Project Title: **Used car Price Prediction System**

Student Name: **Riya Shrestha(BCU)**

Student ID: **24128448**

Module Code: **CMP5366**

Module Title: **Data Management and Machine Learning Operations**

Co-ordinator Name: Rupak Koirala Sir

Date: **May 23 2025**

Table of Content

Table of Tables

Tables of Figures

Abstract- This report explains how a system was built to predict used car prices accurately. The project started by studying data from a real-world Used Car Prices dataset on Kaggle. A strong MLOps pipeline was created to handle the data, which included storing it in a database, checking data quality, cleaning and preparing it, and building a prediction model using XGBoost. Tools like Apache Airflow, Docker, Redis, and MLflow helped manage the workflow and track experiments. The final model was made available through an API with FastAPI. The report also looks at important legal and ethical issues like privacy and fairness. In the end, the project produced a working price prediction system and a clear process showing good practices in data and machine learning management.

Introduction

Predicting the price of a used car is a common problem faced by many buyers and sellers. Prices depend on several factors like brand, model year, mileage, fuel type, and accident history. Manually deciding a fair price can be difficult, so using data and machine learning makes this process more accurate and easier.

This project's main goal is to build a predictive regression model to estimate car prices based on these features. This is a problem where the target variable is the car's price. Similar approaches in the past have used machine learning models such as Random Forest and XGBoost with good results. This project not only helps in learning about ml but also shows how it can solve real-life problems in the automotive and marketing industries.

Source Data Analysis and Selection

Dataset 1: *Utah Real Estate Data*

This Dataset represents 4440 property listings from Utah with 14 columns, collected from Realtor.com using Apify's API as obtained via Kaggle where I found this dataset. While the originally meant for property sales created for education and analytical use.

Fig 1. Dataset 1

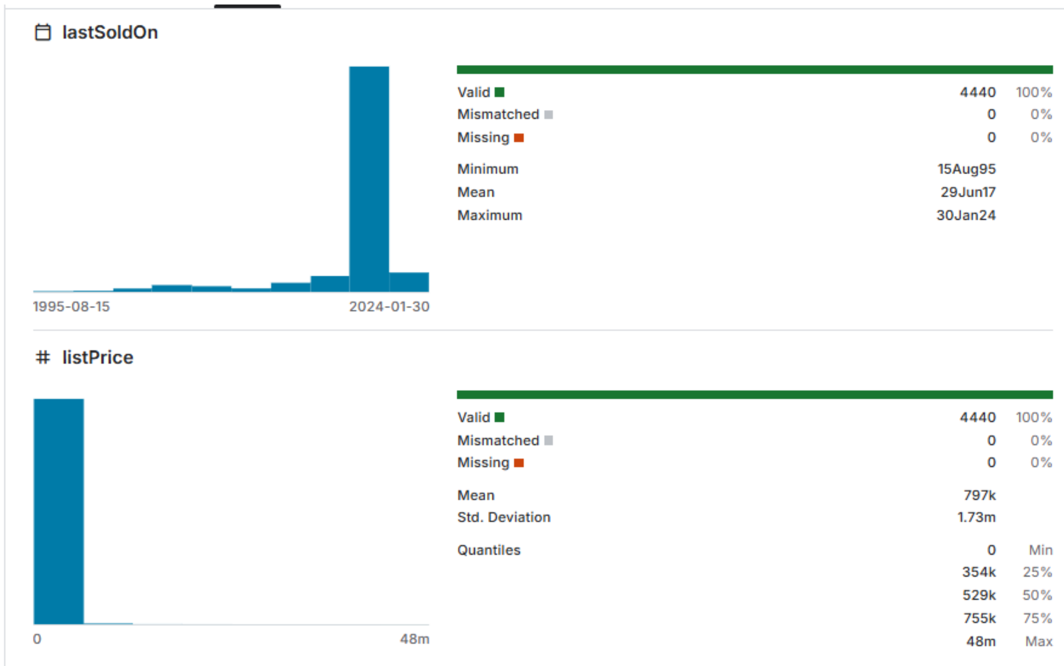


Fig 2. Dataset target variable description

It's structure is simple table flat file, currently stored as downloadable file in CSV format residing physically on system file once downloaded.

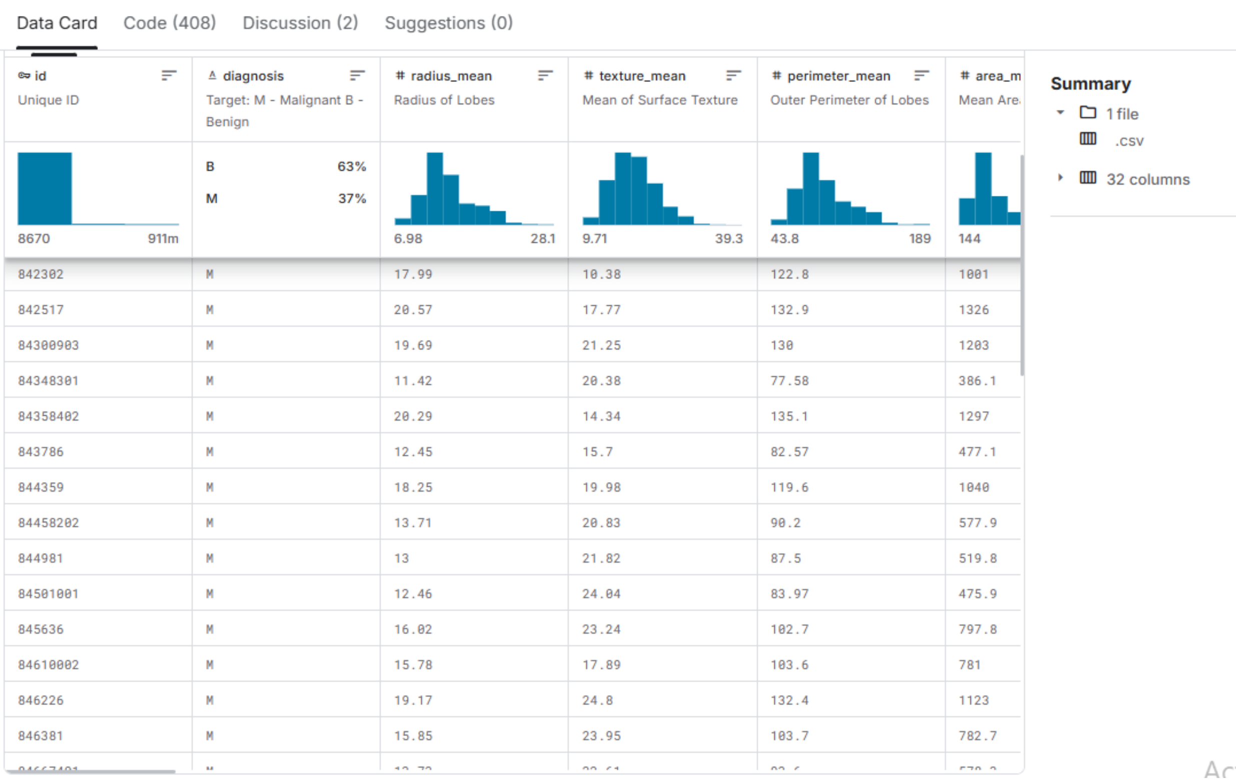
Feature	Feature Description	Data Type
type	Type of property (single-family,land)	String
text	Description of property	String
Year_built	Year the property was built	Integer
beds	Number of bedrooms	Integer
baths	Number of bathrooms	Integer
baths_full	Number of Full bathrooms	Integer
Baths_half	Number of Half bathrooms	Integer
garage	Number of garage sizes	Integer
lot_sqft	Lot size in square feet	Integer
sqft	Property size in square feet	Integer
stories	Number of stories	Integer
lastSoldOn	Date the property was last sold on	Date / String (#####)
listPrice	Listing price of the property	Integer
status	Current status of the property	

Table 1: Feature Description Table of Utah Real State

This dataset is suitable for developing supervised ml models for regression task listPrice column as a target variable remaining as predictor variables . The ListPrice values act as “Ground Truth” reflecting seller listed price from realtor.com. The dataset is clean, has no missing values.

Dataset 2: *Breast Cancer Prediction*

This dataset consists of 569 records with 32 features that describes cell nuclei, originated from the university of Wisconsin hospitals from actual patient to support breast cancer diagnosis by classifying breast tumors as cancerous or not based on cell features. Originally in UCI repository but I obtained this from kaggle.



id Unique ID	diagnosis Target: M - Malignant B - Benign	radius_mean Radius of Lobes	texture_mean Mean of Surface Texture	perimeter_mean Outer Perimeter of Lobes	area_m Mean Are
8670 – 911m		6.98 – 28.1	9.71 – 39.3	43.8 – 189	144
842302	M	17.99	10.38	122.8	1001
842517	M	20.57	17.77	132.9	1326
84300903	M	19.69	21.25	130	1203
84348301	M	11.42	20.38	77.58	386.1
84358402	M	20.29	14.34	135.1	1297
843786	M	12.45	15.7	82.57	477.1
844359	M	18.25	19.98	119.6	1040
84458202	M	13.71	20.83	90.2	577.9
844981	M	13	21.82	87.5	519.8
84501001	M	12.46	24.04	83.97	475.9
845636	M	16.02	23.24	102.7	797.8
84610002	M	15.78	17.89	103.6	781
846226	M	19.17	24.8	132.4	1123
846381	M	15.85	23.95	103.7	782.7

Fig 3. Dataset 2

The dataset structured ada flat CSV file, each row represents a sample and 32 columns as cell features, available for download from kaggle / UCI Repository.

Feature	**Feature Description**	**Data Type**
ID	Unique identifier number for each sample.	Integer
Diagnosis	The target variable; indicates if the tumor is Malignant (M) or Benign (B) .	String
Radius_mean	Average distance from the center to points on the perimeter of the cell nuclei.	Integer
Texture_mean	Average of the standard deviation of gray-scale values in the cell nuclei.	Integer

Perimeter_mean	Average perimeter of the cell nuclei.	Integer
Area_mean	Average area of the cell nuclei.	Integer
Smoothness_mean	Average of the local variation in radius lengths of the cell nuclei.	Integer
Compactness_mean	Average compactness (perimeter^2 / area - 1.0) of the cell nuclei.	Integer
Concavity_mean	Average severity of concave portions of the contour of the cell nuclei.	Integer
Concave Points_mean	Average number of concave portions of the contour of the cell nuclei.	Integer
Symmetry_mean	Average symmetry of the cell nuclei.	Integer
Fractal_dimension_mean	Average "coastline approximation" fractal dimension of the cell nuclei.	Integer
Radius_se	Standard error of the radius measurement.	Integer
Texture_se	Standard error of the texture measurement.	Integer
Perimeter_se	Standard error of the perimeter measurement.	Integer
Area_se	Standard error of the area measurement.	Integer
Smoothness_se	Standard error of the smoothness measurement.	Integer
Compactness_se	Standard error of the compactness measurement.	Integer
Concavity_se	Standard error of the concavity measurement	Inteer
Concave points_se	Standard error of the concave points measurement.	Integer
Symmetry_se	Standard error of the symmetry measurement.	Integer
Fractual_dimension_se	Standard error of the fractal dimension measurement.	Integer
Radius_worst	Mean of the three largest radius values found in the image.	Integer
Texture_worst	Mean of the three largest texture values found in the image	Integer
Perimeter_worst	Mean of the three largest perimeter values found in the image.	Integer
Area_worst	Mean of the three largest area values found in the image.	Integer
Smoothness_worst	Mean of the three largest smoothness values found in the image.	Integer
Compactness_worst	Mean of the three largest compactness values found in the image.	Integer
Concavity_worst	Mean of the three largest concavity values found in the image.	Integer
Concave points_worst	Mean of the three largest concave points values found in the image.	Integer
Symmetry_worst	Mean of the three largest symmetry values found in the image.	Integer
Fractual_dimesnsion_worst	Mean of the three largest fractal dimension values found in the image.	Integer

Table 2: Feature Description Table of Breast Cancer

This dataset is ideal for binary classification task, with target variable is diagnosis (Maligant = M) & (Benign = B). The “Ground Truth” as classification of the tumor is verified through medical diagnosis & biopsy results in university.

Though it contains only 569 rows & no missing values, it is commonly used in ML projects for cancer classification tasks, providing real-life problem.

Dataset 3: *Used Car Prices*

This dataset represents information about used cars, collected from the automotive marketplace website cars.com , with various attributes to predict the price of used vehicles. It was scraped and compiled by the creator, making it available on Kaggle for analysis, research & for buyers to make informed decisions. The dataset was chosen from Kaggle itself.

This dataset stored as a single table flat file within CSV file named used_cars.csv. Each row represents the car and columns represents features, currently stored as downloadable CSV fileswhich resides physically on the user's local system.

Features	Feature Description
ID	Unique identifier for each car listing in the dataset.
Brand	The manufacturer of the car (e.g., Toyota, Ford, BMW).
Model	The specific model name of the car within the brand (e.g.,M4 Base, F-150, A8 L 55).
Model_year	The designated model year of the vehicle (e.g., 2018, 2020).
Milage	The total distance the car has been driven.
Fuel_type	The type of fuel the car's engine uses (e.g., Gasoline, Diesel, Electric, Hybrid).
Engine	Engine specifications, often including horsepower (HP) and sometimes other details like displacement or codes (e.g., 172.0HP 1., 2.7L V6 24).
Transmission	Type of transmission, often indicating Automatic (A/T) and number of speeds (e.g., A/T, 7-Speed A, 10-Speed).
Ext_col	The exterior color of the vehicle.
Int_col	The interior color of the vehicle.
Accident	Indicator of whether the car has a reported accident history .
Clean_title	Indicator of whether the car possesses a "clean" title, meaning no major negative statuses like salvage, flood damage, etc., are officially recorded .
Price	The target variable; the listing price of the used car in dollar.

