Car Resale Value prediction

# PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTERPRENEUSHIP – IT18099

**Submitted by**

|  |  |
| --- | --- |
| **Hyagiriva E A** | **190801030** |
| **Ashraf Ali S** | **190801007** |
| **Keerthi K M** | **190801036** |
| **Nikkila G S** | **190801050** |

**TEAM ID PNT2022TMID53471**

**Of**

**Department of Information Technology**



**SRI VENKATESWARA COLLEGE OF ENGINEERING**

Sriperumbudur,602117

**Mentor Name:** Dr.T.Sukumar

**Industry Mentor Name:** Prof Swetha

# ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to Sri Venkateswara College Of Engineering for providing us with a great opportunity to pursue our Bachelor’s degree in this institution.

It is a matter of immense pleasure to express our sincere thanks to Dr. V.Vidhya, Head of the Departments, Information Technology, Sri Venkateswara College Of Engineering ,for providing the right academic guidance that made our task possible.

We would like to thank our guide Dr. T.Sukumar Associate Professor and Assistant Head of Department of Information Technology , Sri Venkateswara College Of Engineering, for sparing her valuable time to extend help in every step of our project work, which paved the way for smooth progress and the fruitful culmination of the project.

We also would like to thank all the staff members of Information Technology for their support. We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the project work.

# ABSTRACT

This project aims to create a model to forecast the fair prices of used automobiles based on a number of variables, such as vehicle mileage, year of manufacture, gasoline type, transmission, kilometers driven, vehicle type, and engine power. The market for used automobiles can profit from this approach for sellers, purchasers, and automakers. Based on the information entered by the user, it can return a price prediction that is reasonably accurate. Machine learning and data science are used in the model construction process. Utilized automobile listings were used to compile the information.

The research used a variety of regression techniques, including linear regression, polynomial regression, support vector regression, decision tree regression, and random forest regression, to obtain the best level of accuracy. This project first visualized the data to better understand the dataset before beginning to develop the model. To fit the regression and ensure its effectiveness, the dataset was partitioned and altered. R-square was calculated to assess each regression's performance. The resulting model has a greater prediction accuracy compared to earlier studies and incorporates more elements of used cars.

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TOPIC** | **PAGE NO** |
|  | **ACKNOWLEDGEMENT** | 2 |
|  | **ABSTRACT** | 3 |
| **1.** | **INTRODUCTION**   * 1. PROJECT OVERVIEW   2. PURPOSE | 6 |
| **2.** | **LITERATURE SURVEY**   * 1. EXISTING PROBLEM   2. REFERENCES   3. PROBLEM STATEMENT DEFINITION | **7** |
| **3.** | **IDEATION & PROPOSED SOLUTION**   * 1. EMPATHY MAP CANVAS   2. IDEATION & BRAINSTORMING   3. PROPOSED SOLUTION   4. PROBLEM SOLUTION FIT | **9** |
| **4.** | **REQUIREMENT ANALYSIS**   * 1. FUNCTIONAL REQUIREMENT   2. NON-FUNCTIONAL REQUIREMENT | **16** |
| **5.** | **PROJECT DESIGN**   * 1. DATA FLOW DIAGRAMS   2. SOLUTION & TECHNICAL ARCHITECTURE   3. USER STORIES | 17 |
| **6.** | **PROJECT PLANNING & SCHEDULING**   * 1. SPRINT PLANNING AND ESTIMATION   2. SPRINT DELIVERY SCHEDULE   3. REPORTS FROM JIRA | **23** |

|  |  |  |
| --- | --- | --- |
| **7.** | **CODING & SOLUTIONING**   * 1. FEATURE 1   2. FEATURE 2 | **26** |
| **8.** | **TESTING**  8.1 TEST CASES  8.1 USER ACCEPTANCE TESTING | 33 |
| **9.** | **RESULTS**  9.1 PERFORMANCE METRICS | **35** |
| **10.** | **ADVANTAGES AND DISDVANTAGES** | **36** |
| **11.** | **CONCLUSION** | 37 |
| **12.** | **FUTURE SCOPE** | 37 |
| **13.** | **APPENDIX** | 38 |

**1.INTRODUCTION**

# PROJECT OVERVIEW:

* + - The primary goal of developing a system to predict the resale value of cars is to gain practical experience with Python and Data Science.
    - Car resale value prediction is a system that estimates a car's resale value based on the user-provided parameters.
    - The user fills out the form with the car's information, and the value at which it will be sold is then forecasted.
    - The system operates using the machine learning program's trained dataset to determine the exact worth of the car.

# PURPOSE

* + - The sole purpose of this general-purpose system for estimating resale value is to estimate the amount that the user can probably acquire.
    - To help the user acquire an estimated value before reselling the car and avoid making a deal at a loss, we strive to anticipate the amount of resale with an accuracy of best 70–75%.

# 2.LITERATURE SURVEY

* 1. **EXISTING PROBLEM**

The problem is defined as an optimised method of estimating insurance costs based on the manufacturer with some extra expenses paid by the government in the form of taxes. Used car sales are rising globally as a result of the time and energy-intensive nature of the current methods for cost estimation, the rising cost of new cars, and consumers' reluctance to purchase new vehicles owing to a lack of cash. Customers can buy a new automobile with confidence knowing that the money they invest will be worthwhile because the industry's prices for new cars are set by. To accurately assess the worthiness of the car using a range of features, a used car price prediction system is required. Despite the fact that some websites provide this service, their prediction strategy might not be the greatest. Additionally, many methods and algorithms may enhance the accuracy of forecasting a used car's actual market value. When purchasing or selling, it's critical to understand their true market value.

**2.2.REFERENCES**

1. W. Q. Zhao, "(2009). Evaluation of product model design based on bp neural network", Computer Engineering & Design, vol. 30, no. 24, pp. 5715-5711.
2. H. Zhang, "(2002). A study on bp networks with combined activation functions", Journal of Ocean University of Qingdao.
3. You Zhou, "(2014). Research on Second-hand car's Value Assessment Methods Based on the Replacement Cost Method", Liaoning University of Technology.
4. T. Deguchi, T. Takahashi and N. Ishii, "(2014). On temporal summation in chaotic neural network with incremental learning", International Journal of Software Innovation, vol. 2, no. 4, pp. 13, 2014.
   1. **PROBLEM STATEMENT DEFNITION**

|  |  |
| --- | --- |
| Who does the problem affect? | Car resale value prediction is used by people who want to sell their car for a decent price. It is most common for working people business people etc. |
| What is the issue? | Car resale value prediction is very important in this modern period where everyone owns a car. Many people are ready to sell their cars once they find a new model or once their car gets old. But the problem Is that they do not know the exact price of their car. Many just approximately quote a price and they just complete a deal; it is either a loss for the seller or a loss for the buyer. This model brings a solution to it, this model helps in predicting the exact or the most appropriate price for the vehicle so that it is not a loss for the seller as well as the buyer |
| Why is it important that we fix the problem? | It is important to fix this problem because many people do not predict the right prize for their vehicle and either the seller or the buyer tends to face a loss. So this web application helps users get a estimated price of their car and no one gets to face any sorts of loss. |
| Which solution can be used to address this issue? | A machine learning powered web application model with the strong building of algorithm that helps to identify and predict the price of the vehicle with the identification of the vehicle condition from various parameters entered by the user. |
| What methodology used to solve the issue? | Supervised and Unsupervised machine learning, Data mining, Python web application interface – Flask , IBM Cloud. |

**3.IDEATION AND PROPOSED SOLUTION**

* 1. **EMPATHY MAP CANVAS Empathy Map :**

An empathy map is a straightforward, simple-to-understand picture that

summarizes information about a user's actions and views. It is a helpful tool that enables

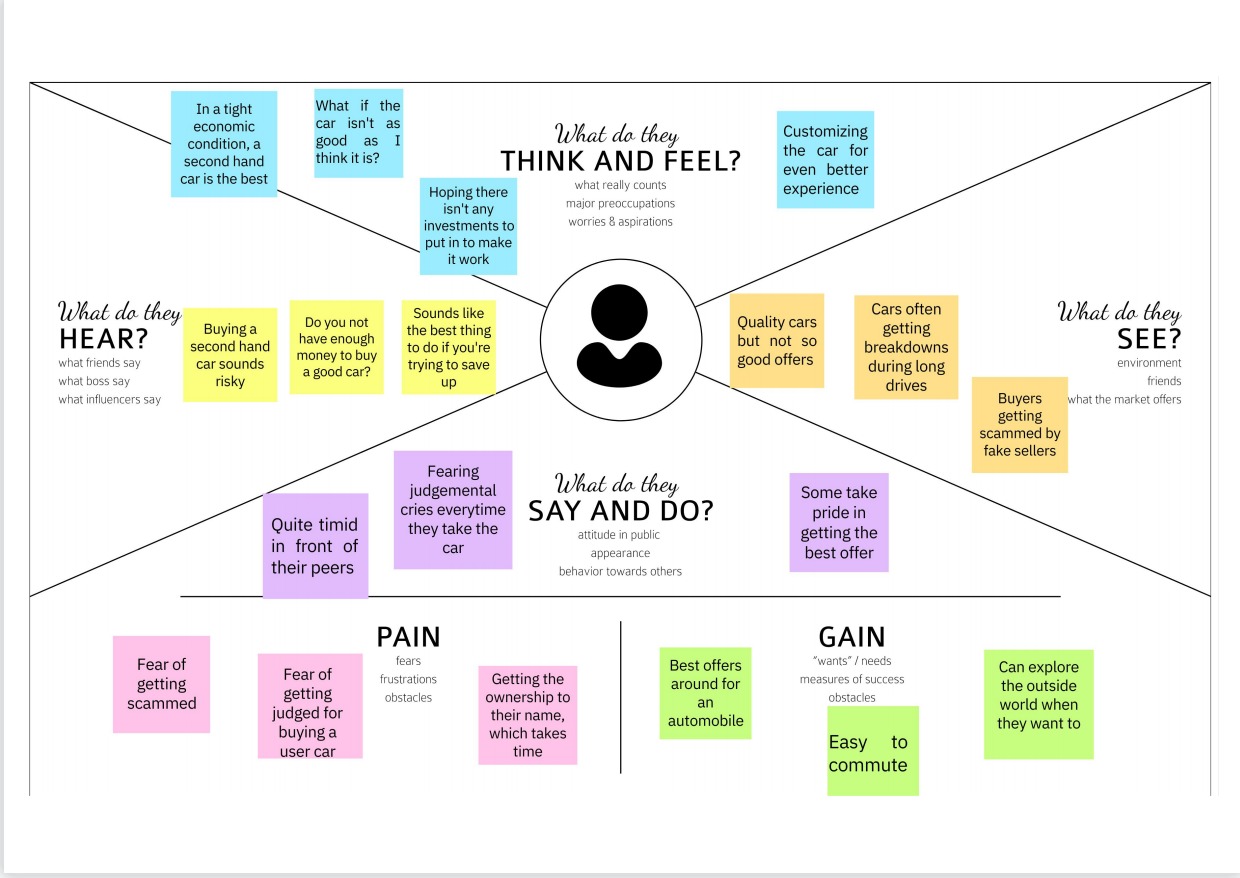
teams to comprehend their users more fully.

