CAR RESALE VALUE PREDICTION

Team Members:

Malepati Ashritha
Abhinayaa K
Ayyagari Mihika
Aishwarya V

INDEX

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Over the past decade, the production of vehicles has been consistently expanding. As a result, the trade-in automobile market started growing, which is now a booming sector of the economy. The new method of utilising online platforms meets the need for both the customer and the merchant to be better informed about the trends and examples that determine the value of a used automobile. Car resale value prediction system is built with the objective of predicting the correct value of used cars that enables the users to sell the cars remotely with accurate valuation and without human intervention in the process to eliminate biased valuation. The price prediction is done based on various factors like year of manufacturing, model type, vehicle type, fuel type, horsepower of vehicle and number of kilometres travelled. In the used car market, this model can be advantageous to vendors, purchasers, and automobile manufacturers. Based on the data that users give, the system outputs a price that is relatively accurate. The model is developed using machine learning algorithms. The system works on Regression algorithm that predicts precise value of a used car. The dataset used was scraped from listings of used cars. The dataset was divided and modified to fit the regression, to ensure the performance of the regression.

1.2 PURPOSE

The goal of a car resale value prediction system is to forecast the accurate worth of used automobiles, allowing users to sell the vehicles remotely with accurate valuation and without the need for human participation to prevent biased pricing. This resale value prediction system is made for general purpose to predict the price that can be roughly acquired by the user. The most important factors for the forecast are brand and model, period of use of the vehicle, mileage of the vehicle, gear type, fuel type used in the vehicle, as well as fuel utilisation per mile, which significantly influences cost of a vehicle because of continuous changes in the cost of fuel. The prediction of the vehicle value has been done precisely considering various characteristics as well as with the assistance of reference data.

2. LITERATURE SURVEY

2.1 EXISTING SYSTEM

The existing system involves a procedure where a vendor chooses a price arbitrarily and the buyer is unaware of the car and its current market value. In actuality, the vendor is also ignorant of the car's current value or the appropriate selling price. And then the existing systems with the machine learning approaches are made where certain features are mapped together inorder to obtain a price for resale car but it's not always feasible as it is not a real time solution. Machine learning models are implemented with only few features which makes the accuracy to be inappropriate. So, the future work would include a website or web application with enhanced features that predicts the value of a car with high accuracy.

2.2 REFERENCES

[1] Ganesh, Mukkesh & Venkatasubbu, Pattabiraman. (2019). Used Cars Price Prediction using Supervised Learning Techniques. International Journal of Engineering and Advanced Technology. 9. 216-223. 10.35940/ijeat.A1042.1291S319.

[2] Shonda Kuiper. Introduction to Multiple Regression: How Much Is Your Car Worth? Journal of Statistics Education.

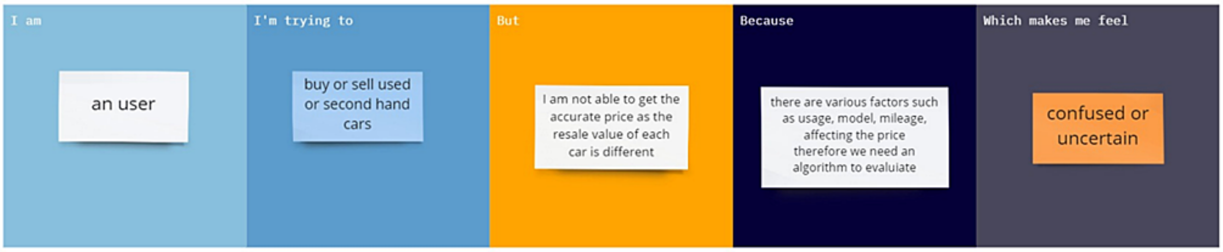
[3] Ketan Agrahari,Ayush Chaubey, Mamoor Khan & Manas Srivastava. Car Price Prediction Using Machine Learning. International Journal of Innovative Research in Technology

[4] Sameerchand Pudaruth. Predicting the Price of Used Cars using Machine Learning Techniques. International Journal of Information & Computation Technology.

[5] Chen, Chuancan ,Hao, Lulu, Xu, Cong. Comparative analysis of used car price evaluation models. AIP Conference Proceedings, Volume 1839, Issue 1, id.020165

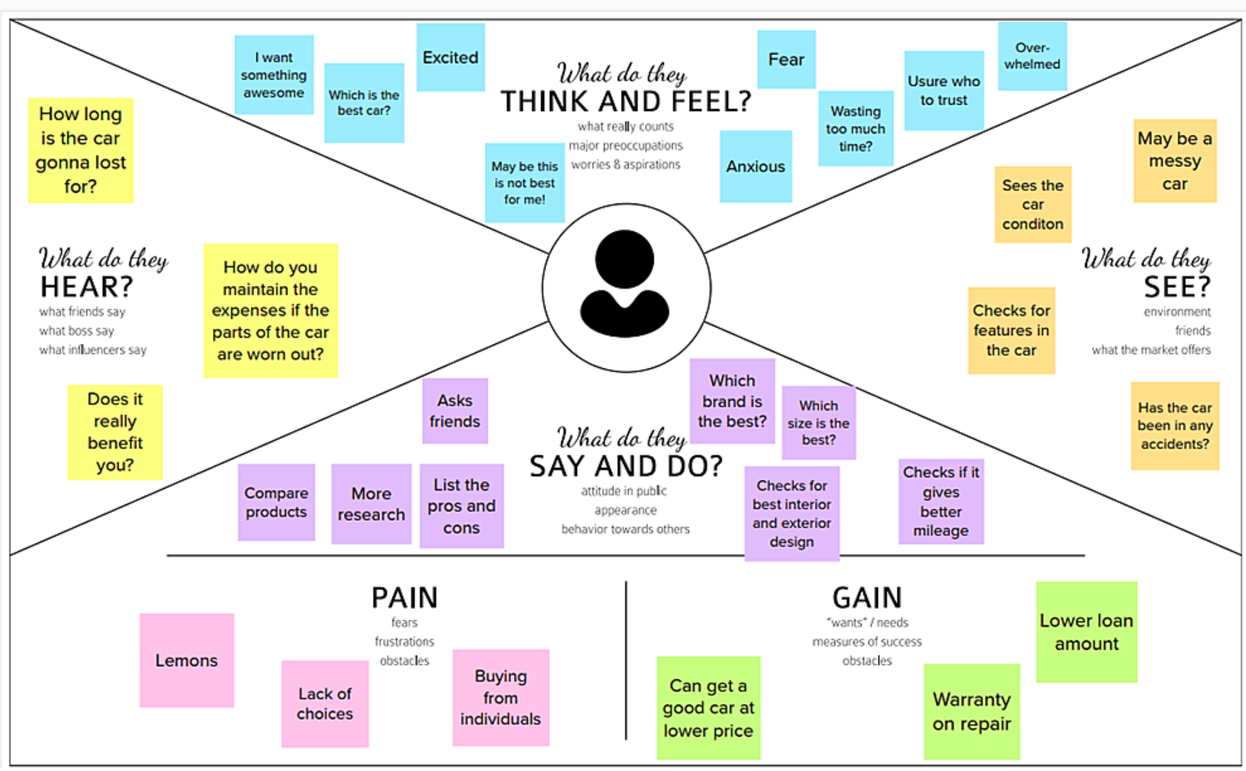
[6] Jency M. Shah, Dr. Ronak Panchal. Novel approach of Machine learning algorithms in car dataset. International Research Journal of Modernizatio