Table 3: Feature Description Table of Used car price

This dataset is well suited for supervised regression task price as target variable and 12 other columns as predictors. The listed price serves as the "Ground Truth" though it is not independently verified reflecting real market conditions. Due to its origin from web scraping, it highly contains inconsistencies and missing values presenting realistic challenge in ML tasks.

	0
brand	0
model	0
model_year	0
milage	0
fuel_type	170
engine	0
transmission	0
ext_col	0
int_col	0
accident	113
clean_title	596
price	0

dtype: int64

```
Data types of columns:
brand           object
model           object
model_year       int64
milage          object
fuel_type       object
engine          object
transmission    object
ext_col         object
int_col         object
accident        object
clean_title     object
price           object
dtype: object
```

Fig 4. Dataset Missing values & Data types

	brand	model	model_year	milage	fuel_type	engine	transmission	ext_col	int_col	accident	clean_title	price
0	Ford	Utility Police Interceptor Base	2013	51,000 mi.	E85 Flex Fuel	300.0HP 3.7L V6 Cylinder Engine Flex Fuel Capa...	6-Speed A/T	Black	Black	At least 1 accident or damage reported	Yes	$10,300
1	Hyundai	Palisade SEL	2021	34,742 mi.	Gasoline	3.8L V6 24V GDI DOHC	8-Speed Automatic	Moonlight Cloud	Gray	At least 1 accident or damage reported	Yes	$38,005
2	Lexus	RX 350 RX 350	2022	22,372 mi.	Gasoline	3.5 Liter DOHC	Automatic	Blue	Black	None reported	NaN	$54,598
3	INFINITI	Q50 Hybrid Sport	2015	88,900 mi.	Hybrid	354.0HP 3.5L V6 Cylinder Engine Gas/Electric H...	7-Speed A/T	Black	Black	None reported	Yes	$15,500
4	Audi	Q3 45 S line Premium Plus	2021	9,835 mi.	Gasoline	2.0L I4 16V GDI DOHC Turbo	8-Speed Automatic	Glacier White Metallic	Black	None reported	NaN	$34,999

Fig 5. Dataset 3

5. Dataset Selection and Justification

I chose Used Car Prices dataset for it's realistic and practical applicability in predicting second-hand vehicles and suffiicent size(4009 rows).Compared to Breast Cancer dataset(569 rows, 0 missing values, 30 features) and Utah Real Estate (4440 entries, 0 missing values, 14 features), It aligns better with the used car dataset (12 features, many missing values), making it a practical and ethical choice for end-to-end process of handlin, processing, and modeling imperfect data.

Data Storage Strategy

I choose star schema for my time series dataset which will be made in a MariaDB database called usedcar, runs inside a Docker container. This schema has one central fact table and five dimension tables to help remove duplicate data and make storage more efficient as shown in figure.

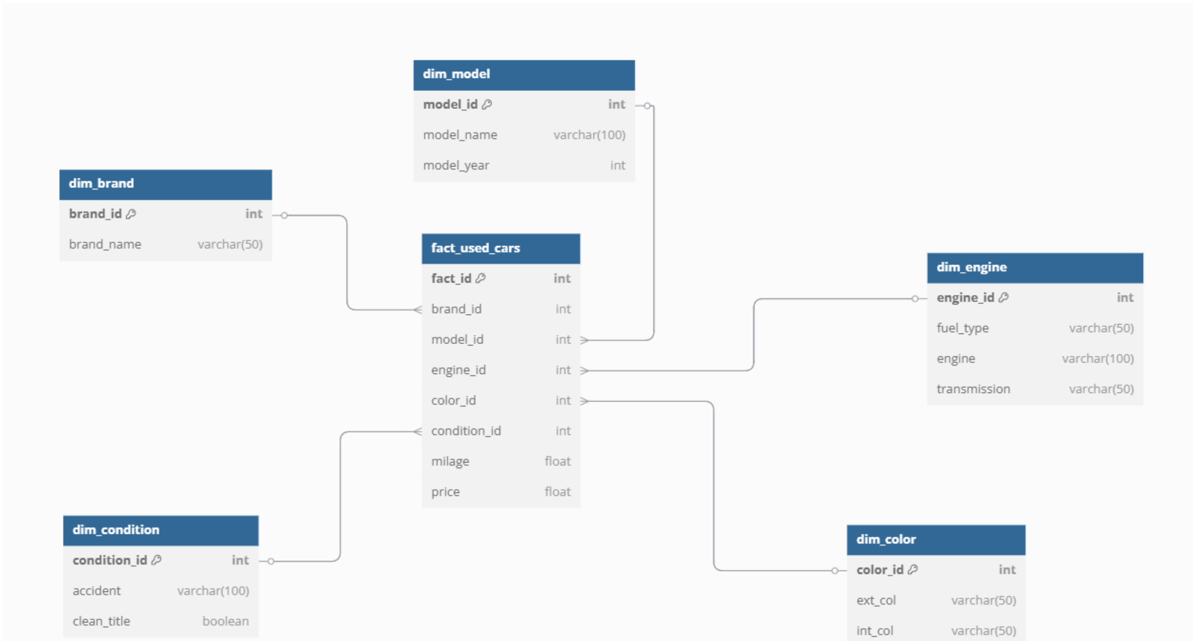


Fig 6: Logical Star Schema diagram

used_cars	
string	brand
Object	model
int	model_year
Object	milage
Object	fuel_type
Object	engine
Object	transmission
Object	ext_col
Object	int_col
Object	accident
Object	clean_title
Object	price

Fig 7. Used Car Price Dataset Datatype

However, since machine learning models work better with flat data, the data will be denormalized into one big table after the ETL process where missing values were imputed.

Table

Aspects	Star Schema	One Big Table	Snowflake Schema
Structure	Central fact table with multiple dimension tables (brand, model, year, mileage, etc.)	All data combined in a single large table with all features	Snowflake adds more normalization layers, increasing complexity
Data Redundancy	Low redundancy due to normalization	High redundancy, data duplicated across rows	Snowflake reduces redundancy further but is more complex
Query Performance	Faster for queries that aggregate or filter by dimensions	Faster for machine learning models needing all data in one row	Snowflake slower than star schema due to more joins
Maintenance	Easier to maintain and update dimension tables	Simpler structure but harder to maintain data consistency	Complex maintenance due to multiple layers

Table 4. Logical Schema Comparison Table

After preprocessing, the clean data will be saved in Redis, which is a fast in-memory database. This makes it easier to use the data in Great Expectations for validation and during model training. The trained model will also be stored in mlflow so it can be quickly used later for monitoring and predictions.

Final MLOps Pipeline Plan

This section details the data analytics pipeline of Usedcar price predictions. The followings below are procedures for pipelines;

1.Pipeline Overview

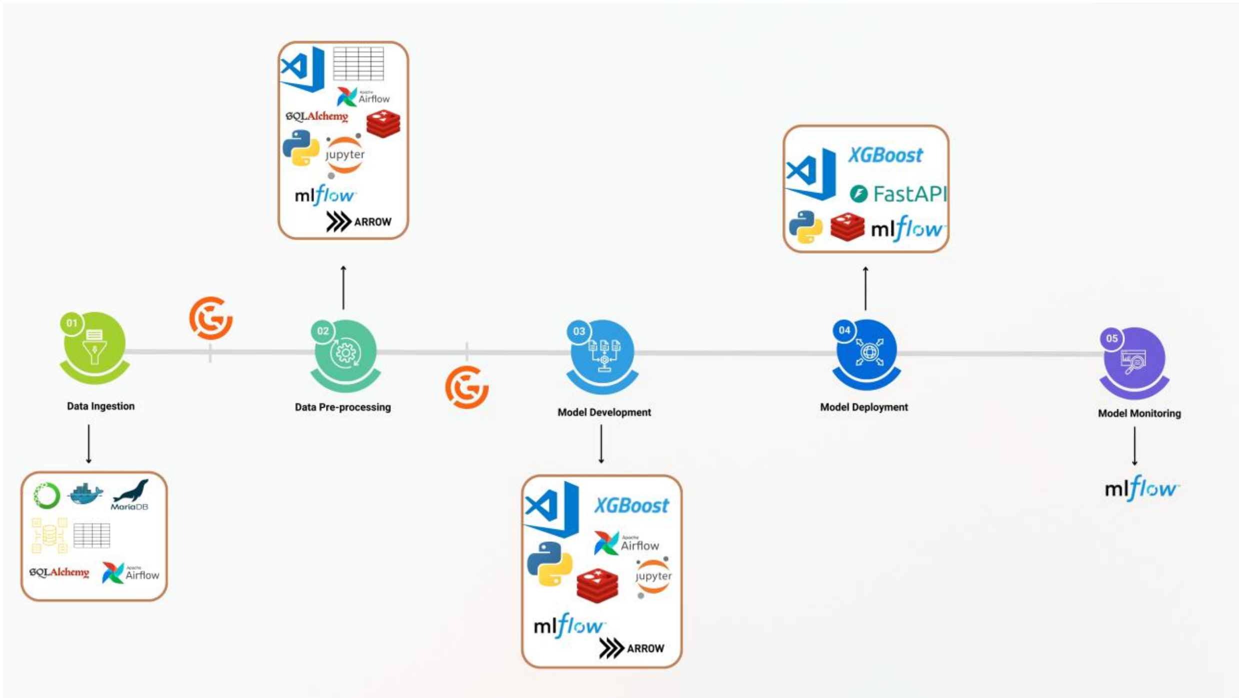


Fig 8. Usedcar Highlevel Pipeline

The pipeline begins with data ingestion from CSV to MariaDB Table, followed by preprocessing and validation using Great Expectations. The preprocessed data is then modeled, monitored and logged for quality assurance, ensuring reliable and consistent data throughout the pipeline.

2. Data Ingestion

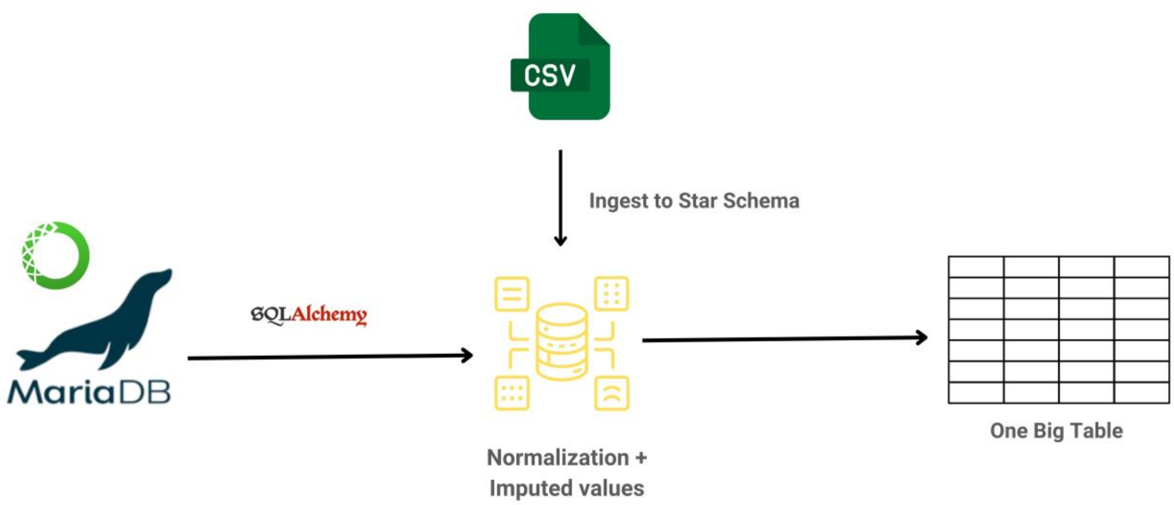


Fig 9.Data Ingestion Process

The used_cars.csv file is loaded into a MariaDB database inside star schema table, normalizing it and again, denormalized to one Big Table ready for validation and preprocessing as shown in figure. Docker along with Anaconda ensure consistent execution of ingestion in controlled environment inside container. Data engineers manage this process for smooth running, while data scientists, analysts, and ML engineer**s** benefit from its outputs for building models. Tools Include : Anaconda, Docker, MariaDB, Python, SQL Alchemy.