It's important to comprehend both the actual issue and the individual who is experiencing

it in order to develop a workable solution. Participants learn to think about situations from

the user's perspective, including goals and problems, through the exercise of constructing

the map.



* 1. **IDEATION AND BRAINSTORMING**

**Brainstorm:**

In a free and open setting, brainstorming enables team members to engage in the

innovative problem-solving process. Prioritizing quantity over quality, unconventional ideas

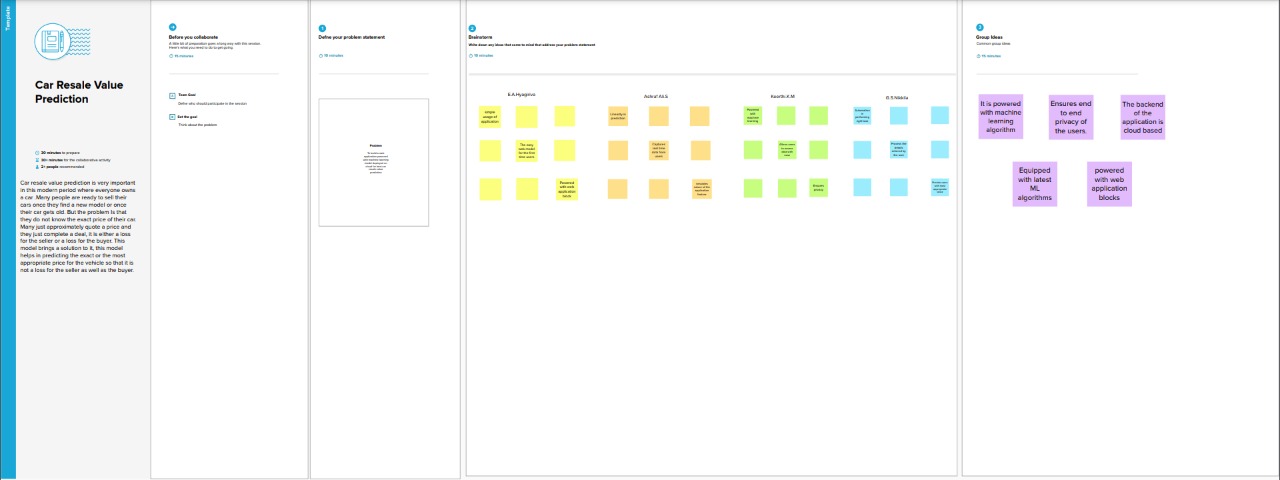
are welcomed and expanded upon, and everyone is urged to cooperate in order to produce

a wealth of creative solutions. Utilize this template during your own brainstorming meetings

to allow your team to use their creativity and begin developing notions even if you are not

in the same room.

**Step-1: Brainstorm, Idea Listing and Grouping**



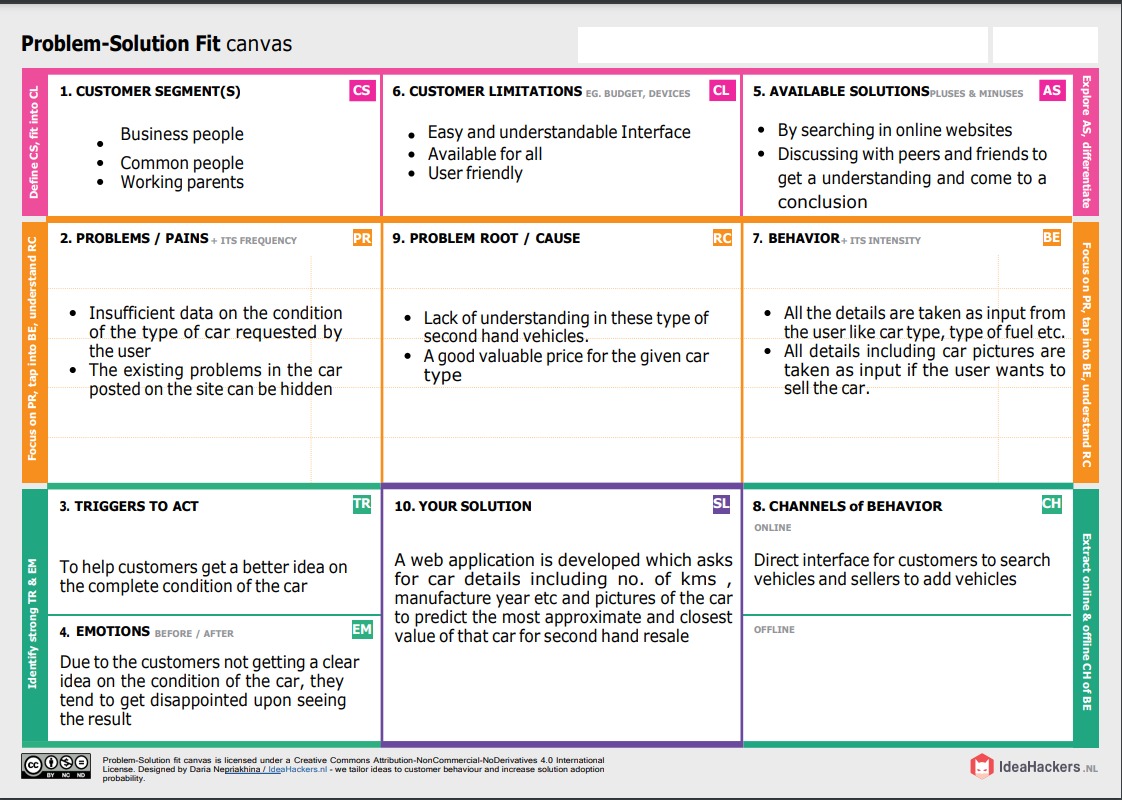
**Step-2: Idea Prioritization**



* 1. **PROPOSED SOLUTION**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Parameter** | **Description** |
| **1.** | Problem Statement (Problem to be solved) | * Car resale value prediction is very important in this modern period where everyone owns a car.Many people are ready to sell their cars once they find a new model or once their car gets old. But the problem Is that they do not know the exact price of their car. Many just approximately quote a price and they just complete a deal, it is either a loss for the seller or a loss for the buyer. This model brings a solution to it, this model helps in predicting the exact or the most appropriate price for the vehicle so that it is not a loss for the seller as well as the buyer. |
| **2.** | Idea / Solution description | * Almost all the existing car details and their most common type of selling details are stored in a csv format. * This data is then loaded, preprocessed in order to remove null values, segregate the dependent and independent variables, encode the needed columns, create analysis maps, split the data into training and testing data, choose the model which can suit this problem, train the model with the training data, test the accuracy with the test data against predicted data and save the model to integrate it with a web app. * A web app is built which renders a form for the user to enter the attributes. The saved model is loaded and the entered values are fed into the loaded model and the predicted results are returned to the user. * The model is then deployed into the cloud for the web app to request from the deployed model. |
| **3.** | Novelty / Uniqueness | * The seller must add all the details about the vehicle such as kilometers driven , year of manufacture, model , brand etc, and add pictures of the car. * Most of the vehicle service centers have a tool that estimates the amount of life remaining in a tire until it completely wears out, so in this web application it is recommended for the sellers to also mention those percentages and submit a copy of their last 1year service history to give the buyer a clear idea on the condition of the car as well as to get the most appropriate price for their vehicle. |
| **4.** | Social Impact/ Customer Satisfaction | * Personalize the UI experience * Improves accurate result as expected * Cloud deployed Machine Learning Model λ * Accurate prediction at good time complexity |
| **5.** | Business Model (Revenue Model) | * Solutions prospects of improvement * Suits for better saving of involvements * Economic Development * Easy interface |
| **6.** | Scalability of the Solution | * Since the machine learning model is saved and deployed in a cloud environment the app is fast to response to user’s requests or queries or to accept multiple user submission of the details of their vehicles and predict the result and return the response to the users. The web app is deployed in a auto scaling environment |

* 1. **PROBLEM SOLUTION FIT**



**4.REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENTS**

|  |  |
| --- | --- |
| **REQUIREMENT ID** | **REQUIREMENT DESCRIPTION** |
| **1** | Users should be able to see the options in which they enter the car details such as kilometers driven, engine condition etc. when they enter the home URL. |
| **2** | Users should select all the details with all the necessary attributes filled and should be appropriately notified when some values are missing. |
| **3** | The app should accept the test values and fees those inputs into pre-saved trained machine learning model and should return the prediction result. |
| **4** | The app should redirect the users to the appropriate page based on the prediction result. |

