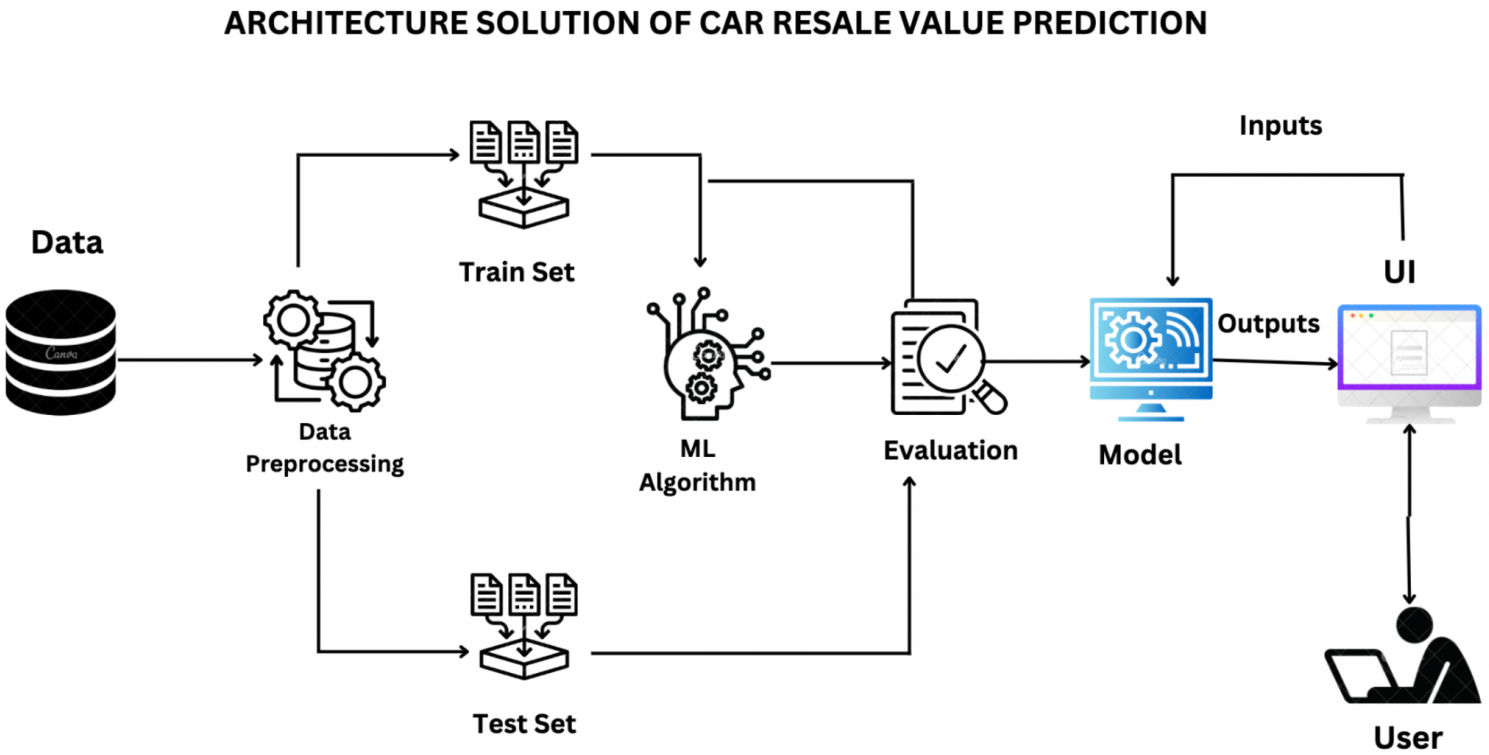
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Predicts the car resale value of used cars. With a simple UI. Easy to navigate. Buyers and sellers pages will not be together.
NFR-2	**Security**	Data of the users are protected. Only legit users and admins can view the information the user wishes to publish.
NFR-3	**Reliability**	Reliability is defined as the probability that a product, system, or service will perform its intended function adequately for a specified period of time, or will operate in a defined environment without failure.
NFR-4	**Performance**	It is the amount of work accomplished by a computer system. The word performance in computer performance means “How well is the computer doing the work it is supposed to do?”.
NFR-5	**Availability**	Availability means the probability that a system is operational at a given time, i.e. the amount of time a device is actually operating as the percentage of total time it should be operating.
NFR-6	**Scalability**	Scalability refers to the ability of an organization to perform well under an increased or expanding workload.

5. PROJECT DESIGN

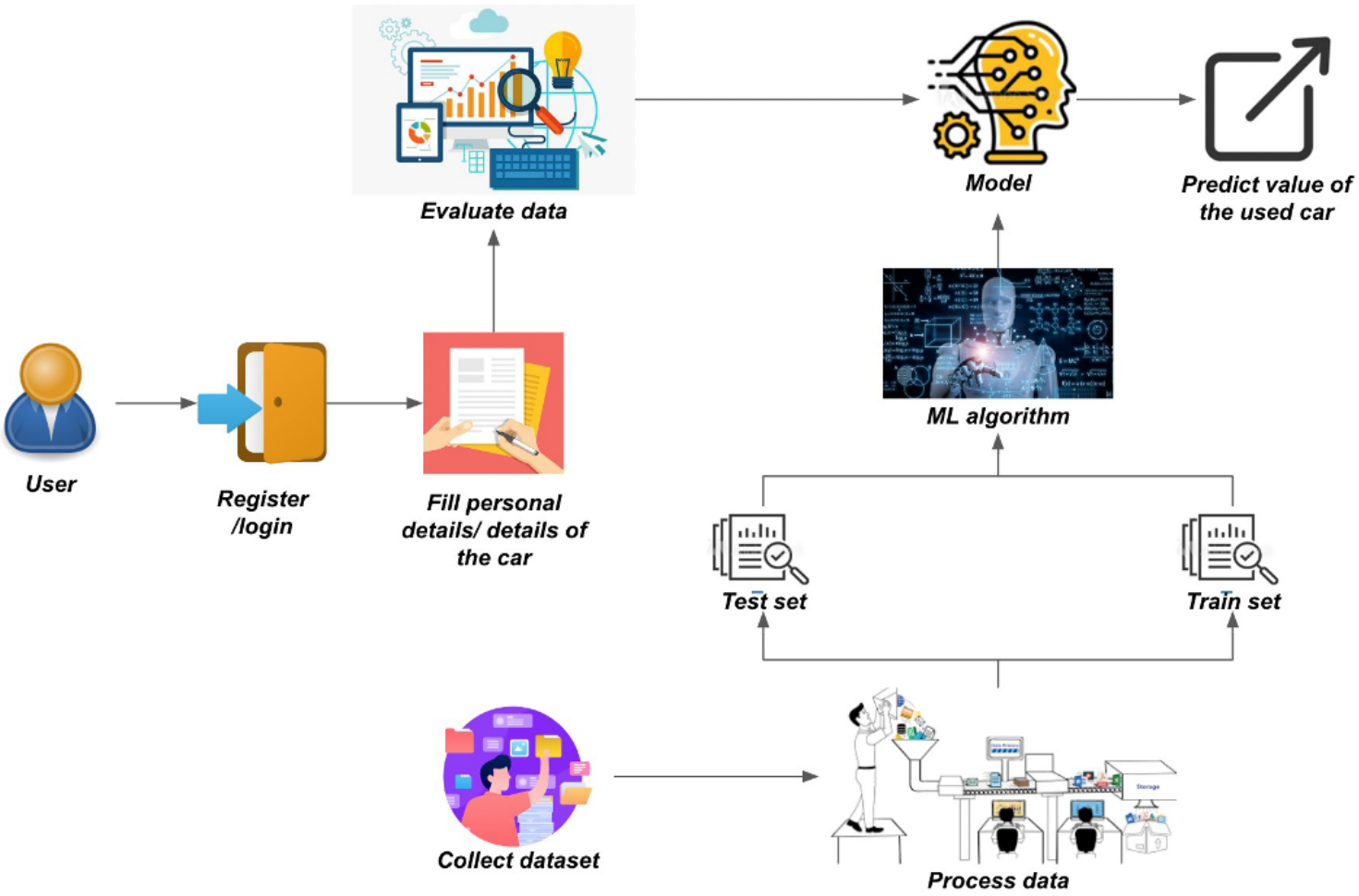
5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 Technical Architecture



5.4 Components and Technologies

S.No	Component	Description	Technology
1.	User Interface	The user interface is the point at which human users interact with a computer, website or application. The goal of effective UI is to make the user's experience easy and intuitive, requiring minimum effort on the user's part to receive the maximum desired outcome.	HTML, CSS, JavaScript / React Js etc.
2.	Application Logic-1	Logic for a process in the application	python
3.	Database	A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).	Structured Query Language (SQL)
4.	Cloud Database	A cloud database is a database built to run in a public or hybrid cloud environment to help organize, store, and manage data within an organization. Cloud databases can be offered as a managed database-as-a-service (DBaaS) or deployed on a cloud-based virtual machine and self-managed by an in-house IT team.	IBM DB2
5.	File Storage	File storage—also called file-level or file-based storage—is a hierarchical storage methodology used to organize and store data on a computer hard drive or on a network-attached storage (NAS) device.	IBM Block Storage
6.	Machine Learning Model	A machine learning model is a file that has been trained to recognize certain types of patterns. You train a model over a set of data, providing it an algorithm that it can use to reason over and learn from those data	K-nearest neighbor, random forest,support vector machine.
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud	Local, Cloud Foundry, Kubernetes, etc.

5.5 Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask, scikit learn,tensor flow
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	Encryptions, decryptions
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	My SQL
4.	Availability	Justify the availability of applications (e.g. use of load balancers, distributed servers etc.)	IBM Watson- can be accessed easily
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Flask-handle multiple requests

5.6 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	HomePage	USN-1	Description about car resalePro cess	I can get an idea about how car resale works.	Low	Sprint-3
		USN-2	Details about the required data		Low	Sprint-3
	Registration	USN-3	As a user, I can register for the application by entering my username, email, phone number, and password, and confirming my password and verifying it.	I can access my account.	Moderate	Sprint-3
		USN-4	As a user, I will receive a confirmation mail once I've registered for the application.	I can receive a confirmation OTP upon registration for verification.	High	Sprint-3
	Login	USN-5	As a user, I can log in to the web application by entering my email id & password.	I can log in successfully.	High	Sprint-2
	Main Page	USN-6	As a user, I submit my car model with a release date.	I can access the page and can submit the car details.	Moderate	Sprint-4
	Price prediction	USN-7	The predicted resale price for the given car model will be displayed.	I got a predicted resale price successfully for my car model.	High	Sprint-4

Admin	Data collection	USN-8	Collect the required data for theCar resale prediction.		High	Sprint-1
	Data preprocessing	USN-9	Clean and analyze the data to avoid duplications	As a result, I get the desired dataset to get trained.	High	Sprint-1
	Model Building	USN-10	Build the model using a RandomForest regression to classify the data.	Successfully trained the model.	High	Sprint-1
	Deploy the model	USN-11	Deployment of ML model using IBM Watson Studio, object storage.	Deployed successfully.	High	Sprint-2
	Integrate the web app with the IBM model	USN-12	Use flask for the integration purpose.	Created the web app successfully.	Moderate	Sprint-2

6. PROJECT PLANNING & SCHEDULING

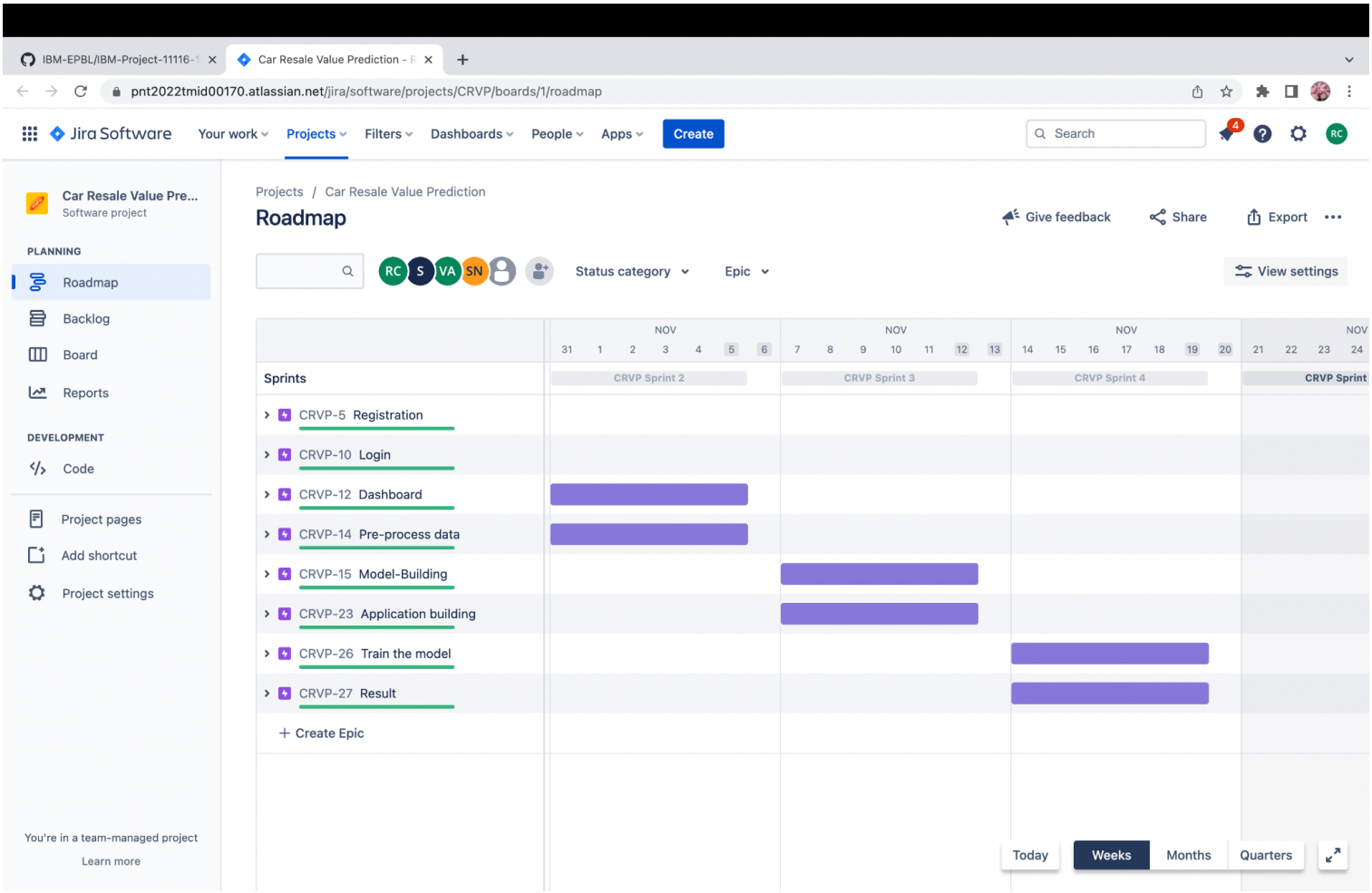
6.1 Sprint Planning & Estimation

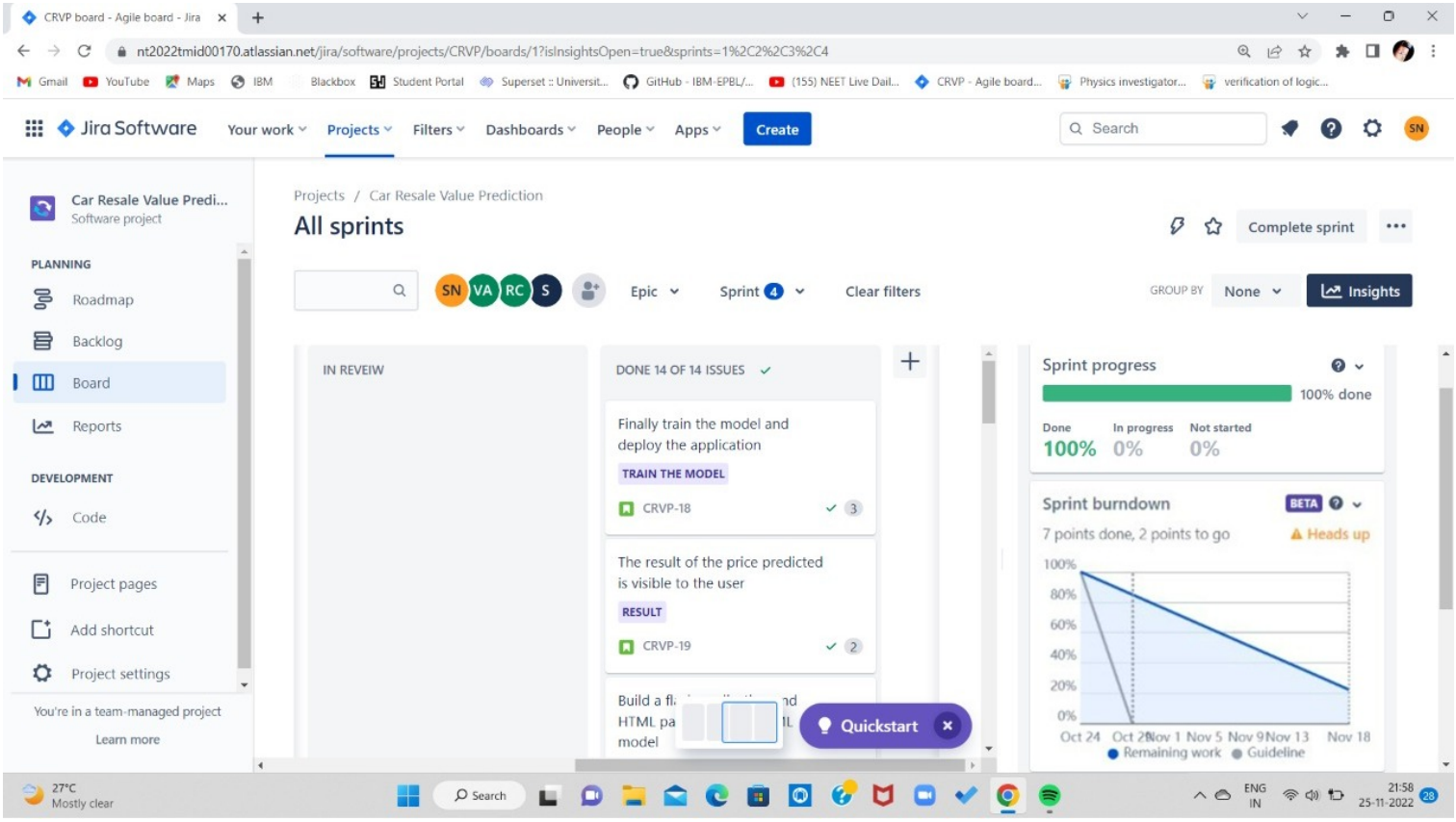
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Sanjana Suresh
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Sharanya N
Sprint-1	Registration	USN-3	As a user, i can register for the application through google	1	Low	Vijay Adhira
Sprint-1	Login	USN-1	As a user, I can log into the application by entering email & password	2	High	Rose Mary C
Sprint-2	Dashboard	USN-1	As a user I fill in the details of the car I want to sell or the details of the type of car I want to purchase	3	High	Sanjana Suresh Rose Mary C
Sprint-2	Pre-process data	USN-1	Collect dataset	2	High	Sharanya N Vijay Adhira
		USN-2	Import libraries	1	Medium	Sanjana Suresh Rose Mary C
		USN-3	Read and clean dataset	2	High	Sharanya N Vijay Adhira
Sprint-3	Model-Building	USN-1	Split data into dependent and independent variables	2	High	Sharanya Vijay Adhira
		USN-2	Choose an appropriate model for building (random forest regression)	2	High	Sanjana Suresh Rose Mary C
Sprint-3	Application building	USN-1	Build a flask application and HTML page. Deploy the ML model	2	High	Sanjana Suresh Rose Mary C
		USN-2	Execute and test	2	Medium	Sharanya N Vijay Adhira
Sprint-4	Train the model	USN-1	Finally train the model and deploy the application	3	Medium	Rose Mary C Sharanya N
Sprint-4	Result	USN-1	The result of the price predicted is visible to the user	2	High	Sanjana Suresh Vijay Adhira

