Car Dheko – Used Car Price Prediction

Objective:

To develop a machine learning model that accurately predicts the resale price of used cars listed on platforms like Car Dekho, based on various car features and market factors.

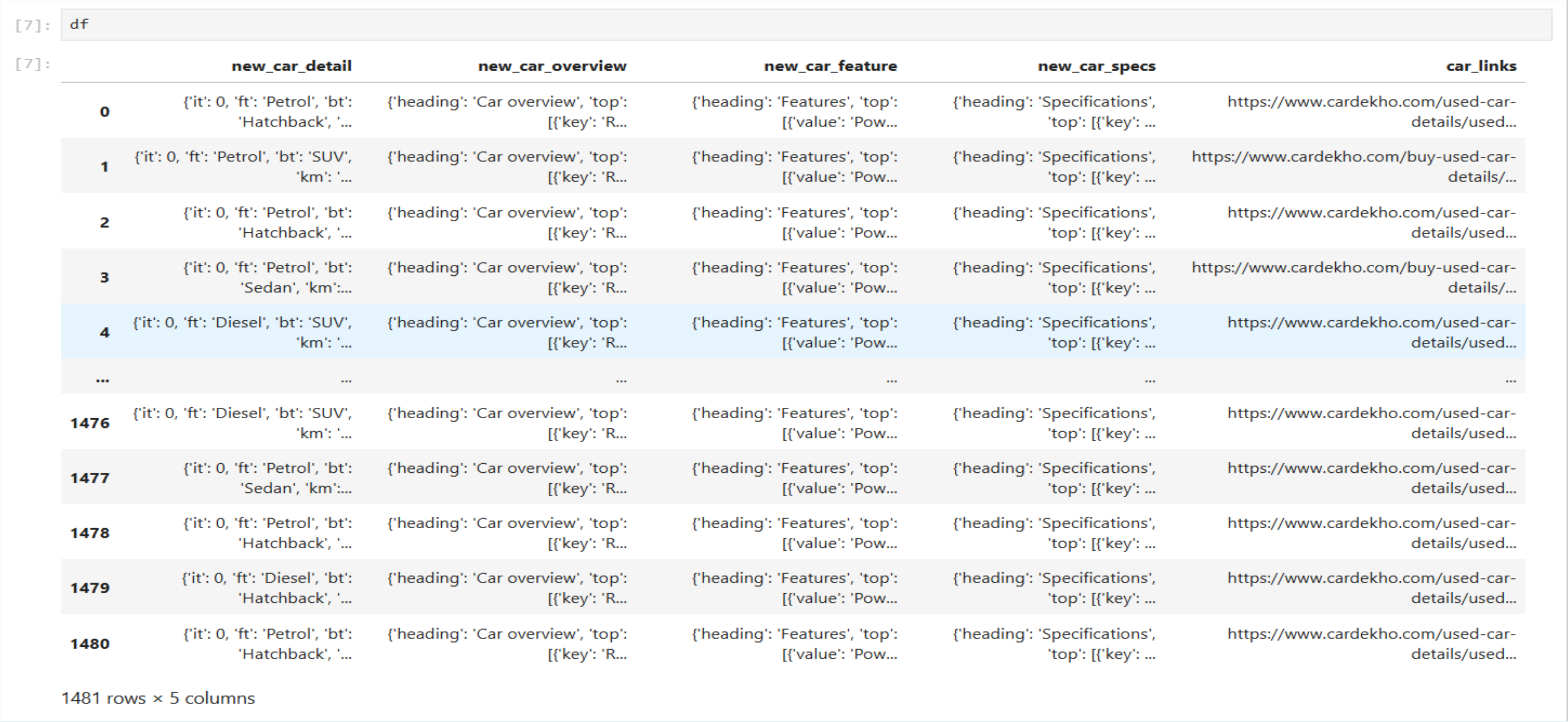
Key Goals:

- Collect and clean real-world used car data (e.g., brand, model, year, kilometers driven, body type, transmission, fuel type, mileage, city, etc.)
- Perform exploratory data analysis (EDA) to understand data distribution, detect outliers, and handle missing values. Visual aids
- Apply appropriate feature engineering and encoding techniques to prepare the dataset for modeling.
- Train, validate, and compare different regression models (e.g., Random Forest, Gradient Boosting) to identify the best-performing model.
- Evaluate model performance using metrics such as R² Score, RMSE, and cross-validation.
- Deploy the final model using a user-friendly Streamlit web application that allows users to input car features and receive a price prediction instantly.