2.3 PROBLEM STATEMENT DEFINITION



3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING

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

To predict the price of used cars using machine learning models.

2

Brainstorm

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

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Malepati Ashritha

- Brand of the car
- Type of car(petrol, diesel)
- Year of manufacturing
- Budget
- Type of transmission (automatic or manual)
- Model of the car
- Mileage

Abhinayaa.K

- Name of the manufacturer
- Insurance
- AC or Non-AC
- Brake system
- Body type
- Engine
- Ownership cost

Ayyagari Mihika

- Battery system
- Registration Certificate
- Condition of the car
- Comfort check
- Maintaneance Record
- Reading of the car
- Gear type

Aishwarya

- Reading of the car
- Air bags
- Engine location
- Performance of the car
- Drive wheel
- Ownership
- Cylinder number

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, 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

- Car resale price can be predicted by using regresion technique
- Collect the details of the car
- The regression model takes sometime to validate the data using train set
- Data should be trained before processed
- The data should be tested before processed
- The result provided should be accurate

TIP

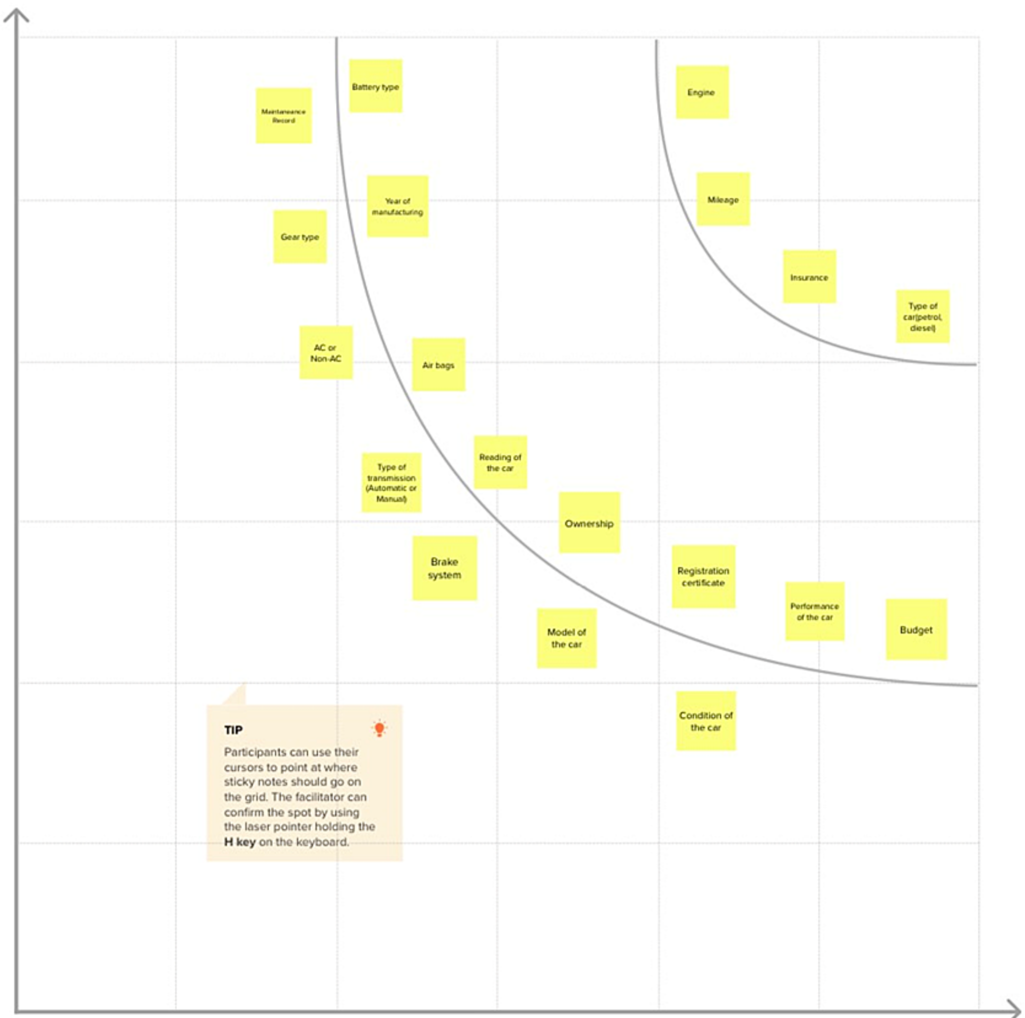
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

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



TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To predict the resale value of second hand cars and used cars. Value of a car depends on various factors such as usage, mileage, registration, model and several other factors. The problem to be solved here is to be able to predict the resale value of any used car by combining all the contributed factors.
2.	Idea / Solution description	To implement a UI design and apply Machine Learning algorithms for predicting the resale value of a car and to display it to the user.
3.	Novelty / Uniqueness	The dataset used should be clean inorder to carry out the analysis and the solution should give the highest possible accuracy.
4.	Social Impact / Customer Satisfaction	UI design is very much attractive and user friendly. Any user can use this system irrespective of their device compatibility and the prediction of car’s value will also depend on certain social factors providing the user best accurate value.