6.2 Sprint Delivery Schedule

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	6	6 Days	24 Oct 2022	29 Oct 2022	6	29 Oct 2022
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	8	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022	5	19 Nov 2022

6.3 Reports from JIRA

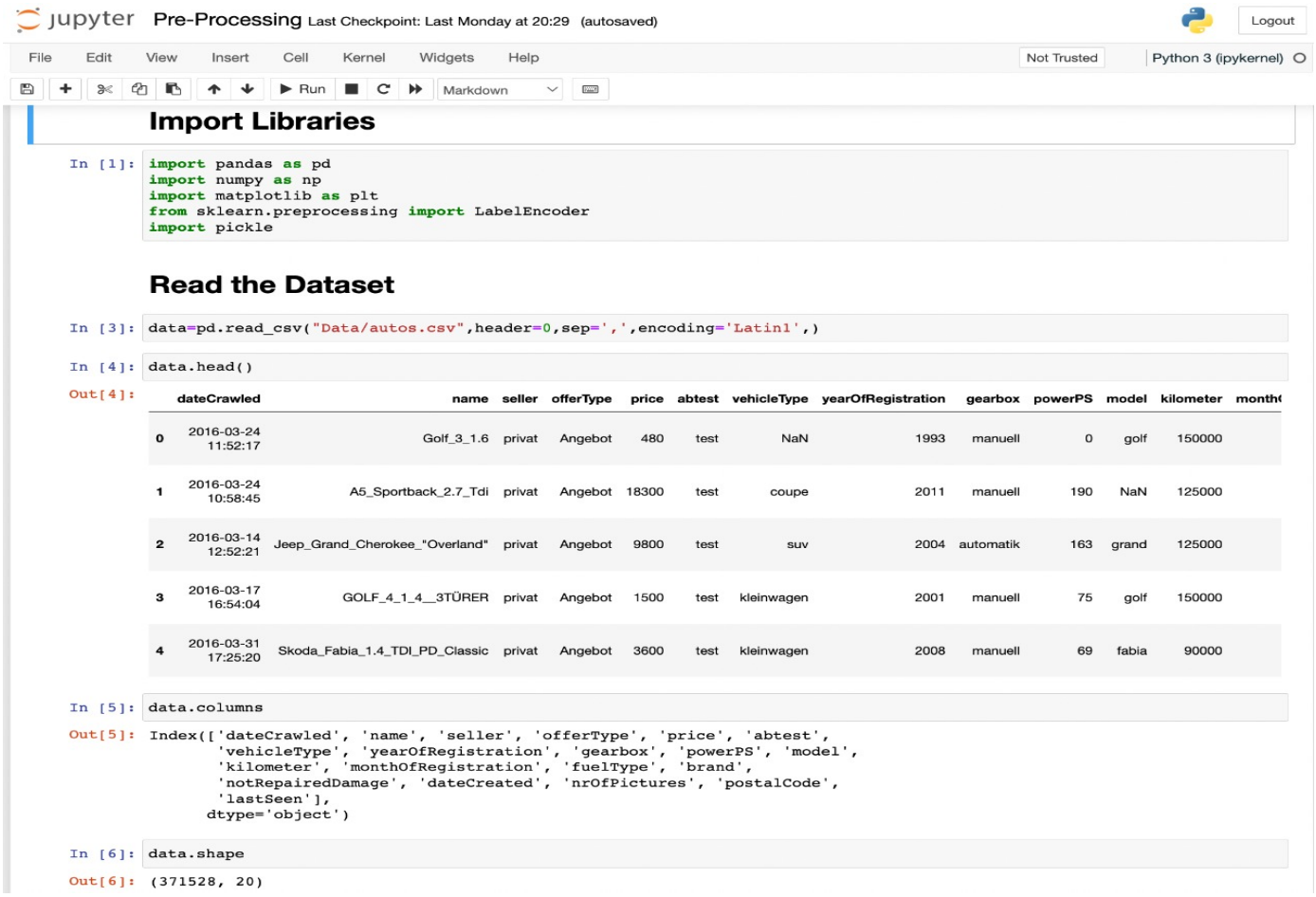



CRVP board - Agile board - Jira
nt2022tmid00170.atlassian.net/jira/software/projects/CRVP/boards/1?isInsightsOpen=true&sprints=1%2C2%2C3%2C4
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Car Resale Value Predi...
Software project
PLANNING
Roadmap
Backlog
Board
Reports
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Car Resale Value Prediction
All sprints
Complete sprint
Epic
Sprint 4
Clear filters
GROUP BY
None
Insights
IN REVEIW
DONE 14 OF 14 ISSUES
Finally train the model and deploy the application
TRAIN THE MODEL
CRVP-18
The result of the price predicted is visible to the user
RESULT
CRVP-19
Quickstart
Sprint progress
100% done
Done
100%
In progress
0%
Not started
0%
Sprint burndown
BETA
7 points done, 2 points to go
Heads up
100%
80%
60%
40%
20%
0%
Oct 24
Nov 1
Nov 5
Nov 9
Nov 13
Nov 18
Remaining work
Guideline
27°C
Mostly clear
Search
ENG
IN
21:58
25-11-2022

7. CODING & SOLUTIONING

7.1 Feature 1

We have trained the ML model for predicting the resale value car using Random Forest Regressor with the accuracy of 86.7%.



jupyter Pre-Processing Last Checkpoint: Last Monday at 20:29 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel)

Run Markdown

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
```

Read the Dataset

In [3]:
```
data=pd.read_csv("Data/autos.csv",header=0,sep=',',encoding='Latin1',)
```

In [4]:
```
data.head()
```

Out[4]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthO
0	2016-03-24 11:52:17	Golf_3_1.6	privat	Angebot	480	test	NaN	1993	manuell	0	golf	150000	
1	2016-03-24 10:58:45	A5_Sportback_2.7_Tdi	privat	Angebot	18300	test	coupe	2011	manuell	190	NaN	125000	
2	2016-03-14 12:52:21	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800	test	suv	2004	automatik	163	grand	125000	
3	2016-03-17 16:54:04	GOLF_4_1_4__3TÜRER	privat	Angebot	1500	test	kleinwagen	2001	manuell	75	golf	150000	
4	2016-03-31 17:25:20	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600	test	kleinwagen	2008	manuell	69	fabia	90000	

In [5]:
```
data.columns
```

Out[5]:
```
Index(['dateCrawled', 'name', 'seller', 'offerType', 'price', 'abtest',
       'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model',
       'kilometer', 'monthOfRegistration', 'fuelType', 'brand',
       'notRepairedDamage', 'dateCreated', 'nrOfPictures', 'postalCode',
       'lastSeen'],
      dtype='object')
```

In [6]:
```
data.shape
```

Out[6]:
```
(371528, 20)
```

Checking null values

In [7]:
```
data.isna().sum()
```

Out[7]:
```
dateCrawled                0
name                       0
seller                     0
offerType                  0
price                      0
abtest                     0
vehicleType            37869
yearOfRegistration         0
gearbox                20209
powerPS                    0
model                  20484
kilometer                  0
monthOfRegistration        0
fuelType               33386
brand                      0
notRepairedDamage      72060
dateCreated                0
nrOfPictures               0
postalCode                 0
lastSeen                   0
dtype: int64
```

Checking the Unique values

In [8]:
```
data.nunique()
```

Out[8]:
```
dateCrawled            280500
name                   233531
seller                      2
offerType                   2
price                    5597
abtest                      2
vehicleType                 8
yearOfRegistration        155
gearbox                     2
powerPS                   794
model                     251
kilometer                  13
monthOfRegistration        13
fuelType                    7
brand                      40
notRepairedDamage           2
dateCreated               114
nrOfPictures                1
postalCode               8150
lastSeen               182806
dtype: int64
```

Cleaning The Dataset

In [9]:
```
print(data.seller.value_counts())
```

```
privat        371525
gewerblich         3
Name: seller, dtype: int64
```

In [10]:
```
data[data.seller !='gewerblich']
```

Out[10]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	m
0	2016-03-24 11:52:17	Golf_3_1.6	privat	Angebot	480	test	NaN	1993	manuell	0	
1	2016-03-24 10:58:45	A5_Sportback_2.7_Tdi	privat	Angebot	18300	test	coupe	2011	manuell	190	
2	2016-03-14 12:52:21	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800	test	suv	2004	automatik	163	g
3	2016-03-17 16:54:04	GOLF_4_1_4__3TÜRER	privat	Angebot	1500	test	kleinwagen	2001	manuell	75	
4	2016-03-31 17:25:20	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600	test	kleinwagen	2008	manuell	69	
...	...	...	...	...	...	...	...	...	...	...	
371523	2016-03-14 17:48:27	Suche_t4___vito_ab_6_sitze	privat	Angebot	2200	test	NaN	2005	NaN	0	
371524	2016-03-05 19:56:21	Smart_smart_leistungssteigerung_100ps	privat	Angebot	1199	test	cabrio	2000	automatik	101	fo
371525	2016-03-19 18:57:12	Volkswagen_Multivan_T4_TDI_7DC_UY2	privat	Angebot	9200	test	bus	1996	manuell	102	transp
371526	2016-03-20 19:41:08	VW_Golf_Kombi_1_9l_TDI	privat	Angebot	3400	test	kombi	2002	manuell	100	
371527	2016-03-07 19:39:19	BMW_M135i_vollausgestattet_NP_52.720____Euro	privat	Angebot	28990	control	limousine	2013	manuell	320	m_

371525 rows × 20 columns

In [11]:
```
data=data.drop('seller',1)
```

In [12]:
```
print(data.offerType.value_counts())
```

```
Angebot     371516
Gesuch          12
Name: offerType, dtype: int64
```

In [13]:
```
data[data.offerType !='Gesuch']
```

Out[13]:

	dateCrawled	name	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	k
0	2016-03-24 11:52:17	Golf_3_1.6	Angebot	480	test	NaN	1993	manuell	0	golf	
1	2016-03-24 10:58:45	A5_Sportback_2.7_Tdi	Angebot	18300	test	coupe	2011	manuell	190	NaN	
2	2016-03-14 12:52:21	Jeep_Grand_Cherokee_"Overland"	Angebot	9800	test	suv	2004	automatik	163	grand	
3	2016-03-17 16:54:04	GOLF_4_1_4__3TÜRER	Angebot	1500	test	kleinwagen	2001	manuell	75	golf	
4	2016-03-31 17:25:20	Skoda_Fabia_1.4_TDI_PD_Classic	Angebot	3600	test	kleinwagen	2008	manuell	69	fabia	
...	...	...	...	...	...	...	...	...	...	...	
371523	2016-03-14 17:48:27	Suche_t4___vito_ab_6_sitze	Angebot	2200	test	NaN	2005	NaN	0	NaN	
371524	2016-03-05 19:56:21	Smart_smart_leistungssteigerung_100ps	Angebot	1199	test	cabrio	2000	automatik	101	fortwo	
371525	2016-03-19 18:57:12	Volkswagen_Multivan_T4_TDI_7DC_UY2	Angebot	9200	test	bus	1996	manuell	102	transporter	
371526	2016-03-20 19:41:08	VW_Golf_Kombi_1_9l_TDI	Angebot	3400	test	kombi	2002	manuell	100	golf	
371527	2016-03-07 19:39:19	BMW_M135i_vollausgestattet_NP_52.720____Euro	Angebot	28990	control	limousine	2013	manuell	320	m_reihe	

371516 rows × 19 columns

In [14]:
```
data=data.drop('offerType',1)
```

```
/var/folders/hf/2qd6lpqs4r7_19kpypql9pyw0000gq/T/ipykernel_18905/2361284133.py:1: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
  data=data.drop('offerType',1)
```

In [15]:
```
print(data.shape)
```

```
(371528, 18)
```

In [16]:
```
data=data[(data.powerPS>50)&(data.powerPS<900)]
```

In [17]:
```
print(data.shape)
```

```
(319709, 18)
```

In [18]:
```
data=data[(data.yearOfRegistration>=1950)&(data.yearOfRegistration<2017)]
```

In [19]:
```
print(data.shape)
```

```
(309171, 18)
```

In [20]:
```
data.drop(['name','abtest','dateCrawled','nrOfPictures','lastSeen','postalCode','dateCreated',],axis='columns',inplace=
```

In [21]:
```
new_dataset=data.copy()
```

In [22]:
```
new_dataset=new_dataset.drop_duplicates(['price','vehicleType','yearOfRegistration','gearbox','powerPS','model','kilome
```

In [23]:
```
new_dataset.count()
```

Out[23]:
```
price                285145
vehicleType          274091
yearOfRegistration   285145
gearbox              280031
powerPS              285145
model                273813
kilometer            285145
monthOfRegistration  285145
fuelType             269665
brand                285145
notRepairedDamage    243965
dtype: int64
```

In [24]:
```
new_dataset.gearbox.replace(('manuell','automatik'),('manual','automatic'),inplace=True)
new_dataset.fuelType.replace(('benzin','andere','elektro'),('petrol','others','electric'),inplace=True)
new_dataset.vehicleType.replace(('kleinwagen','cabrio','kombi','andere'),('small car','convertible','combination','othe
new_dataset.notRepairedDamage.replace(('ja','nein'),('Yes','No'),inplace=True)
```

In [25]:
```
new_dataset=new_dataset[(new_dataset.price >= 100)&(new_dataset.price <=150000)]
```

In [26]:
```
new_dataset['notRepairedDamage'].fillna(value='not-declared',inplace=True)
new_dataset['fuelType'].fillna(value='not-declared',inplace=True)
new_dataset['gearbox'].fillna(value='not-declared',inplace=True)
new_dataset['vehicleType'].fillna(value='not-declared',inplace=True)
new_dataset['model'].fillna(value='not-declared',inplace=True)
```

In [27]:
```
new_dataset.isna().sum()
```

Out[27]:
```
price                0
vehicleType          0
yearOfRegistration   0
gearbox              0
powerPS              0
model                0
kilometer            0
monthOfRegistration  0
fuelType             0
brand                0
notRepairedDamage    0
dtype: int64
```

In [28]:
```
new_dataset.head()
```

Out[28]:

	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	notRepairedDamage
1	18300	coupe	2011	manual	190	not-declared	125000	5	diesel	audi	Yes
2	9800	suv	2004	automatic	163	grand	125000	8	diesel	jeep	not-declared
3	1500	small car	2001	manual	75	golf	150000	6	petrol	volkswagen	No
4	3600	small car	2008	manual	69	fabia	90000	7	diesel	skoda	No
5	650	limousine	1995	manual	102	3er	150000	10	petrol	bmw	Yes

In [29]:
```
new_dataset.tail()
```

Out[29]:

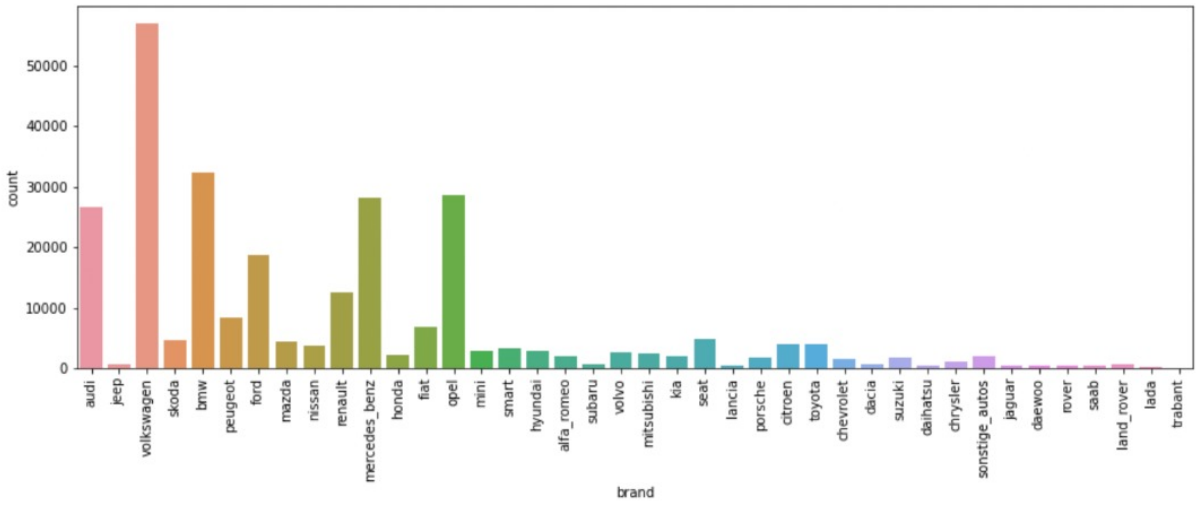
	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	notRepairedDamage
371520	3200	limousine	2004	manual	225	leon	150000	5	petrol	seat	Yes
371524	1199	convertible	2000	automatic	101	fortwo	125000	3	petrol	smart	No
371525	9200	bus	1996	manual	102	transporter	150000	3	diesel	volkswagen	No
371526	3400	combination	2002	manual	100	golf	150000	6	diesel	volkswagen	not-declared
371527	28990	limousine	2013	manual	320	m_reihe	50000	8	petrol	bmw	No

Visualization

In [30]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [31]:
```
plt.figure(figsize = (15, 5))
sns.countplot(x=new_dataset['brand'])
plt.xticks(rotation=90)
```

Out[31]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37, 38, 39]),
 [Text(0, 0, 'audi'),
  Text(1, 0, 'jeep'),
  Text(2, 0, 'volkswagen'),
  Text(3, 0, 'skoda'),
  Text(4, 0, 'bmw'),
  Text(5, 0, 'peugeot'),
  Text(6, 0, 'ford'),
  Text(7, 0, 'mazda'),
  Text(8, 0, 'nissan'),
  Text(9, 0, 'renault'),
  Text(10, 0, 'mercedes_benz'),
  Text(11, 0, 'honda'),
  Text(12, 0, 'fiat'),
  Text(13, 0, 'opel'),
  Text(14, 0, 'mini'),
  Text(15, 0, 'smart'),
  Text(16, 0, 'hyundai'),
  Text(17, 0, 'alfa_romeo'),
  Text(18, 0, 'subaru'),
  Text(19, 0, 'volvo'),
  Text(20, 0, 'mitsubishi'),
  Text(21, 0, 'kia'),
  Text(22, 0, 'seat'),
  Text(23, 0, 'lancia'),
  Text(24, 0, 'porsche'),
  Text(25, 0, 'citroen'),
  Text(26, 0, 'toyota'),
  Text(27, 0, 'chevrolet'),
  Text(28, 0, 'dacia'),
  Text(29, 0, 'suzuki'),
  Text(30, 0, 'daihatsu'),
  Text(31, 0, 'chrysler'),
  Text(32, 0, 'sonstige_autos'),
  Text(33, 0, 'jaguar'),
  Text(34, 0, 'daewoo'),
  Text(35, 0, 'rover'),
  Text(36, 0, 'saab'),
  Text(37, 0, 'land_rover'),
  Text(38, 0, 'lada'),
  Text(39, 0, 'trabant')])
```