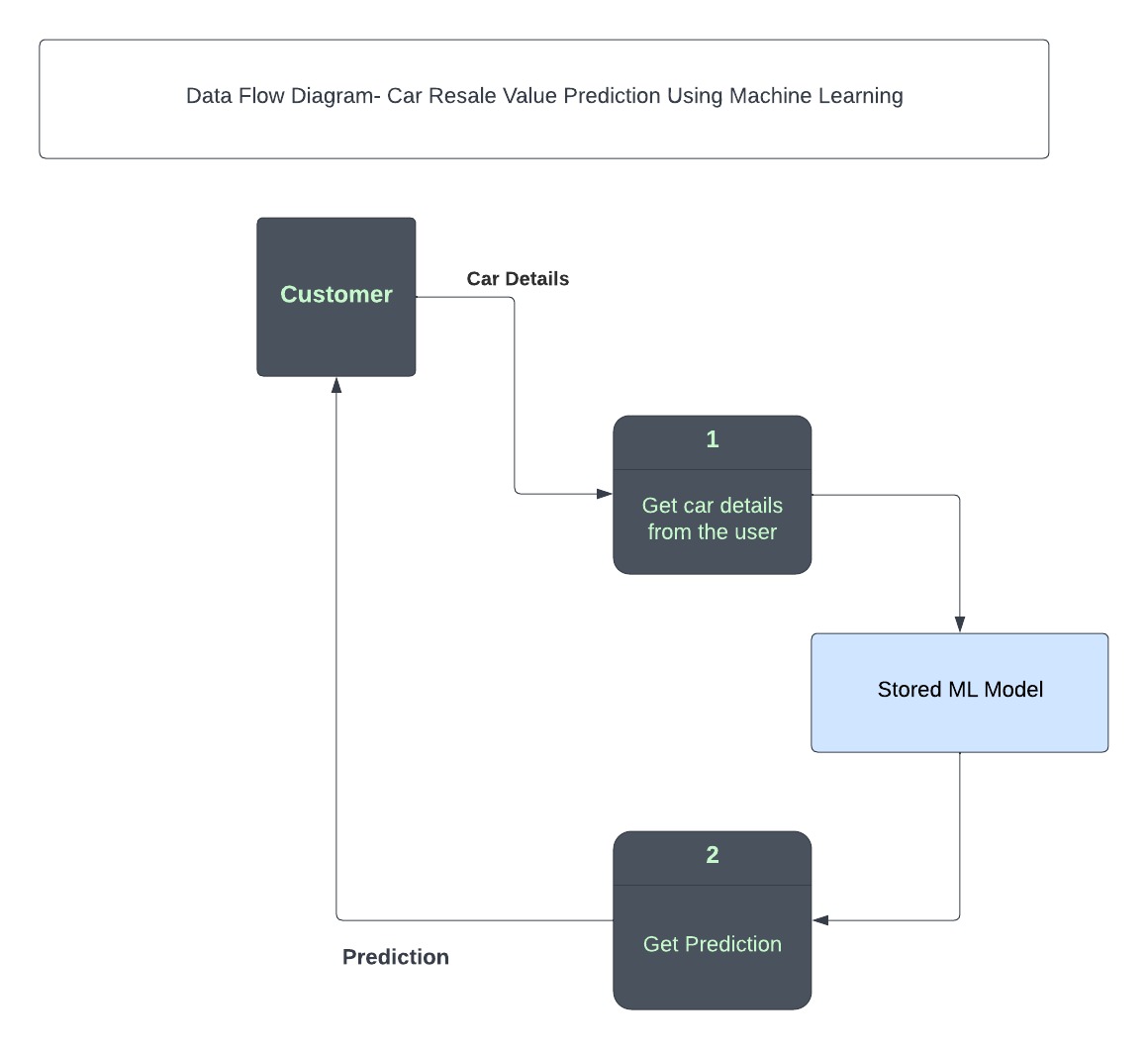
**4.2 NON-FUNCTIONAL REQUIREMENTS**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Indicates how effectively and easy users can learn and use a system. |
| NFR-2 | **Security** | Assures all data inside the system or its part will be protected against malware attacks or  unauthorized access. |
| NFR-3 | **Reliability** | Specifies the probability of the software performing without failure for a specific  number of uses or amount of time. |
| NFR-4 | **Performance** | Deals with the measure of the system’s  response time under different load conditions. |
| NFR-5 | **Availability** | Describes how likely the system is accessible  for a user at a given point in time. |
| NFR-6 | **Scalability** | Accesses the highest workload under which the system will still meet the performance  requirements. |

* 1. **N**

1. **PROJECT DESIGN**
   1. **DATA FLOW DIAGRAMS**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right value of the resale car of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



* 1. **SOLUTION AND TECHNICAL ARCHITECTURE i.SOLUTION ARCHITECTURE**

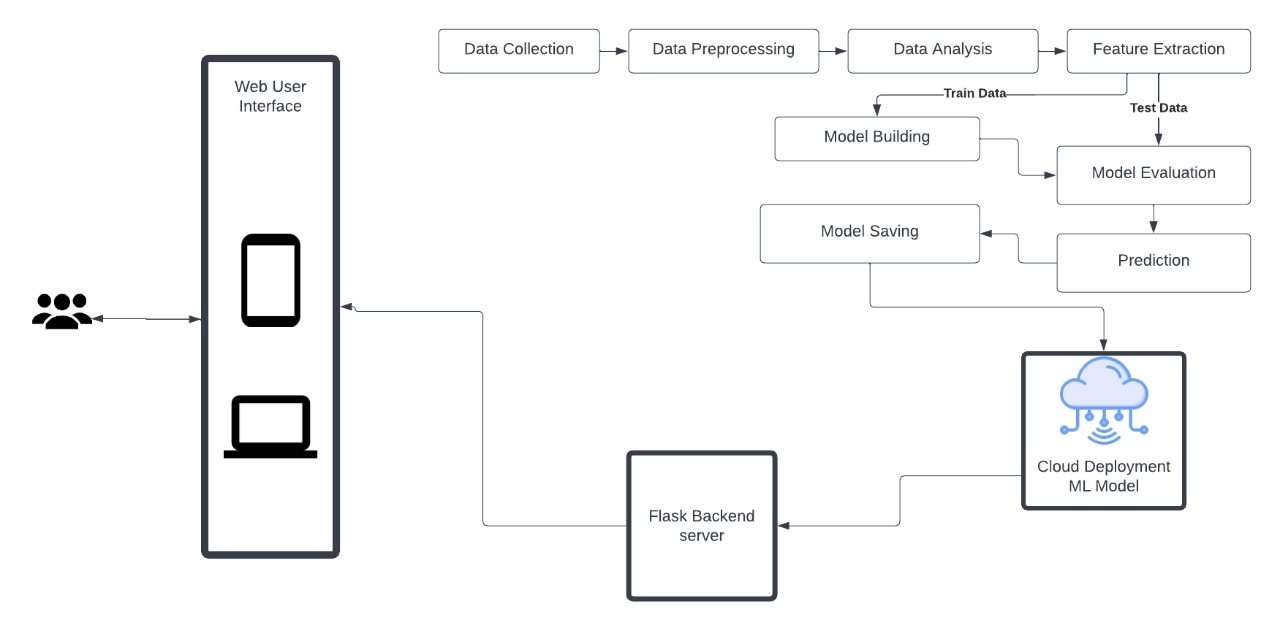
**FUNCTIONAL REQUIREMENTS**

|  |  |
| --- | --- |
| **REQUIREMENT ID** | **REQUIREMENT DESCRIPTION** |
| **1** | Users should be able to see the options in which they enter the car details such as kilometers driven, engine condition etc. when they enter the home URL. |
| **2** | Users should select all the details with all the necessary attributes filled and should be appropriately notified when some values are missing. |
| **3** | The app should accept the test values and fees those inputs into pre-saved trained machine learning model and should return the prediction result. |
| **4** | The app should redirect the users to the appropriate page based on the prediction result. |

**NON-FUNCTIONAL REQUIREMENTS**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Indicates how effectively and easy users can learn and use a system. |
| NFR-2 | **Security** | Assures all data inside the system or its part will be protected against malware attacks or  unauthorized access. |
| NFR-3 | **Reliability** | Specifies the probability of the software performing without failure for a specific  number of uses or amount of time. |
| NFR-4 | **Performance** | Deals with the measure of the system’s  response time under different load conditions. |
| NFR-5 | **Availability** | Describes how likely the system is accessible  for a user at a given point in time. |
| NFR-6 | **Scalability** | Accesses the highest workload under which the system will still meet the performance  requirements. |

**ii. TECHNICAL ARCHITECTURE**



**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with application e.g.  Web Ui only | HTML, CSS,Python, Flask |
| 2. | Application Logic-1 | Load the data set and find the test data and train data | Python |
| 3. | Application Logic-2 | Logic for a process in the  application | Pandas, numpy, sklearn |
| 4. | Application Logic-3 | Logic for a process in the  application | flask |
| 5. | Database | Data Type,  Configurations etc. | Dataset |
| 6. | Cloud Database | Database Service on  Cloud | IBM Cloud |
| 7. | File Storage | File storage requirements | IBM Block Storage or Other Storage Service or Local  Filesystem |
| 8. | External API-1 | Purpose of External API  used in the application | IBM cloud API, etc. |
| 9. | Machine Learning  Model | Purpose of Machine  Learning Model | Regression Model. |
| 10. | Infrastructure (Server  / Cloud) | Application Deployment on Local System / Cloud Local Server Configuration:  Cloud Server  Configuration : | Local, Cloud Foundry, Kubernetes, etc. |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source  Frameworks | List the open-source  frameworks used | Technology of  Opensource framework |
| 2. | Scalable Architecture | Justify the scalability of architecture (3 – tier, -  services) | Machine Learning |
| 3. | Availability | Justify the availability of application (e.g. use of load balancers, distributed  servers etc.) | Machine Learning |