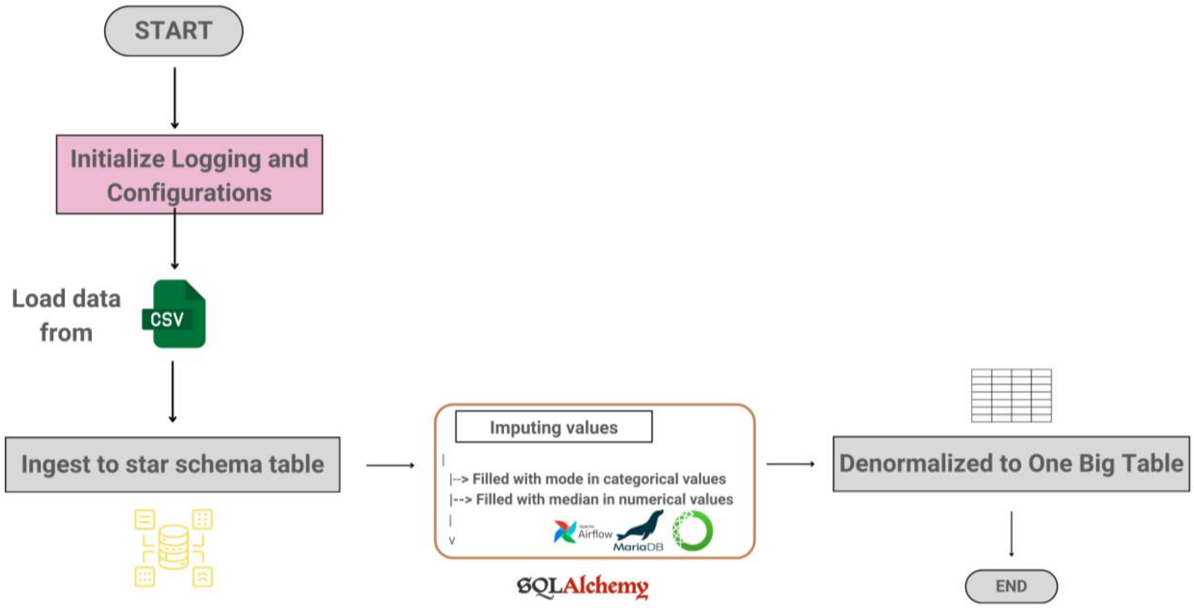


Fig 10. Ingestion Code Execution Process

3. Great Expectation Before

In this step, Great expectation and pandas are used to validate key column to meet the requirement of data quality need for better prediction model like car age should be between 1970-2024, price, brand column has no null values, millages has mi. at the end. This stage is essential to catch data quality issues early, which prevents problems during pre-processing or model training. Data engineers typically handle this step. Tools Include: MariaDB, Pandas, Great Expectations.

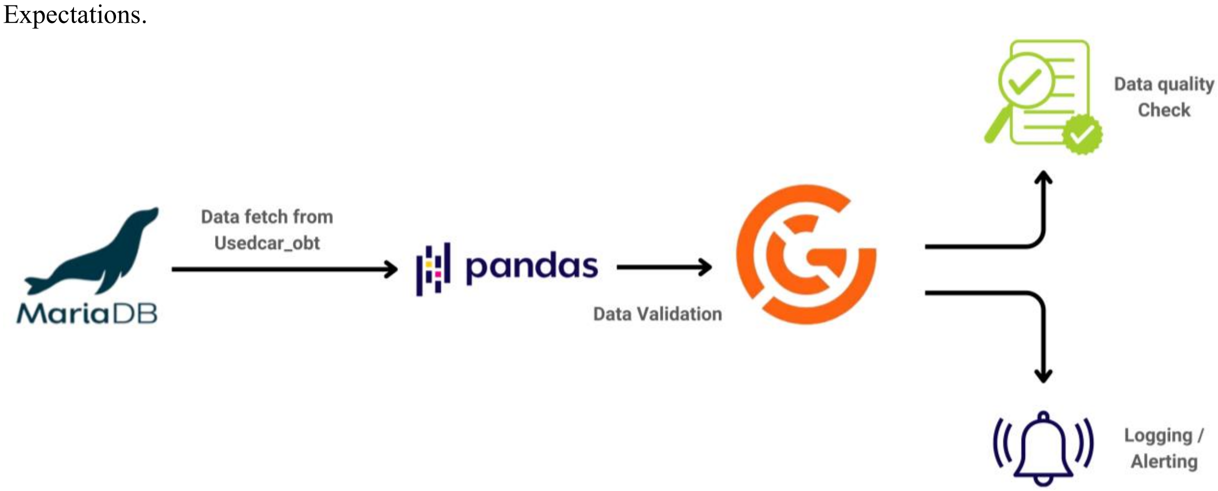


Fig 11. Great Expectation using raw data

4. Data Pre-processing

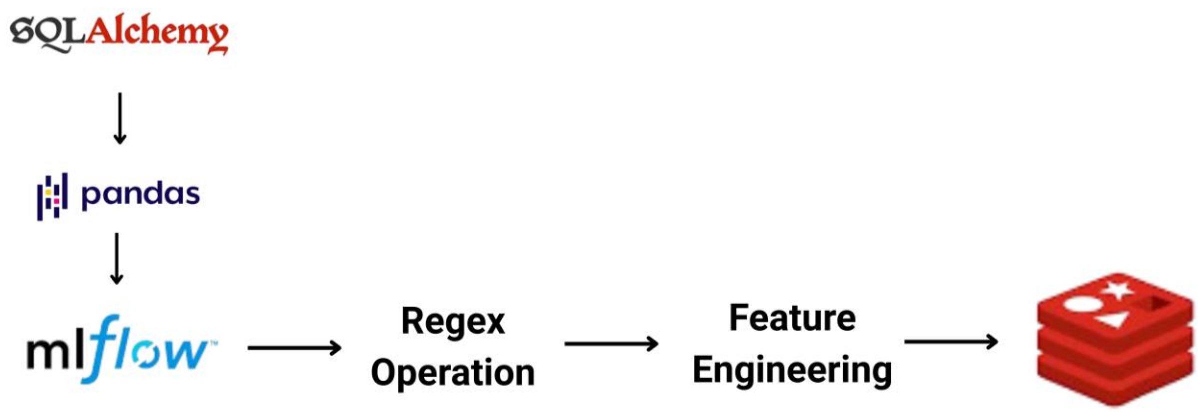


Fig 12. Preprocessing Process

Since missing values are imputed during, normalization, this process removes incorrect datatypes, unwanted characters like mi. , HP, $ & categorical value are encoded into numerical values. Feature engineering would be involved calculating car's age . This Step is cruical for better model performance and analysis. The cleaned data is stored in redis. Data Scientists designs logic, handle features, and choose cleaning methods; Data Engineers help automate and improve the process. Tools Include: Pandas, Redis, MLflow, SQLAlchemy, Python, Pyarrow,Sklearn.

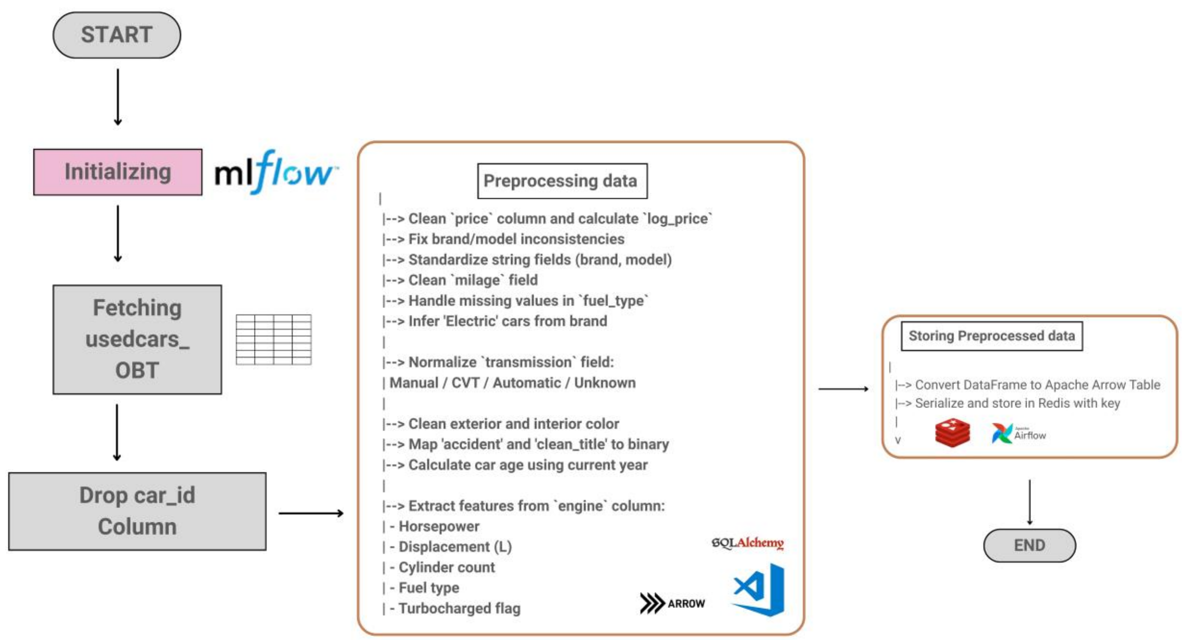


Fig 13.Data Preprocessing Code Process

5. Great Expectation after

This process again performs the validation same way but with processed data stored in redis ensuring data has been cleaned appropriately for improving model performance and feature selection. Data Scientists typically handle this step. Tools Include: Redis , Pandas, Great Expectations.

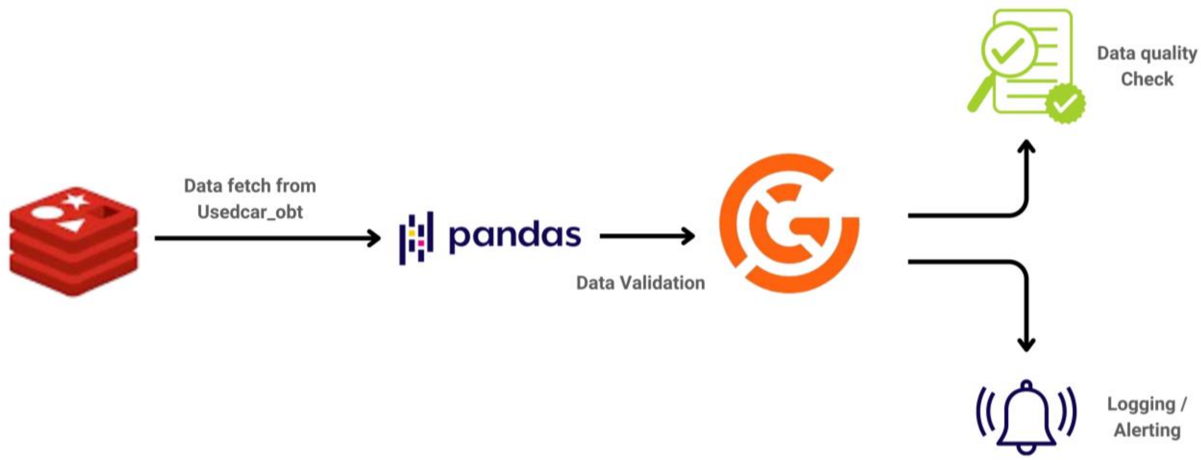


Fig 14. GE after Preprocessing

6. Model Development

Fig 15. Model training process

This stage involves using pre-processed data stored in redis to train, optimize, evaluate the regression model for predicting car price as shown in fig15/16. The model is retrained and registered in MLflow for future deployment.

Data scientists handle model building, tuning, and explainability, with ML engineers supporting infrastructure and tracking. The deployment team, business stakeholders, and analysts benefit from accurate predictions. Tools Include: Redis, Pyarrow, Sklearn, Python, XGBoost, Mlflow.

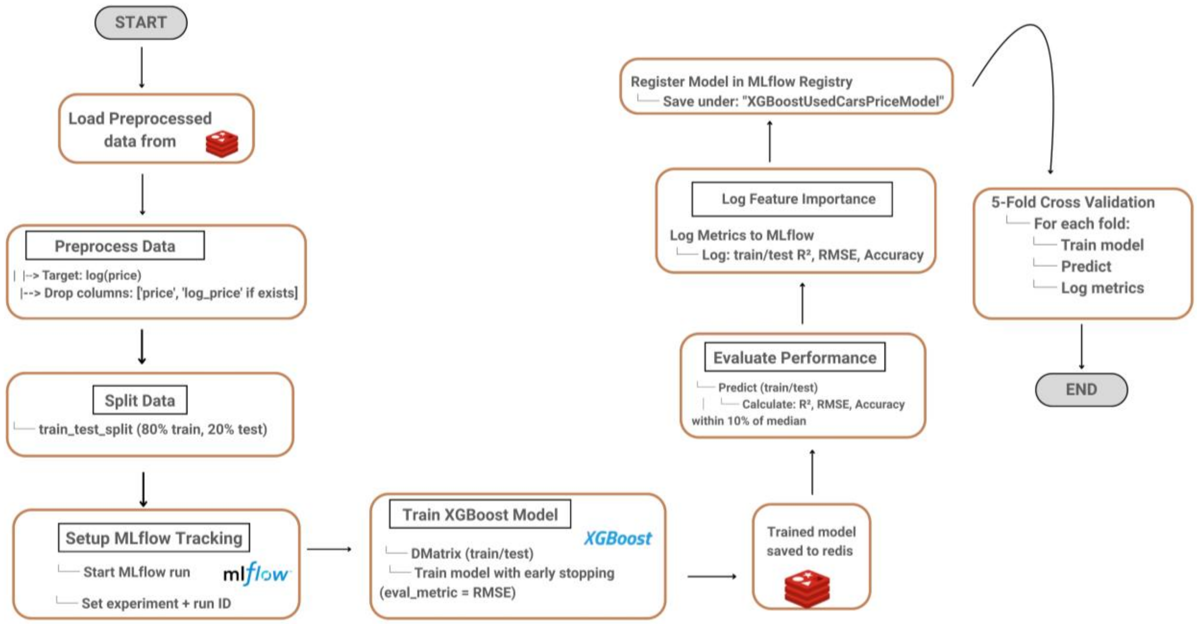


Fig 16.Model development Code Process

7. Model Deployment

Fig 17. Model deployment process

This process takes the model saved in mlflow to operate using Fast API so it can serve as prediction system. The API will be hosted on a server or cloud for users. ML Engineers and DevOps will work together to manage deployment and infrastructure. Tools Include: FastAPI, MLFlow, Python, XGBoost.