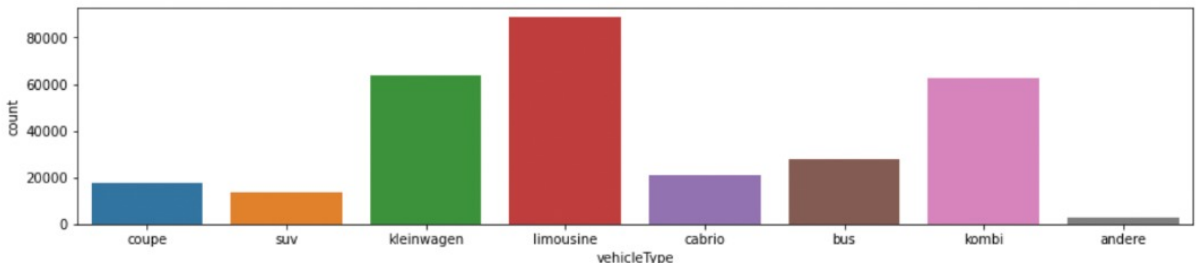


In [32]:
```
print(data.vehicleType.value_counts())
plt.figure(figsize = (15, 3))
sns.countplot(data['vehicleType'])
```

```
limousine     88521
kleinwagen    63898
kombi         62640
bus           27706
cabrio        21249
coupe         17538
suv           13674
andere         2523
Name: vehicleType, dtype: int64
```

```
/Users/rosemarychittilappilly/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, a
nd passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

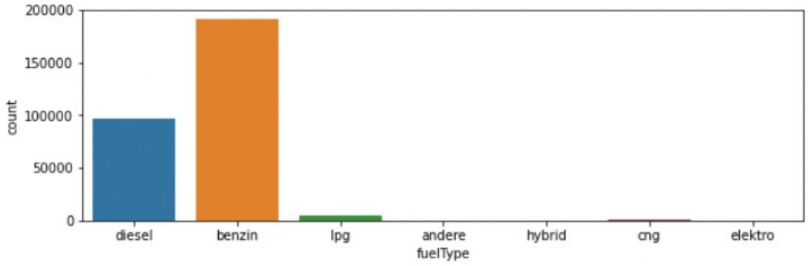
Out[32]: <AxesSubplot:xlabel='vehicleType', ylabel='count'>



In [33]:
```
plt.figure(figsize = (10, 3))
sns.countplot(data['fuelType'])
```

```
/Users/rosemarychittilappilly/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, a
nd passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[33]: <AxesSubplot:xlabel='fuelType', ylabel='count'>



In [34]:
```
sns.pairplot(new_dataset)
```

Out[34]: <seaborn.axisgrid.PairGrid at 0x7f92897adbe0>

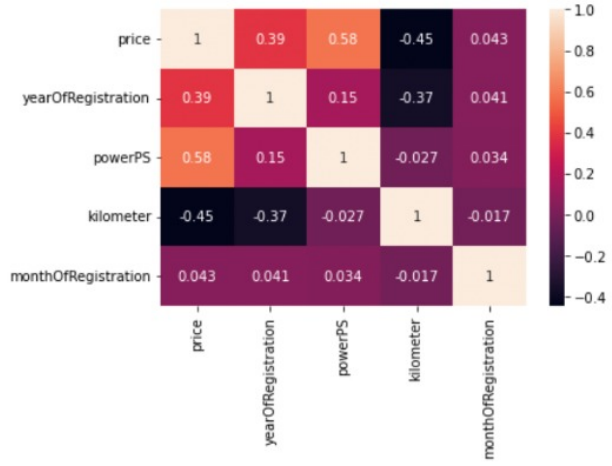
In [35]:
```
new_dataset.corr()
```

Out[35]:

	price	yearOfRegistration	powerPS	kilometer	monthOfRegistration
price	1.000000	0.389244	0.581269	-0.446352	0.042642
yearOfRegistration	0.389244	1.000000	0.152223	-0.370298	0.041436
powerPS	0.581269	0.152223	1.000000	-0.027243	0.034140
kilometer	-0.446352	-0.370298	-0.027243	1.000000	-0.017340
monthOfRegistration	0.042642	0.041436	0.034140	-0.017340	1.000000

In [36]:
```
sns.heatmap(new_dataset.corr(),annot=True)
```

Out[36]: <AxesSubplot:>



Saving the cleaned dataset

In [37]:
```
new_dataset.to_csv("autos_preprocessed.csv")
```

Label encoding the categorical data

In [38]:
```
labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

mapper = {}
for i in labels:
    mapper[i] = LabelEncoder()
    mapper[i].fit(new_dataset[i])
    tr = mapper[i].transform(new_dataset[i])
    np.save(str('classes'+i+'.npy'), mapper[i].classes_)
    new_dataset.loc[:, i+'_labels'] = pd.Series(tr, index=new_dataset.index)

labeled = new_dataset[['price', 'yearOfRegistration','powerPS','kilometer','monthOfRegistration']
                  +[x+"_labels" for x in labels]]

print(labeled.columns)
```

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

In [39]:
```
new_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 278578 entries, 1 to 371527
Data columns (total 17 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   price                     278578 non-null  int64
 1   vehicleType               278578 non-null  object
 2   yearOfRegistration        278578 non-null  int64
 3   gearbox                   278578 non-null  object
 4   powerPS                   278578 non-null  int64
 5   model                     278578 non-null  object
 6   kilometer                 278578 non-null  int64
 7   monthOfRegistration       278578 non-null  int64
 8   fuelType                  278578 non-null  object
 9   brand                     278578 non-null  object
 10  notRepairedDamage         278578 non-null  object
 11  gearbox_labels            278578 non-null  int64
 12  notRepairedDamage_labels  278578 non-null  int64
 13  model_labels              278578 non-null  int64
 14  brand_labels              278578 non-null  int64
 15  fuelType_labels           278578 non-null  int64
 16  vehicleType_labels        278578 non-null  int64
dtypes: int64(11), object(6)
memory usage: 46.3+ MB
```

Splitting Data Into Independent And Dependent Variables

In [40]:
```
Y = labeled.iloc[:,0].values
```

In [41]:
```
X= labeled.iloc[:,1:].values
```

In [42]:
```
Y=Y.reshape(-1,1)
```

In [43]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X = sc.fit_transform(X)
X
```

Out[43]:
```
array([[ 1.15477685,  0.98363346, -0.00840187, ..., -1.42763225,
        -1.43892059, -0.27453985],
       [ 0.09592185,  0.54636171, -0.00840187, ..., -0.45869425,
        -1.43892059,  1.74235787],
       [-0.35787314, -0.87882029,  0.62686435, ...,  1.33011436,
         0.74532082,  1.33897833],
       ...,
       [-1.11419814, -0.44154854,  0.62686435, ...,  1.33011436,
        -1.43892059, -1.48467848],
       [-0.20660814, -0.47393904,  0.62686435, ...,  1.33011436,
        -1.43892059, -1.08129894],
       [ 1.45730685,  3.08901596, -1.9142005 , ..., -1.35309856,
         0.74532082,  0.1288397 ]])
```

In [45]:
```
from sklearn.preprocessing import scale
Y=scale(Y)
Y
```

Out[45]:
```
array([[ 1.42198263],
       [ 0.40444993],
       [-0.58914082],
       ...,
       [ 0.3326241 ],
       [-0.36169233],
       [ 2.70167964]])
```

In [48]:
```
from sklearn.model_selection import cross_val_score, train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state = 3)

print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", Y_train.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", Y_test.shape)
```

```
Number transactions X_train dataset:  (195004, 10)
Number transactions y_train dataset:  (195004, 1)
Number transactions X_test dataset:  (83574, 10)
Number transactions y_test dataset:  (83574, 1)
```

In [49]:
```
X_train
```

Out[49]:
```
array([[-0.20660814, -0.21481504,  0.62686435, ..., -0.75682902,
        -1.43892059, -1.08129894],
       [-0.66040314,  1.03221921,  0.62686435, ..., -0.0114921 ,
         0.74532082, -0.67791939],
       [ 1.15477685, -0.39296279, -1.9142005 , ...,  1.33011436,
        -1.43892059,  0.1288397 ],
       ...,
       [-1.41672814, -0.44154854,  0.62686435, ..., -1.35309856,
         0.74532082,  0.1288397 ],
       [-1.26546314, -1.12174904,  0.62686435, ...,  0.28664267,
         0.74532082,  1.33897833],
       [ 1.30604185,  2.97564921, -1.9142005 , ..., -1.42763225,
        -1.43892059, -1.08129894]])
```

In [50]:
```
Y_train
```

Out[50]:
```
array([[-0.58914082],
       [-0.08635996],
       [ 0.78752107],
       ...,
       [-0.67293763],
       [-0.67293763],
       [ 5.21678105]])
```

In [51]:
```
X_test
```

Out[51]:
```
array([[ 0.70098185,  0.17387096,  0.62686435, ..., -1.42763225,
        -1.43892059,  0.1288397 ],
       [-0.50913814, -1.12174904,  0.62686435, ...,  1.33011436,
         0.74532082,  1.33897833],
       [ 0.09592185, -1.21892054, -0.00840187, ...,  1.33011436,
         0.74532082,  1.33897833],
       ...,
       [-0.35787314, -0.44154854,  0.62686435, ..., -0.0114921 ,
         0.74532082,  0.1288397 ],
       [-0.05534314,  0.27104246,  0.62686435, ..., -1.35309856,
         0.74532082, -0.67791939],
       [-0.66040314, -0.87882029,  0.62686435, ...,  0.28664267,
         0.74532082,  1.33897833]])
```

In [52]:
```
Y_test
```

Out[52]:
```
array([[ 0.42839188],
       [-0.58914082],
       [-0.44560885],
       ...,
       [-0.61308276],
       [-0.17015677],
       [-0.67569095]])
```

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the Dataset

In [2]:
```
data=pd.read_csv("autos_preprocessed.csv",header=0,sep=',',encoding='Latin1',)
```

In [3]:
```
data.head()
```

Out[3]:

	Unnamed: 0	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	notRepairedDamage
0	1	18300	coupe	2011	manual	190	not-declared	125000	5	diesel	audi	Yes
1	2	9800	suv	2004	automatic	163	grand	125000	8	diesel	jeep	not-declared
2	3	1500	small car	2001	manual	75	golf	150000	6	petrol	volkswagen	No
3	4	3600	small car	2008	manual	69	fabia	90000	7	diesel	skoda	No
4	5	650	limousine	1995	manual	102	3er	150000	10	petrol	bmw	Yes

Label encoding the categorical data

In [4]:
```
labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

mapper = {}
for i in labels:
    mapper[i] = LabelEncoder()
    mapper[i].fit(data[i])
    tr = mapper[i].transform(data[i])
    np.save(str('classes'+i+'.npy'), mapper[i].classes_)
    data.loc[:, i+'_labels'] = pd.Series(tr, index=data.index)

labeled = data[['price', 'yearOfRegistration','powerPS','kilometer','monthOfRegistration']
                  +[x+"_labels" for x in labels]]

print(labeled.columns)
```

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 278578 entries, 0 to 278577
Data columns (total 18 columns):
 #   Column                    Non-Null Count   Dtype 
---  ------                    --------------   ----- 
 0   Unnamed: 0                278578 non-null  int64 
 1   price                     278578 non-null  int64 
 2   vehicleType               278578 non-null  object
 3   yearOfRegistration        278578 non-null  int64 
 4   gearbox                   278578 non-null  object
 5   powerPS                   278578 non-null  int64 
 6   model                     278578 non-null  object
 7   kilometer                 278578 non-null  int64 
 8   monthOfRegistration       278578 non-null  int64 
 9   fuelType                  278578 non-null  object
 10  brand                     278578 non-null  object
 11  notRepairedDamage         278578 non-null  object
 12  gearbox_labels            278578 non-null  int64 
 13  notRepairedDamage_labels  278578 non-null  int64 
 14  model_labels              278578 non-null  int64 
 15  brand_labels              278578 non-null  int64 
 16  fuelType_labels           278578 non-null  int64 
 17  vehicleType_labels        278578 non-null  int64 
dtypes: int64(12), object(6)
memory usage: 38.3+ MB
```

Splitting Data Into Independent And Dependent Variables

In [6]:
```
Y = labeled.iloc[:,0].values
```

In [7]:
```
X= labeled.iloc[:,1:].values
```

In [8]:
```
Y=Y.reshape(-1,1)
```

In [9]:
```
from sklearn.model_selection import cross_val_score, train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state = 3)
```

In [10]:
```
X_train
```

Out[10]:
```
array([[  2002,    116, 150000, ...,     10,      1,      1],
       [  1999,    193, 150000, ...,     20,      7,      2],
       [  2011,    105,  50000, ...,     38,      1,      4],
       ...,
       [  1994,    102, 150000, ...,      2,      7,      4],
       [  1995,     60, 150000, ...,     24,      7,      7],
       [  2012,    313,  50000, ...,      1,      1,      1]])
```

In [11]:
```
Y_train
```

Out[11]:
```
array([[ 1500],
       [ 5700],
       [13000],
       ...,
       [  800],
       [  800],
       [50000]])
```

In [12]:
```
X_test
```

Out[12]:
```
array([[  2008,    140, 150000, ...,      1,      1,      4],
       [  2000,     60, 150000, ...,     38,      7,      7],
       [  2004,     54, 125000, ...,     38,      7,      7],
       ...,
       [  2001,    102, 150000, ...,     20,      7,      4],
       [  2003,    146, 150000, ...,      2,      7,      2],
       [  1999,     75, 150000, ...,     24,      7,      7]])
```

In [13]:
```
Y_test
```

Out[13]:
```
array([[10000],
       [ 1500],
       [ 2699],
       ...,
       [ 1300],
       [ 5000],
       [  777]])
```

Choose The Appropriate Model

In [14]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

In [15]:
```
regressor = RandomForestRegressor(n_estimators=1000,max_depth=10,random_state=34)
```

In [16]:
```
regressor.fit(X_train, np.ravel(Y_train,order='C'))
```

Out[16]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)
```

Check The Metrics Of The Model

In [18]:
```
y_pred = regressor.predict(X_test)
```

In [19]:
```
r2=r2_score(Y_test,y_pred)
print("R2_score:",r2)
```

```
R2_score: 0.834527626497731
```

In [20]:
```
Adjusted_R2=1-(1-r2*((X_test.shape[0]-1)/(X_test.shape[0]-X_test.shape[1]-1)))
print("Adjusted R2:",Adjusted_R2)
```

```
Adjusted R2: 0.8346274945764857
```

In [21]:
```
from sklearn.metrics import mean_squared_error
import math
```

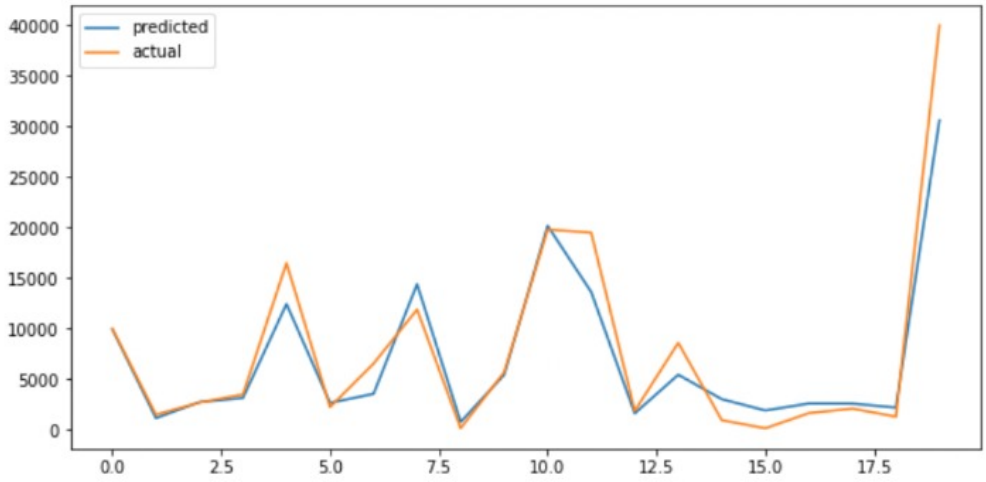
In [22]:
```
MSE=mean_squared_error(Y_test,y_pred)
print("MSE:",MSE)
```

```
MSE: 11837192.971239958
```

In [23]:
```
RMSE=math.sqrt(MSE)
print("RMSE:",RMSE)
```

```
RMSE: 3440.5221945570934
```

In [24]:
```
plt.figure(figsize=(10,5))
plt.plot(y_pred[0:20])
plt.plot(np.array(Y_test[0:20]))
plt.legend(["predicted","actual"])
plt.show()
```



Save the model

In [25]:
```
filename='resale_model.sav'
pickle.dump(regressor,open(filename,'wb'))
```

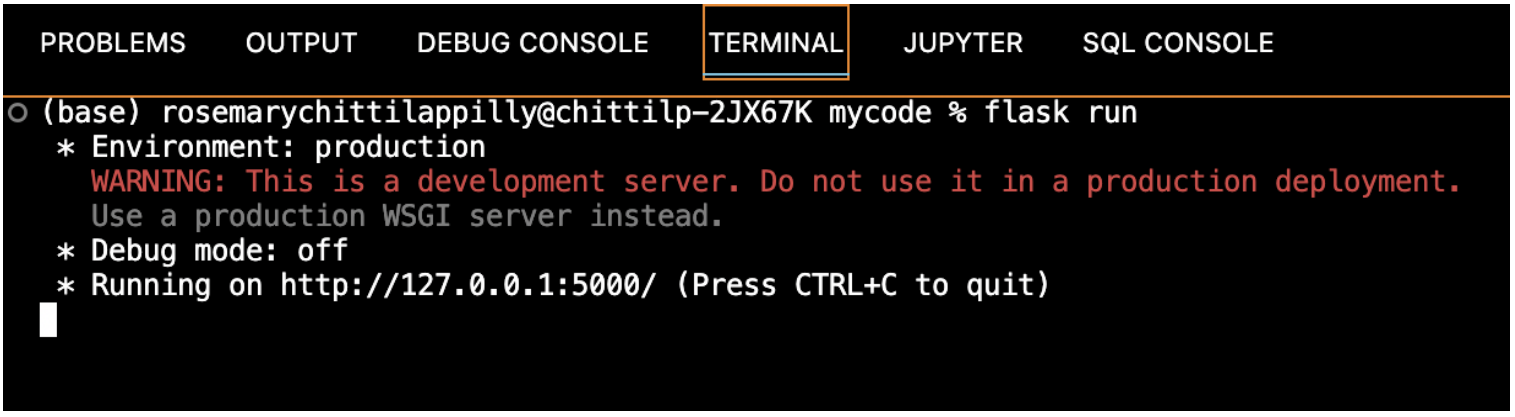
7.2 Feature 2

We have created an Application with Login/Register page, HomePage and Predict Page.