**5.3USER STORIES**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Numb**  **er** | **User Story / Task** | **Acceptance criteria** | **Priori ty** | **Release** |
| Customer | Dashboard | USN-1 | Entering the car details  in the webpage |  | High | Sprint-1 |
| Customer (Web- user) | Process | USN-2 | As a user, I can enter the car details for which I want to know the resale value. |  | Medi um | Sprint-2 |
| Customer Care Executive | Maintenance | USN-3 | As a executive, I can rectify Customer’s Problems as well as  Comments | I can interact through commen ts | High | Sprint-4 |

**6.PROJECT PLANNING AND SCHEDULING**

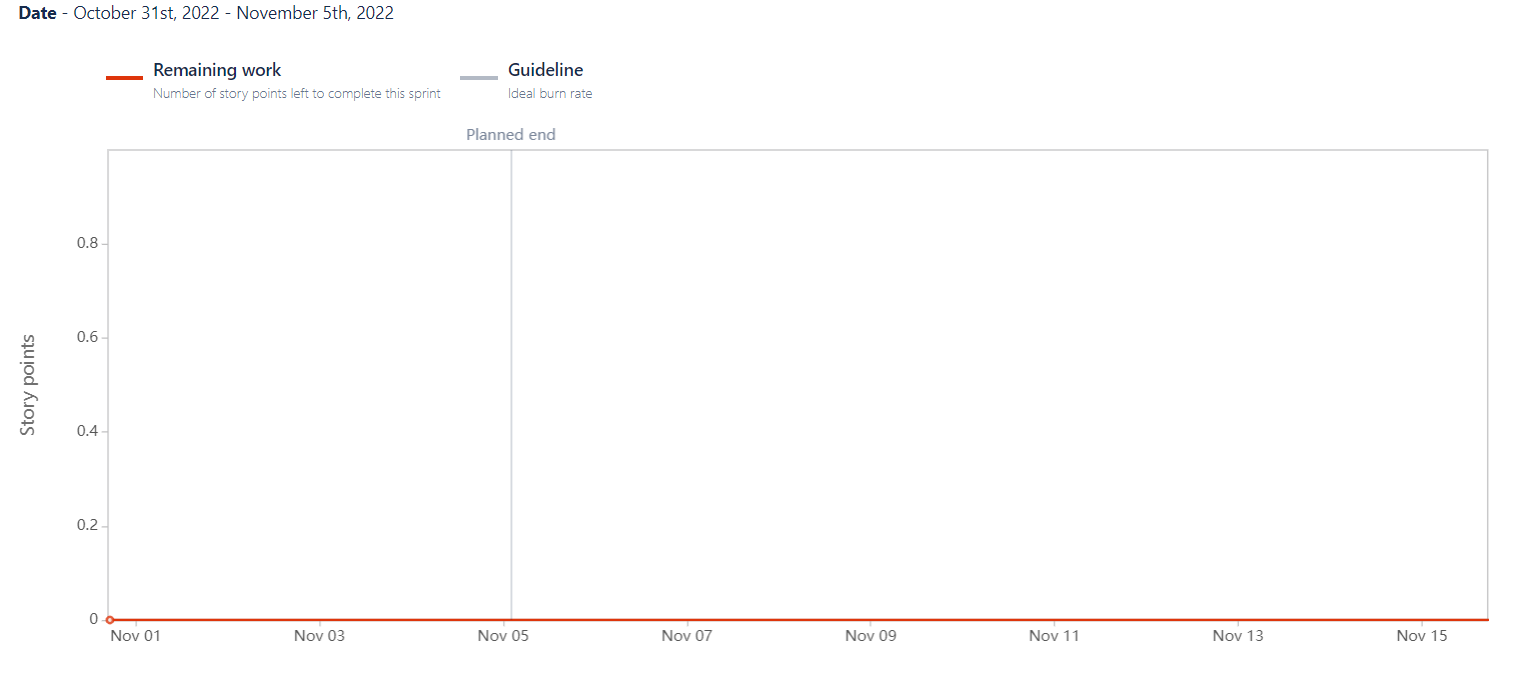
* 1. **SPRINT PLANNING AND ESTIMATION**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Data collection | USN-1 | Collect dataset | 1 | low | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-1 | Preprocess data | USN-2 | Read and clean the dataset | 2 | low | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-2 | Model building | USN-3 | Splitting into independent and dependent variables | 3 | medium | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
| Sprint-2 | regression | USN-4 | Applying regression model | 3 | medium | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-3 | Application building | USN-5 | Build the python flask application and HTML page | 5 | High | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-3 | Testing | USN-6 | Execute the code and test | 5 | high | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-4 | Training the model /Integrating flask | USN-7 | Training the model on IBM cloud and integrate flask | 5 | high | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |

* 1. **SPRINT DELIVERY SCHEDULE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Complet ed (as on Planned**  **End Date)** | **Sprint Release Date (Actual)** |
| Sprint-1 | 15 | 5 Days | 24  Oct2022 | 29 Oct2022 | 15 | 29Oct 2022 |
| Sprint-2 | 15 | 5 Days | 31  Oct2022 | 05 Nov2022 | 15 | 05Nov 2022 |
| Sprint-3 | 15 | 5 Days | 07 Nov  2022 | 12 Nov2022 | 15 | 12Nov 2022 |
| Sprint-4 | 15 | 5 Days | 14 Nov  2022 | 19 Nov2022 | 15 | 19Nov 2022 |

* 1. **REPORTS FROM JIRA**



**7.CODING AND SOLUTIONING**

**7.1 FEATURE 1**

import pandas as pd import numpy as np import matplotlib as plt

from sklearn.preprocessing import LabelEncoder import pickle

#Load the dataset

df = pd.read\_csv(r"E:\car\_resale\Data\autos.csv", header=0, sep=',', encoding='Latin1', )

#print all the different sellers print(df.seller.value\_counts())

#remove the seller type haveing only 3 car df[df.seller != 'gewerblich']

#now all the sellers are same so we can get rid of this column df=df.drop(columns=['seller']) #1 refer the columns & 0 refer the index

#print all different seller print(df.offerType.value\_counts())

#remove the offers type having only 12 listings df[df.offerType != 'Gesuch']

#now all offer are sameso we can get rid this collumn df=df.drop(columns=['offerType']) # 1 refer the columns & 0 refer the index

'''car having power les then 50ps and above 900ps seems a little suspicious, let's remove them and see what we have got now'''

print(df.shape)

df = df[(df.powerPS > 50) & (df.powerPS < 900)] print(df.shape)

#around 50000 cars ahave been removed which could have introunduced error toour data #Simlarly, filtering our the cars having registeration years not in the mentioned range #print(df.shape)

df = df[(df.yearOfRegistration >= 1950) & (df.yearOfRegistration < 2017)] print(df.shape)

#not much of a difference but still, 10000 rows have been reduced. it's better to #get rid of faulty data instead of keeping them just to increase the size

'''removing irrelevent columns which are either the same for all the cars in the dataset, or can introduce bias, so removing them too..'''

df.drop(['name', 'abtest', 'dateCrawled', 'nrOfPictures', 'lastSeen', 'postalCode', 'dateCreated'], axis='columns', inplace=True)

'''dropping the duplicates from the dataframe and stroing it in a new

here all row having same value in all the mentioned columns will be deleted and by defult, only first occurance of any such row is kept'''

new\_df = df.copy()

new\_df = new\_df.drop\_duplicates(['price', 'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS',

'model', 'kilometer', 'monthOfRegistration', 'fuelType', 'notRepairedDamage'])

#As the dataset contained same german words for many features, changing them to engilsh new\_df.gearbox.replace(('manuell','automatik'), ('manual','automatic'), inplace=True) new\_df.fuelType.replace(('benzin','andere','elektro'), ('petrol','others','electric'), inplace=True) new\_df.vehicleType.replace(('kleinwagen','cabrio','kombi','andere'),

('small car','canvertible','combination','others'), inplace=True) new\_df.notRepairedDamage.replace(('ja','nein'), ('Yes','No'), inplace=True) #### Removing the outliers

new\_df = new\_df[(new\_df.price >= 100) & (new\_df.price <= 150000)]

''' Filling NaN values for columns whose data might not be there with the information provider, which might lead to some variance but our model but we will still be able to give some estimate to the user'''

new\_df['notRepairedDamage'].fillna(value='not-declared',inplace=True) new\_df['fuelType'].fillna(value='not-declared',inplace=True) new\_df['gearbox'].fillna(value='not-declared',inplace=True) new\_df['vehicleType'].fillna(value='not-declared',inplace=True) new\_df['model'].fillna(value='not-declared',inplace=True)

#can save the csv for future purpose. new\_df.to\_csv("autos\_preprocessed.csv")

#Columns which contain categorical values, which we'll need to convert via label encoding labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

'''looping over the labels to the label encoding for all at once and saveing the LABEL ENCODING FILES'''

mapper = {} for i in labels:

mapper[i] = LabelEncoder() mapper[i].fit(new\_df[i])

tr = mapper[i].transform(new\_df[i]) np.save(str('classes'+i+'.npy'), mapper[i].classes\_)

print(i,":", mapper[i])

new\_df.loc[:, i+ '\_labels'] = pd.Series(tr, index=new\_df.index)

#Final data to be put in a new dataframe called "LABELED", labeled = new\_df[

[

'price', 'yearOfRegistration', 'powerPS', 'kilometer', 'monthOfRegistration'

] + [x+"\_labels" for x in labels]] print(labeled.columns)

#Storing price in Y and reset of the data in X Y = labeled.iloc[:,0].values

X = labeled.iloc[:,1:].values #need to reshape the Y values Y = Y.reshape(-1,1)

#traing data and test data

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)

#Model building and fitting

from sklearn.metrics import r2\_score

Model=LGBMRegressor(boosting\_type="gbdt",learning\_rate=0.07,metric="rmse",n\_estimators=300,objective="root\_mean\_squared\_error",random\_state=42,reg\_sqrt=True)

model.fit(X\_train, Y\_train)

Y\_pred = model.predict(X\_test)

find\_scores(Y\_test, Y\_pred, X\_train)

filename = 'resale\_model.pkl'

pickle.dump(regressor, open(filename, 'wb'))

The System is defined in the python language that predicts the amount of resale value based on the given information.The system works on the trained dataset of the machine learning program that evaluates the precise value of the car.User can enter details only of fields like purchase price of car,kilometers driven,fuel of car,year of purchase.