8. Model Monitoring

After deployment the model's performance will be monitored and tracked in mlflow to ensure its performance if any changes in data patterns or market trends affect predictions. Monitoring this ensures the model is accurate & responsive, with timely retraining if needed.

Initial Pipeline Implementation

1.Setup and Configuration

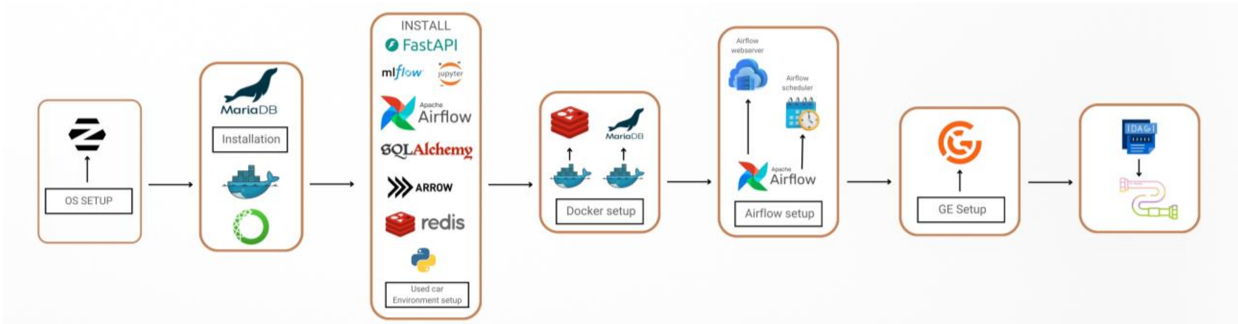


Fig 18.Server Setup

1.1 Linux

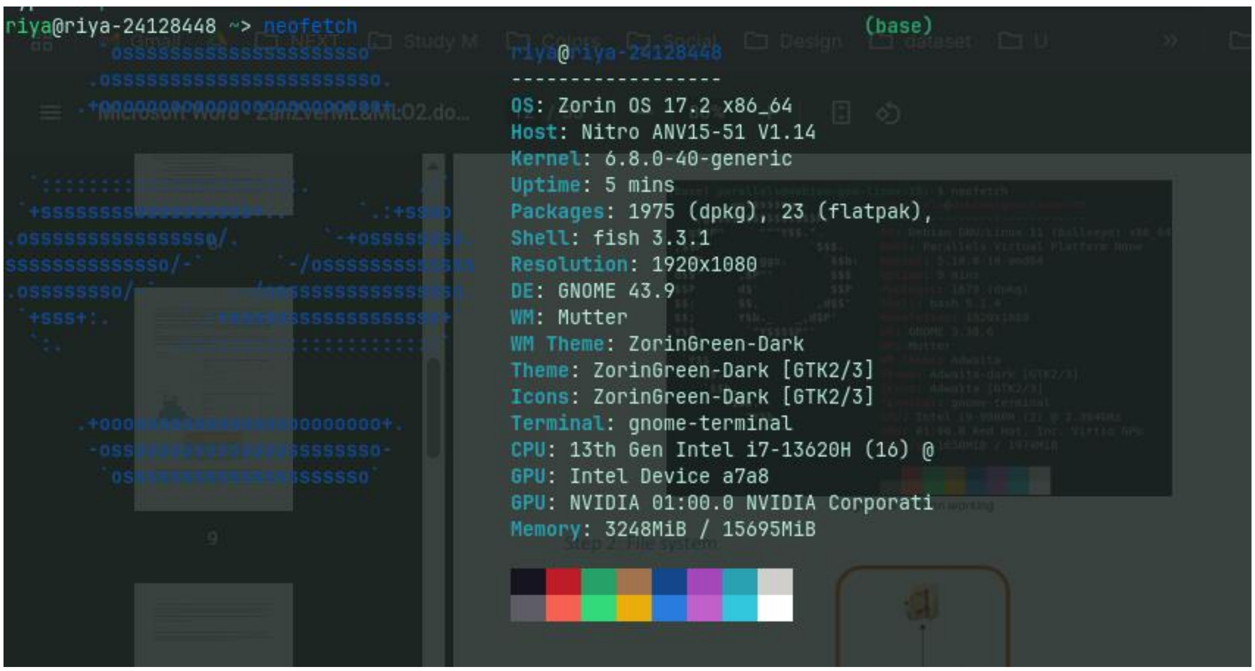


Fig 19.Zorin Setup

1.2 Docker / MariaDB / Anaconda

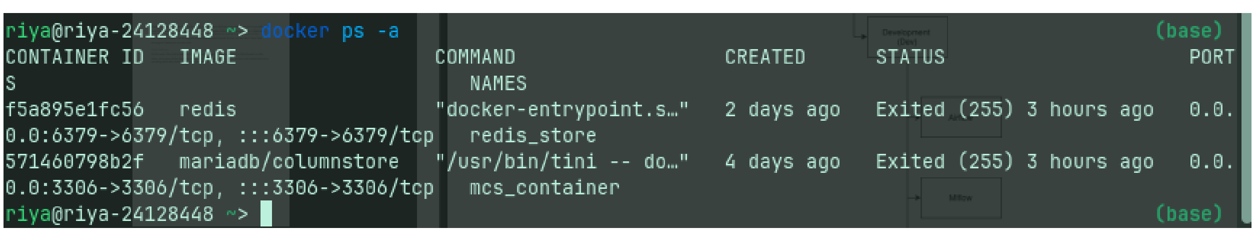


Fig 20.Docker Setup

```
riya@riya-24128448 ~> docker exec -it mcs_container bash                                  (base)
[root@571460798b2f /]# mariadb -u riya --password=riyastha12#
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 11.1.1-MariaDB-log MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Fig 21.MariaDB Setup

```
riya@riya-24128448 ~ [2]> anaconda --version                                              (base)
anaconda Command line client (version 1.12.3)
```

Fig 22. Anaconda Setup

1.3 Environment Setup

```
riya@riya-24128448 ~> conda activate                                                      (base)
riya@riya-24128448 ~> conda activate usedcar                                              (base)
```

Fig 23.Usedcar environment Setup

```
riya@riya-24128448 ~> airflow                                                          (usedcar)
Usage: airflow [-h] GROUP_OR_COMMAND ...

Positional Arguments:
  GROUP_OR_COMMAND

    Groups
      config         View configuration
      connections    Manage connections
      dags           Manage DAGs
      db             Database operations
      jobs           Manage jobs
      pools          Manage pools
      providers      Display providers
```

Fig 24.AirflowSetup

```
riya@riya-24128448 ~ [2]> fastapi --version                                            (usedcar)
FastAPI CLI version: 0.0.5
riya@riya-24128448 ~> jupyter notebook --version                                       (usedcar)
7.2.2
riya@riya-24128448 ~> mlflow --version                                                 (usedcar)
mlflow, version 2.15.1
riya@riya-24128448 ~> python --version                                                 (usedcar)
Python 3.8.20
riya@riya-24128448 ~> redis-cli                                                        (usedcar)
```

Fig 25. Required Tools Setup

1.4 Container Setup

Fig 26. Container Setup

1.5 Airflow Setup

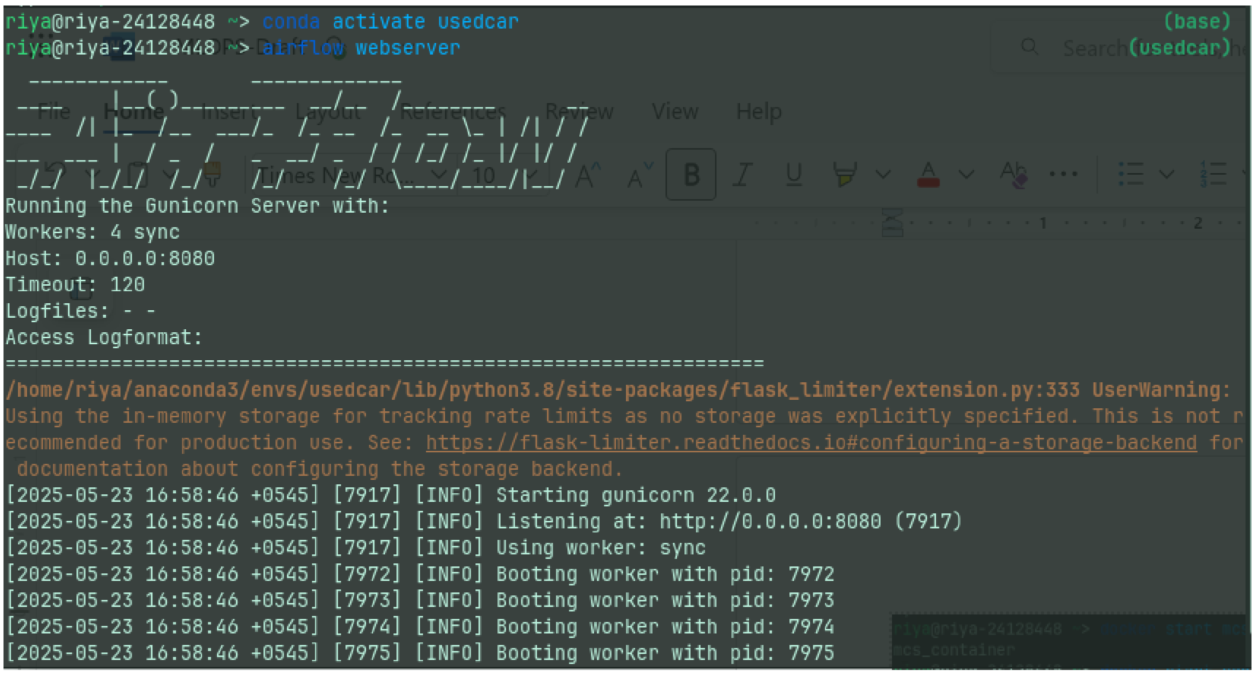


Fig :27 airflow webserver Setup

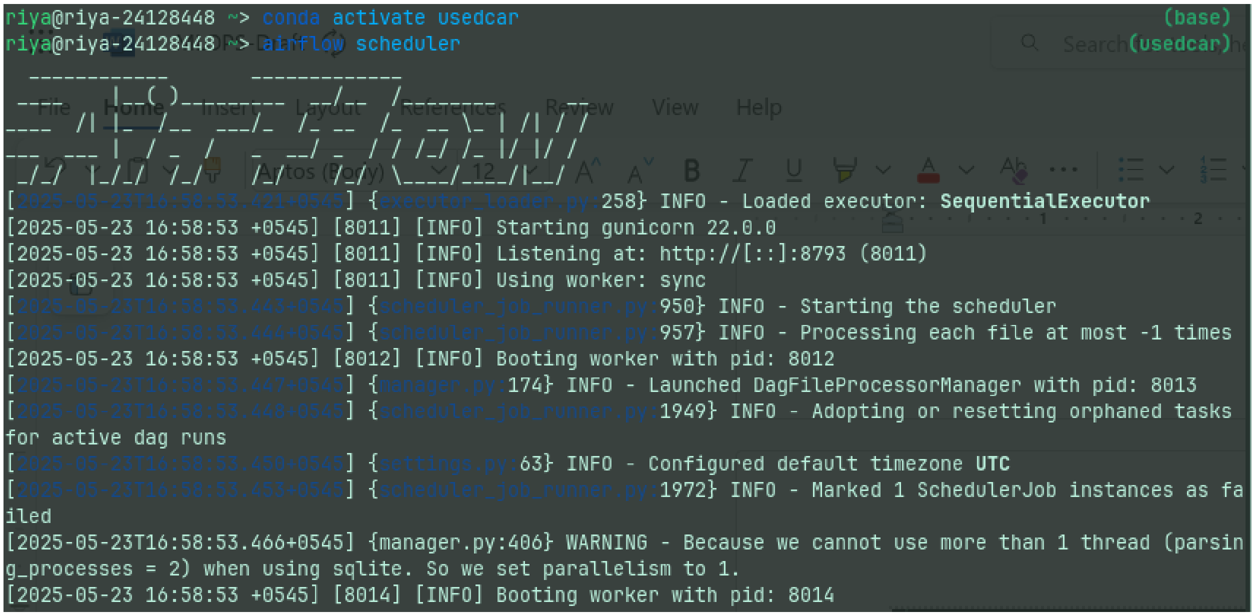


Fig 28: airflow scheduler Setup

1.6 Great Expectations

Fig 29. Great Expectation Setup

1.7 Pipeline Implementation

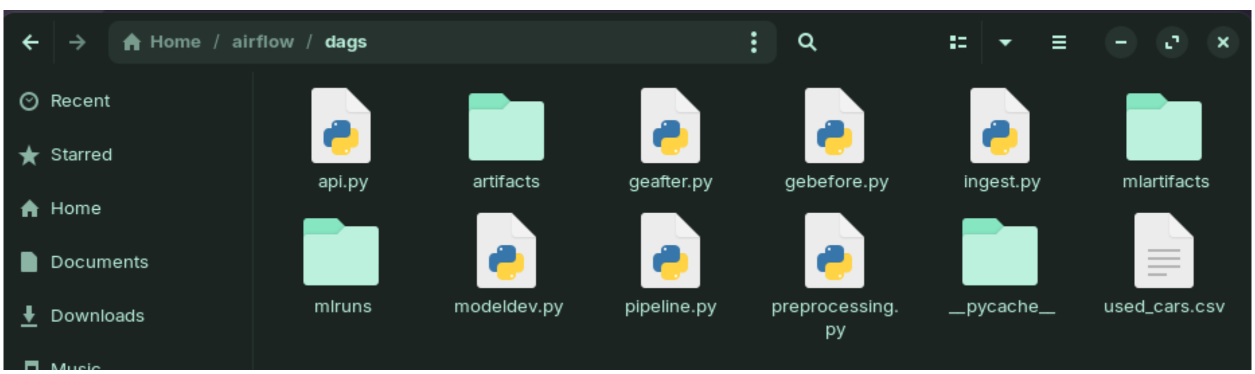


Fig 30: Project file inside dags