```
import sqlite3
from flask import Flask, redirect, url_for, render_template, request, session
import pandas as pd
import numpy as np
import pickle
from sklearn.preprocessing import LabelEncoder
import requests

API_KEY = "8KsrtRbjivnSJzQkVxyUOf7nQW2lGImM_X9fWB7c4XSF"
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}

def register_user_to_db(username,email,contact,password):
   con = sqlite3.connect('database.db')
   cur = con.cursor()
   cur.execute('INSERT INTO users(username,email,contact,password) values
(?,?,?,?)', (username,email,contact,password))
   con.commit()
   con.close()

def check_user(username, password):
   con = sqlite3.connect('database.db')
   cur = con.cursor()
   cur.execute('Select username,password FROM users WHERE username=? and
password=?', (username, password))

   result = cur.fetchone()
   if result:
       return True
   else:
       return False

app = Flask(__name__)
filename = 'resale_model.sav'
model_rand = pickle.load(open(filename, 'rb'))

app.secret_key = "r@nd0mSk_1"

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

@app.route('/register', methods=["POST", "GET"])
def register():
   if request.method == 'POST':
       username = request.form['username']
       email = request.form['email']
       contact = request.form['contact']
       password = request.form['password']
```

```
        register_user_to_db(username,email,contact,password)
        return render_template('login.html')
      # return redirect(url_for('index'))

    else:
        return render_template('register.html')

@app.route('/login', methods=["POST", "GET"])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        print(check_user(username, password))
        if check_user(username, password):
            session['username'] = username
        #return render_template('resalepredict.html')
        return redirect(url_for('predict'))
    else:
      #  return redirect(url_for('index'))
         return render_template('login.html')

@app.route('/predict', methods=['POST', "GET"])
def predict():
    if 'username' in session:
        return render_template('resalepredict.html', username=session['username'])
    else:
        return render_template('login.html',info='Username or Password is wrong!')

@app.route('/y_predict', methods=['GET', 'POST'])
def y_predict():
    regyear = int(request.args.get('regyear'))
    powerps = float(request.args.get('powerps'))
    kms = float(request.args.get('kms'))
    regmonth = int(request.args.get('regmonth'))
    gearbox = request.args.get('geartype')
    damage = request.args.get('damage')
    model = request.args.get('model')
    brand= request.args.get('brand')
    fuelType = request.args.get('fuelType')
    vehicletype= request.args.get('vehicletype')

    new_row={'yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer':kms,
'monthofRegistration': regmonth,
             'gearbox': gearbox, 'notRepairedDamage': damage,'model':model,
             'brand':brand, 'fuelType': fuelType,'vehicleType': vehicletype}
    print(new_row)
    new_dataset = pd.DataFrame(columns =['vehicleType', 'yearOfRegistration',
'gearbox', 'powerPS', 'model',
                                         'kilometer', 'monthofRegistration', 'fuelType',
'brand', 'notRepairedDamage'])
    new_dataset= new_dataset.append(new_row, ignore_index = True)
    labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType',
'vehicleType']
    mapper = {}
    for i in labels:
        mapper[i] = LabelEncoder()
        # mapper[i].classes = np.load('../IBM carproject/'+str('classes'+i+'.npy'),
allow_pickle=True)
        mapper[i].classes = np.load('../mycode/'+str('classes'+i+'.npy'),
allow_pickle=True)
        tr = mapper[i].fit_transform(new_dataset[i])
```

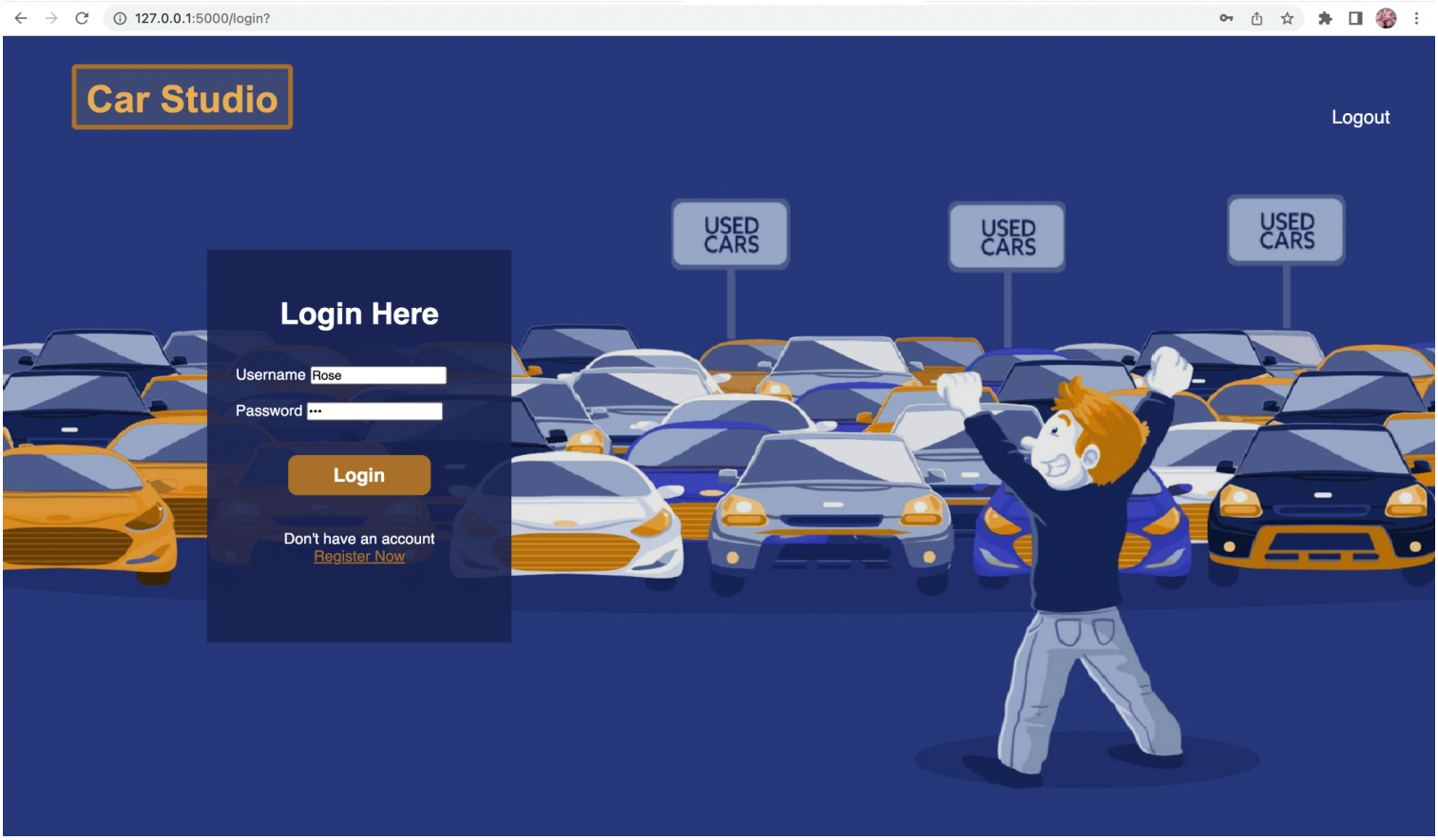
```
      new_dataset.loc[:,i + '_labels'] = pd.Series(tr, index = new_dataset.index)

   labeled = new_dataset[
['yearOfRegistration','powerPS','kilometer','monthofRegistration']+[x+'_labels' for
x in labels]]

   X = labeled.values
   print(X)

   y_prediction = model_rand.predict(X)
   print(y_prediction)
   return render_template('predict.html', predict='The resale value predicted is
{:.2f} $'.format(y_prediction[0]))

   payload_scoring = {"input_data": [{"field": [['yearOfRegistration', 'powerPS',
'kilometer', 'monthOfRegistration','gearbox_labels', 'notRepairedDamage_labels',
'model_labels','brand_labels', 'fuelType_labels', 'vehicleType_labels']], "values":
X}]}
   response_scoring =
requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/8048356c-7b59-4d1
5-abdf-fbff12c7b88a/predictions?version=2022-11-22', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
   print("Scoring response")
   predictions =response_scoring.json()
   predict = predictions['predictions'][0]['values'][0][0]
   print("Final prediction :",predict)

   return render_template('predict.html',predict='{:.2f} $'.format(predict))

@app.route('/logout')
def logout():
   session.clear()
   return redirect(url_for('index'))

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



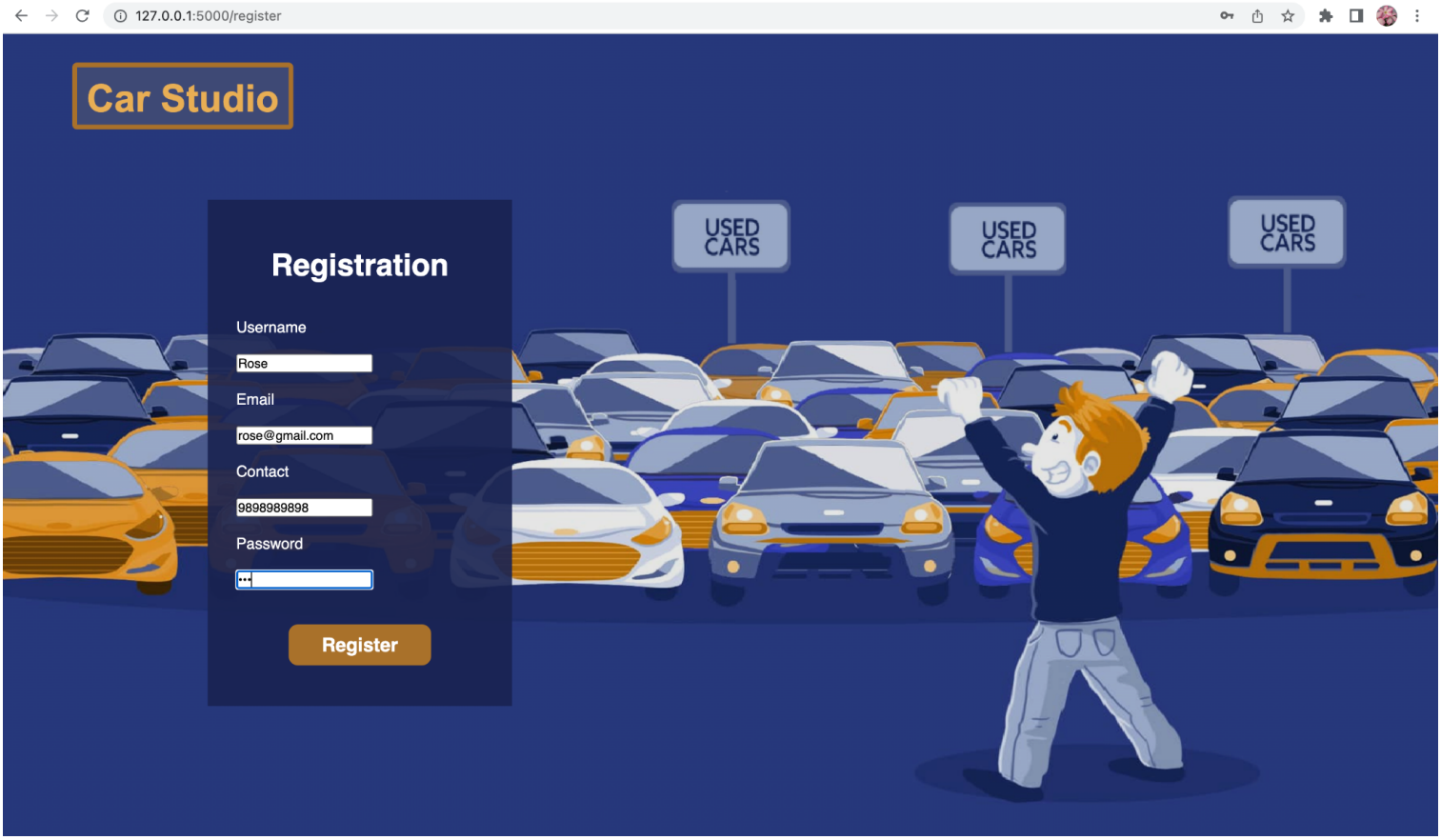
Output Screenshots:

Intro Page

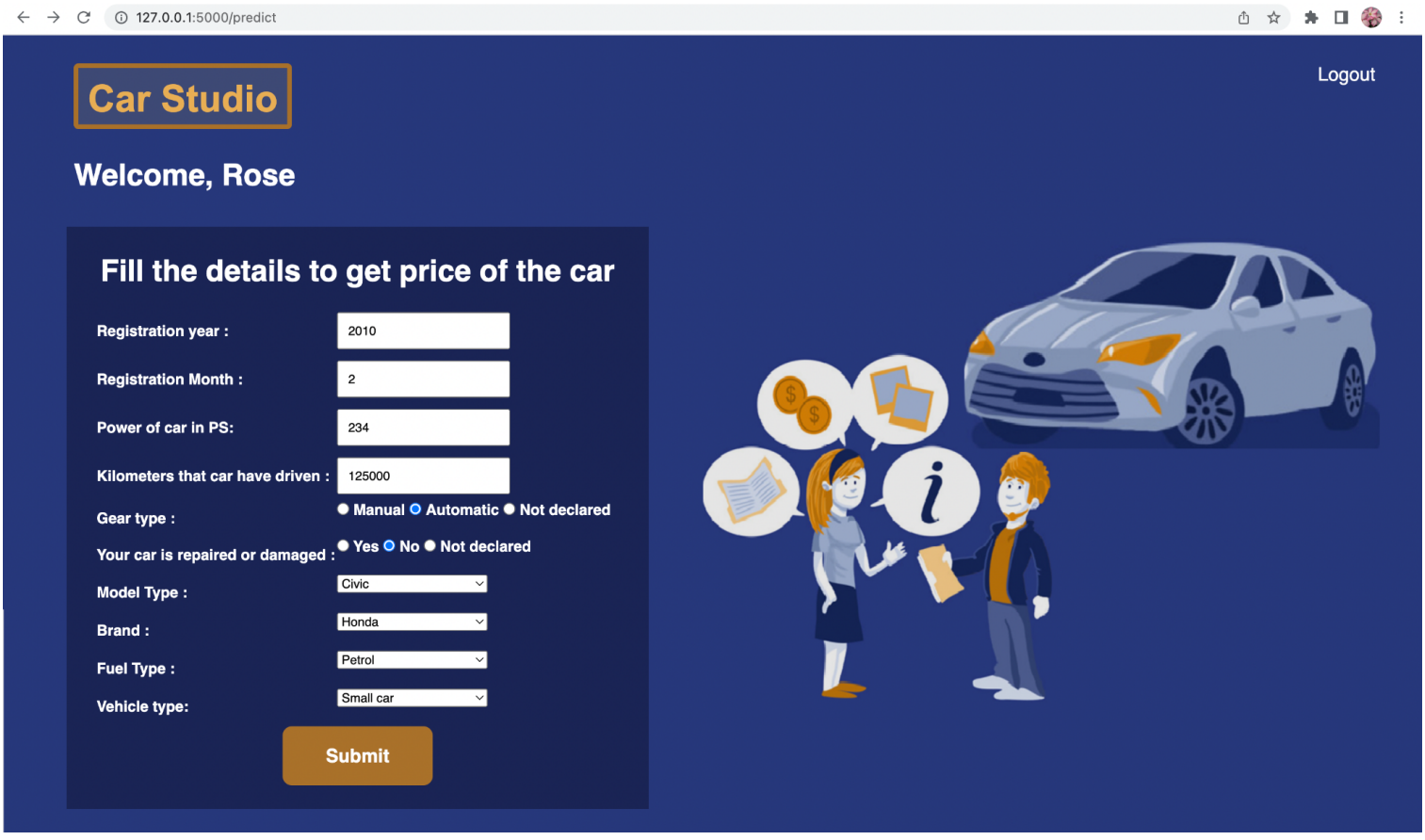
Login Page



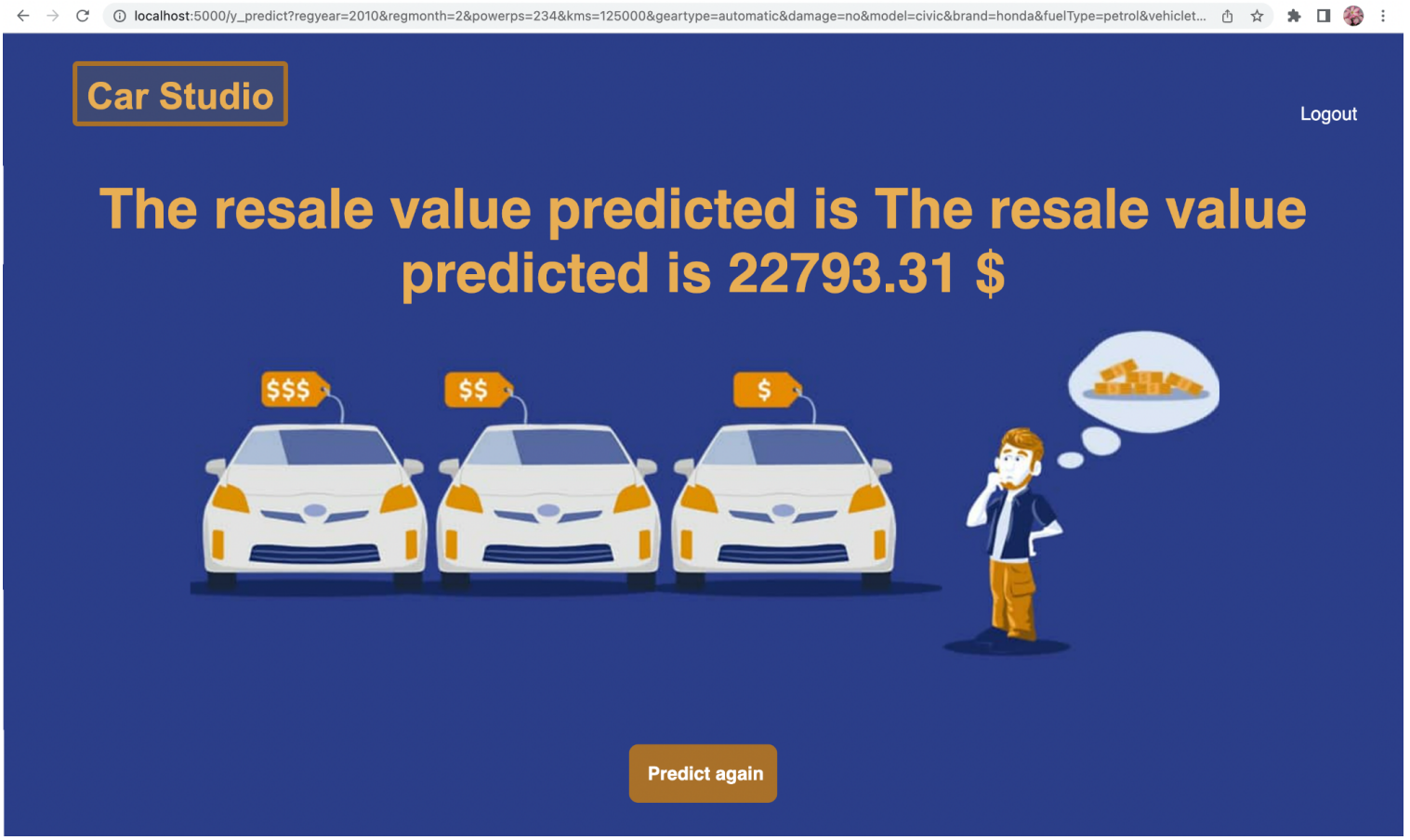
Registration Page



Fill details Page



Predicted Page



8. TESTING

8.1 UTA Initiation

USER ACCEPTANCE TESTING CHECKLIST:

1. **Identify the key stakeholders**

   The stakeholders of this product would be the end user, or customer who is trying to calculate the value of the second hand car.