**7.1 FEATURE 2**

import pandas as pd

import numpy as np

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

import pickle

from sklearn.preprocessing import LabelEncoder

import requests

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

def load\_model(file='C:/Users/Hyagiriva/OneDrive/Desktop/docs/nalaiya thiran/New folder/final1/Final Deliverables/Model Building/resale\_model.sav'):

return pickle.load(open(file, 'rb'))

@app.route('/')

def index():

return render\_template('templates/value.html')

@app.route('/predict\_page')

def predict\_page():

return render\_template('value.html')

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

def predict():

reg\_year = int(request.args.get('regyear'))

powerps = float(request.args.get('powerps'))

kms = float(request.args.get('kms'))

reg\_month = 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')

fuel\_type = request.args.get('fuelType')

veh\_type = request.args.get('vehicletype')

new\_row = {'yearOfReg': reg\_year, 'powerPS': powerps, 'kilometer': kms,

'monthOfRegistration': reg\_month, 'gearbox': gearbox,

'notRepairedDamage': damage,

'model': model, 'brand': brand, 'fuelType': fuel\_type,

'vehicletype': veh\_type}

print(new\_row)

new\_df = pd.DataFrame(columns=['vehicletype', 'yearOfReg', 'gearbox',

'powerPS', 'model', 'kilometer', 'monthOfRegistration', 'fuelType',

'brand', 'notRepairedDamage'])

new\_df = new\_df.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(

str('classes'+i+'.npy'), allow\_pickle=True)

transform = mapper[i].fit\_transform(new\_df[i])

new\_df.loc[:, i+'\_labels'] = pd.Series(transform, index=new\_df.index)

labeled = new\_df[['yearOfReg', 'powerPS', 'kilometer',

'monthOfRegistration'] + [x+'\_labels' for x in labels]]

X = labeled.values.tolist()

print('\n\n', X)

predict = reg\_model.predict(X)

payload\_scoring = {"input\_data": [{"fields": [['yearOfReg', 'powerPS', 'kilometer', 'monthOfRegistration', 'gearbox\_labels',

'notRepairedDamage\_labels', 'model\_labels', 'brand\_labels', 'fuelType\_labels', 'vehicletype\_labels']], "values": X}]}

print("Final prediction :", predict)

return render\_template('predict.html', predict=predict)

if \_\_name\_\_ == '\_\_main\_\_':

reg\_model = load\_model()

app.run(host='localhost', debug=True, threaded=False)

Upon from submission,the data is sent to the ML model via Flask API and the model responds with a predicted resale value of the car based on user input.The prediction is displayed on the web page using a render template .Thus,with minimal information and without human intervention or manual examination,a user can predict the resale value of his car.

**8.TESTING**

* 1. **TEST CASES**

**1.)Metrics:**

**Values:**

**Regression Model: LGBM Regressor**

'mae': 1327.549477341283,

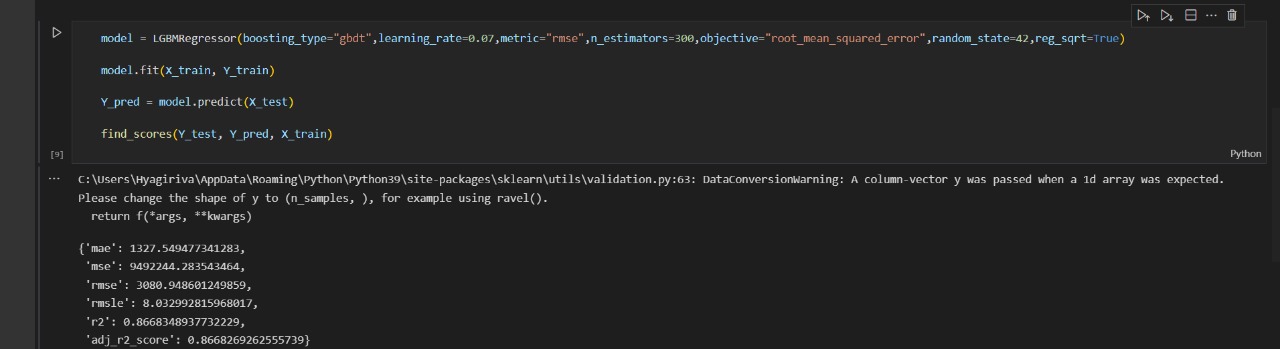
'mse': 9492244.283543464,

'rmse': 3080.948601249859,

'rmsle': 8.032992815968017,

'r2': 0.8668348937732229,

'adj\_r2\_score': 0.8668269262555739



**2.)Tune the model:**

**Hyperparameter Tuning:**

1) Learning Rate: [0.01, 0.03, 0.05, 0.07]

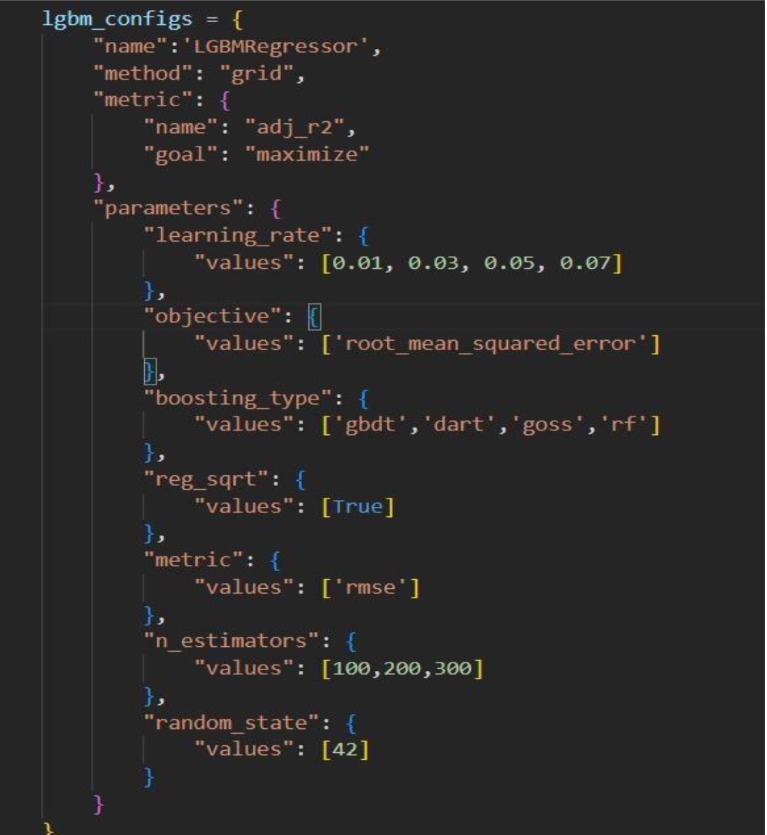
2) Boosting Type: ['gbdt','dart','goss','rf']

3) Number of Estimators: [100,200,300]

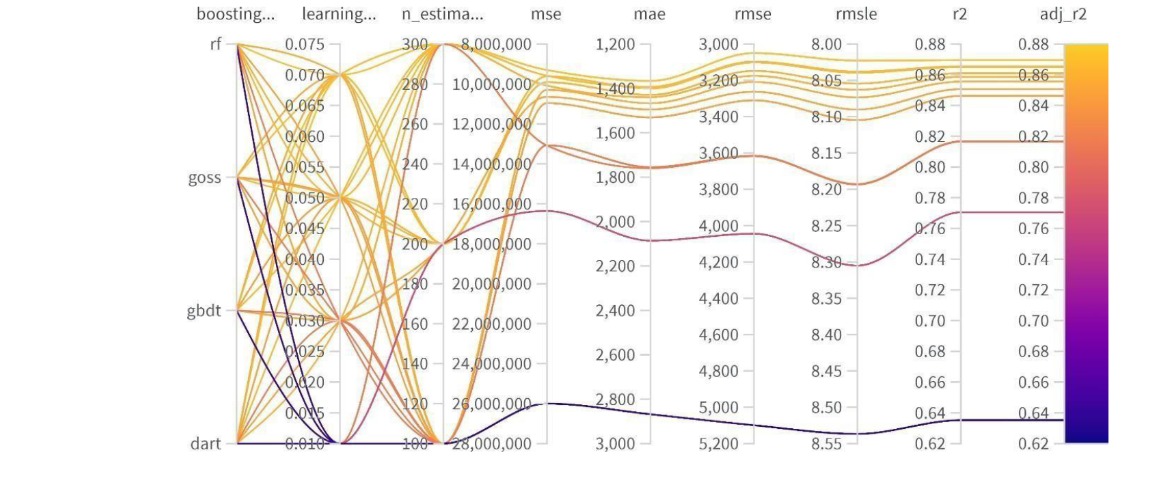
**Validation Method:** Grid Search Cross Validation