```
MariaDB [(none)]> show tables;
ERROR 1046 (3D000): No database selected
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| calpontsys         |
| columnstore_info   |
| dbexample1         |
| infinidb_querystats |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test               |
| usedcar            |
+--------------------+
10 rows in set (0.005 sec)
```

Fig 31: Usedcar database

```
MariaDB [(none)]> use usedcar
Reading table information for completion of table and column name
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [usedcar]> DESCRIBE fact_used_cars;
+--------------+------------+------+-----+---------+-------+
| Field        | Type       | Null | Key | Default | Extra |
+--------------+------------+------+-----+---------+-------+
| car_id       | bigint(20) | YES  |     | NULL    |       |
| model_year   | bigint(20) | YES  |     | NULL    |       |
| milage       | text       | YES  |     | NULL    |       |
| fuel_type    | text       | YES  |     | NULL    |       |
| transmission | text       | YES  |     | NULL    |       |
| price        | text       | YES  |     | NULL    |       |
| brand_id     | bigint(20) | YES  |     | NULL    |       |
| model_id     | bigint(20) | YES  |     | NULL    |       |
| engine_id    | bigint(20) | YES  |     | NULL    |       |
| ext_color_id | bigint(20) | YES  |     | NULL    |       |
| int_color_id | bigint(20) | YES  |     | NULL    |       |
| condition_id | bigint(20) | YES  |     | NULL    |       |
+--------------+------------+------+-----+---------+-------+
12 rows in set (0.001 sec)
```

Fig 32: Usedcars Fact table

```
MariaDB [usedcar]> DESCRIBE dim_brand;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| brand_id   | int(11)     | NO   | PRI | NULL    | auto_increment |
| brand_name | varchar(50) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```

Fig 33: Dimension brand Table

```
MariaDB [usedcar]> DESCRIBE dim_engine;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| engine_id    | int(11)      | NO   | PRI | NULL    | auto_increment |
| fuel_type    | varchar(50)  | YES  |     | NULL    |                |
| engine       | varchar(100) | YES  |     | NULL    |                |
| transmission | varchar(50)  | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
4 rows in set (0.001 sec)
```

Fig 34: Dimension engine Table

```
MariaDB [usedcar]> DESCRIBE dim_color;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| color_id | int(11)     | NO   | PRI | NULL    | auto_increment |
| ext_col  | varchar(50) | YES  |     | NULL    |                |
| int_col  | varchar(50) | YES  |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
3 rows in set (0.001 sec)
```

Fig 35: Dimension Color Table

```
MariaDB [usedcar]> DESCRIBE dim_condition;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| condition_id | int(11)      | NO   | PRI | NULL    | auto_increment |
| accident     | varchar(100) | YES  |     | NULL    |                |
| clean_title  | tinyint(1)   | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
3 rows in set (0.001 sec)
```

Fig 36: Dimension ConditionTable

```
MariaDB [usedcar]> DESCRIBE dim_model;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| model_id   | int(11)      | NO   | PRI | NULL    | auto_increment |
| model_name | varchar(100) | YES  |     | NULL    |                |
| model_year | int(11)      | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.001 sec)
```

Fig 37: Dimension Model Table

```
MariaDB [usedcar]> DESCRIBE usedcars_obt;
+--------------+------------+------+-----+---------+-------+
| Field        | Type       | Null | Key | Default | Extra |
+--------------+------------+------+-----+---------+-------+
| brand        | text       | YES  |     | NULL    |       |
| model        | text       | YES  |     | NULL    |       |
| model_year   | bigint(20) | YES  |     | NULL    |       |
| milage       | text       | YES  |     | NULL    |       |
| fuel_type    | text       | YES  |     | NULL    |       |
| engine       | text       | YES  |     | NULL    |       |
| transmission | text       | YES  |     | NULL    |       |
| ext_col      | text       | YES  |     | NULL    |       |
| int_col      | text       | YES  |     | NULL    |       |
| accident     | text       | YES  |     | NULL    |       |
| clean_title  | bigint(20) | YES  |     | NULL    |       |
| price        | text       | YES  |     | NULL    |       |
| car_id       | bigint(20) | YES  |     | NULL    |       |
+--------------+------------+------+-----+---------+-------+
13 rows in set (0.001 sec)
```