   Similarly the company hosting the website would also become a stakeholder as the performance of the website dictates their credibility as a company.

2. **Communicate the business intent, objectives and acceptance criteria of the system**

   Business Intent

   - To create a system that proves useful to assess with accuracy the resale value of a car.

   Objectives

   - To create an intuitive and easy to use user interface.
   - To provide accurate predictions of car prices.
   - To create a stable and reliable website.
   - To provide for customer satisfaction**.**

   Acceptance criteria

   - The user must be able to register at the website with no issues.
   - The user should be able to login successfully.
   - The user should be able to submit details of the car.
   - The user must be able to view the predicted value of the car.
   - The predicted value must be accurate.

3. **Agree on User Acceptance Testing team**

   The testing team has been decided and will compose of the following members:

   - Sanjana Suresh
   - Rose Mary Chittilappilly
   - Sharanya N
   - Vijay Adhira C

4. **Agree on documentation to support User Acceptance Testing**

   The documentation will be done meticulously and is planned to be stored in the form of an excel file and word document which will be converted into a pdf file later on.

5. **Agree on decision making structures**

   All decisions will be taken by mutual agreement within the testing team after thorough discussion.

6. **Agree on User Acceptance Testing team resources**

   All testing team members are in possession of a laptop each, adequate system configurations, and an environment with all the variables that are required for testing purposes including testing software. All users are also in access to Cloud resources as well.

7. **Initiate User Acceptance Testing training**

   All team members have been thoroughly trained by attending classes and what have been provided as videos, as well as using other online materials

8. **Form an initial project plan for User Acceptance Testing**

   The following steps are going to be undertaken during acceptance testing:

- Determine whether the business intent and the user expectations have been captured and are measurable.
- Verify that business requirements have been captured.
- Verify that all requirement types have been included.
- Ensure that the scope is clear and relevant.
- Capture and verify the business processes.
- Evaluate the current documentation and its sustainability to serve as a test basis.

8. 2 UTA Execution: Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	BUG resolution status	After bug resolution	Executed By
IntroPage_TC_OO1	Functional	Intro Page	Verify user is able to see the Login button and if it is functional	Install and setup visual studio code. Create a Introduction page with a login button.	1.Enter URL and click go 2.Click on My Account dropdown button 3.Move to login page	http://127.0.0.1:5000/	The Home page with the login button should be displayed. Clicking on the login button take the user to the login page	Working as expected	PASS	The Login button is clear and legible to the user. The login button is functional.	Y				Sanjana Suresh Rose Mary C
LoginPage_TC_OO2	UI	Login Page	Verify the UI elements in Login Page.	Create login page with input boxes so the user may input details.	1.Enter URL 2. Click on login button 3.Verify login with below UI elements: a.Username text box b.password text box c.Login button d. Register now link	http://127.0.0.1:5000/login	Application should show below UI elements: a.username text box b.password text box c.Login button with orange colour d.Register now link	Working as expected	PASS	Login page has the necessary details. A recovery link could be added	Y				Sharanya N Vijay Adhira
LoginPage_TC_OO3	Functional	Login Page	Verify user is able to log into application with Valid credentials	Create login page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/login?) 2.Enter Valid username in username text box 3.Enter valid password in password text box 4.Click on login button	Username: rose password: 123	User should navigate Fill details page	Working as expected	PASS	Successfully created login page.	Y				Vijay Adhira Shanjana Suresh
LoginPage_TC_OO4	Functional	Login page	Verify user is able to log into application with Invalid credentials	Create login page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/login?) 2.Enter Valid username in username text box 3.Enter Invalid password in password text box 4.Click on login button	Username: rose password: 123456789	Application should show 'Incorrect Username or password ' validation message.	Working as expected	PASS	Successfully created login page.	Y				Rose Mary C Sharanya N
LoginPage_TC_OO4	Functional	Login page	Verify user is able to log into application with Invalid credentials	Create login page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/login?) 2.Enter Invalid username in username text box 3.Enter Invalid password in password text box 4.Click on login button	Username: Rose cpcp password: 123456789	Application should show 'Incorrect Username or password ' validation message.	Working as expected	PASS	Successfully created login page.	Y				Sanjana Suresh Rose Mary C
LoginPage_TC_OO5	Functional	Login page	Verify user is able to log into application with Invalid credentials	Create login page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/login?) 2.Enter Invalid username in username text box 3.Enter Valid password in password text box 4.Click on login button	Username: Rose cpcp password: 123	Application should show 'Incorrect email or password ' validation message.	Working as expected	PASS	Successfully created page.	Y				Sharanya N Vijay Adhira
LoginPage_TC_OO6	Functional	Login page	Verify user is able to enter the registeration page on clicking the register now page	Create login page with a link to the registeration page	1.Enter URL(http://127.0.0.1:5000/login?) 2.click on Register now link	http://127.0.0.1:5000/regist	User should be navigated to register page	Working as expected	PASS	Successfully created login page.	Y				Vijay Adhira Shanjana Suresh
RegistrationPage_TC_OO7	UI	Registration Page	Verify the UI elements in Registration page.	Create Registration page with input boxes so the user may input details.	1.Enter URL (http://127.0.0.1:5000/register) 2.Verify login with below UI elements: a.Username text box b. Email ID text vox c. Mobile number text box d.password text box e.Register button	http://127.0.0.1:5000/register	Application should show below UI elements: a.Username text box b. Email ID text vox c. Mobile number text box d.password text box e.Register button	Working as expected	PASS	Registration page has the necessary details.	Y				Rose Mary C Sharanya N
RegistrationPage_TC_OO8	Functional	Registration Page	Verify if the user is able to enter details	Create Registration page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/register) 2.Enter Valid username in username text box 3. Enter Valid email ID in emailID text box 4. Enter Valid mobile number in contacts text box 5.Enter valid password in password text box 6.Click on register button	Username: rose email ID: rose@gmail.com contact: 9898989898 password: 123	User should navigate to back to login page	Working as expected	PASS	Successfully created Registration page.	Y				Sanjana Suresh Rose Mary C
RegistrationPage_TC_OO9	Functional	Registration Page	Verify if the user is able to enter invalid emailID	Create Registration page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/register) 2.Enter Valid username in username text box 3. Enter Inalid email ID in emailID text box 4. Enter Valid mobile number in contacts text box 5.Enter valid password in password text box 6.Click on register button	Username: rose email ID: rose@ contact: 9898989898 password: 123	Enter a valid Email ID	Working as expected	PASS	Successfully created Registration page.	Y				Sharanya N Vijay Adhira
RegistrationPage_TC_OO10	Functional	Registration Page	Verify if the user is able to enter invalid contact	Create Registration page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/register) 2.Enter Valid username in username text box 3. Enter Valid email ID in emailID text box 4. Enter Invalid mobile number in contacts text box 5.Enter valid password in password text box 6.Click on register button	Username: rose email ID: rose@gmail.com contact: 989898 password: 123	Enter a valid mobile number	Working as expected	PASS	Successfully created Registration page.	Y				Vijay Adhira Shanjana Suresh
RegistrationPage_TC_OO11	Functional	Registration Page	Verify if the user is able to enter invalid username	Create Registration page with input boxes so the user may input details. Connect the page with the database storing all users details	1.Enter URL(http://127.0.0.1:5000/register) 2.Enter Invalid username in username text box 3. Enter Valid email ID in emailID text box 4. Enter Valid mobile number in contacts text box 5.Enter valid password in password text box 6.Click on register button	Username: 123## email ID: rose@gmail.com contact: 9898989898 password: 123	Enter a valid username	Working as expected	PASS	Successfully created Registration page.	Y				Rose Mary C Sharanya N
FillDetailsPage_TC_OO12	UI	Fill Details Page	Verify the UI elements in Fill details page.	Create Fill Details page with input boxes, dropdown boxes and radio buttons so the user may input details.	1.Enter URL (http://127.0.0.1:5000/predict) 2.Verify login with below UI elements: a. Registration year text box b. Registration Month text vox c. Power of car in PS text box d. Gear type radio button e. Your car is repaired or damaged radio button f. Model Type dropdown g. Brand dropdown h. Fuel Type dropdown i. Vehicle type dropdown j. Submit button	http://127.0.0.1:5000/predict	Application should show below UI elements: a. Registration year text box b. Registration Month text vox c. Power of car in PS text box d. Gear type radio button e. Your car is repaired or damaged radio button f. Model Type dropdown g. Brand dropdown h. Fuel Type dropdown i. Vehicle type dropdown j. Submit button	Working as expected	PASS	Fill Details page collects the necessary details.	Y				Sanjana Suresh Rose Mary C
FillDetailsPage_TC_OO13	Functional	Fill Details Page	Verify if the user is able to enter Valid details	Create Fill Details page with input boxes, dropdown boxes and radio buttons so the user may input details.	1.Enter URL (http://127.0.0.1:5000/predict) 2. Enter Valid Registration year in text box 3. Enter Valid Registration Month in text vox 4. Enter Valid Power of car in PS in text box 5. Enter Valid Kilometers driven in text box	Registration year : 2010 (numeric) Registration Month :2 (numeric) Power of car in PS :234 (numeric) Kilometers driven: 125000 (numeric)	User is navigated to prediction page	Working as expected	PASS	Successfully created fill details page.	Y				Sharanya N Vijay Adhira
FillDetailsPage_TC_OO14	Functional	Fill Details Page	Verify if the user is able to enter Valid details	Create Fill Details page with input boxes, dropdown boxes and radio buttons so the user may input details.	1.Enter URL (http://127.0.0.1:5000/predict) 2. Enter Valid Registration year in text box 3. Enter Invalid Registration Month in text vox 4. Enter Valid Power of car in PS in text box 5. Enter Valid Kilometers driven in text box	Registration year : 2010 (numeric) Registration Month :february Power of car in PS :234 (numeric) Kilometers driven: 125000 (numeric)	Indicates to user to enter the month as a numeric	Internal server error	FAIL (now resolved)	Add a condition that informs the user of what is wrong	Y	BUG-1	Resolved	Test Passed	Vijay Adhira Shanjana Suresh
FillDetailsPage_TC_OO15	Functional	Fill Details Page	Verify if the user is able to submit details on clicking submit	Create Fill Details page for the user to input details and a submit button to submit the details .	1.Enter URL (http://127.0.0.1:5000/predict) 2. Enter Valid Registration year in text box 3. Enter Invalid Registration Month in text vox 4. Enter Valid Power of car in PS in text box 5. Enter Valid Kilometers driven in text box 6. click submit button	Registration year : 2010 (numeric) Registration Month :2 (numeric) Power of car in PS :234 (numeric) Kilometers driven: 125000 (numeric)	User is navigated to prediction page	Working as expected	PASS	Successfully created fill details page.	Y				Rose Mary C Sharanya N
PredictPage_TC_OO16	UI	Predict Page	Verify the UI elements in Predict page.	Create predict page that displays the price of the used car based on the details provided by the user in fill details page	1. Enter URL (http://localhost:5000/y_predict?) 2. Verify with below UI elements: a. The predicted value displayed b. logout link c. Predict again button	http://localhost:5000/y_predict?	The value predicted is displayed	Working as expected	PASS	Successfully created prediction page.	Y				Sanjana Suresh Rose Mary C
PredictPage_TC_OO17	Functional	Predict Page	Verify if the predicted value is displayed	Create predict page that displays the price of the used car based on the details provided by the user in fill details page	1. Enter details correctly in fill details page 2. View the result produced	http://localhost:5000/y_predict?	The value predicted is displayed	Working as expected	PASS	Successfully created prediction page.	Y				Sharanya N Vijay Adhira
PredictPage_TC_OO18	Functional	Predict Page	Verify if on clicking the predict now button the user is navigated to fill details page	Create predict page that displays the price of the used car based on the details provided by the user in fill details page	1. Enter details correctly in fill details page 2. View the result produced 3. Click on predict now button	http://localhost:5000/y_predict?	The user is navigated to fill details page	Working as expected	FAIL (now resolved)	The user was navigated to login page instead of fill details page	Y	BUG-2	Resolved	Test Passed	Vijay Adhira Shanjana Suresh
LogoutPage_TC_0019	Functional	Logout Page	Verify if the user is navigated to the intro page on clicking logout	Create a Introduction page	Click logout on the top right corner	http://127.0.0.1:5000/	The user is navicated to Intro page	Working as expected	PASS	Successfully created prediction page.	Y				Rose Mary C Sharanya N

	Test Scenarios
1	Verify user is able to see the Login button and if it is functional.
2	Verify the UI elements in Login Page.
3	Verify user is able to log into application with Valid credentials.
4	Verify user is able to log into application with Invalid credentials.
5	Verify user is able to log into application with Invalid credentials.
6	Verify user is able to log into application with Invalid credentials.
7	Verify user is able to enter the registration page on clicking the register now page.
8	Verify the UI elements in Registration page.
9	Verify if the user is able to enter details.
10	Verify if the user is able to enter invalid email ID.
11	Verify if the user is able to enter invalid contact.
12	Verify if the user is able to enter invalid username.
13	Verify the UI elements in Fill details page.
14	Verify if the user is able to enter Valid details.
15	Verify if the user is able to enter Valid details.
16	Verify if the user is able to submit details on clicking submit.
17	Verify the UI elements in Predict page.
18	Verify if the predicted value is displayed.
19	Verify if on clicking the predict now button the user is navigated to fill details page.
20	Verify if the user is navigated to the intro page on clicking logout

8.3 User Acceptance Testing Report Submission

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Car Resale Value Prediction 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	1	2	2	0	5
Duplicate	1	0	1	0	2
External	2	0	1	0	3
Fixed	0	1	1	0	2
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	4	3	5	0	12

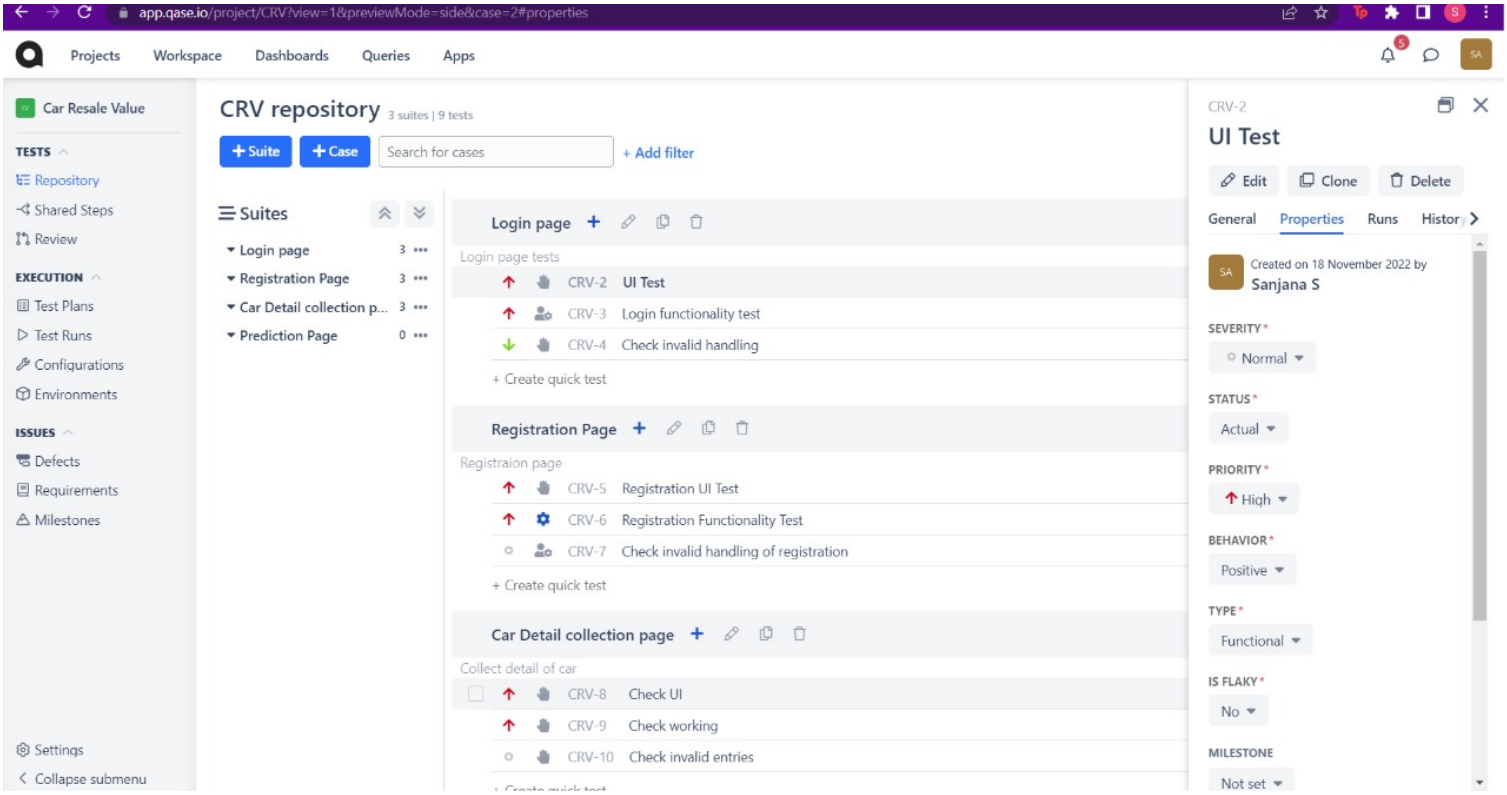
3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Home page	1	0	0	1
Login page	6	0	0	6

Registration page	5	0	0	5
Fill Details page	4	0	1	3
Prediction page	3	0	0	3
Logout	1	0	0	1
Exception Report	4	0	2	2
Final Report Output	4	0	0	4

8.4 Utilization of Test management tools



8.5 User Acceptance Testing – Design Document

It is important to ensure the test design for UAT follows the below steps in order to ensure that the UAT provides the desired outcome.