**Best Parameters:** Learning Rate – 0.07 Boosting Type – ‘gbdt’

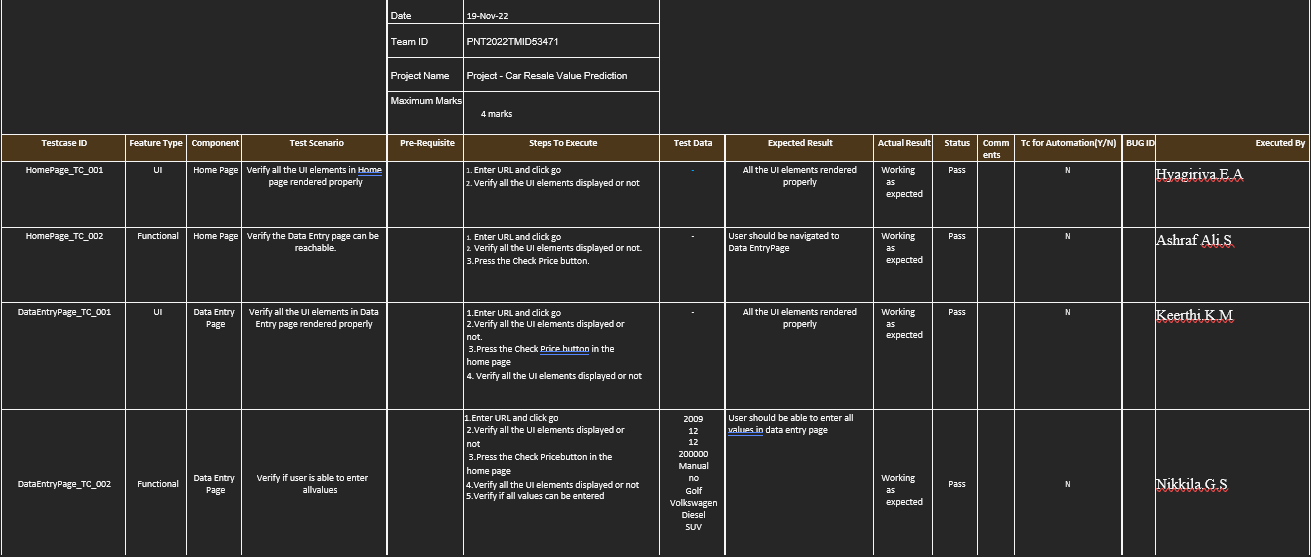
Number of Estimators - 300

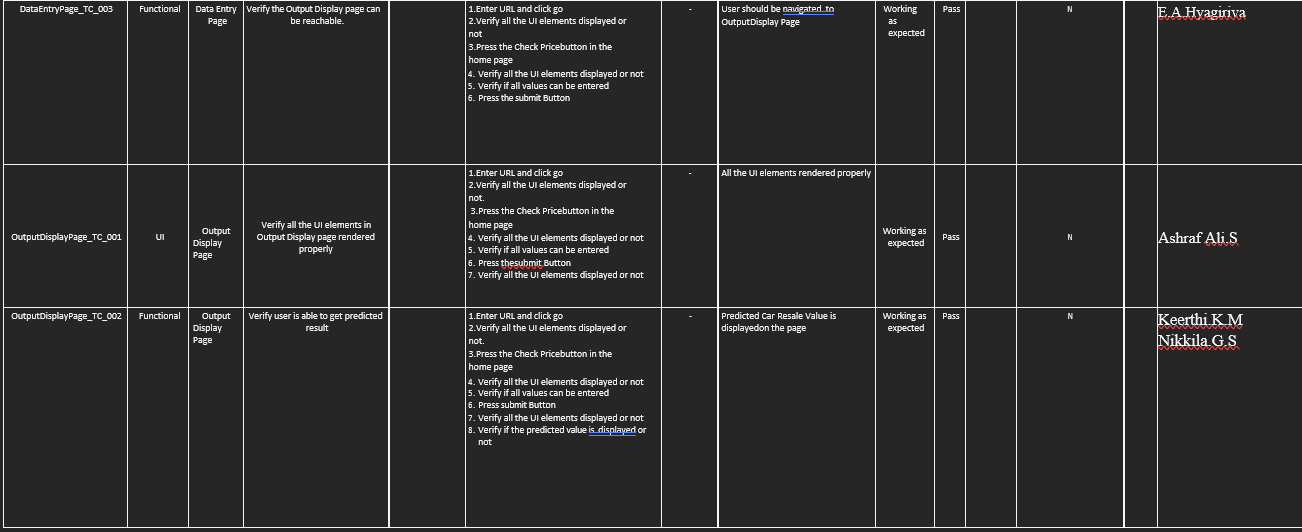


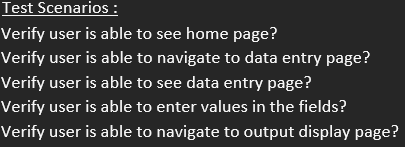
**Wandb sweep:**



* 1. **USER ACCEPTANCE TESTING :**

****





* 1. **RESULTS**

**9.1 PERFORMANCE METRICS**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Data collection | USN-1 | Collect dataset | 1 | low | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-1 | Preprocess data | USN-2 | Read and clean the dataset | 2 | low | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-2 | Model building | USN-3 | Splitting into independent and dependent variables | 3 | medium | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
| Sprint-2 | regression | USN-4 | Applying regression model | 3 | medium | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-3 | Application building | USN-5 | Build the python flask application and HTML page | 5 | High | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-3 | Testing | USN-6 | Execute the code and test | 5 | high | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |
| Sprint-4 | Training the model /Integrating flask | USN-7 | Training the model on IBM cloud and integrate flask | 5 | high | Hyagiriva.E.A  Ashraf Ali.S  Keerthi.K.M  Nikkila.G.S |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Complet ed (as on Planned**  **End Date)** | **Sprint Release Date (Actual)** |
| Sprint-1 | 15 | 5 Days | 24  Oct2022 | 29 Oct2022 | 15 | 29Oct 2022 |
| Sprint-2 | 15 | 5 Days | 31  Oct2022 | 05 Nov2022 | 15 | 05Nov 2022 |
| Sprint-3 | 15 | 5 Days | 07 Nov  2022 | 12 Nov2022 | 15 | 12Nov 2022 |
| Sprint-4 | 15 | 5 Days | 14 Nov  2022 | 19 Nov2022 | 15 | 19Nov 2022 |

* 1. **ADVANTAGES AND DISDVANTAGES**

1. **ADVANTAGES**
   * Value for the money pre-owned vehicles have lower sticker prices, better value for the money spent.
   * Slower rate of depreciation.
   * Reduced insurance and registration costs.
   * Lower loan amount to be borrowed
   * Higher inflation
2. **DISDVANTAGES**
   * Little to No Financing
   * Little to No Warranty
   * New Models Not Available
   * No Accurate Prediction
   1. **CONCLUSION**

The large number of factors that must be taken into account for an effective prediction

makes predicting car prices a difficult undertaking. Although one of the techniques that

improves prediction performance is data cleansing, it is insufficient in the case of

complicated data sets like the one used in this study. The accuracy of a single

machine algorithm applied to the data set was less than 50%. Because of this, the ensemble

of various machine learning algorithms has been suggested, and this combination of

ML methods improves approximate price prediction. When compared to the use of a

single machine learning method, this represents a significant improvement. However,

the suggested system has the disadvantage of using significantly more computational

resources than a single machine learning method.

* 1. **FUTURE SCOPE**

Once sufficient data has been gathered, efficient deep learning techniques like RNNs

or LSTMs (Long Short-Term Memory) can be used. The accuracy and RMSE can both

be significantly improved. By using CNN to recognise dents, scratches, and other

physical flaws in a car's photos, one may also forecast a car's resale value that is more

pertinent to its physical state.

* 1. **APPENDIX CODING**

**Source code:**

**Resale value prediction final.ipynb**

import pandas as pd import numpy as np import matplotlib as plt

from sklearn.preprocessing import LabelEncoder import pickle

#Load the dataset

df = pd.read\_csv(r"E:\car\_resale\Data\autos.csv", header=0, sep=',', encoding='Latin1', )

#print all the different sellers print(df.seller.value\_counts())

#remove the seller type haveing only 3 car df[df.seller != 'gewerblich']

#now all the sellers are same so we can get rid of this column df=df.drop(columns=['seller']) #1 refer the columns & 0 refer the index

#print all different seller print(df.offerType.value\_counts())

#remove the offers type having only 12 listings df[df.offerType != 'Gesuch']

#now all offer are sameso we can get rid this collumn df=df.drop(columns=['offerType']) # 1 refer the columns & 0 refer the index

'''car having power les then 50ps and above 900ps seems a little suspicious, let's remove them and see what we have got now'''

print(df.shape)

df = df[(df.powerPS > 50) & (df.powerPS < 900)] print(df.shape)

#around 50000 cars ahave been removed which could have introunduced error toour data #Simlarly, filtering our the cars having registeration years not in the mentioned range #print(df.shape)

df = df[(df.yearOfRegistration >= 1950) & (df.yearOfRegistration < 2017)] print(df.shape)

#not much of a difference but still, 10000 rows have been reduced. it's better to #get rid of faulty data instead of keeping them just to increase the size

'''removing irrelevent columns which are either the same for all the cars in the dataset, or can introduce bias, so removing them too..'''

df.drop(['name', 'abtest', 'dateCrawled', 'nrOfPictures', 'lastSeen', 'postalCode', 'dateCreated'], axis='columns', inplace=True)

'''dropping the duplicates from the dataframe and stroing it in a new

here all row having same value in all the mentioned columns will be deleted and by defult, only first occurance of any such row is kept'''

new\_df = df.copy()

new\_df = new\_df.drop\_duplicates(['price', 'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS',

'model', 'kilometer', 'monthOfRegistration', 'fuelType', 'notRepairedDamage'])

#As the dataset contained same german words for many features, changing them to engilsh new\_df.gearbox.replace(('manuell','automatik'), ('manual','automatic'), inplace=True) new\_df.fuelType.replace(('benzin','andere','elektro'), ('petrol','others','electric'), inplace=True) new\_df.vehicleType.replace(('kleinwagen','cabrio','kombi','andere'),

('small car','canvertible','combination','others'), inplace=True) new\_df.notRepairedDamage.replace(('ja','nein'), ('Yes','No'), inplace=True) #### Removing the outliers

new\_df = new\_df[(new\_df.price >= 100) & (new\_df.price <= 150000)]