5.	Business Model (Revenue Model)	In terms of Business, since the solution is deployed in the cloud, it is accessible to everyone. Software is developed understanding the use case of Machine Learning in automotive industry to build a prediction model.
6.	Scalability of the Solution	In order to provide a scalable solution we will have a feature set with various factors contributing to the price of a car as attributes such that even with increase in the dataset the solution will be able to give an accurate prediction. And as the software is deployed in the cloud even the mobile user can access it.

3.4 PROBLEM FIT SOLUTION

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

People who are looking to buy or sell used cars.

6. CUSTOMER CONSTRAINTS CC

Lack of user-friendly, reliable and free technology, Lack of efficient algorithms, Lack of availability of secure and easy UI.

5. AVAILABLE SOLUTIONS AS

There are existing solutions that can predict the value for used cars but they are not very efficient and reliable. Also, with increase in dataset and factors algorithms might not perform well. So, the existing solutions lack the accuracy.

Explore AS, differentiate

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

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

To develop a feature set with the factors contributing to the change of price and to implement an algorithm to predict the resale value of a car.

9. PROBLEM ROOT CAUSE RC

The need for secondhand cars and the people wishing to sell their used cars. Customers have to do it in order to get a satisfactory value for the used cars.

7. BEHAVIOUR BE

Customers feel uncertain about the price and use the available technologies to get the resale value of a car.

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

Identify strong TR &EM

3. TRIGGERS

TR

Coming across the need of knowing the price of a secondhand car

4. EMOTIONS: BEFORE / AFTER EM

Before: Uncertain, worried and confused.

After: Relieved, clear and happy.

10. YOUR SOLUTION

SL

To build a reliable technology that can address all the customer needs to predict the resale value of a car with all the factors contributing to the change of value of a car ensuring efficient functioning and results.

8.CHANNELS of BEHAVIOUR

CH

8.1 **ONLINE**

Can access the website and upload the details of their car and usage to get the resale value with the current condition.

8.2 **OFFLINE**

Customers can seek into the details and condition of a car to get an approximate value.

Identify strong TR &EM

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

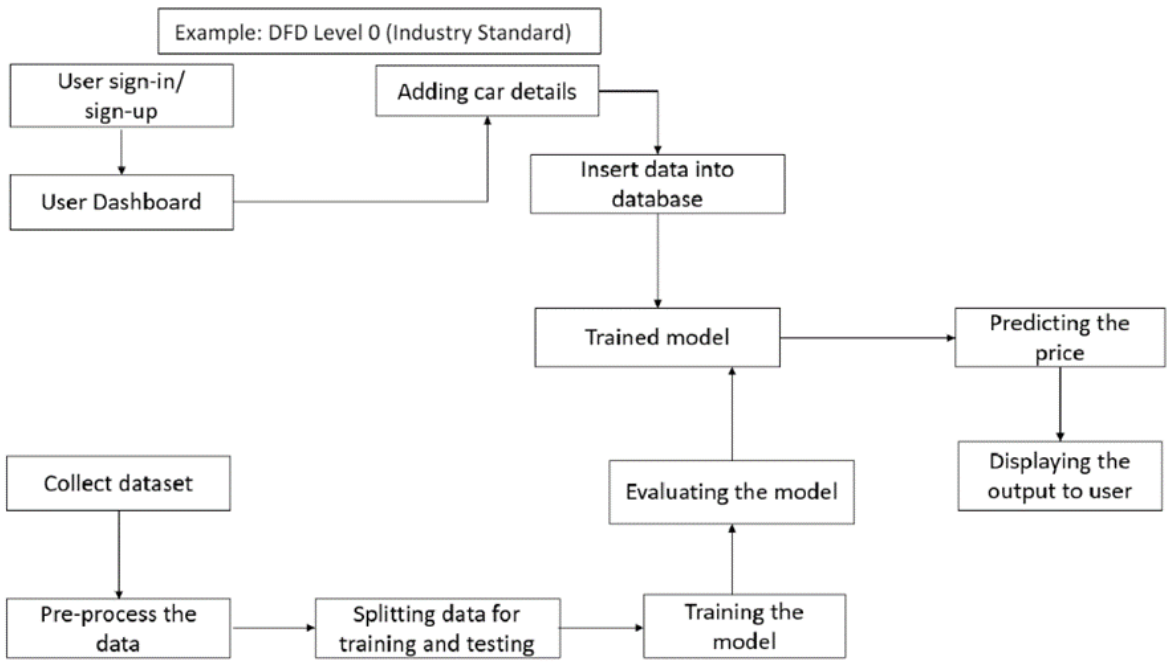
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	User can register through email with credentials such as Username, password, number.
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Availability of Car model	The user can search and select the car from variety of car models available and can compare with other cars as per their wish.
FR-4	Condition of the car	Condition of the car such as usage, mileage, model etc are examined to predict the value of the car.
FR-5	Predicting the price	After looking into all the feature sets of the car, its resale value is predicted.
FR-6	Invoice	Once the price is finalized, user can make a payment and can get the invoice along with the documents.