1. **Establish the entry criteria for User Acceptance Testing**

   The tests should cover the following and pass these tests successfully:

   - Login functionality
   - Login page user interface
   - Login page invalid input
   - Registration page functionality
   - Registration page user interface
   - Registration page invalid input handling
   - Detail fetching functionality
   - Detail fetching UI
   - Detail Fetching invalid input handling

2. **Review test scripts where available.**

   The tests scripts have been thoroughly reviewed

3. **Define the User Acceptance Testing strategy.**

   We will attempt one test case at a time in order of priority

4. **Review existing test conditions where available and write new ones**

   Whenever the test case fails, a patch is introduced to fix it and it is made to pass.

5. **Ensure that the tests cover all the requirements.**

   Care has been taken to ensure that the requirements are met adequately.

9. RESULTS

9.1 Performance Metrics

Check The Metrics Of The Model

In [17]:
```
y_pred = regressor.predict(X_test)
```

In [18]:
```
r2=r2_score(Y_test,y_pred)
print("R2_score:",r2)
```

```
R2_score: 0.834527626497731
```

In [19]:
```
Adjusted_R2=1-(1-r2*((X_test.shape[0]-1)/(X_test.shape[0]-X_test.shape[1]-1)))
print("Adjusted R2:",Adjusted_R2)
```

```
Adjusted R2: 0.8346274945764857
```

In [20]:
```
from sklearn.metrics import mean_squared_error
import math
```

In [21]:
```
MSE=mean_squared_error(Y_test,y_pred)
print("MSE:",MSE)
```

```
MSE: 11837192.971239958
```

In [22]:
```
RMSE=math.sqrt(MSE)
print("RMSE:",RMSE)
```

```
RMSE: 3440.5221945570934
```

9.2 Hyperparameter Tuning

In [14]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

In [15]:
```
regressor = RandomForestRegressor(n_estimators=1000,max_depth=10,random_state=34)
```

In [16]:
```
regressor.fit(X_train, np.ravel(Y_train,order='C'))
```

Out[16]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)
```

In [17]:
```
y_pred = regressor.predict(X_test)
```

In [18]:
```
r2=r2_score(Y_test,y_pred)
print("R2_score:",r2)
```

```
R2_score: 0.834527626497731
```

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

- Application is easy to learn
- Good at learning complex and non-linear relationships.
- Highly explainable and easy to interpret.
- Robust to outliers.
- No feature scaling is required.
- Application is free
- No need to commission a car resale agent to estimate resale value

10.2 Disadvantages

- Consume more time. User needs to fill all the required details
- Requires high computational power.
- Does not work for cars of different distribution

11. CONCLUSION

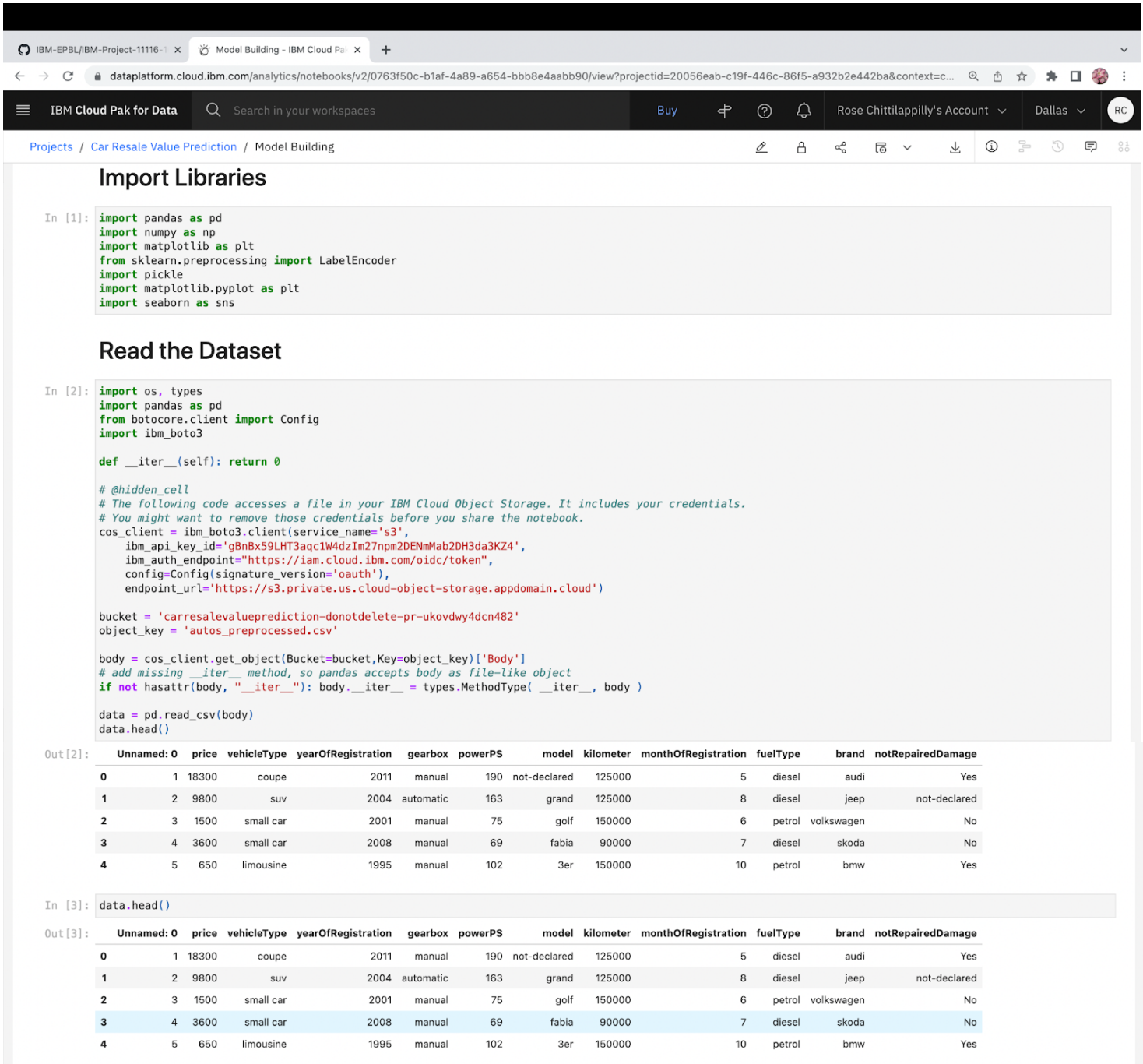
As we can see in the cross-validation scores table, linear regressor and random forest are the ones that perform better. Saying that one model is objectively better than another is difficult. Random forests are almost always preferable to linear regressors because they don't need much preprocessing and sometimes they produce good results even in presence of outliers. In our case, the differences in performance between linear regressor and random forest are not enough to justify the exaggeratedly high number of attributes introduced by one hot encoding.

12. FUTURE SCOPE

In the 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 on car prices which can help to improve the accuracy of the machine learning model. We can build an android app as a user interface for interacting with users. For better performance, we intend to carefully design deep learning network structures, employ adaptive learning rates, and train on data clusters rather than the entire dataset.

13. APPENDIX

13.1 Source Code



IBM-EPBL/IBM-Project-11116-1 | Model Building - IBM Cloud Pak

dataplatform.cloud.ibm.com/analytics/notebooks/v2/0763f50c-b1af-4a89-a654-bbb8e4aabb90/view?projectid=20056eab-c19f-446c-86f5-a932b2e442ba&context=c...

IBM Cloud Pak for Data | Search in your workspaces | Buy | Rose Chittilappilly's Account | Dallas | RC

Projects / Car Resale Value Prediction / Model Building

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the Dataset

In [2]:
```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='gBnBx59LHT3aqc1W4dzIm27npm2DENmMab2DH3da3KZ4',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'carresalevalueprediction-donotdelete-pr-ukovdwy4dcn482'
object_key = 'autos_preprocessed.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

data = pd.read_csv(body)
data.head()
```

Out[2]:

	Unnamed: 0	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	notRepairedDamage
0	1	18300	coupe	2011	manual	190	not-declared	125000	5	diesel	audi	Yes
1	2	9800	suv	2004	automatic	163	grand	125000	8	diesel	jeep	not-declared
2	3	1500	small car	2001	manual	75	golf	150000	6	petrol	volkswagen	No
3	4	3600	small car	2008	manual	69	fabia	90000	7	diesel	skoda	No
4	5	650	limousine	1995	manual	102	3er	150000	10	petrol	bmw	Yes

In [3]:
```
data.head()
```

Out[3]:

	Unnamed: 0	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	notRepairedDamage
0	1	18300	coupe	2011	manual	190	not-declared	125000	5	diesel	audi	Yes
1	2	9800	suv	2004	automatic	163	grand	125000	8	diesel	jeep	not-declared
2	3	1500	small car	2001	manual	75	golf	150000	6	petrol	volkswagen	No
3	4	3600	small car	2008	manual	69	fabia	90000	7	diesel	skoda	No
4	5	650	limousine	1995	manual	102	3er	150000	10	petrol	bmw	Yes

Label encoding the categorical data

In [4]:
```
labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

mapper = {}
for i in labels:
    mapper[i] = LabelEncoder()
    mapper[i].fit(data[i])
    tr = mapper[i].transform(data[i])
    np.save(str('classes'+i+'.npy'), mapper[i].classes_)
    data.loc[:, i+'_labels'] = pd.Series(tr, index=data.index)

labeled = data[['price', 'yearOfRegistration','powerPS','kilometer','monthOfRegistration']
                +[x+"_labels" for x in labels]]

print(labeled.columns)
```

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 278578 entries, 0 to 278577
Data columns (total 18 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   Unnamed: 0                278578 non-null  int64
 1   price                     278578 non-null  int64
 2   vehicleType               278578 non-null  object
 3   yearOfRegistration        278578 non-null  int64
 4   gearbox                   278578 non-null  object
 5   powerPS                   278578 non-null  int64
 6   model                     278578 non-null  object
 7   kilometer                 278578 non-null  int64
 8   monthOfRegistration       278578 non-null  int64
 9   fuelType                  278578 non-null  object
 10  brand                     278578 non-null  object
 11  notRepairedDamage         278578 non-null  object
 12  gearbox_labels            278578 non-null  int64
 13  notRepairedDamage_labels  278578 non-null  int64
 14  model_labels              278578 non-null  int64
 15  brand_labels              278578 non-null  int64
 16  fuelType_labels           278578 non-null  int64
 17  vehicleType_labels        278578 non-null  int64
dtypes: int64(12), object(6)
memory usage: 38.3+ MB
```

Splitting Data Into Independent And Dependent Variables

In [6]:
```
Y = labeled.iloc[:,0].values
```

In [7]:
```
X= labeled.iloc[:,1:].values
```

In [8]:
```
Y=Y.reshape(-1,1)
```

In [9]:
```
from sklearn.model_selection import cross_val_score, train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state = 3)
```

In [10]:
```
X_train
```

Out[10]:
```
array([[  2002,    116, 150000, ...,     10,      1,      1],
       [  1999,    193, 150000, ...,     20,      7,      2],
       [  2011,    105,  50000, ...,     38,      1,      4],
       ...,
       [  1994,    102, 150000, ...,      2,      7,      4],
       [  1995,     60, 150000, ...,     24,      7,      7],
       [  2012,    313,  50000, ...,      1,      1,      1]])
```

In [11]:
```
Y_train
```

Out[11]:
```
array([[ 1500],
       [ 5700],
       [13000],
       ...,
       [  800],
       [  800],
       [50000]])
```

In [12]:
```
X_test
```

Out[12]:
```
array([[  2008,    140, 150000, ...,      1,      1,      4],
       [  2000,     60, 150000, ...,     38,      7,      7],
       [  2004,     54, 125000, ...,     38,      7,      7],
       ...,
       [  2001,    102, 150000, ...,     20,      7,      4],
       [  2003,    146, 150000, ...,      2,      7,      2],
       [  1999,     75, 150000, ...,     24,      7,      7]])
```

In [13]:
```
Y_test
```

Out[13]:
```
array([[10000],
       [ 1500],
       [ 2699],
       ...,
       [ 1300],
       [ 5000],
       [  777]])
```

Choose The Appropriate Model

In [14]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

In [15]:
```
regressor = RandomForestRegressor(n_estimators=1000,max_depth=10,random_state=34)
```

In [16]:
```
regressor.fit(X_train, np.ravel(Y_train,order='C'))
```

Out[16]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)
```

Check The Metrics Of The Model

In [17]:
```
y_pred = regressor.predict(X_test)
```

In [18]:
```
r2=r2_score(Y_test,y_pred)
print("R2_score:",r2)
```

```
R2_score: 0.834527626497731
```

In [19]:
```
Adjusted_R2=1-(1-r2*((X_test.shape[0]-1)/(X_test.shape[0]-X_test.shape[1]-1)))
print("Adjusted R2:",Adjusted_R2)
```

```
Adjusted R2: 0.8346274945764857
```

In [20]:
```
from sklearn.metrics import mean_squared_error
import math
```

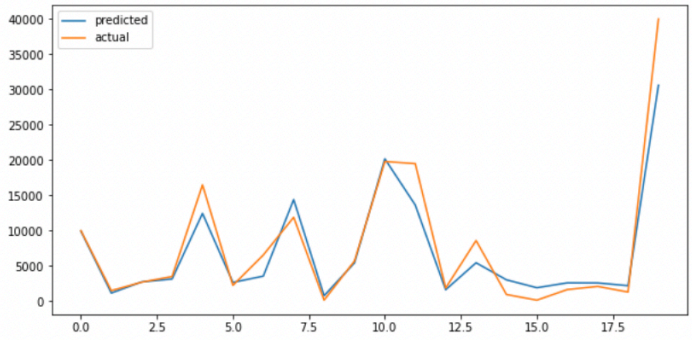
In [21]:
```
MSE=mean_squared_error(Y_test,y_pred)
print("MSE:",MSE)
```

```
MSE: 11837192.971239958
```

In [22]:
```
RMSE=math.sqrt(MSE)
print("RMSE:",RMSE)
```

```
RMSE: 3440.5221945570934
```

In [23]:
```
plt.figure(figsize=(10,5))
plt.plot(y_pred[0:20])
plt.plot(np.array(Y_test[0:20]))
plt.legend(["predicted","actual"])
plt.show()
```



Save the model

In [24]:
```
filename='resale_model.sav'
pickle.dump(regressor,open(filename,'wb'))
```

Deployment

In [25]:
```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (4.8.2)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.3.4)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (0.10.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (1.20.3)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (1.15.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (2.0.4)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from importlib-metadata->ibm_watson_machine_learning) (3.6.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging->ibm_watson_machine_learning) (3.0.4)
```

In [26]:
```
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
```

Authentication and set space

In [27]:
```
wml_credentials = {
    "apikey" : "8KsrtRbjivnSJzQkVxyUOf7nQW2lGImM_X9fWB7c4XSF",
    "url" : "https://us-south.ml.cloud.ibm.com" #For Dallas region
}
```

In [28]:
```
client =APIClient(wml_credentials)
```

In [29]:
```
client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
------------------------------------  ----------  ------------------------
ID                                    NAME        CREATED
c0dabc89-2195-4957-8a89-d8152ff10049  Prediction  2022-11-22T04:57:21.843Z
------------------------------------  ----------  ------------------------
```

In [30]:
```
space_uid= "c0dabc89-2195-4957-8a89-d8152ff10049"
```

In [31]:
```
client.set.default_space(space_uid)
```

Out[31]:
```
'SUCCESS'
```

In [32]:
```
client.software_specifications.list()
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
autoai-kb_rt22.2-py3.10         125b6d9a-5b1f-5e8d-972a-b251688ccf40  base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6            1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                 20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda         26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                        295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8             2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6           2b73a275-7cbf-420b-a912-eae7f436e0bc  base
kernel-spark3.3-py3.9           2b7961e2-e3b1-5a8c-a491-482c8368839a  base
pytorch_1.2-py3.6               2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                 2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt      32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37            36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                 390d21f8-e58b-4fac-9c55-d7ceda621326  base
autoai-ts_rt22.2-py3.10         396b2e83-0953-5b86-9a55-7ce1628a406f  base
xgboost_0.82-py3.6              39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt      40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
pytorch-onnx_rt22.2-py3.10      40e73f55-783a-5535-b3fa-0c8b94291431  base
default_r36py38                 41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9          4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                  42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
pmml-3.0_4.3                    493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6           49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6              4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6          50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
autoai-ts_3.9-py3.8             52c57136-80fa-572e-8728-a5e7cbb42cde  base
spark-mllib_2.4-scala_2.11      55a70f99-7320-4be5-9fb9-9edb5a443af5  base
spark-mllib_3.0                 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9  base
autoai-obm_2.0                  5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1               5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
cuda-py3.8                      5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e  base
runtime-22.2-py3.10-xc          5e8cddff-db4a-5a6a-b8aa-2d4af9864dab  base
autoai-kb_3.1-py3.7             632d4b22-10aa-5180-88f0-f52dfb6444d7  base
------------------------------  ------------------------------------  ----
Note: Only first 50 records were displayed. To display more use 'limit' parameter.
```

In [33]:
```
software_spec_uid = client.software_specifications.get_uid_by_name('runtime-22.1-py3.9')
software_spec_uid
```

Out[33]:
```
'12b83a17-24d8-5082-900f-0ab31fbfd3cb'
```

Save and Deploy the model

In [34]:
```
import sklearn
sklearn.__version__
```

Out[34]:
```
'1.0.2'
```

In [35]:
```
MODEL_NAME='prediction_model'
DEPLOYMENT_NAME='prediction'
DEMO_MODEL= regressor
```

In [36]:
```
model_props= {
    client.repository.ModelMetaNames.NAME: "MODEL_NAME",
    client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

In [37]:
```
#save model
model_details=client.repository.store_model(
    model=DEMO_MODEL,
    meta_props=model_props,
    training_data= X_train,
    training_target= Y_train
)
```

In [38]:
```
model_details
```

Out[38]:

```
{'entity': {'hybrid_pipeline_software_specs': [],
  'label_column': 'l0',
  'schemas': {'input': [{'fields': [{'name': 'f0', 'type': 'int'},
      {'name': 'f1', 'type': 'int'},
      {'name': 'f2', 'type': 'int'},
      {'name': 'f3', 'type': 'int'},
      {'name': 'f4', 'type': 'int'},
      {'name': 'f5', 'type': 'int'},
      {'name': 'f6', 'type': 'int'},
      {'name': 'f7', 'type': 'int'},
      {'name': 'f8', 'type': 'int'},
      {'name': 'f9', 'type': 'int'}],
     'id': '1',
     'type': 'struct'}],
   'output': []},
  'software_spec': {'id': '12b83a17-24d8-5082-900f-0ab31fbfd3cb',
   'name': 'runtime-22.1-py3.9'},
  'type': 'scikit-learn_1.0'},
 'metadata': {'created_at': '2022-11-22T06:34:43.161Z',
  'id': '6505dbb1-e921-485f-8e0b-2cce1cd37659',
  'modified_at': '2022-11-22T06:35:26.008Z',
  'name': 'MODEL_NAME',
  'owner': 'IBMid-6610045SLT',
  'resource_key': '8b30ea94-b43b-4aaa-9a62-eed747c87b95',
  'space_id': 'c0dabc89-2195-4957-8a89-d8152ff10049'},
 'system': {'warnings': []}}
```