''' Filling NaN values for columns whose data might not be there with the information provider, which might lead to some variance but our model but we will still be able to give some estimate to the user'''

new\_df['notRepairedDamage'].fillna(value='not-declared',inplace=True) new\_df['fuelType'].fillna(value='not-declared',inplace=True) new\_df['gearbox'].fillna(value='not-declared',inplace=True) new\_df['vehicleType'].fillna(value='not-declared',inplace=True) new\_df['model'].fillna(value='not-declared',inplace=True)

#can save the csv for future purpose. new\_df.to\_csv("autos\_preprocessed.csv")

#Columns which contain categorical values, which we'll need to convert via label encoding labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

'''looping over the labels to the label encoding for all at once and saveing the LABEL ENCODING FILES'''

mapper = {} for i in labels:

mapper[i] = LabelEncoder() mapper[i].fit(new\_df[i])

tr = mapper[i].transform(new\_df[i]) np.save(str('classes'+i+'.npy'), mapper[i].classes\_)

print(i,":", mapper[i])

new\_df.loc[:, i+ '\_labels'] = pd.Series(tr, index=new\_df.index)

#Final data to be put in a new dataframe called "LABELED", labeled = new\_df[

[

'price', 'yearOfRegistration', 'powerPS', 'kilometer', 'monthOfRegistration'

] + [x+"\_labels" for x in labels]] print(labeled.columns)

#Storing price in Y and reset of the data in X Y = labeled.iloc[:,0].values

X = labeled.iloc[:,1:].values #need to reshape the Y values Y = Y.reshape(-1,1)

#traing data and test data

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)

#Model building and fitting

from sklearn.metrics import r2\_score

Model=LGBMRegressor(boosting\_type="gbdt",learning\_rate=0.07,metric="rmse",n\_estimators=300,objective="root\_mean\_squared\_error",random\_state=42,reg\_sqrt=True)

model.fit(X\_train, Y\_train)

Y\_pred = model.predict(X\_test)

find\_scores(Y\_test, Y\_pred, X\_train)

filename = 'resale\_model.pkl'

pickle.dump(regressor, open(filename, 'wb'))

x

**Resale\_flask.py**

import pandas as pd

import numpy as np

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

import pickle

from sklearn.preprocessing import LabelEncoder

import requests

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

def load\_model(file='C:/Users/Hyagiriva/OneDrive/Desktop/docs/nalaiya thiran/New folder/final1/Final Deliverables/Model Building/resale\_model.sav'):

return pickle.load(open(file, 'rb'))

@app.route('/')

def index():

return render\_template('templates/value.html')

@app.route('/predict\_page')

def predict\_page():

return render\_template('value.html')

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

def predict():

reg\_year = int(request.args.get('regyear'))

powerps = float(request.args.get('powerps'))

kms = float(request.args.get('kms'))

reg\_month = 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')

fuel\_type = request.args.get('fuelType')

veh\_type = request.args.get('vehicletype')

new\_row = {'yearOfReg': reg\_year, 'powerPS': powerps, 'kilometer': kms,

'monthOfRegistration': reg\_month, 'gearbox': gearbox,

'notRepairedDamage': damage,

'model': model, 'brand': brand, 'fuelType': fuel\_type,

'vehicletype': veh\_type}

print(new\_row)

new\_df = pd.DataFrame(columns=['vehicletype', 'yearOfReg', 'gearbox',

'powerPS', 'model', 'kilometer', 'monthOfRegistration', 'fuelType',

'brand', 'notRepairedDamage'])

new\_df = new\_df.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(

str('classes'+i+'.npy'), allow\_pickle=True)

transform = mapper[i].fit\_transform(new\_df[i])

new\_df.loc[:, i+'\_labels'] = pd.Series(transform, index=new\_df.index)

labeled = new\_df[['yearOfReg', 'powerPS', 'kilometer',

'monthOfRegistration'] + [x+'\_labels' for x in labels]]

X = labeled.values.tolist()

print('\n\n', X)

predict = reg\_model.predict(X)

payload\_scoring = {"input\_data": [{"fields": [['yearOfReg', 'powerPS', 'kilometer', 'monthOfRegistration', 'gearbox\_labels',

'notRepairedDamage\_labels', 'model\_labels', 'brand\_labels', 'fuelType\_labels', 'vehicletype\_labels']], "values": X}]}

print("Final prediction :", predict)

return render\_template('predict.html', predict=predict)

if \_\_name\_\_ == '\_\_main\_\_':

reg\_model = load\_model()

app.run(host='localhost', debug=True, threaded=False)