4.2 NON-FUNCTIONAL REQUIREMENTS

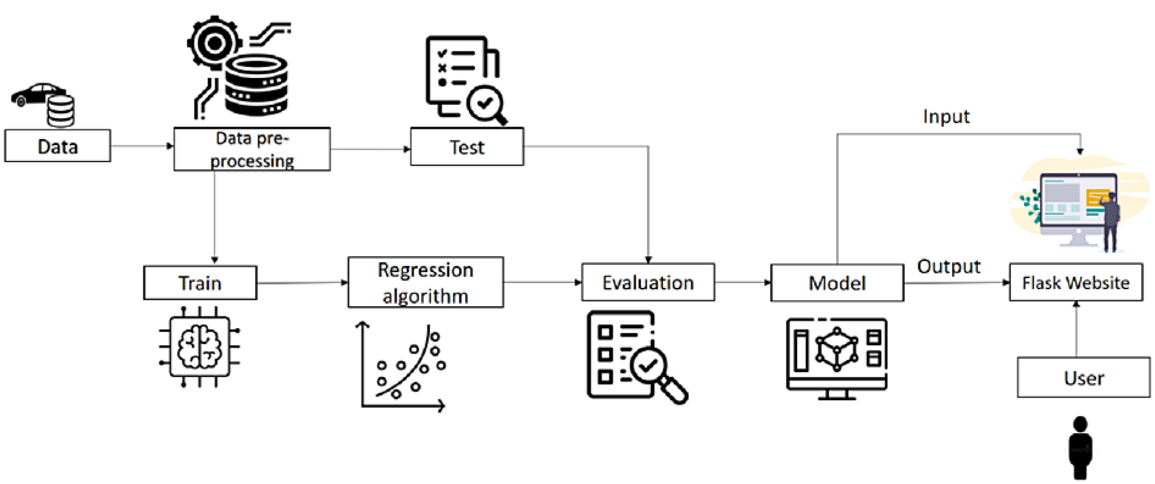
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	User Interface of the application be environment friendly providing user good interaction.
NFR-2	**Security**	User details, car registrations and the site should be securely managed, and the users should be authenticated.
NFR-3	**Reliability**	Application should be reliable in terms of its information, operations.
NFR-4	**Performance**	Application should be able to provide a quick solution, based upon the condition of the car app has to value the car and generate a invoice to the customer based on their interest in the car.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE



Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User interaction with application, Web UI	HTML, CSS, JavaScript
2.	Application Logic-1	Collection and pre-processing of data	Python
3.	Application Logic-2	Mapping the feature set and predicting the value	Python
4.	Application Logic-3	Sending invoice, updates and alerts to users	IBM Watson Assistant, STT Service
5.	Database	Data Type, Configurations provided by users	MySQL, NoSQL
6.	Cloud Database	Handles database services on Cloud	IBM DB2, IBM Cloudant
7.	File Storage	File storage requirements ensuring high performance even with large data	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	API offers the interface for the users to connect	IBM Watson API, etc.
9.	External API-2	API provides the transparency and meets the requirements for the performance	IBM Watson
10.	External API-3	Communication with both user and the application is taken care by API's	IBM API Connect etc.
11.	Machine Learning Model	Implements a feature set and predicts the value	Regression Models, Feature Differentiation Model, Object Recognition Model, etc.
12.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Kubernetes, etc

Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	A software for which the source code is made freely available and modified according to the requirement	Tensorflow, Jupiter Notebook
2.	Security Implementations	Secure monitoring of features provided by the users	IBM encryption services
3.	Scalable Architecture	Able to match the conditions provided by the user to the available feature map	Machine Learning models
4.	Availability	Usage of data and availability of information regarding cars and price can be tracked anytime	Tensorflow, API's
5.	Performance	High performance for any large data and can quickly predict the values.	API and regression models

5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer 1	Index page	USN-1	As a user, I can enter and view the home page	I can access the home page	Low	Sprint-2
	Page redirection	USN-2	As a user, I can be redirected to the data entry page from home page	I can move to the data entry page	Medium	Sprint-2
	Data entry page	USN-3	As a user, I can enter details of car in the fields	I can submit the details	High	Sprint-1
	Resale value prediction	USN-4	As a user, I can expect the predicted price	I can view the predicted the price	High	Sprint-1
Customer 2	Index page	USN-7	As a user, I can enter and view the home page	I can access the home page	Low	Sprint-2
	Page redirection	USN-8	As a user, I can be redirected to the data entry page from home page	I can move to the data entry page	Medium	Sprint-2
	Data entry page	USN-9	As a user, I can enter details of car in the fields	I can submit the details	High	Sprint-1
	Resale value prediction	USN-10	As a user, I can expect the predicted price	I can view the predicted the price	High	Sprint-1

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	Pre-process data	USN-1	Collect Dataset	1	Low	Abhinayaa K
Sprint-1		USN-2	Import required libraries	1	Low	Ayyagari Mihika
Sprint-1		USN-3	Read and clean data sets	2	Low	Malepati Ashritha
Sprint-2	Model building	USN-1	Split data into independent and dependentvariables	3	Medium	Aishwarya V
Sprint-2		USN-2	Apply using regression model	3	Medium	Abhinayaa.K
Sprint-3	Application building	USN-1	Build python flask application and HTML page	5	High	Ayyagari Mihika & Malepati Ashritha
Sprint-3		USN-2	Execute and test	5	High	Aishwarya V
Sprint-4	Training the model	USN-1	Train machine learning model	5	High	Abhinayaa.K & Aishwarya V
Sprint-4		USN-2	Integrate flask	5	High	Malepati Ashritha