Fig 38: One big table

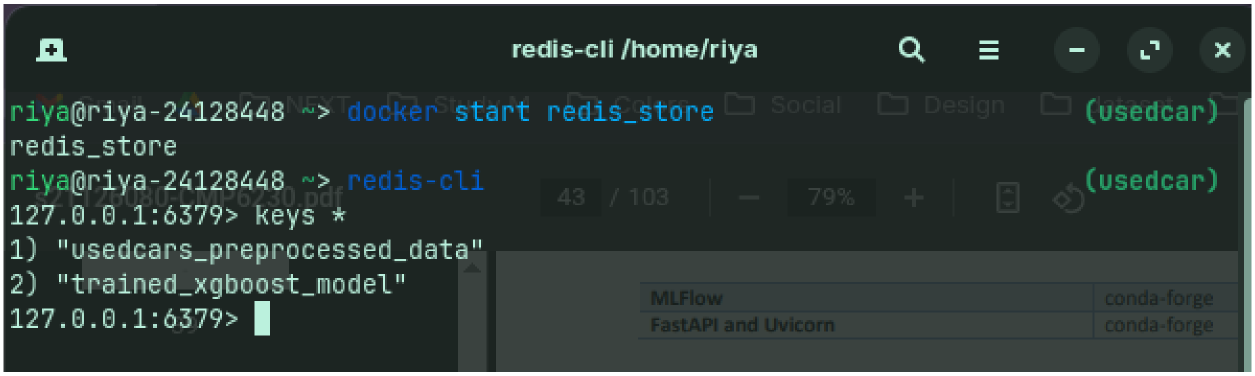


Fig 39: Redis container

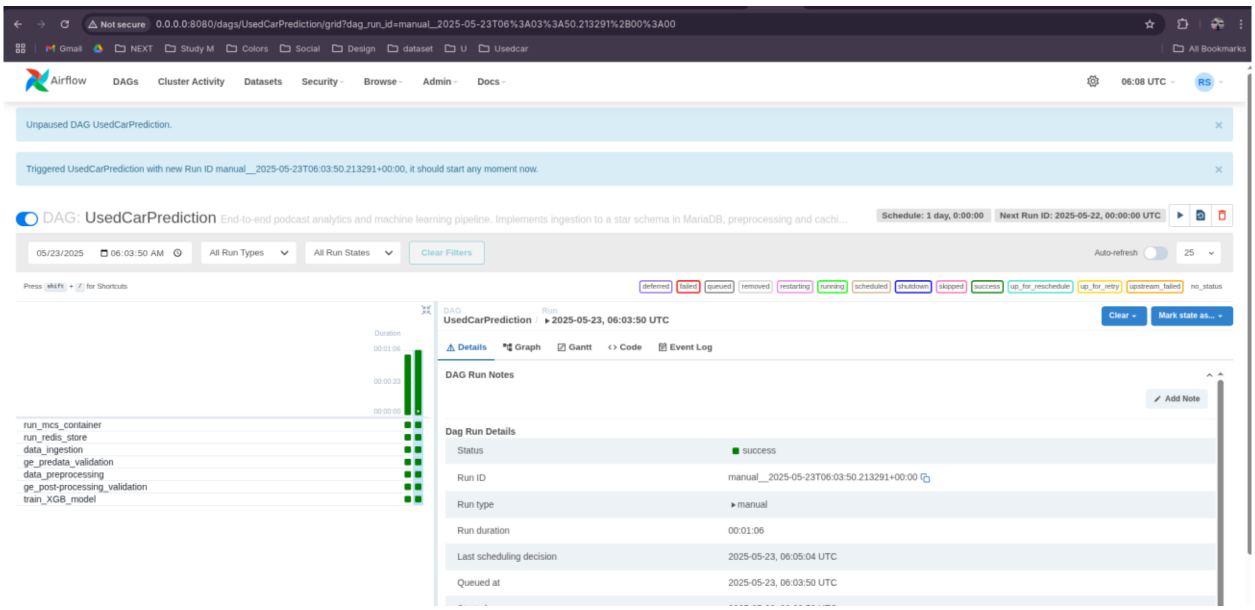


Fig 40:Trained model airflow

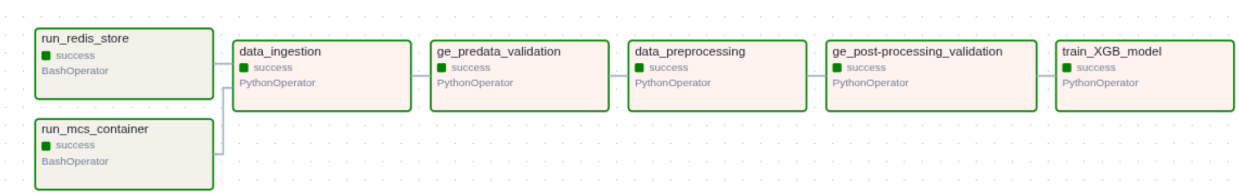


Fig 41: airflow model graph

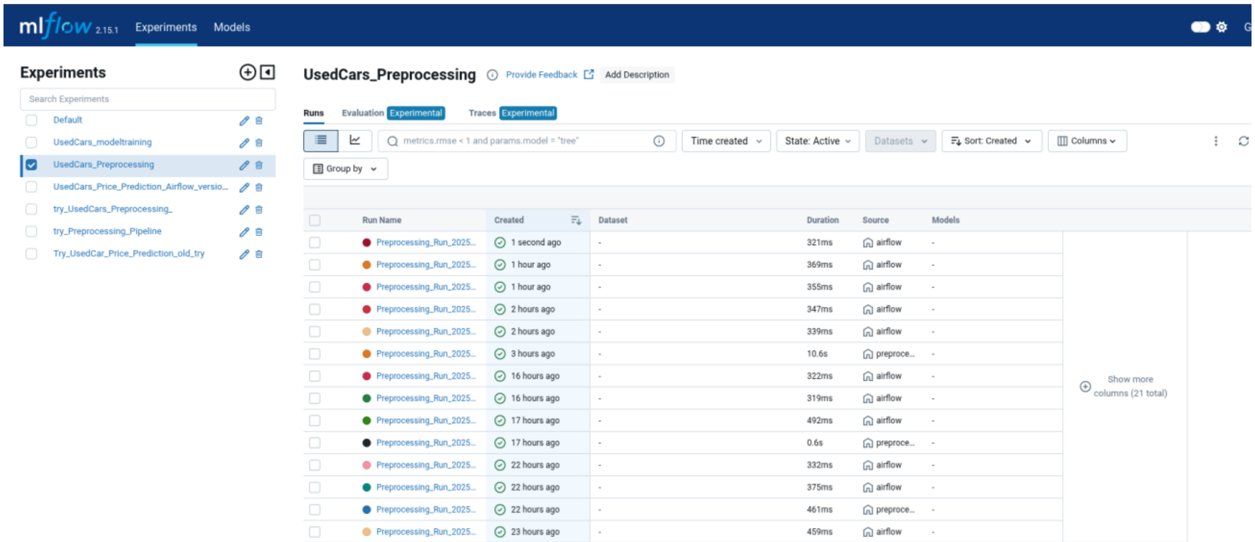


Fig 42: Preprocessing Mlflow

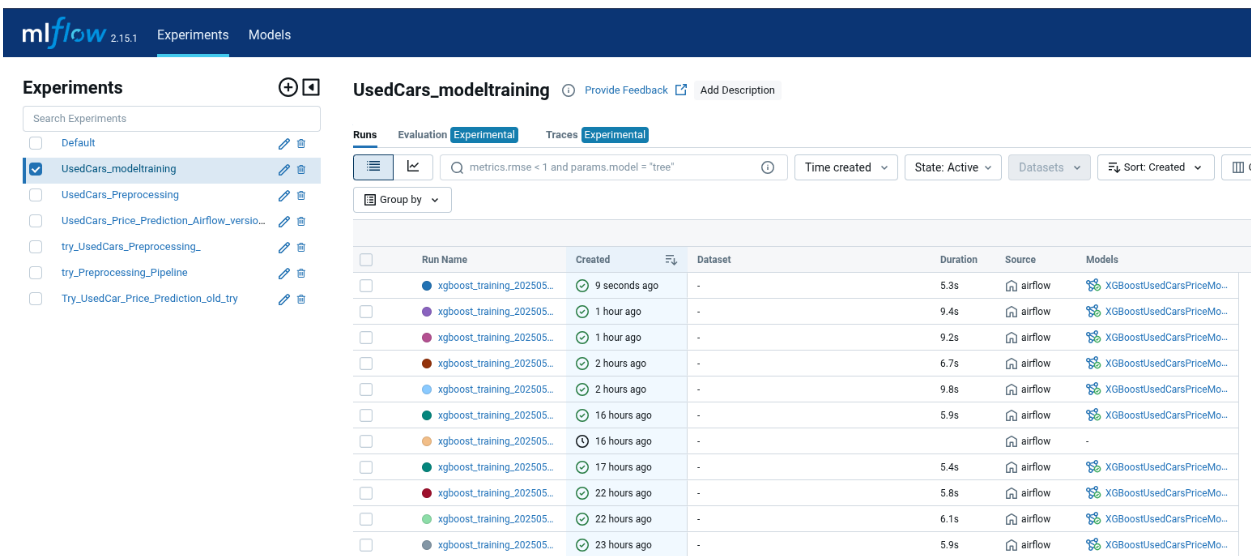


Fig 43: Training model mlflow

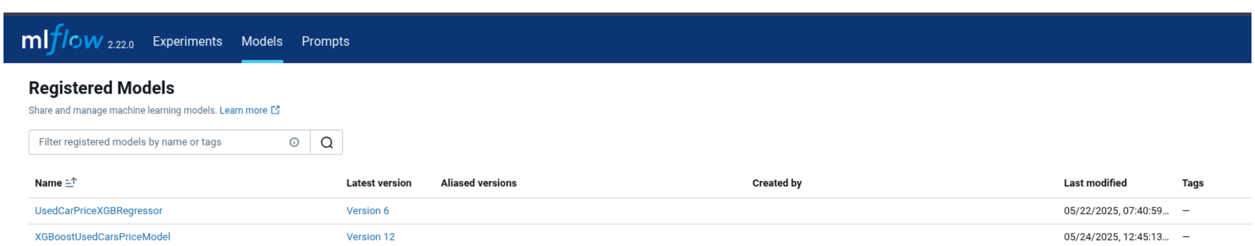


Fig 44: Training model Version

Fig 45: Model predictive value

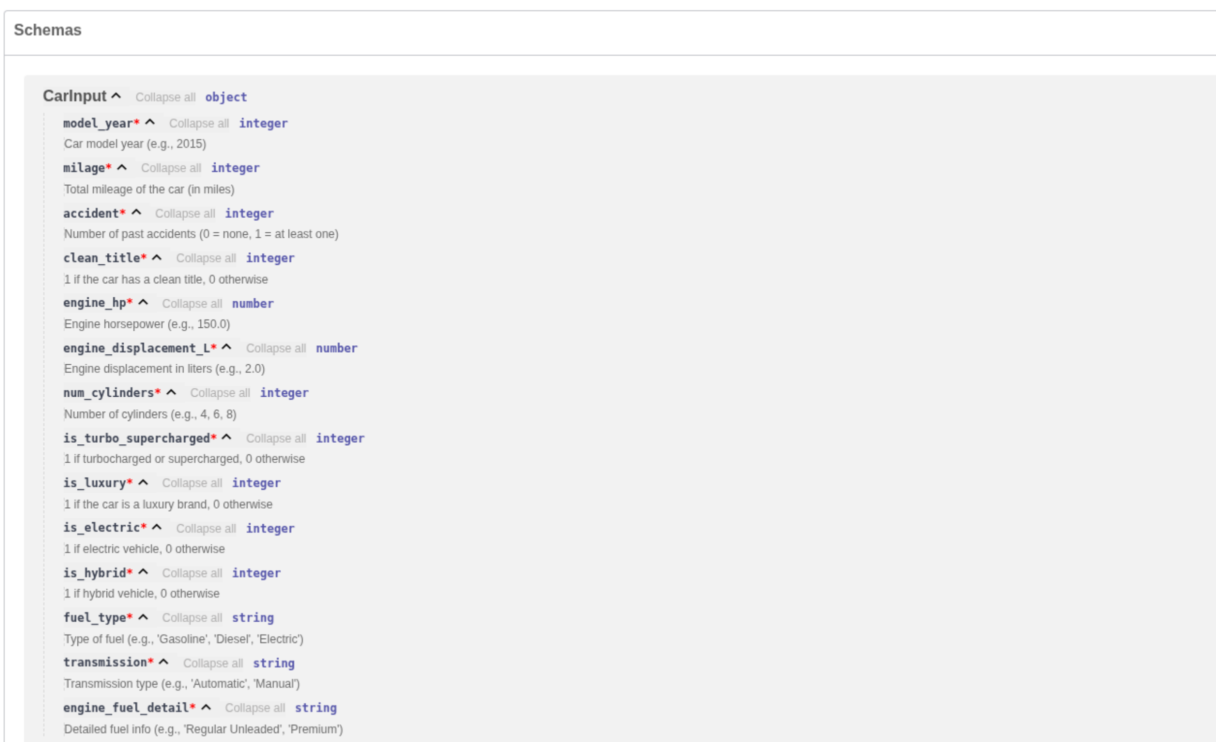


Fig 46: Input Format

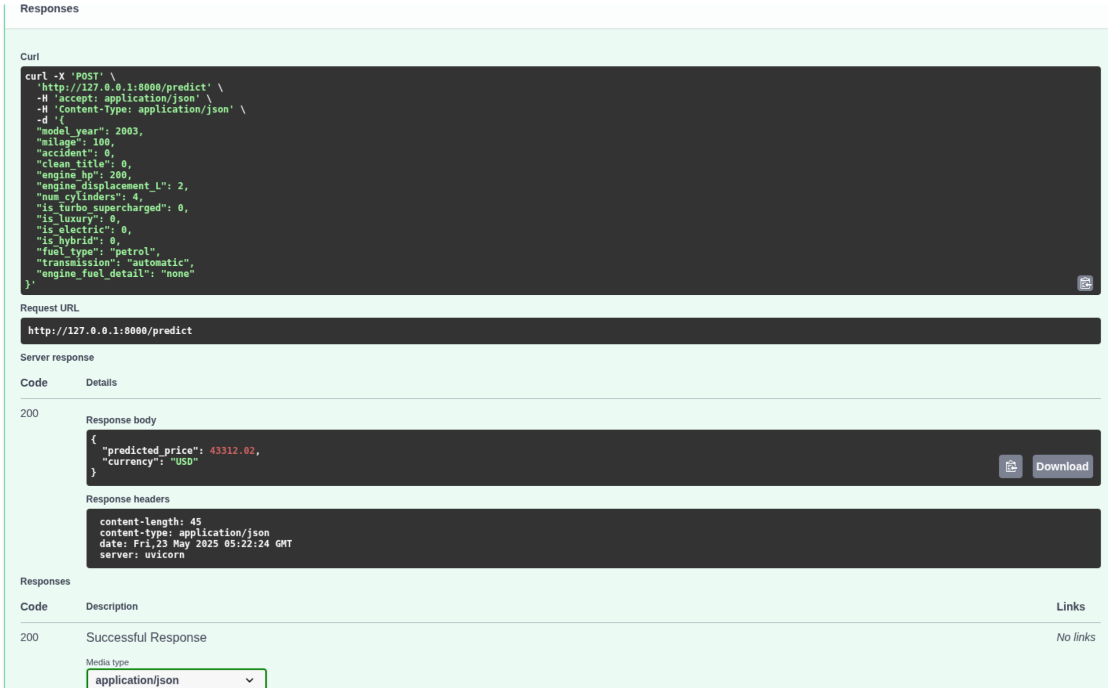


Fig 47: Prediction

Exploratory Data Analysis and Insights

1. Using Raw Data

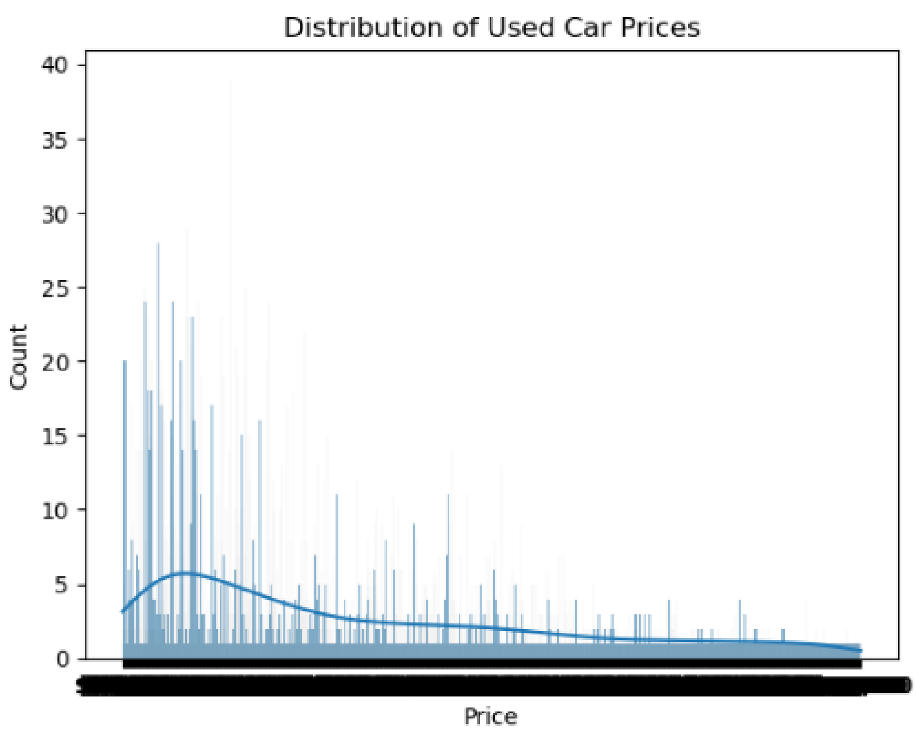


Fig 48: Car Price Distribution

The distribution of used car prices is right-skewed, indicating that most cars are priced on the lower end, with fewer cars having very high prices. This suggests that affordable used cars dominate the market, while luxury or high-end cars are relatively rare in the dataset.