Methodology

Data Source

The raw data was stored in an Excel file, where each row represented a unique car listing, and key car features such as fuel type, body type, kilometers driven, transmission, ownership, brand, model, model year, and price details were embedded as nested JSON-like dictionaries inside individual cells.



```
[7]: df
```

[7]:

	new_car_detail	new_car_overview	new_car_feature	new_car_specs	car_links
0	{'it': 0, 'ft': 'Petrol', 'bt': 'Hatchback', '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
1	{'it': 0, 'ft': 'Petrol', 'bt': 'SUV', 'km': '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/buy-used-car-details/...
2	{'it': 0, 'ft': 'Petrol', 'bt': 'Hatchback', '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
3	{'it': 0, 'ft': 'Petrol', 'bt': 'Sedan', 'km':...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/buy-used-car-details/...
4	{'it': 0, 'ft': 'Diesel', 'bt': 'SUV', 'km': '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
...	...	...	...	...	...
1476	{'it': 0, 'ft': 'Diesel', 'bt': 'SUV', 'km': '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
1477	{'it': 0, 'ft': 'Petrol', 'bt': 'Sedan', 'km':...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
1478	{'it': 0, 'ft': 'Petrol', 'bt': 'Hatchback', '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
1479	{'it': 0, 'ft': 'Diesel', 'bt': 'Hatchback', '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...
1480	{'it': 0, 'ft': 'Petrol', 'bt': 'Hatchback', '...	{'heading': 'Car overview', 'top': [{'key': 'R...	{'heading': 'Features', 'top': [{'value': 'Pow...	{'heading': 'Specifications', 'top': [{'key': ...	https://www.cardekho.com/used-car-details/used...