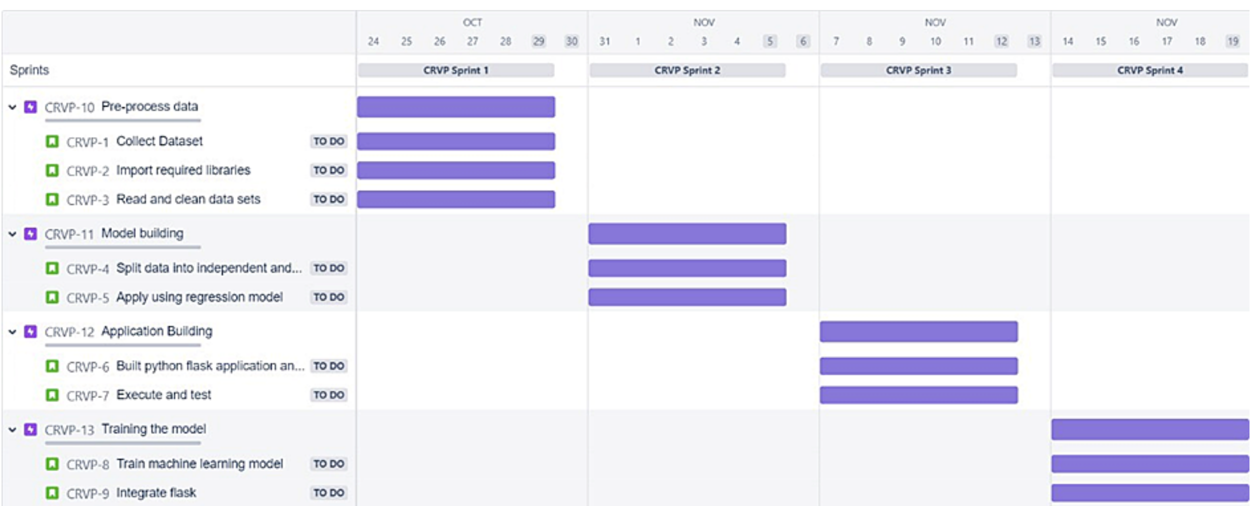
6.2 SPRINT DELIVERY & SCHEDULE

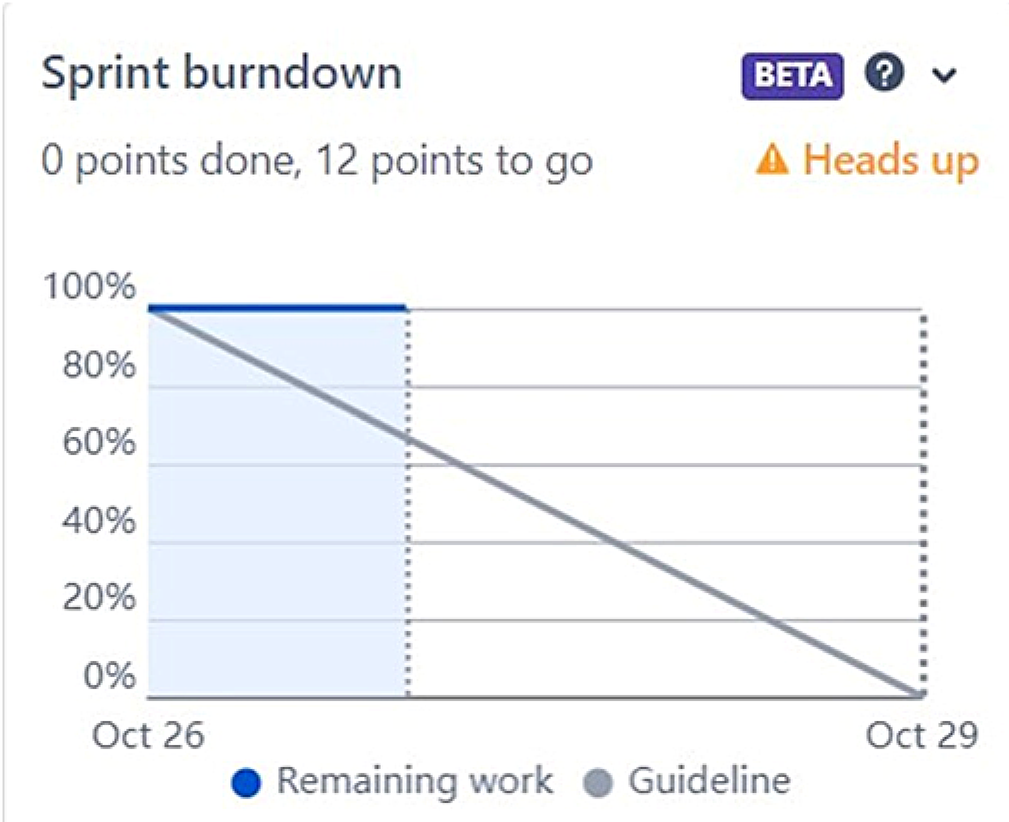
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	07 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publications etc.	15 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	16 SEPTEMBER2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	17 SEPTEMBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	23 SEPTEMBER 2022
Problem Solution Fit	Prepare problem - solution fit document.	28 SEPTEMBER 2022
Solution Architecture	Prepare solution architecture document.	30 SEPTEMBER 2022

Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	5 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	10 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	15 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	11 OCTOBER 2022
Prepare Milestone & ActivityList	Prepare the milestones & activity list of the project.	21 OCTOBER 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS

6.3 REPORTS FROM JIRA




Sprint burndown
BETA
0 points done, 12 points to go
Heads up
100%
80%
60%
40%
20%
0%
Oct 26
Oct 29
Remaining work
Guideline

7. CODING & SOLUTIONING

7.1 FEATURE 1

```
In [31]: labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']
```

```
In [32]: 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)

         gearbox : LabelEncoder()
         notRepairedDamage : LabelEncoder()
         model : LabelEncoder()
         brand : LabelEncoder()
         fuelType : LabelEncoder()
         vehicleType : LabelEncoder()
```

```
In [33]: labeled=new_df[ ['price'
                          ,'yearOfRegistration'
                          ,'powerPS'
                          ,'kilometer'
                          ,'monthOfRegistration'
                         ]
                        + [x+"_labels" for x in labels]]
```

```
In [34]: print(labeled.columns)

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

7.2 FEATURE 2

Label Encoding

```
In [48]: 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')
```

Score Evaluation