In [39]:
```
model_id= client.repository.get_model_id(model_details)
```

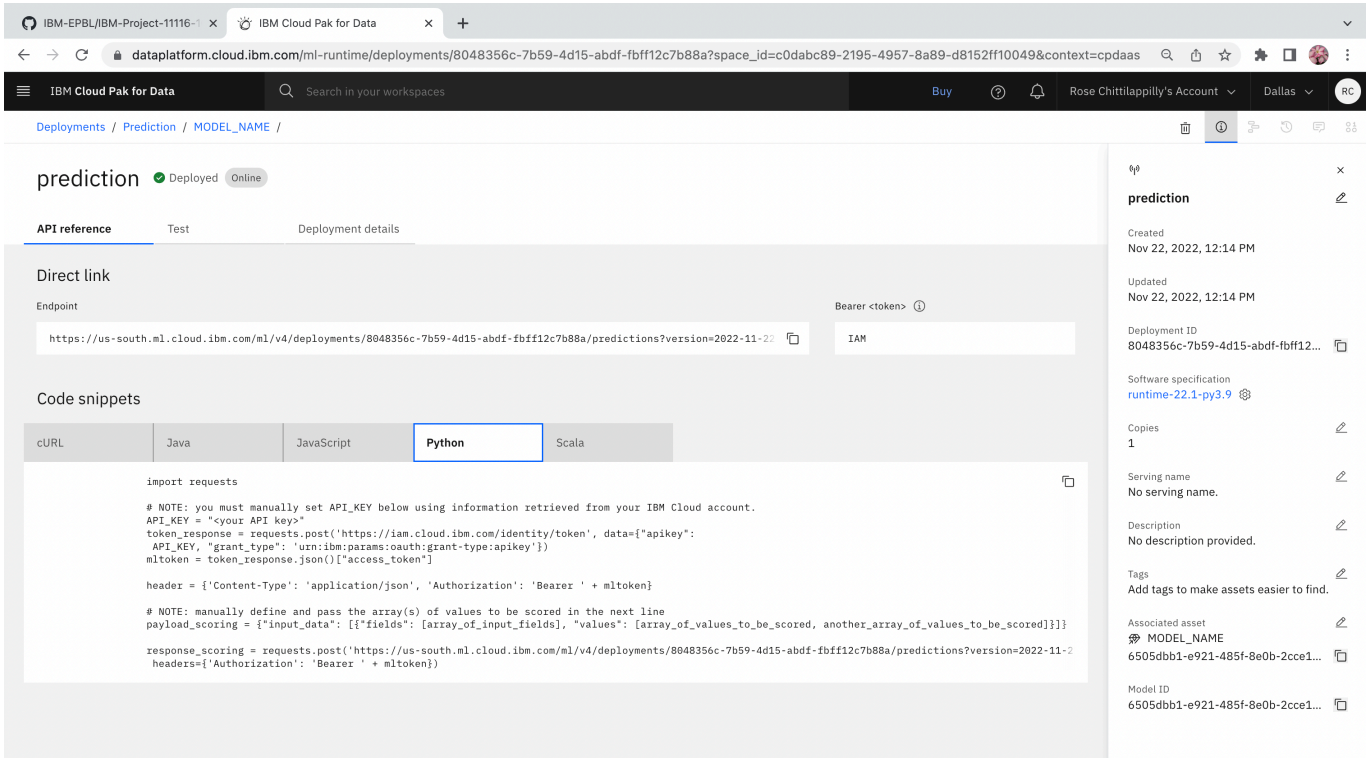
In [40]:
```
model_id
```

Out[40]:
```
'6505dbb1-e921-485f-8e0b-2cce1cd37659'
```

In [42]:
```
#deploying props
#set mets
deployment_props={
    client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

In [45]:
```
#deploy
deployment= client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)
```

```
#######################################################################################

Synchronous deployment creation for uid: '6505dbb1-e921-485f-8e0b-2cce1cd37659' started

#######################################################################################


initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready


------------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='8048356c-7b59-4d15-abdf-fbff12c7b88a'
------------------------------------------------------------------------------------------------
```



app.py

```
import sqlite3
from flask import Flask, redirect, url_for, render_template, request, session
import pandas as pd
import numpy as np
import pickle
from sklearn.preprocessing import LabelEncoder
import requests

API_KEY = "8KsrtRbjivnSJzQkVxyUOf7nQW2lGImM_X9fWB7c4XSF"
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}


def register_user_to_db(username,email,contact,password):
    con = sqlite3.connect('database.db')
    cur = con.cursor()
    cur.execute('INSERT INTO users(username,email,contact,password) values (?,?,?,?)',
(username,email,contact,password))
    con.commit()
    con.close()

def check_user(username, password):
    con = sqlite3.connect('database.db')
    cur = con.cursor()
    cur.execute('Select username,password FROM users WHERE username=? and password=?',
(username, password))

    result = cur.fetchone()
    if result:
        return True
    else:
        return False
```

```
app = Flask(__name__)
filename = 'resale_model.sav'
model_rand = pickle.load(open(filename, 'rb'))

app.secret_key = "r@nd0mSk_1"

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

@app.route('/register', methods=["POST", "GET"])
def register():
   if request.method == 'POST':
       username = request.form['username']
       email = request.form['email']
       contact = request.form['contact']
       password = request.form['password']

       register_user_to_db(username,email,contact,password)
       return render_template('login.html')
     # return redirect(url_for('index'))


   else:
       return render_template('register.html')

@app.route('/login', methods=["POST", "GET"])
def login():
   if request.method == 'POST':
       username = request.form['username']
       password = request.form['password']
       print(check_user(username, password))
       if check_user(username, password):
           session['username'] = username
       #return render_template('resalepredict.html')
       return redirect(url_for('predict'))
   else:
     #  return redirect(url_for('index'))
        return render_template('login.html')



@app.route('/predict', methods=['POST', "GET"])
def predict():
   if 'username' in session:
       return render_template('resalepredict.html', username=session['username'])
   else:
       return render_template('login.html',info='Username or Password is wrong!')

@app.route('/y_predict', methods=['GET', 'POST'])
def y_predict():
   regyear = int(request.args.get('regyear'))
   powerps = float(request.args.get('powerps'))
   kms = float(request.args.get('kms'))
   regmonth = int(request.args.get('regmonth'))
   gearbox = request.args.get('geartype')
   damage = request.args.get('damage')
   model = request.args.get('model')
   brand= request.args.get('brand')
   fuelType = request.args.get('fuelType')
   vehicletype= request.args.get('vehicletype')

   new_row={'yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer':kms,
'monthofRegistration': regmonth,
           'gearbox': gearbox, 'notRepairedDamage': damage,'model':model,
           'brand':brand, 'fuelType': fuelType,'vehicleType': vehicletype}
   print(new_row)
   new_dataset = pd.DataFrame(columns =['vehicleType', 'yearOfRegistration', 'gearbox',
'powerPS', 'model',
                                    'kilometer', 'monthofRegistration', 'fuelType', 'brand',
'notRepairedDamage'])
   new_dataset= new_dataset.append(new_row, ignore_index = True)
   labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']
```

```
    mapper = {}
    for i in labels:
        mapper[i] = LabelEncoder()
        # mapper[i].classes = np.load('../IBM carproject/'+str('classes'+i+'.npy'),
allow_pickle=True)
        mapper[i].classes = np.load('../mycode/'+str('classes'+i+'.npy'), allow_pickle=True)
        tr = mapper[i].fit_transform(new_dataset[i])
        new_dataset.loc[:,i + '_labels'] = pd.Series(tr, index = new_dataset.index)

    labeled = new_dataset[
['yearOfRegistration','powerPS','kilometer','monthofRegistration']+[x+'_labels' for x in
labels]]

    X = labeled.values
    print(X)

    y_prediction = model_rand.predict(X)
    print(y_prediction)
    return render_template('predict.html', predict='The resale value predicted is {:.2f}
$'.format(y_prediction[0]))

    payload_scoring = {"input_data": [{"field": [['yearOfRegistration', 'powerPS', 'kilometer',
'monthOfRegistration','gearbox_labels', 'notRepairedDamage_labels',
'model_labels','brand_labels', 'fuelType_labels', 'vehicleType_labels']], "values": X}]}
    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/8048356c-7b59-4d15-abdf-
fbff12c7b88a/predictions?version=2022-11-22', json=payload_scoring, headers={'Authorization':
'Bearer ' + mltoken})
    print("Scoring response")
    predictions =response_scoring.json()
    predict = predictions['predictions'][0]['values'][0][0]
    print("Final prediction :",predict)

    return render_template('predict.html',predict='{:.2f} $'.format(predict))

@app.route('/logout')
def logout():
    session.clear()
    return redirect(url_for('index'))

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

Templates :

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" href="../static/css/style3.css">
    <meta charset="UTF-8">
    <title>Login page</title>
</head>
<body>
<div class="main">

  <div class="navbar">
    <div class="icon">
      <h2 class="logo">Car Studio </h2>
    </div>
  </div>

  <a href="/logout" class="logout">Logout</a>
    <div class="container">
      <center><h1>Login Here</h1></center><br><br>
      <form action="/login" method="post" autocomplete="off">
        <div class="mb-3">
```

```
          <label for="username" class="form-label">Username</label>
          <input type="text" class="form-control" id="username" name="username" required>
        </div>
        <br>
        <div class="mb-3">
          <label for="password" class="form-label">Password</label>
          <input type="password" class="form-control" id="password" name="password" required>
        </div>
        <br>
        <br>
        <center><button type="submit" class="submit_btn">Login</button></center>
        <br>
      </form>
      <br>
      <center>Don't have an account</center>
      <center><a href="/register" class="reg_btn">Register Now</a></center>
      <br>
      <center><h5 class="info">{{info}}</h5></center>
    </div>
</div>
</body>
</html>
```

register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <link rel="stylesheet" href="../static/css/style4.css">
    <meta charset="UTF-8">
    <title>Register Now</title>
</head>
<body>
<div class="main">

  <div class="navbar">
    <div class="icon">
      <h2 class="logo">Car Studio </h2>
    </div>
  </div>

<div class="container">
     <center><h1>Registration</h1></center><br>

     <br>
  <form action="/register" method="post" autocomplete="off">

  <div class="mb-3">
    <label for="Username" class="form-label">Username</label><br><br>
    <input type="text" class="form-control" placeholder="username" name="username" required>

  </div>
  <br>
  <div class="mb-3">
    <label for="email" class="form-label">Email</label><br><br>
    <input type="email" class="form-control" placeholder="Email" name="email" required>
  </div>
  <br>
  <div class="mb-3">
    <label for="contact" class="form-label">Contact</label><br><br>
    <input type="tel" class="form-control" placeholder="Enter 10-digit number" maxlength="10"
name="contact" pattern="[0-9]{10}" required>
  </div>
  <br>
  <div class="mb-3">
    <label for="Password" class="form-label">Password</label><br><br>
    <input type="password" class="form-control" placeholder="Password" name="password"
required>
```

```
  </div>
  <br>
  <br>
  <center><button type="submit" class="btn btn-primary">Register</button></center>
  </form>
</div>
</div>
</body>
</html>
```

resaleintro.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Car resale value Prediction</title>
    <link rel="stylesheet" href="../static/css/style.css">
  </head>
  <body>
    <form action="/login" method="GET">
      <div class="main">

        <div class="navbar">
          <div class="icon">
            <h2 class="logo">Car Studio </h2>
          </div>
        </div>

        <div class="content">
          <h1>Car resale value Prediction</h1>
          <p class="par">The Best website to predict accurate values of your used car </p>
        </div>

      <div class= "footer">
        <center>
          <button class="cn"><a href="./login" class="btn btn-primary btn-
sm"></a><strong>PREDICT NOW</strong></a></button>
        </center>
      </div>