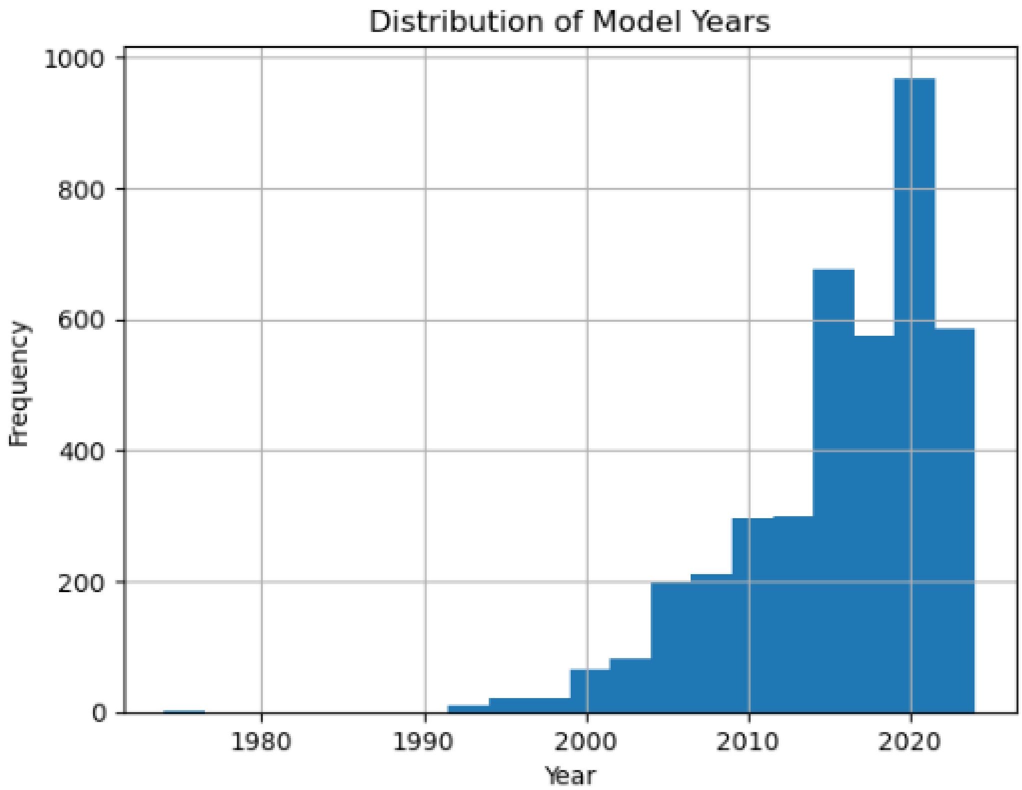


Fig 49: Model Years Distribution

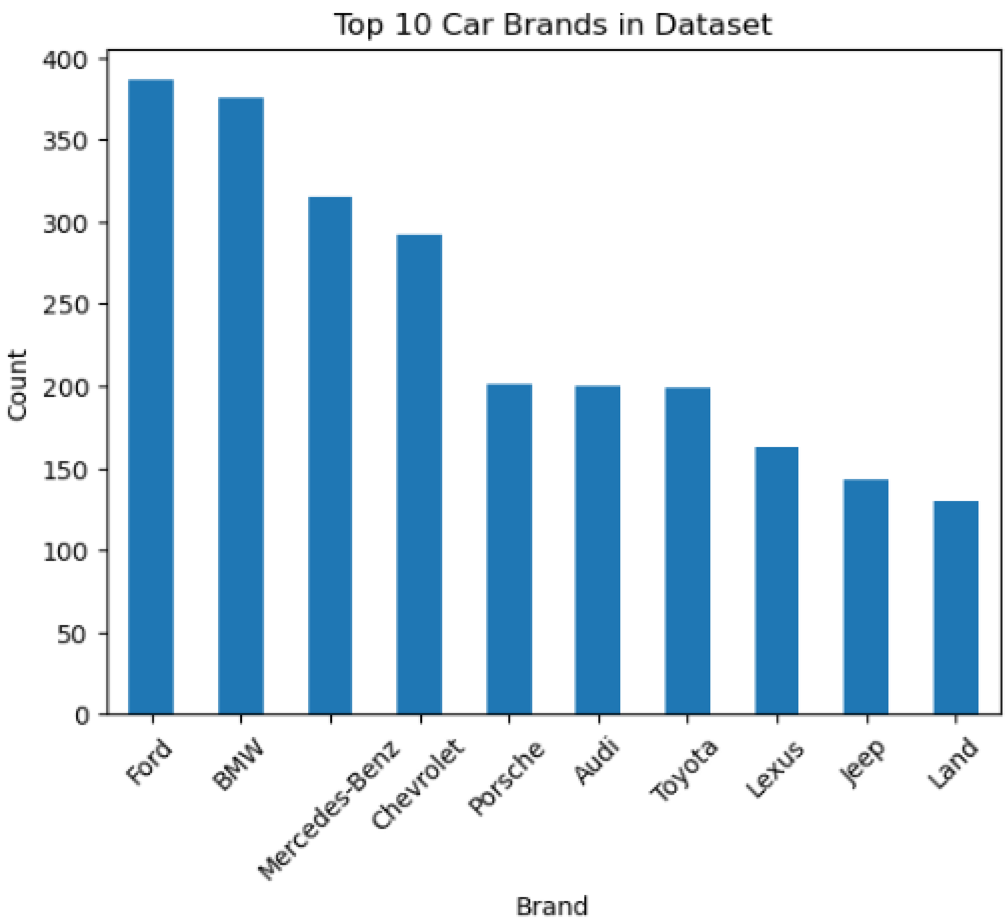


Fig 50: Top 10 Brands

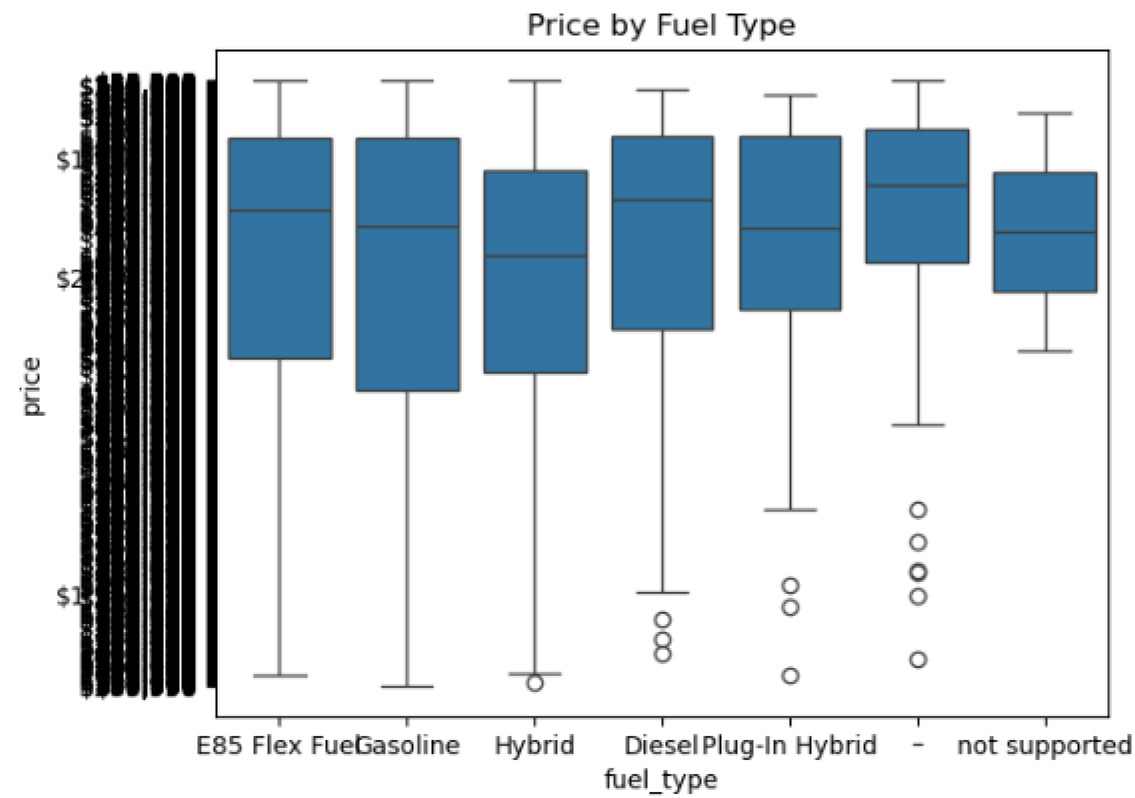


Fig 51: Boxplot of price by fuel

This chart shows how car prices differ based on fuel type. Cars using plug-in hybrid and diesel fuels generally cost more and show a wider price range. Regular gasoline and flex-fuel cars are more consistent in price. Some fuel types also have a few very expensive outliers.

2. Using Preprocessed Data

```
=== Dataset Info ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4009 entries, 0 to 4008
Data columns (total 36 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   model_year                    4009 non-null   int64  
 1   milage                        4009 non-null   float64
 2   accident                      4009 non-null   int64  
 3   clean_title                   4009 non-null   float64
 4   price                         4009 non-null   float64
 5   log_price                     4009 non-null   float64
 6   car_age                       4009 non-null   float64
 7   engine_hp                     4009 non-null   float64
 8   engine_displacement_L         4009 non-null   float64
 9   num_cylinders                 4009 non-null   float64
 10  is_turbo_supercharged         4009 non-null   int64  
 11  milage_per_year               4009 non-null   float64
 12  is_luxury                     4009 non-null   int64  
 13  is_electric                   4009 non-null   int64  
 14  is_hybrid                     4009 non-null   int64  
 15  is_common_color               4009 non-null   int64  
 16  fuel_type_E85 Flex Fuel       4009 non-null   bool   
 17  fuel_type_Electric            4009 non-null   bool   
 18  fuel_type_Gasoline            4009 non-null   bool   
 19  fuel_type_Hybrid              4009 non-null   bool   
 20  fuel_type_Plug-In Hybrid      4009 non-null   bool   
 21  fuel_type_Unknown             4009 non-null   bool   
 22  fuel_type_not supported       4009 non-null   bool   
 23  transmission_CVT              4009 non-null   bool   
 24  transmission_Manual           4009 non-null   bool   
 25  transmission_Unknown          4009 non-null   bool   
 26  engine_fuel_detail_Electric   4009 non-null   bool   
 27  engine_fuel_detail_Flex Fuel  4009 non-null   bool   
 28  engine_fuel_detail_Gasoline   4009 non-null   bool   
 29  engine_fuel_detail_Hybrid     4009 non-null   bool   
 30  engine_fuel_detail_Unknown    4009 non-null   bool   
 31  brand_freq                    4009 non-null   float64
 32  model_freq                    4009 non-null   float64
 33  ext_col_freq                  4009 non-null   float64
 34  int_col_freq                  4009 non-null   float64
 35  brand_model_freq              4009 non-null   float64
dtypes: bool(15), float64(14), int64(7)
memory usage: 716.6 KB
None
```

Fig 52: Feature Datatype

```
=== Descriptive Stats ===
        model_year       milage     accident  clean_title         price  \
count  4009.000000  4009.000000  4009.000000       4009.0  4.009000e+03
mean   2015.515590    -0.006787     0.245947          0.0  4.455319e+04
std       6.104816     0.973526     0.430701          0.0  7.871064e+04
min    1974.000000    -1.225526     0.000000          0.0  2.000000e+03
25%    2012.000000    -0.796969     0.000000          0.0  1.720000e+04
50%    2017.000000    -0.228390     0.000000          0.0  3.100000e+04
75%    2020.000000     0.561913     0.000000          0.0  4.999000e+04
max    2024.000000     3.016068     1.000000          0.0  2.954083e+06

         log_price      car_age    engine_hp  engine_displacement_L  \
count  4009.000000  4009.000000  4009.000000            4009.000000
mean     10.302401    -0.004180    -0.009254              -0.001205
std       0.850058     0.983581     0.954062               0.989581
min       7.601402    -1.226137    -1.785279              -1.614869
25%       9.752723    -0.734661    -0.545033              -0.729761
50%      10.341775    -0.243184    -0.161791              -0.139689
75%      10.819598     0.575943     0.486072               0.671659
max      14.898699     3.020220     3.123145               2.220597

       num_cylinders  ...  milage_per_year    is_luxury  is_electric  \
count    4009.000000  ...      4009.000000  4009.000000  4009.000000
mean       -0.010350  ...        -0.010263     0.380893     0.022449
std         0.953936  ...         0.954692     0.485667     0.148159
min        -1.494604  ...        -1.551534     0.000000     0.000000
25%        -0.134207  ...        -0.723117     0.000000     0.000000
50%        -0.134207  ...        -0.118045     0.000000     0.000000
75%         1.226189  ...         0.559538     1.000000     0.000000
max         2.586586  ...         2.945471     1.000000     1.000000

         is_hybrid  is_common_color   brand_freq   model_freq  ext_col_freq  \
count  4009.000000      4009.000000  4009.000000  4009.000000   4009.000000
mean      0.056872         0.900723     0.047526     0.001120      0.129518
std       0.231627         0.299070     0.031217     0.001196      0.083282
min       0.000000         0.000000     0.000249     0.000249      0.000249
25%       0.000000         1.000000     0.018957     0.000249      0.065104
50%       0.000000         1.000000     0.040659     0.000748      0.123722
75%       0.000000         1.000000     0.078573     0.001247      0.203542
max       1.000000         1.000000     0.096283     0.007483      0.226490

       int_col_freq  brand_model_freq
count   4009.000000       4009.000000
mean       0.293231          0.001119
std        0.218849          0.001196
min        0.000249          0.000249
25%        0.117735          0.000249
50%        0.506111          0.000748
75%        0.506111          0.001247
max        0.506111          0.007483

[8 rows x 21 columns]
```

Fig 53: Preprocessed data head

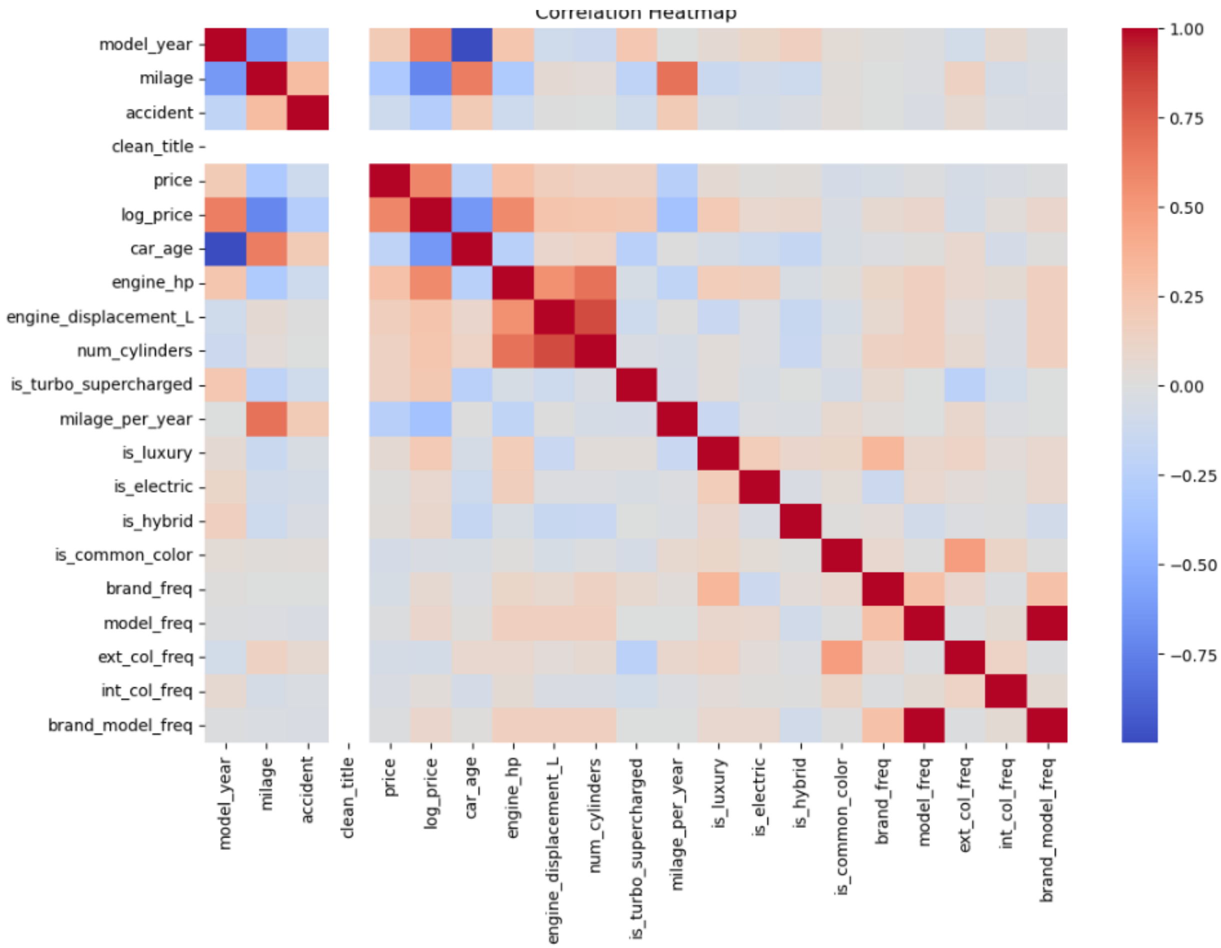


Fig 54: Heatmap

```
=== Top features correlated with log_price ===
log_price                1.000000
model_year               0.625211
price                    0.594027
engine_hp                0.573624
engine_displacement_L    0.256221
num_cylinders            0.241404
is_turbo_supercharged    0.220425
is_luxury                0.210508
brand_model_freq         0.100730
model_freq               0.100466
Name: log_price, dtype: float64
```