```
In [49]: def find_scores(Y_actual, Y_pred, X_train):
             mae = mean_absolute_error(Y_actual, Y_pred)
             mse = mean_squared_error(Y_actual, Y_pred)
             rmse = np.sqrt(mse)
             rmsle = np.log(rmse)
             r2 = r2_score(Y_actual, Y_pred)
             n, k = X_train.shape
             adj_r2_score = 1 - ((1-r2)*(n-1)/(n-k-1))

             wandb.log({"mae": mae, "mse": mse, 'rmse':rmse, 'rmsle':rmsle, 'r2':r2, 'adj_r2':adj_r2_score})
```

7.3 FEATURE 3

LGBM Regressor

```
In [55]: def LGBM_regressor():
    config_defaults = {
                'objective':'root_mean_squared_error',
                'reg_sqrt': True,
                'metric':'rmse',
                'random_state':42
            }
    wandb.init(config=config_defaults)
    config = wandb.config

    X = labeled.iloc[:,1:].values
    Y = labeled.iloc[:,0].values.reshape(-1,1)

    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4, random_state=42)

    model = LGBMRegressor(
      learning_rate=config.learning_rate,
      n_estimators = config.n_estimators,
      random_state = config.random_state)

    model.fit(X_train, Y_train)

    Y_pred = model.predict(X_test)

    find_scores(Y_test, Y_pred, X_train)
```

```
In [56]: lgbm_configs = {
    "name":'LGBMRegressor',
    "method": "grid",
    "metric": {
        "name": "adj_r2",
        "goal": "maximize"
    },
    "parameters": {
        "learning_rate": {
            "values": [0.01, 0.03, 0.05, 0.07]
        },
        "objective": {
```

7.4 FEATURE 4

```
@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('Result\\'+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]]
```

8. TESTING

8.1 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	Executed By
IndexPage_TC_OO1	UI	Index page	User must enter the url to navigate into a index page of car resale value prediction	Proper Internet connection/System downloaded browser	1.Enter URL and click go 2.Click on Check price	http://localhost:5000	Index page should be displayed along with the check price button	Working as expected	Pass	-	No	-	Abhinayaa K, Ayyagari Mihika, Malepati Ashritha, Aishwarya V
InputPage_TC_OO2	UI	Input Page	Verify the UI elements in input page	Proper Internet connection/System downloaded browser	1.Redirected to input page 2. Verify below UI elements: a.registration month,year b.model and brand type c. gear and fuel type d.power and kilometers driven e. damaged/repaired	http://localhost:5000	Application should show below UI elements: a.registration month,year b.model and brand type c. gear and fuel type d.power and kilometers driven e. damaged/repaired	Working as expected	Pass	-	No	-	Abhinayaa K, Ayyagari Mihika, Malepati Ashritha, Aishwarya V
InputPage_TC_OO3	UI/Functional	Input page	Verify user is able to get the value for the given instances	Proper Internet connection/System downloaded browser	Enter the details and ensure the following: 1. Choose appropriate value in the dropdowns 2. Enter numbers in the required fields 3. Choose proper option to radio buttons 4.Click on submit button	http://localhost:5000	User should be navigated to result page with a prediction	Working as expected	Pass	-	No	-	Abhinayaa K, Ayyagari Mihika, Malepati Ashritha, Aishwarya V
ResultPage_TC_OO4	Functional/Result	Final result page	Verify whether the user is able to get a predicted value for the input given	Proper Internet connection/System downloaded browser	1. Check the resale value of a car for given details	Input fields	The predict value of the car is displayed.	Working as expected	Pass	-	No	-	Abhinayaa K, Ayyagari Mihika, Malepati Ashritha, Aishwarya V

8.2 USER ACCEPTANCE TESTING

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 reportshows the numberof resolved or closed bugs at each severity level,and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	2	2	1	15
Duplicate	2	0	0	1	3
External	1	1	0	0	2
Fixed	15	2	4	5	26
Not Reproduced	0	1	0	0	1
Skipped	0	0	0	1	1
Won't Fix	1	0	2	1	4
Totals	29	6	8	9	52

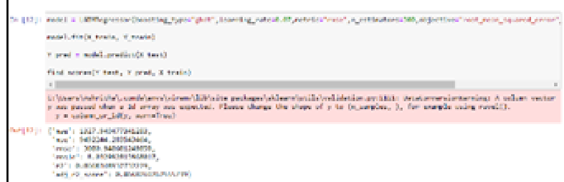
3.Test Case Analysis

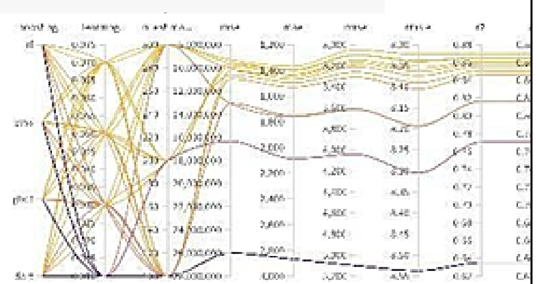
Thisreport shows the number of test casesthat have passed,failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	12	0	0	12
Security	2	0	0	2
Outsource Shipping	1	0	0	1
Exception Reporting	4	0	0	4
Final ReportOutput	6	0	0	6
Version Control	2	0	0	2

9. RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** LGBM Regressor MAE: 1327.56 MSE: 9492244.25 RMSE: 3080.93 RMSLE: 8.05 R2 Score: 0.8664 Adjusted R2 Score: 0.8666	

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 –
			
			lgbm_configs = {
			"name":'LGBMRegressor',
			"method": "grid",
			"metric": {
			"name": "adj_r2",
			"goal": "maximize"
			},
			"parameters": {
			"learning_rate": {
			"values": [0.01, 0.03, 0.05, 0.07]
			},
			"objective": {
			"values": ['root_mean_squared_error']
			},
			"boosting_type": {
			"values": ['gbdt','dart','goss','rf']
			},
			"reg_sqrt": {
			"values": [True]
			},
			"metric": {
			"values": ['rmse']
			},
			"n_estimators": {
			"values": [100,200,300]
			},
			"random_state": {
			"values": [42]
			}
			}
			}
			

		0.07 Boosting Type – ‘gbdt’ Number of Estimators - 300		

SCREENSHOTS

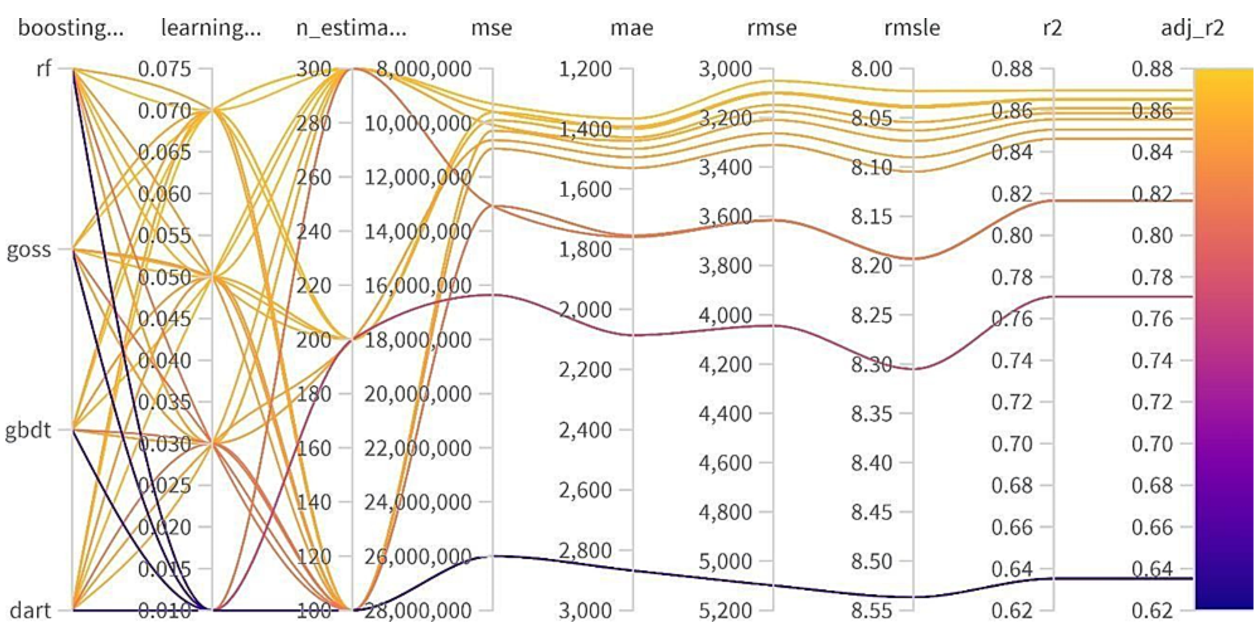
1. **Metrics**