1481 rows × 5 columns

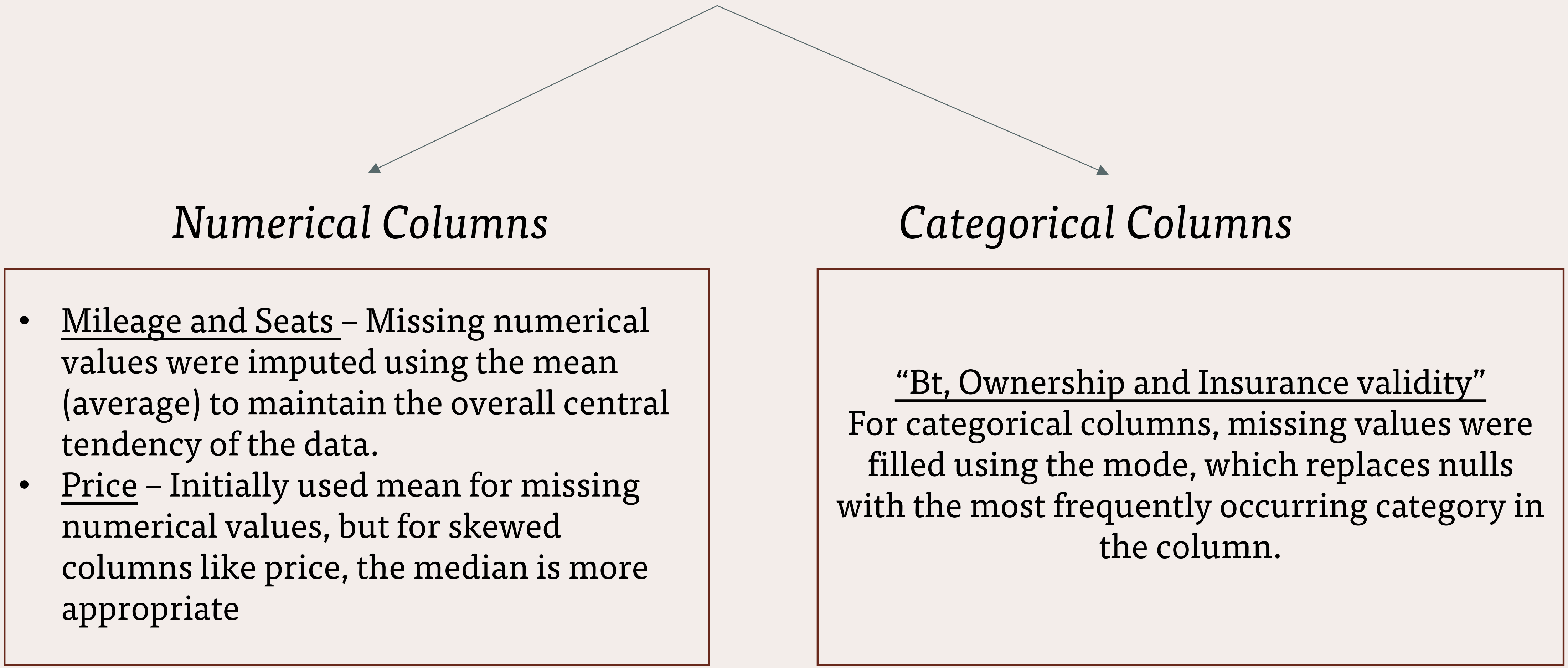
Data Normalization

- The car details were stored as JSON text within a single column, not in separate columns.
- Used Python (json.loads + pd.json_normalize) to parse each JSON string and expand it into individual columns.
- Parsed each cell's JSON text into a Python dictionary
- Normalized the nested dictionaries to create a flat tabular dataset.
- Combine all the normalized data for further cleaning.
- Obtained a clean, structured dataset with one column per feature, ready for preprocessing and modeling.

Data Processing

Null imputation, Standardising datas, Encoding, Normalising, Removing Outliers

Handling missing Values



Numerical Columns

- Mileage and Seats – Missing numerical values were imputed using the mean (average) to maintain the overall central tendency of the data.
- Price – Initially used mean for missing numerical values, but for skewed columns like price, the median is more appropriate

Categorical Columns

“Bt, Ownership and Insurance validity”
For categorical columns, missing values were filled using the mode, which replaces nulls with the most frequently occurring category in the column.

Removing of outliers

- Outliers were treated using the IQR method. Values falling below Q1 – 1.5*IQR or above* Q3 + *1.5*IQR were identified as outliers and capped to reduce their impact on mean-based imputation and scaling.
- For the price column, outliers were handled using the IQR method combined with capping.
- Since the IQR method alone did not fully remove extreme values, capping was applied to limit prices to the upper threshold.
- Some outliers may still appear visually, but the capping ensures they do not overly influence the model.
- Final preprocessing was completed with these adjusted values.