Fig 55: Top Feature correlated with log_price

```
=== Outlier Counts by Feature ===
                        Outlier Count
accident                          986
is_common_color                   398
model_freq                        347
brand_model_freq                  347
is_turbo_supercharged             309
price                             244
is_hybrid                         228
engine_hp                         174
is_electric                        90
car_age                            84
milage_per_year                    75
milage                             69
model_year                         67
log_price                          66
```

Fig 56: Outliers in count

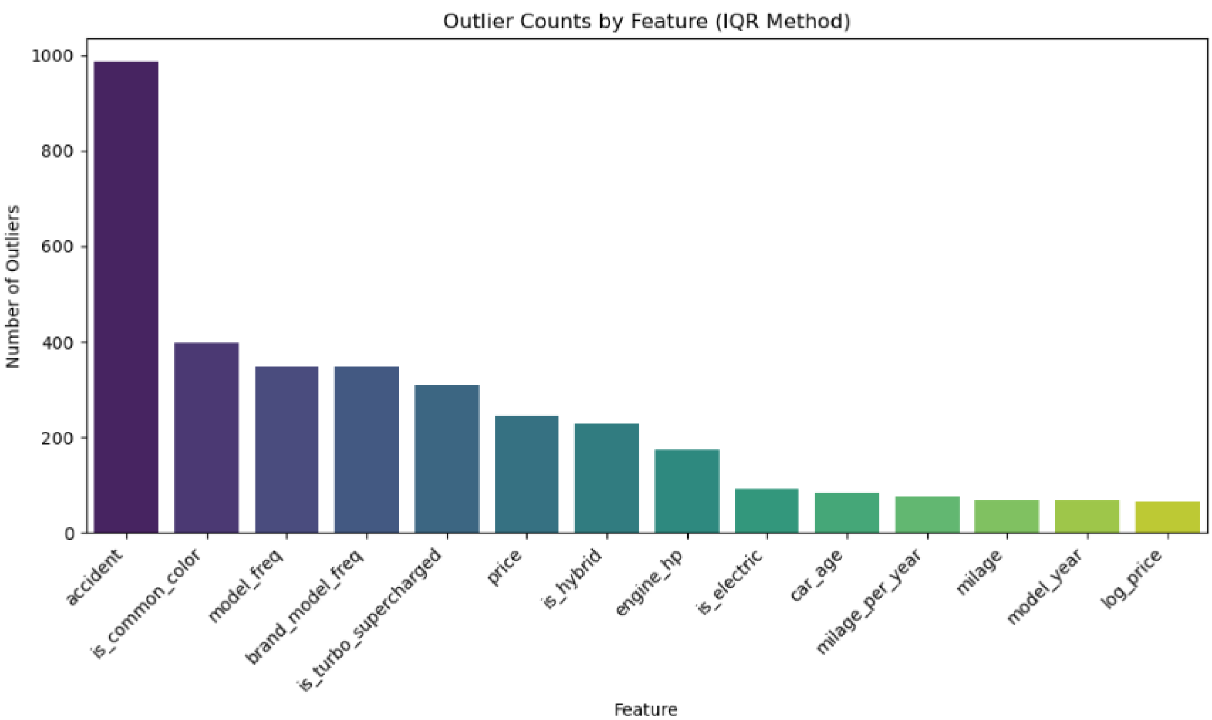


Fig 57: Outliers in Bar Graph

In Outliers in accident, color are high because of imputed value.

Since the ingested was first passed normalized using star schema and then, denormalized to one big table, all the duplicates were removed and missing values were imputed So, the only difference between initial data source and ingested data has no missing values and duplications.

3. Drift

Model drift is when a machine learning model doesn't perform as well as it used to after being used for some time. This happens because the real-world data it sees starts to change. There are two main types: data drift, where the input data changes like new customer habits, and concept drift, where the meaning or pattern behind the data changes like what customers consider important. To deal with this, we can track model's results, retrain it using fresh data, or build models that can learn and update themselves over time.

Legal, Ethical and Security Considerations

1. Data privacy means keeping your personal information safe and only sharing it with people or companies you trust. It's about having control over who sees and uses your data.

2. Data security is the way we protect data from hackers, leaks, or people who shouldn't have access. This includes using passwords, encryption, and safe storage.

3. Data ethics is about using data in a fair and honest way. For example, not spying on people, not using data to trick others, and avoiding bias in AI systems.

4. Data protection laws are rules that companies must follow when they collect and use people's data. A popular one is GDPR in Europe, which gives people rights over their data.

5. Differential Privacy is a data protection technique that allows organizations to extract insights from datasets without revealing individual-level information. It adds statistical noise to the data or query results, making it difficult to identify specific individuals. Real-world applications include Apple's use in iOS data collection and the U.S. Census Bureau's 2020 census. Differential privacy balances utility and privacy, making it crucial for ethical AI and big data analysis.

6. Ethical & Practical Concerns with Mitigating Strategies

Since the dataset was web scraped by creator had some ethical and practical problems that we fixed during preprocessing and model building. To protect privacy, Iremoved sensitive information like car IDs which was formed during star schema. Missing data were imputed in using median and mode values to keep the data reliable. I also fixed different data formats and changed categories into one-hot codes so the model treats them fairly. To reduce bias and make the model fairer, I used a log transformation on the target variable. I tested the model carefully using train-test splits and 5-fold cross-validation to check its accuracy. I used MLflow to track all experiments, making the process clear and repeatable. Redis was used to store data temporarily for fast and safe access. These steps helped make sure the data was handled carefully and ready for building a good, ethical prediction model.

Reflections

This Coursework is a great learning experience that covered the full process of a machine learning project using MLOps. Choosing the right dataset showed how important it is to match data with project goals, especially when dealing with real-world problems like missing and inconsistent data. Careful cleaning and checking the data with tools like Great Expectations was essential.

Building the MLOps setup with tools like Docker, MariaDB, Airflow, Redis, and MLflow was challenging but important for creating a reliable and repeatable pipeline. Designing the Airflow workflow to handle all steps from data loading to model training taught me a lot about automating complex processes.

Training the XGBoost model and deploying it through FastAPI made the project practical and useful. Exploring the data helped improve the model and understand its results. The project also raised important legal and ethical issues, like dealing with biased or web-scraped data, reminding me of the responsibility in building such systems.

The biggest challenge was setting up and fixing problems with all the different tools working together, mainly while I was training model, that was the most hectic for me because model took very long time to train. Later I realized it was stuck in between, the model wasn't trained at all which took time and patience. Also, the right guidance from my coordinator's sample and friends helped me find errors to complete this work.

Further Plan

Although the project is progressing well, there is still more work to be done. Improving the model's performance remains a priority which i haven't done yet so I plan to apply Optuna for hyperparameter tuning to achieve better results. Additionally, I need to enhance the internal pipeline by adding more steps such as data imputation, data splitting, and other preprocessing tasks.I will also address feedback provided in the future to further improve the system.

If possible, I'll create a user-friendly frontend to enhance the overall user experience, which will complement the backend and make the system more accessible.Bias mitigation in the model is not yet fully complete, so I will continue working on reducing bias to ensure fairness and reliability. Currently, the trained model is stored in Redis which needs to be saved in MLflow for model management and versioning.

Finally, I aim to strengthen the pipeline's automation so that it can run independently at scheduled times. The pipeline should also be able to handle new incoming data efficiently, enabling continuous model updates and improvements over time.

References

Najib, T. (2023) *Used car price prediction dataset*, *Kaggle*. Available at: https://www.kaggle.com/datasets/taeefnajib/used-car-price-prediction-dataset/data (Accessed: 25 April 2025).

Kanchana1990 (2024) *Real estate data Utah 2024*, *Kaggle*. Available at: https://www.kaggle.com/datasets/kanchana1990/real-estate-data-utah-2024/data (Accessed: 25 April 2025).

H, M.Y. (2021) *Breast cancer dataset*, *Kaggle*. Available at: https://www.kaggle.com/datasets/yasserh/breast-cancer-dataset (Accessed: 25 April 2025).

N. Monburinon, P. Chertchom, T. Kaewkiriya, S. Rungpheung, S. Buya and P. Boonpou, "Prediction of prices for used car by using regression models," 2018 5th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand, 2018, pp. 115-119, doi: 10.1109/ICBIR.2018.8391177. keywords: {Automobiles;Regression tree analysis;Linear regression;Predictive models;Data models;Vegetation;Forestry;comparative study;multiple linear regression;random forest;gradient boosting;supervised learning},

Adhikary, D.R.D., Sahu, R. and Panda, S.P. (1970) *Prediction of used car prices using machine learning*, *SpringerLink*. Available at: https://link.springer.com/chapter/10.1007/978-981-16-8739-6_11 (Accessed: 25 April 2025).

M. Hankar, M. Birjali and A. Beni-Hssane, "Used Car Price Prediction using Machine Learning: A Case Study," 2022 11th International Symposium on Signal, Image, Video and Communications (ISIVC), El Jadida, Morocco, 2022, pp. 1-4, doi: 10.1109/ISIVC54825.2022.9800719. keywords: {Maximum likelihood estimation;Linear regression;Training data;Production;Pricing;Predictive models;Boosting;regression analysis;prediction;estimation;Avito;machine learning;log transformation;used car price;regression assumptions},

(No date a) *Simple search*. Available at: https://www.diva-portal.org/smash/search.jsf?dswid=4178 (Accessed: 24 May 2025).

C. Barnes. Worldwide Used Car Dealers Industry. Barnes Reports, C. Barnes & Co.,USA, 2011.

R. Caruana, A. Munson, and A. Niculescu-Mizil. Getting the Most Out of Ensemble Selection. In Proc. of the 6th Intern. Conf. on Data Mining, pages 828-833. IEEE Computer Society, Los Alamitos, 2006.

J. Du, L. Xie, and S. Schroeder. PIN Optimal distribution of auction vehicles system: Applying price forecasting, elasticity estimation, and genetic algorithms to used-vehicle distribution. Marketing Science, 28:637-644, 2009.

T. Hastie, R. Tibshirani, and J.H. Friedman. The Elements of Statistical Learning. Springer, New York, 2nd edition, 2009.

(No date) *Https://ieeexplore.ieee.org/document/7741877*. Available at: https://arxiv.org/pdf/1907.10273 (Accessed: 24 May 2025).

Sahiner, B. *et al.* (2023) 'Data Drift in medical machine learning: Implications and potential remedies', *The British Journal of Radiology*, 96(1150). doi:10.1259/bjr.20220878.

Hoens, T.R., Polikar, R. and Chawla, N.V. (2012) *Learning from streaming data with concept drift and imbalance: An Overview - progress in Artificial Intelligence*, *SpringerLink*. Available at: https://link.springer.com/article/10.1007/s13748-011-0008-0 (Accessed: 24 May 2025).

Ljungholm, D.P. (1970) *Autonomous car regulation in the SMART Transportation Infrastructure: Ethical issues, legal liabilities, and privacy concerns*, *Geopolitics, History, and International Relations*. Available at: https://www.ceeol.com/search/article-detail?id=804055 (Accessed: 24 May 2025).

(No date a) *Tristanconference*. Available at: https://tristanconference.org/system/tristan-viii/book_of_abstracts/data/Methodological/TRISTAN8_paper_115.pdf (Accessed: 24 May 2025).

SUPERVISOR, EDP (2022). The History of the General Data Protection Regulation. url: https : / / edps . europa . eu / data - protection / data - protection / legislation / history - general - data - protection - regulation en [Last accessed: 28th Jan. 2022]. Meehan, J, C Aslantas, S Zdonik, N Tatbul and J Du (2017).

'Data Ingestion for the Connected World.' In: CIDR. Zheng, A and A Casari (2018). Feature engineering for machine learning: principles and techniques for data scientists. " O'Reilly Media, Inc." Radovic, M, M Ghalwash, N Filipovic and Z Obradovic (2017). '

Minimum redundancy maximum relevance feature selection approach for temporal gene expression data'. In. Huang, J, YF Li and M Xie (Nov. 2015). 'An empirical analysis of data preprocessing for machine learning-based software cost estimation'. In: vol. 67. Elsevier, pp. 108–127.

Gavazza, A., Lizzeri, A. and Roketskiy, N. (no date) *A quantitative analysis of the used-car market*, *American Economic Review*. Available at: https://www.aeaweb.org/articles?id=10.1257%2Faer.104.11.3668 (Accessed: 24 May 2025).