```
In [12]: model = LGBMRegressor(boosting_type="gbdt",learning_rate=0.07,metric="rmse",n_estimators=300,objective="root_mean_squared_error",

         model.fit(X_train, Y_train)

         Y_pred = model.predict(X_test)

         find_scores(Y_test, Y_pred, X_train)

C:\Users\Ashritha\.conda\envs\virenv\lib\site-packages\sklearn\utils\validation.py:1111: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

Out[12]: {'mae': 1327.549477341283,
          'mse': 9492244.283543464,
          'rmse': 3080.948601249859,
          'rmsle': 8.032992815968017,
          'r2': 0.8668348937732229,
          'adj_r2_score': 0.8668269262555739}
```

2. Tune the model

```
lgbm_configs = {
    "name":'LGBMRegressor',
    "method": "grid",
    "metric": {
        "name": "adj_r2",
        "goal": "maximize"
    },
    "parameters": {
        "learning_rate": {
            "values": [0.01, 0.03, 0.05, 0.07]
        },
        "objective": {
            "values": ['root_mean_squared_error']
        },
        "boosting_type": {
            "values": ['gbdt','dart','goss','rf']
        },
        "reg_sqrt": {
            "values": [True]
        },
        "metric": {
            "values": ['rmse']
        },
        "n_estimators": {
            "values": [100,200,300]
        },
        "random_state": {
            "values": [42]
        }
    }
}
```

Wandb Sweep:



10. ADVANTAGES & DISADVANTAGES

Advantages:

- Able to give accurate and acceptable price for both buyer and seller.
- Have range of option on buying on budget.
- Helps in saving money than giving to brokerage.
- This system helps to reduce installation cost.
- This system is useful to sell the car for reasonable price.

Disadvantages:

- Poor checking and invalid information affect the value of prediction.
- Cars are limited usage vehicles some people only could afford this basis on knowledge-based purchasing.
- Car Resale value cannot be used by the person who does not have access to the internet.
- Very hard to use for targeted range of people.

11. CONCLUSION

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

12.FUTURE SCOPE

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

13. APPENDIX

Jupyter notebook:

```
import pandas as pd

import numpy as np

import matplotlib as plt

from sklearn.preprocessing import LabelEncoder

import pickle

df=pd.read_csv("C:\\Users\\Ashritha\\Documents\\Nalaya thiran\\Data\\autos.csv",
header=0, sep=',', encoding='Latin1',)

df.head()
```

```
df.shape

print(df.seller.value_counts())

df[df.seller != 'gewerblich']

df=df.drop('seller',axis=1)

print(df.offerType.value_counts())

df[df.offerType != 'Gesuch']

df=df.drop('offerType',axis=1)

print(df.shape)

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

print(df.shape)

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

print(df.shape)

df.drop(['name', 'abtest', 'dateCrawled', 'nrOfPictures', 'lastSeen',

        'postalCode','dateCreated'], axis='columns',inplac

e=True)

new_df = df.copy()

new_df = new_df.drop_duplicates ([ 'price', 'vehicleType', 'yearOfRegistration'

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

                              ,'notRepairedDamage'])