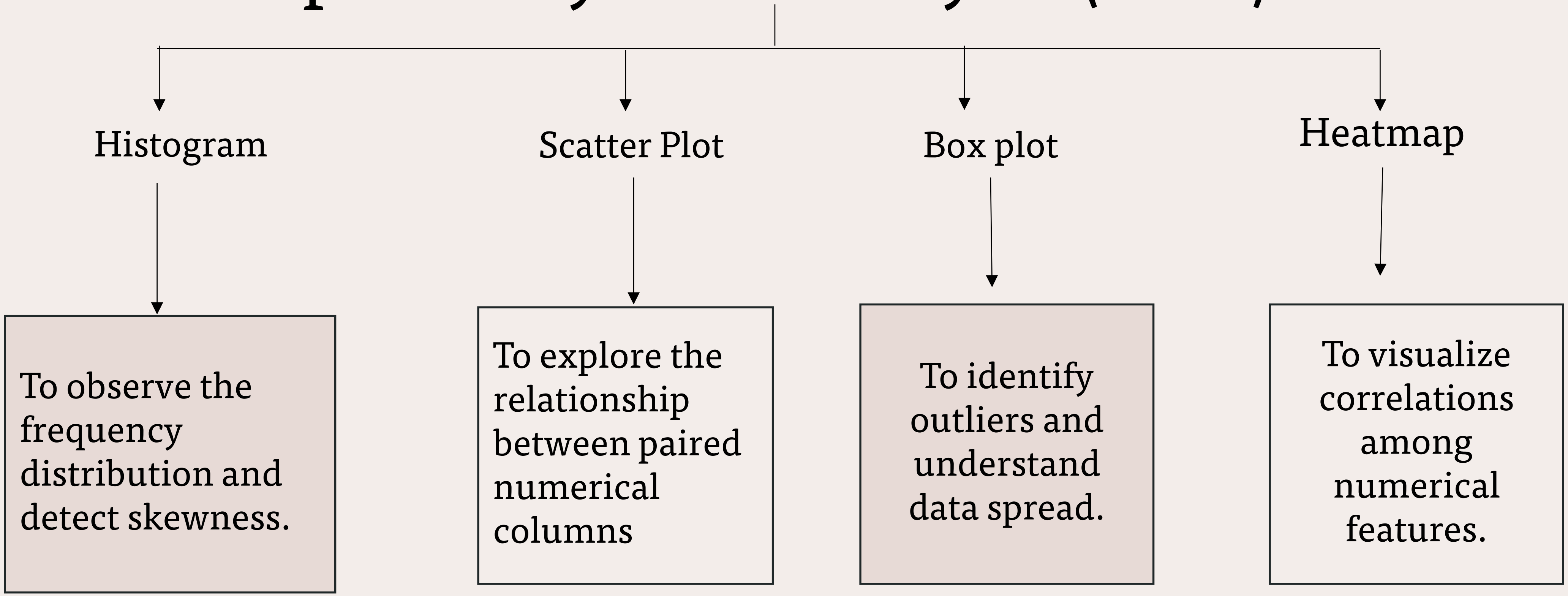
Encoding Categorical variable

Since all categorical variables in the dataset are nominal (no natural ranking), one-hot encoding was applied to convert them into binary columns, avoiding any false ordinal interpretation and making the data suitable for machine learning models

Normalising numerical Feature

Min-Max scaling was applied to normalize numerical columns to a fixed range (0–1). Since the data is not normally distributed and may contain varying scales, Min-Max scaling preserves the shape of the distribution while making features comparable. Standard scaling was not used as it assumes a normal distribution, which our data does not follow

Exploratory Data Analysis (EDA)



Histogram

To observe the frequency distribution and detect skewness.

Scatter Plot

To explore the relationship between paired numerical columns

Box plot

To identify outliers and understand data spread.

Heatmap

To visualize correlations among numerical features.

Model Development

Model Comparison

Trained Model	MAE (Mean absolute error)	RMSE (Mean Squared Error)	R-squared
Linear Regression	1,366,026	42,729	-156
Decision Trees	95,196.41	152,636.19	0.8001
Random Forest	*72,161.17*	*111,877.75*	*0.8926*
Gradient Boosting	101,423.22	141,945.58	0.8271

Model Selection:

Random Forest was selected because it achieved the highest accuracy on test data.
Its ensemble approach combines multiple Decision Trees, which reduces overfitting and captures complex relationships in the data.
This makes it more robust and reliable than a single tree or a simple linear model.

Hyperparameter Tuning:

Randomized SearchCV was used to optimize model hyperparameters.
Compared to Grid SearchCV, Randomized SearchCV samples random combinations from the parameter grid, making it faster and more efficient for large search spaces while still achieving good results.

Hyperparameter tuning	**MAE (Mean absolute error)**	**RMSE (Mean Squared Error)**	**R-squared**
RandomSearchCV	89,147.57	130,418.43	0.8540

Model Deployment using Streamlit

The final Random Forest model was deployed using Streamlit to create an interactive web application.

Users can input features such as [Body Type, Model Year, KM Driven, etc.] through a user-friendly sidebar, and the app predicts the car price instantly.

Streamlit was chosen for its simplicity and ability to quickly turn Python models into shareable web apps without complex backend development.

Thank you