      </div>
    </form>
  </body>
</html>
```

resalepredict.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
<link rel="stylesheet" href="../static/css/style1.css">
<title>Car resale value</title>
</head>
<body>
    <div class="main">
        <a href="/logout" class="logout">Logout</a>

        <div class="navbar">
            <div class="icon">
                <h2 class="logo">Car Studio </h2>
            </div>
        </div>
        <br>
        <br>
        <br>
```

```
        <h1 class="welcome"> Welcome, {{username}}</h1>

        <div class="container">
        <table border="0" align="center">
            <br>
            <center><h1>Fill the details to get price of the car</h1></center>
            <br>
            <form action="http://localhost:5000/y_predict" method="GET" autocomplete="off">
                <tr>
                <td><label for="year" class="form-label"><b>Registration year :
</b></label></td>
                <td><input id="year" maxlength="50" name="regyear" type="text" />
                <br>
                </td>
                </tr>
                <tr>
                <td><label for="month"><b>Registration Month :</b> </label></td>
                <td><input id="month" maxlength="50" name="regmonth" type="text" />
                <br>
                </td>
                </tr>
                <tr>
                <td><label for="power"><b>Power of car in PS: </b></label></td>
                <td><input id="power" maxlength="50" name="powerps" type="text" />
                <br>
                </td>
                </tr>
                <tr>
                <td><label for="kilometer"><b>Kilometers that car have driven :</b>
</label></td>
                <td><input id="kilometer" maxlength="50" name="kms" type="text" />
                <br>
                </td>
                </tr>
                <tr>
                <td><label for="geartype"><b>Gear type :</b> </label></td>
                <td><input type="radio" name="geartype" value="manual"/> <b>Manual </b>
                <input type="radio" name="geartype" value="automatic"/> <b>Automatic  </b>
                <input type="radio" name="geartype" value="not-declared"/> <b>Not declared
</b>
                <br>
                <br>
                </td>
                </tr>
                <tr>
                <td><label for="damage"><b>Your car is repaired or damaged : </b></label></td>
                <td><input type="radio" name="damage" value="yes"/><b> Yes</b>
                <input type="radio" name="damage" value="no"/><b> No </b>
                <input type="radio" name="damage" value="not-declared"/><b> Not declared </b>
                <br>
                <br>
                </td>
                </tr>

                <tr>
                <td><label for="model"><b>Model Type :</b> </label></td>
                <td>
                <select name="model" id="model">
                <option value="" disabled selected hidden>Choose Model Name...</option>
                <option value="1_reihe">1 Reihe </option>
                <option value="100">100 </option>
                <option value="100">100 </option>
                <option value="100">100 </option>
                <option value="145">145 </option>
                <option value="147">147 </option>
                <option value="147">147 </option>
                <option value="147">147 </option>
                <option value="156">156 </option>
                <option value="156">156 </option>
                <option value="156">156 </option>
                <option value="159">159 </option>
                <option value="159">159 </option>
```

```
<option value="1er">1er </option>
<option value="1er">1er </option>
<option value="1er">1er </option>
<option value="2_reihe">2 Reihe </option>
<option value="2_reihe">2 Reihe </option>
<option value="2_reihe">2 Reihe </option>
<option value="200">200 </option>
<option value="3_reihe">3 Reihe </option>
<option value="3_reihe">3 Reihe </option>
<option value="300c">300c </option>
<option value="300c">300c </option>
<option value="3er">3er </option>
<option value="3er">3er </option>
<option value="3er">3er </option>
<option value="4_reihe">4 Reihe </option>
<option value="4_reihe">4 Reihe </option>
<option value="48429">48429 </option>
<option value="5_reihe">5 Reihe </option>
<option value="5_reihe">5 Reihe </option>
<option value="5_reihe">5 Reihe </option>
<option value="500">500 </option>
<option value="500">500 </option>
<option value="5er">5er </option>
<option value="5er">5er </option>
<option value="5er">5er </option>
<option value="6_reihe">6 Reihe </option>
<option value="6_reihe">6 Reihe </option>
<option value="6_reihe">6 Reihe </option>
<option value="601">601 </option>
<option value="6er">6er </option>
<option value="6er">6er </option>
<option value="7er">7er </option>
<option value="7er">7er </option>
<option value="7er">7er </option>
<option value="80">80 </option>
<option value="80">80 </option>
<option value="850">850 </option>
<option value="90">90 </option>
<option value="90">90 </option>
<option value="900">900 </option>
<option value="9000">9000 </option>
<option value="911">911 </option>
<option value="a_klasse">A klasse </option>
<option value="a1">A1 </option>
<option value="a2">A2 </option>
<option value="a3">A3 </option>
<option value="a4">A4 </option>
<option value="a5">A5 </option>
<option value="a6">A6 </option>
<option value="a8">A8 </option>
<option value="accord">Accord </option>
<option value="agila">Agila </option>
<option value="alhambra">Alhambra </option>
<option value="almera">Almera </option>
<option value="altea">Altea </option>
<option value="amarok">Amarok </option>
<option value="andere">Andere </option>
<option value="antara">Antara </option>
<option value="arosa">Arosa </option>
<option value="astra">Astra </option>
<option value="auris">Auris </option>
<option value="avensis">Avensis </option>
<option value="aveo">Aveo </option>
<option value="aygo">Aygo </option>
<option value="b_klasse">B Klasse </option>
<option value="b_max">B Max </option>
<option value="beetle">Beetle </option>
<option value="berlingo">Berlingo </option>
<option value="bora">Bora </option>
<option value="boxster">Boxster </option>
<option value="bravo">Bravo </option>
<option value="c_klasse">C Klasse </option>
```

```
                <option value="c_max">C Max </option>
                <option value="c_reihe">C Reihe </option>
                <option value="c1">C1 </option>
                <option value="c2">C2 </option>
                <option value="c3">C3 </option>
                <option value="c4">C4 </option>
                <option value="c5">C5 </option>
                <option value="caddy">Caddy </option>
                <option value="calibra">Calibra </option>
                <option value="captiva">Captiva </option>
                <option value="carisma">Carisma </option>
                <option value="carnival">Carnival </option>
                <option value="cayenne">Cayenne </option>
                <option value="cc">Cc </option>
                <option value="ceed">Ceed </option>
                <option value="charade">Charade </option>
                <option value="cherokee">Cherokee </option>
                <option value="citigo">Citigo </option>
                <option value="civic">Civic </option>
                <option value="cl">Cl </option>
                <option value="clio">Clio </option>
                <option value="clk">Clk </option>
                <option value="clubman">Clubman </option>
                <option value="colt">Colt </option>
                <option value="combo">Combo </option>
                <option value="cooper">Cooper </option>
                <option value="cordoba">Cordoba </option>
                <option value="corolla">Corolla </option>
                <option value="corsa">Corsa </option>
                <option value="cr_reihe">Cr Reihe </option>
                <option value="croma">Croma </option>
                <option value="crossfire">Crossfire </option>
                <option value="cuore">Cuore </option>
                <option value="cx_reihe">Cx Reihe </option>
                <option value="defender">Defender </option>
                <option value="delta">Delta </option>
                <option value="discovery_sport">Discovery Sport </option>
                <option value="discovery">Discovery </option>
                <option value="doblo">Doblo </option>
                <option value="ducato">Ducato </option>
                <option value="duster">Duster </option>
                <option value="e_klasse">E Klasse </option>
                <option value="elefantino">Elefantino </option>
                <option value="eos">Eos </option>
                <option value="escort">Escort </option>
                <option value="espace">Espace </option>
                <option value="exeo">Exeo </option>
                <option value="fabia">Fabia </option>
                <option value="fiesta">Fiesta </option>
                <option value="focus">Focus </option>
                <option value="forester">Forester </option>
                <option value="forfour">Forfour </option>
                <option value="fortwo">Fortwo </option>
                <option value="fox">Fox </option>
                <option value="freelander">Freelander </option>
                <option value="fusion">Fusion </option>
                <option value="g_klasse">G Klasse </option>
                <option value="galant">Galant </option>
                <option value="galaxy">Galaxy </option>
                <option value="getz">Getz </option>
                <option value="gl">Gl </option>
                <option value="glk">Glk </option>
                <option value="golf">Golf </option>
                <option value="grand">Grand </option>
                <option value="i_reihe">I Reihe </option>
                <option value="i3">I3 </option>
                <option value="ibiza">Ibiza </option>
                <option value="impreza">Impreza </option>
                <option value="insignia">Insignia </option>
                <option value="jazz">Jazz </option>
                <option value="jetta">Jetta </option>
                <option value="jimny">Jimny </option>
```

```
                <option value="juke">Juke </option>
                <option value="justy">Justy </option>
                <option value="ka">Ka </option>
                <option value="kadett">Kadett </option>
                <option value="kaefer">Kaefer </option>
                <option value="kalina">Kalina </option>
                <option value="kalos">Kalos </option>
                <option value="kangoo">Kangoo </option>
                <option value="kappa">Kappa </option>
                <option value="kuga">Kuga </option>
                <option value="laguna">Laguna </option>
                <option value="lancer">Lancer </option>
                <option value="lanos">Lanos </option>
                <option value="legacy">Legacy </option>
                <option value="leon">Leon </option>
                <option value="lodgy">Lodgy </option>
                <option value="logan">Logan </option>
                <option value="lupo">Lupo </option>
                <option value="lybra">Lybra </option>
                <option value="m_klasse">M Klasse </option>
                <option value="m_reihe">M Reihe </option>
                <option value="materia">Materia </option>
                <option value="matiz">Matiz </option>
                <option value="megane">Megane </option>
                <option value="meriva">Meriva </option>
                <option value="micra">Micra </option>
                <option value="mii">Mii </option>
                <option value="modus">Modus </option>
                <option value="mondeo">Mondeo </option>
                <option value="move">Move </option>
                <option value="musa">Musa </option>
                <option value="mustang">Mustang </option>
                <option value="mx_reihe">Mx Reihe </option>
                <option value="navara">Navara </option>
                <option value="niva">Niva </option>
                <option value="note">Note </option>
                <option value="nubira">Nubira </option>
                <option value="octavia">Octavia </option>
                <option value="omega">Omega </option>
                <option value="one">One </option>
                <option value="outlander">Outlander </option>
                <option value="pajero">Pajero </option>
                <option value="panda">Panda </option>
                <option value="passat">Passat </option>
                <option value="phaeton">Phaeton </option>
                <option value="picanto">Picanto </option>
                <option value="polo">Polo </option>
                <option value="primera">Primera </option>
                <option value="ptcruiser">Ptcruiser </option>
                <option value="punto">Punto </option>
                <option value="q3">Q3 </option>
                <option value="q5">Q5 </option>
                <option value="q7">Q7 </option>
                <option value="qashqai">Qashqai </option>
                <option value="r19">R19 </option>
                <option value="range_rover_evoque">Range Rover Evoque </option>
                <option value="range_rover_sport">Range Rover Sport </option>
                <option value="range_rover">Range Rover </option>
                <option value="rangerover">RangeRover </option>
                <option value="rav">Rav </option>
                <option value="rio">Rio </option>
                <option value="roadster">Roadster </option>
                <option value="roomster">Roomster </option>
                <option value="rx_reihe">Rx Reihe </option>
                <option value="s_klasse">S Klasse </option>
                <option value="s_max">S Max </option>
                <option value="s_type">S Type </option>
                <option value="s60">S60 </option>
                <option value="samara">Samara </option>
                <option value="sandero">Sandero </option>
                <option value="santa">Santa </option>
                <option value="scenic">Scenic </option>
```

```
<option value="scirocco">Scirocco </option>
<option value="seicento">Seicento </option>
<option value="serie_1">Serie 1 </option>
<option value="serie_2">Serie 2 </option>
<option value="serie_3">Serie 3 </option>
<option value="sharan">Sharan </option>
<option value="signum">Signum </option>
<option value="sirion">Sirion </option>
<option value="sl">Sl </option>
<option value="slk">Slk </option>
<option value="sorento">Sorento </option>
<option value="spark">Spark </option>
<option value="spider">Spider </option>
<option value="sportage">Sportage </option>
<option value="sprinter">Sprinter </option>
<option value="stilo">Stilo </option>
<option value="superb">Superb </option>
<option value="swift">Swift </option>
<option value="terios">Terios </option>
<option value="tigra">Tigra </option>
<option value="tiguan">Tiguan </option>
<option value="toledo">Toledo </option>
<option value="touareg">Touareg </option>
<option value="touran">Touran </option>
<option value="transit">Transit </option>
<option value="transporter">Transporter </option>
<option value="tt">Tt </option>
<option value="tucson">Tucson </option>
<option value="twingo">Twingo </option>
<option value="up">Up </option>
<option value="v_klasse">V Klasse </option>
<option value="v40">V40 </option>
<option value="v50">V50 </option>
<option value="v60">V60 </option>
<option value="v70">V70 </option>
<option value="vectra">Vectra </option>
<option value="verso">Verso </option>
<option value="viano">Viano </option>
<option value="vito">Vito </option>
<option value="vivaro">Vivaro </option>
<option value="voyager">Voyager </option>
<option value="wrangler">Wrangler </option>
<option value="x_reihe">X Reihe </option>
<option value="x_trail">X Trail </option>
<option value="x_type">X Type </option>
<option value="xc_reihe">Xc Reihe </option>
<option value="yaris">Yaris </option>
<option value="yeti">Yeti </option>
<option value="ypsilon">Ypsilon </option>
<option value="z_reihe">Z Reihe </option>
<option value="zafira">Zafira </option>

</select>
<br>
<br>
</td>
</tr>

<tr>
<td><label for="brand"><b>Brand :</b></label></td>
<td>
<select name="brand" id="brand">
<option value="" disabled selected hidden>Choose Brand Name...</option>
<option value="alfa_romeo">Alfa Romeo </option>
<option value="audi">Audi </option>
<option value="bmw">Bmw </option>
<option value="chevrolet">Chevrolet </option>
<option value="chrysler">Chrysler </option>
<option value="citroen">Citroen </option>
<option value="dacia">Dacia </option>
<option value="daewoo">Daewoo </option>
<option value="daihatsu">Daihatsu </option>
```

```
<option value="fiat">Fiat </option>
<option value="ford">Ford </option>
<option value="honda">Honda </option>
<option value="hyundai">Hyundai </option>
<option value="jaguar">Jaguar </option>
<option value="jeep">Jeep </option>
<option value="kia">Kia </option>
<option value="lada">Lada </option>
<option value="lancia">Lancia </option>
<option value="land_rover">Land Rover </option>
<option value="mazda">Mazda </option>
<option value="mercedes_benz">Mercedes Benz </option>
<option value="mini">Mini </option>
<option value="mitsubishi">Mitsubishi </option>
<option value="nissan">Nissan </option>
<option value="opel">Opel </option>
<option value="peugeot">Peugeot </option>
<option value="porsche">Porsche </option>
<option value="renault">Renault </option>
<option value="rover">Rover </option>
<option value="saab">Saab </option>
<option value="seat">Seat </option>
<option value="skoda">Skoda </option>
<option value="smart">Smart </option>
<option value="sonstige_autos">Sonstige Autos </option>
<option value="subaru">Subaru </option>
<option value="suzuki">Suzuki </option>
<option value="toyota">Toyota </option>
<option value="trabant">Trabant </option>
<option value="volkswagen">Volkswagen </option>
<option value="volvo">Volvo </option>

</select>
<br>
<br>
</td>
</tr>

<tr>
<td><label for="fuelType"><b>Fuel Type :</b></label></td>
<td>
<select name="fuelType" id="brand">
<option value="" disabled selected hidden>Choose Fuel Type...</option>
<option value="petrol"> Petrol </option>
<option value="diesel"> Diesel </option>
<option value="not-declared"> Not Declared </option>
<option value="lpg">LPG </option>
<option value="cng">CNG </option>
<option value="hybrid">Hybrid </option>
<option value="others">Others </option>
<option value="electric">Electric </option>
</select>
<br>
<br>
</td>
</tr>

<tr>
<td><label for="vehicletype"><b>Vehicle type:</b></label></td>
<td>
<select name="vehicletype" id="vehicle" >
<option value="" disabled selected hidden>Choose Vehicle Type...</option>
<option value="coupe">Coupe </option>
<option value="suv">SUV </option>
<option value="kleinwagen">Kleinwagen </option>
<option value="limousine">Limousine </option>
<option value="cabrio">Cabrio </option>
<option value="bus">Bus </option>
<option value="kombi">Kombi </option>
<option value="andere">Andere </option>
<option value="small car">Small car </option>
</select>
```

```
                <br>
                <br>
                </td>
                </tr>
                </tbody>
                </table>
                <center><input name="Submit" type="Submit" value="Submit"
id="button"/></center>
            </form>
        </div>
    </div>
</body>
</html>
```

predict.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../static/css/style2.css">
    <title>Car Resale Predicted Value</title>
</head>
<body>
    <div class="main">
        <div class="navbar">
            <div class="icon">
                <h2 class="logo">Car Studio </h2>
            </div>
        </div>

        <a href="/logout" class="logout">Logout</a>

        <div class="text-box">
          <h1>The resale value predicted is {{predict}}</h1>
        </div>

        <div class="footer">
            <form action="/predict" method="GET">
                <center><input name="Submit" type="Submit" value="Predict again" id="button"
/></center>
            </form>
        </div>
    </div>
</body>
</html>
```

Static :

Login: style3.css

```
*{
    margin: 0;
    padding: 0;
```

```
    font-family: sans-serif;
}
.main{
    background-image: linear-gradient(rgba(0,0,0,0.1),rgba(0,0,0,0.1)),url(../Images/bg2.png);
    background-position: center;
    background-size: cover;
    height:100vh;
}
.navbar {
    width: auto;
    height: 75px;
    margin: auto;
    margin-left: 5%;
}

.icon {
    margin-top: 30px;
    width: auto;
    float: left;
    height: 60px;
    border: 5px solid #A97129;
    border-radius: 5px;
    background: #3E4975;
}

.logo {
    color: #E8AE54;
    font-size: 40px;
    font-family: Arial;
    padding-left: 20px;
    padding-right: 20px;
    float: left;
    padding: 10px;
}

.logout{
    color: white;
    float: right;
    text-decoration: none;
    font-style: normal;
    margin-right: 50px;
    /* margin-top: 30px; */
    font-size: 20px;
}

.logout:hover {
    font-size: 25px;
}

.container{
    width:320px;
    height: 420px;;
    color: #fff;
    top:42%;
    left:25%;
    background: rgba(24, 35, 81, 0.9);
    position: relative;
    transform: translate(-50%,-50%);
    box-sizing:border-box;
    padding: 50px 30px;
}

.main button[type="submit"]{
    background: #A97129;
    border: none;
    width: 150px;
    height: auto;
    color: #fff;
    font-weight: bold;
    font-size: 20px;
    padding: 10px;
    border-radius: 10px;
```

```
}

.main button[type="submit"]:hover {
    background: #E8AE54;
}
.reg_btn{
    color: #A97129;
}
.reg_btn:hover {
    color: #E8AE54;
}

.info{
    color: #E8AE54;
}
```

Register: style4.css

```
*{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
}
.main{

    background-image: linear-gradient(rgba(0,0,0,0.1),rgba(0,0,0,0.1)),url(../Images/bg2.png);
    background-position: center;
    background-size: cover;
    height:100vh;
}

.navbar {
    width: auto;
    height: 75px;
    margin: auto;
    margin-left: 5%;
}

.icon {
    margin-top: 30px;
    width: auto;
    float: left;
    height: 60px;
    border: 5px solid #A97129;
    border-radius: 5px;
    background: #3E4975;
}

.logo {
    color: #E8AE54;
    font-size: 40px;
    font-family: Arial;
    padding-left: 20px;
    padding-right: 20px;
    float: left;
    padding: 10px;
}

.container{
    width:320px;
    height:530px;
    color: #fff;
    top: 42%;
    left: 25%;
    background: rgba(24, 35, 81, 0.9);
    position: relative;
    transform: translate(-50%, -50%);
    box-sizing: border-box;
    padding: 50px 30px;

}
```

```css
.main button[type="submit"] {
    background: #A97129;
    border: none;
    width: 150px;
    height: auto;
    color: #fff;
    font-weight: bold;
    font-size: 20px;
    padding: 10px;
    border-radius: 10px;
}

.main button[type="submit"]:hover {
    background: #E8AE54;
}
```

Resaleintro: style.css

```css
*{
  margin: 0;
  padding: 0;
}
.main{
    padding: auto;
    background-image: linear-gradient(rgba(0,0,0,0.1),rgba(0,0,0,0.1)),url(../Images/bg1.png);
    background-position: center;
    background-size: cover;
    height:100vh;
}
.navbar{
    width: auto;
    height: 75px;
    margin: auto;
    margin-left: 5%;

}
.icon{
    margin-top: 30px;
    width:auto;
    float: left;
    height: 60px;
    border: 5px solid #A97129;
    border-radius: 5px;
    background: #3E4975;
    /* border-color:#3E4975 */
}
.logo{
    color: #E8AE54;
    font-size: 40px;
    font-family: Arial;
    padding-left: 20px;
    padding-right: 20px;
    float:left;
    padding:10px;
}
.content{
    width: 1200px;
    height: auto;
    color: #fff;
    position: relative;
    margin: auto;
}
.content h1 {
    font-family: 'Times New Roman';
    font-size: 60px;
    padding-left: 20px;
    margin-top: 3%;
    letter-spacing: 2px;
    text-align: center;
}
```

```css
.content .par{
    padding-left: 20px;
    padding-bottom: 25px;
    font-family: Arial;
    letter-spacing: 1.2px;
    line-height: 30px;
    text-align: center;
}
.footer {
    position: relative;
    margin: auto;
    margin-top:25%;
    padding-top: 20px;
}
.cn {
    margin-top: 40px;
    padding: 20px;
    width: auto;
    height: auto;
    background: #A97129;
    border: none;
    margin-bottom: 10px;
    font-size: 20px;
    border-radius: 10px;
    cursor: pointer;
}
a {
    text-decoration: none;
    color: #fff;
}
.cn:hover {
    background-color: #E8AE54;

}
```

Resalepredict: style1.css

```css
*{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
}
.main{
    background-image: linear-gradient(rgba(0,0,0,0.1),rgba(0,0,0,0.1)),url(../Images/bg3.png);
    background-position: center;
    background-size: cover;
    height:100vh;
}
.navbar {
  width: auto;
  height: 75px;
  margin: auto;
  margin-left: 5%;
}
.icon {
  margin-top: 30px;
  width: auto;
  float: left;
  height: 60px;
  border: 5px solid #A97129;
  border-radius: 5px;
  background: #3E4975;
}
.logo {
  color: #E8AE54;
  font-size: 40px;
  font-family: Arial;
  padding-left: 20px;
  padding-right: 20px;
  float: left;
  padding: 10px;
```

```
}
.welcome{
    margin-left: 5%;
    color: #fff;
}
.container{
    width:620px;
    height:620px;
    color: #fff;
    display: block;
    top:40%;
    left:25%;
    background:rgba(24,35,81,0.9);
    position: relative;
    transform: translate(-50%,-50%);
    box-sizing:border-box;
    padding: 10px 30px;
}
a{
    color:white;
    float:right;
    text-decoration:none;
    font-style:normal;
    margin-right:50px;
    margin-top: 30px;
    font-size:20px;
}
 a:hover{
    font-size:25px;
 }
textarea {
  width: 50%;
  height: 20px;
  padding: 10px 20px;
  box-sizing: border-box;
  border: 2px solid #ccc;
  border-radius: 4px;
  background-color: #f8f8f8;
  resize: none;
}
input[type=text] {
  transition: width 0.4s ease-in-out;
}
input[type=text] {
  width: 55%;
  height: 10%;
  padding: 10px 10px;
  margin: 5px 0;
}
#model{
 width: 55%;
}
#brand{
width:55%;
}
#vehicle{
width:55%;
}
#button{
  margin: 0;
  text-align:center;
  color: #fff;
  background: #A97129;
  padding: 20px;
  width: 160px;
  height: auto;
  border: none;
  font-size: 20px;
  border-radius: 10px;
  font-weight: bold;
}
#button:hover
```

```
{
  background: #E8AE54;
}
```

Predict: style2.css

```
* {
  margin: 0;
  padding: 0;
  font-family: sans-serif;
}
.main{
  min-height: 100vh;
  background-image: url(../Images/bg4.png),linear-gradient(#67B8D0, #C6D0CD);
  background-position: center;
  background-size: cover;
  background-repeat: no-repeat;
}
.navbar {
  width: auto;
  height: 75px;
  margin: auto;
  margin-left: 5%;
}
.icon {
  margin-top: 30px;
  width: auto;
  float: left;
  height: 60px;
  border: 5px solid #A97129;
  border-radius: 5px;
  background: #3E4975;
}
.logo {
  color: #E8AE54;
  font-size: 40px;
  font-family: Arial;
  padding-left: 20px;
  padding-right: 20px;
  float: left;
  padding: 10px;
}
.logout {
  color: white;
  float: right;
  text-decoration: none;
  font-style: normal;
  margin-right: 50px;
  font-size: 20px;
}
.logout:hover {
  font-size: 25px;
}
.text-box{
  text-align: center;
  position: relative;
  color:  #E8AE54;
  top:50%;
  font-family: 'Courier New',monospace;
}
.text-box h1{
  margin-top: 80px;
  font-size: 60px;
}
.footer {
  position: relative;
  margin: auto;
  margin-top: 25%;
```

```
  padding-top: 20px;
}
#button {
  margin: 0;
  text-align: center;
  color: #fff;
  background: #A97129;
  padding: 20px;
  width: 160px;
  height: auto;
  border: none;
  font-size: 20px;
  border-radius: 10px;
  font-weight: bold;
}
#button:hover {
  background: #E8AE54;
}
```

13.2 Github Link

https://github.com/IBM-EPBL/IBM-Project-11116-1659264566.git

13.3 Project Demo Link