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', 'convertible', 'combination', 'others'), inplace=True)
new_df.notRepairedDamage.replace(('ja', 'nein'), ('Yes', 'No'),inplace=True)
new_df = new_df[(new_df.price >= 100) & (new_df.price <= 150000)]
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)
new_df.to_csv("autos_preprocessed.csv")
labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']
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)
labeled=new_df[ ['price'
             ,'yearOfRegistration'
             ,'powerPS'
             ,'kilometer'
             ,'monthOfRegistration'
            ]
```

```
              + [x+"_labels" for x in labels]]
print(labeled.columns)
Y = labeled.iloc[:,0].values
X = labeled.iloc[:,1:].values
Y=Y.reshape(-1,1)
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)
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)
def find_scores(Y_actual, Y_pred, X_train):
```

```
    mae = mean_absolute_error(Y_actual, Y_pred)
    mse = mean_squared_error(Y_actual, Y_pred)
    rmse = np.sqrt(mse)
    rmsle = np.log(rmse)
    r2 = r2_score(Y_actual, Y_pred)
    n, k = X_train.shape
    adj_r2_score = 1 - ((1-r2)*(n-1)/(n-k-1))

    wandb.log({"mae": mae, "mse": mse, 'rmse':rmse, 'rmsle':rmsle, 'r2':r2,
'adj_r2':adj_r2_score})
def LGBM_regressor():
    config_defaults = {
            'objective':'root_mean_squared_error',
            'reg_sqrt': True,
            'metric':'rmse',
            'random_state':42
        }
    wandb.init(config=config_defaults)
    config = wandb.config

    X = labeled.iloc[:,1:].values
    Y = labeled.iloc[:,0].values.reshape(-1,1)

    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4,
```

```
random_state=42)

    model = LGBMRegressor(
      learning_rate=config.learning_rate,
      n_estimators = config.n_estimators,
      random_state = config.random_state)

    model.fit(X_train, Y_train)

    Y_pred = model.predict(X_test)

    find_scores(Y_test, Y_pred, X_train)
lgbm_configs = {
    "name":'LGBMRegressor',
    "method": "grid",
    "metric": {
        "name": "adj_r2",
        "goal": "maximize"
    },
    "parameters": {
        "learning_rate": {
            "values": [0.01, 0.03, 0.05, 0.07]
        },
        "objective": {
```

```
            "values": ['root_mean_squared_error']
        },
        "boosting_type": {
            "values": ['gbdt','dart','goss','rf']
        },
        "reg_sqrt": {
            "values": [True]
        },
        "metric": {
            "values": ['rmse']
        },
        "n_estimators": {
            "values": [100,200,300]
        },
        "random_state": {
            "values": [42]
        }
    }
}

sweep_id = wandb.sweep(sweep=lgbm_configs, project="car_resale_value")
wandb.agent(sweep_id=sweep_id, function=LGBM_regressor)
def find_scores(Y_actual, Y_pred, X_train):
    scores = dict()
```

```
    mae = mean_absolute_error(Y_actual, Y_pred)
    mse = mean_squared_error(Y_actual, Y_pred)
    rmse = np.sqrt(mse)
    rmsle = np.log(rmse)
    r2 = r2_score(Y_actual, Y_pred)
    n, k = X_train.shape
    adj_r2_score = 1 - ((1-r2)*(n-1)/(n-k-1))

    scores['mae']=mae
    scores['mse']=mse
    scores['rmse']=rmse
    scores['rmsle']=rmsle
    scores['r2']=r2
    scores['adj_r2_score']=adj_r2_score

    return scores
X = labeled.iloc[:,1:].values
Y = labeled.iloc[:,0].values.reshape(-1,1)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4,
random_state=42)
model =
LGBMRegressor(boosting_type="gbdt",learning_rate=0.07,metric="rmse",n_estim
ators=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)
pickle.dump(model, open('resale_model1.sav', 'wb'))
```

INTEGRATE_FLASK:

```
# Import Libraries
import pickle

import numpy as np
import pandas as pd
import requests
from sklearn.preprocessing import LabelEncoder

from flask import Flask, Response, render_template, request

# NOTE: you must manually set API_KEY below using information retrieved
from your IBM Cloud account.
API_KEY = "MIfDRZYQhDHWH7dNHo2oQrSY2ajDfwJGV8PLQI9NIX36"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

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


app = Flask(__name__)#initiate flask app
```

```
def load_model(file='C:\\Users\\Ashritha\\Documents\\Nalaya thiran\\Car resale
pred\\resale_model1.sav'):#load the saved model
    return pickle.load(open(file, 'rb'))

@app.route('/')
def index():#main page
    return render_template('car.html')

@app.route('/predict_page')
def predict_page():#predicting 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('Result\\'+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)

    # NOTE: manually define and pass the array(s) of values to be scored in the
next line
    payload_scoring = {"input_data": [{"field": [['yearOfReg', 'powerPS',
'kilometer', 'monthOfRegistration','gearbox_labels', 'notRepairedDamage_labels',
'model_labels','brand_labels', 'fuelType_labels', 'vehicletype_labels']], "values":
X}]}
    #payload_scoring = {"input_data": [{"fields": [array_of_input_fields],
"values": [array_of_values_to_be_scored,
```

```
another_array_of_values_to_be_scored]}]}

        response_scoring = requests.post('https://eu-de.ml.cloud.ibm.com/ml/v4/deployments/99a4f93d-9a11-4878-95ed-d5395db2f283/predictions?version=2022-11-16', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
        predictions = response_scoring.json()
        print(response_scoring.json())
        predict = predictions['predictions'][0]['values'][0][0]
        print("Final prediction :",predict)

        return render_template('predict.html',predict=predict)

if __name__=='__main__':
        reg_model = load_model()#load the saved model
        app.run(host='localhost', debug=True, threaded=False)
```

Github Repo:

https://github.com/IBM-EPBL/IBM-Project-39392-1660410627

Video link:

https://drive.google.com/file/d/14U2NEaGsgtFGJ2vpry_CwnGpoavxiLEL/view?usp=share_link