**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">

<title>Car Resale Predicted Value</title>

</head>

<body style="background-color:coral;">

<div class="text-box">

<h1 style="color: black;">The Predicted Car Resale Value is </h1>

<h1 style="color: black;">{{predict}}</h1>

</div>

</body>

</html>

**Value.html**

<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

<link rel="stylesheet" href="../static/css/value.css">

<title>Car resale value</title>

</head>

<body>

<section class="form">

<form action="http://localhost:5000/predict" method="GET">

<table>

<tbody>

<h1>Predict Resale Value of Your Car</h1>

<tr>

<td><label for="year" padding:10px>Registration year: </label></td>

<td><input id="year" maxlength="50" name="regyear" type="text" />

<br>

<br>

</td>

</tr>

<tr>

<td><label for="month">Registration Month: </label></td>

<td><input id="month" maxlength="50" name="regmonth" type="text" />

<br>

<br>

</td>

</tr>

<tr>

<td><label for="power">Power of car in PS: </label></td>

<td><input id="power" maxlength="50" name="powerps" type="text" />

<br>

<br>

</td>

</tr>

<tr>

<td><label for="kilometer">Kilometers that car have driven: </label></td>

<td><input id="kilometer" maxlength="50" name="kms" type="text" />

<br>

<br>

</td>

</tr>

<tr>

<td><label for="geartype">Gear type: </label></td>

<td><input type="radio" name="geartype" value="manual" /> Manual

<input type="radio" name="geartype" value="automatic" /> Automatic

<input type="radio" name="geartype" value="not-declared" /> Not declared

<br>

<br>

</td>

</tr>

<tr>

<td><label for="damage">Your car is repaired or damaged: </label></td>

<td><input type="radio" name="damage" value="yes" /> Yes

<input type="radio" name="damage" value="no" /> No

<input type="radio" name="damage" value="not-declared" /> Not declared

<br>

<br>

</td>

</tr>

<tr>

<td><label for="model">Model Type: </label></td>

<td>

<select name="model" id="model">

<option value="" disabled selected hidden>Choose Model Name...</option>

<option value="golf">Golf </option>

<option value="grand">Grand </option>

<option value="fabia">Fabia </option>

<option value="3er">3er </option>

<option value="2\_reihe">2 Reihe </option>

<option value="andere">Andere </option>

<option value="c\_max">C Max </option>

<option value="3\_reihe">3 Reihe </option>

<option value="passat">Passat </option>

<option value="navara">Navara </option>

<option value="ka">Ka </option>

<option value="polo">Polo </option>

<option value="twingo">Twingo </option>

<option value="a\_klasse">A klasse </option>

<option value="scirocco">Scirocco </option>

<option value="5er">5er </option>

<option value="meriva">Meriva </option>

<option value="arosa">Arosa </option>

<option value="c4">C4 </option>

<option value="civic">Civic </option>

<option value="transporter">Transporter </option>

<option value="punto">Punto </option>

<option value="e\_klasse">E Klasse </option>

<option value="clio">Clio </option>

<option value="kadett">Kadett </option>

<option value="kangoo">Kangoo </option>

<option value="corsa">Corsa </option>

<option value="one">One </option>

<option value="fortwo">Fortwo </option>

<option value="1er">1er </option>

<option value="b\_klasse">B Klasse </option>

<option value="signum">Signum </option>

<option value="astra">Astra </option>

<option value="a8">A8 </option>

<option value="jetta">Jetta </option>

<option value="fiesta">Fiesta </option>

<option value="c\_klasse">C Klasse </option>

<option value="micra">Micra </option>

<option value="vito">Vito </option>

<option value="sprinter">Sprinter </option>

<option value="156">156 </option>

<option value="escort">Escort </option>

<option value="forester">Forester </option>

<option value="xc\_reihe">Xc Reihe </option>

<option value="scenic">Scenic </option>

<option value="a4">A4 </option>

<option value="a1">A1 </option>

<option value="insignia">Insignia </option>

<option value="combo">Combo </option>

<option value="focus">Focus </option>

<option value="tt">Tt </option>

<option value="a6">A6 </option>

<option value="jazz">Jazz </option>

<option value="omega">Omega </option>

<option value="slk">Slk </option>

<option value="7er">7er </option>

<option value="80">80 </option>

<option value="147">147 </option>

<option value="glk">Glk </option>

<option value="100">100 </option>

<option value="z\_reihe">Z Reihe </option>

<option value="sportage">Sportage </option>

<option value="sorento">Sorento </option>

<option value="v40">V40 </option>

<option value="5er">5er </option>

<option value="ibiza">Ibiza </option>

<option value="3er">3er </option>

<option value="mustang">Mustang </option>

<option value="eos">Eos </option>

<option value="touran">Touran </option>

<option value="getz">Getz </option>

<option value="a3">A3 </option>

<option value="almera">Almera </option>

<option value="megane">Megane </option>

<option value="7er">7er </option>

<option value="1er">1er </option>

<option value="lupo">Lupo </option>

<option value="r19">R19 </option>

<option value="zafira">Zafira </option>

<option value="caddy">Caddy </option>

<option value="2\_reihe">2 Reihe </option>

<option value="mondeo">Mondeo </option>

<option value="cordoba">Cordoba </option>

<option value="colt">Colt </option>

<option value="impreza">Impreza </option>

<option value="vectra">Vectra </option>

<option value="berlingo">Berlingo </option>

<option value="80">80 </option>

<option value="m\_klasse">M Klasse </option>

<option value="tiguan">Tiguan </option>

<option value="i\_reihe">I Reihe </option>

<option value="espace">Espace </option>

<option value="sharan">Sharan </option>

<option value="6\_reihe">6 Reihe </option>

<option value="panda">Panda </option>

<option value="up">Up </option>

<option value="seicento">Seicento </option>

<option value="ceed">Ceed </option>

<option value="5\_reihe">5 Reihe </option>

<option value="yeti">Yeti </option>

<option value="octavia">Octavia </option>

<option value="mii">Mii </option>

<option value="rx\_reihe">Rx Reihe </option>

<option value="6er">6er </option>

<option value="modus">Modus </option>

<option value="fox">Fox </option>

<option value="matiz">Matiz </option>

<option value="beetle">Beetle </option>

<option value="c1">C1 </option>

<option value="rio">Rio </option>

<option value="touareg">Touareg </option>

<option value="logan">Logan </option>

<option value="spider">Spider </option>

<option value="cuore">Cuore </option>

<option value="s\_max">S Max </option>

<option value="a2">A2 </option>

<option value="x\_reihe">X Reihe </option>

<option value="a5">A5 </option>

<option value="galaxy">Galaxy </option>

<option value="c3">C3 </option>

<option value="viano">Viano </option>

<option value="s\_klasse">S Klasse </option>

<option value="1\_reihe">1 Reihe </option>

<option value="avensis">Avensis </option>

<option value="sl">Sl </option>

<option value="roomster">Roomster </option>

<option value="q5">Q5 </option>

<option value="kaefer">Kaefer </option>

<option value="santa">Santa </option>

<option value="cooper">Cooper </option>

<option value="leon">Leon </option>

<option value="4\_reihe">4 Reihe </option>

<option value="500">500 </option>

<option value="laguna">Laguna </option>

<option value="ptcruiser">Ptcruiser </option>

<option value="clk">Clk </option>

<option value="primera">Primera </option>

<option value="exeo">Exeo </option>

<option value="159">159 </option>

<option value="transit">Transit </option>

<option value="juke">Juke </option>

<option value="qashqai">Qashqai </option>

<option value="carisma">Carisma </option>

<option value="accord">Accord </option>

<option value="corolla">Corolla </option>

<option value="lanos">Lanos </option>

<option value="phaeton">Phaeton </option>

<option value="boxster">Boxster </option>

<option value="verso">Verso </option>

<option value="swift">Swift </option>

<option value="rav">Rav </option>

<option value="kuga">Kuga </option>

<option value="picanto">Picanto </option>

<option value="kalos">Kalos </option>

<option value="superb">Superb </option>

<option value="stilo">Stilo </option>

<option value="alhambra">Alhambra </option>

<option value="911">911 </option>

<option value="mx\_reihe">Mx Reihe </option>

<option value="m\_reihe">M Reihe </option>

<option value="roadster">Roadster </option>

<option value="ypsilon">Ypsilon </option>

<option value="cayenne">Cayenne </option>

<option value="galant">Galant </option>

<option value="justy">Justy </option>

<option value="90">90 </option>

<option value="sirion">Sirion </option>

<option value="crossfire">Crossfire </option>

<option value="6\_reihe">6 Reihe </option>

<option value="agila">Agila </option>

<option value="duster">Duster </option>

<option value="cr\_reihe">Cr Reihe </option>

<option value="v50">V50 </option>

<option value="discovery">Discovery </option>

<option value="c\_reihe">C Reihe </option>

<option value="v\_klasse">V Klasse </option>

<option value="yaris">Yaris </option>

<option value="c5">C5 </option>

<option value="aygo">Aygo </option>

<option value="cc">Cc </option>

<option value="carnival">Carnival </option>

<option value="fusion">Fusion </option>

<option value="bora">Bora </option>

<option value="forfour">Forfour </option>

<option value="100">100 </option>

<option value="cl">Cl </option>

<option value="tigra">Tigra </option>

<option value="156">156 </option>

<option value="300c">300c </option>

<option value="100">100 </option>

<option value="147">147 </option>

<option value="q3">Q3 </option>

<option value="spark">Spark </option>

<option value="v70">V70 </option>

<option value="x\_type">X Type </option>

<option value="5\_reihe">5 Reihe </option>

<option value="ducato">Ducato </option>

<option value="s\_type">S Type </option>

<option value="x\_trail">X Trail </option>

<option value="toledo">Toledo </option>

<option value="altea">Altea </option>

<option value="7er">7er </option>

<option value="voyager">Voyager </option>

<option value="calibra">Calibra </option>

<option value="bravo">Bravo </option>

<option value="range\_rover">Range Rover </option>

<option value="antara">Antara </option>

<option value="tucson">Tucson </option>

<option value="q7">Q7 </option>

<option value="citigo">Citigo </option>

<option value="jimny">Jimny </option>

<option value="cx\_reihe">Cx Reihe </option>

<option value="wrangler">Wrangler </option>

<option value="lybra">Lybra </option>

<option value="range\_rover\_sport">Range Rover Sport </option>

<option value="lancer">Lancer </option>

<option value="159">159 </option>

<option value="freelander">Freelander </option>

<option value="captiva">Captiva </option>

<option value="c2">C2 </option>

<option value="500">500 </option>

<option value="range\_rover\_evoque">Range Rover Evoque </option>

<option value="sandero">Sandero </option>

<option value="note">Note </option>

<option value="900">900 </option>

<option value="147">147 </option>

<option value="defender">Defender </option>

<option value="cherokee">Cherokee </option>

<option value="clubman">Clubman </option>

<option value="samara">Samara </option>

<option value="2\_reihe">2 Reihe </option>

<option value="1er">1er </option>

<option value="3er">3er </option>

<option value="601">601 </option>

<option value="3\_reihe">3 Reihe </option>

<option value="4\_reihe">4 Reihe </option>

<option value="5er">5er </option>

<option value="6\_reihe">6 Reihe </option>

<option value="legacy">Legacy </option>

<option value="pajero">Pajero </option>

<option value="auris">Auris </option>

<option value="niva">Niva </option>

<option value="5\_reihe">5 Reihe </option>

<option value="s60">S60 </option>

<option value="nubira">Nubira </option>

<option value="vivaro">Vivaro </option>

<option value="g\_klasse">G Klasse </option>

<option value="lodgy">Lodgy </option>

<option value="850">850 </option>

<option value="serie\_2">Serie 2 </option>

<option value="6er">6er </option>

<option value="charade">Charade </option>

<option value="croma">Croma </option>

<option value="outlander">Outlander </option>

<option value="gl">Gl </option>

<option value="doblo">Doblo </option>

<option value="musa">Musa </option>

<option value="amarok">Amarok </option>

<option value="156">156 </option>

<option value="move">Move </option>

<option value="9000">9000 </option>

<option value="v60">V60 </option>

<option value="145">145 </option>

<option value="aveo">Aveo </option>

<option value="200">200 </option>

<option value="300c">300c </option>

<option value="b\_max">B Max </option>

<option value="delta">Delta </option>

<option value="terios">Terios </option>

<option value="rangerover">RangeRover </option>

<option value="90">90 </option>

<option value="materia">Materia </option>

<option value="kalina">Kalina </option>

<option value="elefantino">Elefantino </option>

<option value="i3">I3 </option>

<option value="kappa">Kappa </option>

<option value="serie\_3">Serie 3 </option>

<option value="48429">48429 </option>

<option value="serie\_1">Serie 1 </option>

<option value="discovery\_sport">Discovery Sport </option>

</select>

<br>

<br>

</td>

</tr>

<tr>

<td><label for="brand">Brand :</label></td>

<td>

<select name="brand" id="brand">

<option value="" disabled selected hidden>Choose Brand Name...</option>

<option value="volkswagen">Volkswagen </option>

<option value="audi">Audi </option>

<option value="jeep">Jeep </option>

<option value="skoda">Skoda </option>

<option value="bmw">Bmw </option>

<option value="peugeot">Peugeot </option>

<option value="ford">Ford </option>

<option value="mazda">Mazda </option>

<option value="nissan">Nissan </option>

<option value="renault">Renault </option>

<option value="mercedes\_benz">Mercedes Benz </option>

<option value="opel">Opel </option>

<option value="seat">Seat </option>

<option value="citroen">Citroen </option>

<option value="honda">Honda </option>

<option value="fiat">Fiat </option>

<option value="mini">Mini </option>

<option value="smart">Smart </option>

<option value="hyundai">Hyundai </option>

<option value="sonstige\_autos">Sonstige Autos </option>

<option value="alfa\_romeo">Alfa Romeo </option>

<option value="subaru">Subaru </option>

<option value="volvo">Volvo </option>

<option value="mitsubishi">Mitsubishi </option>

<option value="kia">Kia </option>

<option value="suzuki">Suzuki </option>

<option value="lancia">Lancia </option>

<option value="porsche">Porsche </option>

<option value="toyota">Toyota </option>

<option value="chevrolet">Chevrolet </option>

<option value="dacia">Dacia </option>

<option value="daihatsu">Daihatsu </option>

<option value="trabant">Trabant </option>

<option value="saab">Saab </option>

<option value="chrysler">Chrysler </option>

<option value="jaguar">Jaguar </option>

<option value="daewoo">Daewoo </option>

<option value="rover">Rover </option>

<option value="land\_rover">Land Rover </option>

<option value="lada">Lada </option>

</select>

<br>

<br>

</td>

</tr>

<tr>

<td><label for="fuelType">Fuel Type :</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">Vehicle type:</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="volkswagen">Volkswagen </option>

</select>

<br>

<br>

</td>

</tr>

</tbody>

</table>

<input name="Submit" type="Submit" value="Submit" id="button" />

</form>

</section>

</body>

</html>

**GitHub & Project Demo Link**

[Github](https://github.com/IBM-EPBL/IBM-Project-3879-1658668833) [Demowork](https://vimeo.com/772726861)