Car Dheko - Used Car Price Prediction

Project Report

By

MUGESH SURYA PRADEEP

INTRODUCTION

- In this project, the task is to enhancing the customer experience and optimizing the pricing process for used cars by developing a machine learning model. This model, based on historical car price data, will take into account various features such as the car's make, model, year, fuel type, and transmission type. The goal is to predict the prices of used cars accurately and integrate this model into an interactive web application using Streamlit. The app will enable customers and sales representatives to input car details and receive real-time price predictions, making the car buying and selling process more efficient .

APPROACH

1. **Import files and Data wrangling:**

- Load the datasets from multiple cities, which are in unstructured format (likely messy, containing text, JSON-like entries, or other formats) from Excel files.
- Use libraries like pandas to import data from each city's dataset for further processing.
- For unstructured data that includes JSON-like structures (e.g., dictionaries or lists embedded as strings), use ast.literal_eval to safely evaluate these strings into Python objects.
- Once JSON-like data is parsed, use pandas.json_normalize() to convert nested JSON objects into a flat, structured dataframe.
- This ensures that all nested values are properly spread across multiple columns, creating a clean and organized structure.
- After converting each city's dataset into a structured format, add a new column named ‘City’ and assign the respective city name as the value for all rows in the dataset.
- Use pandas.concat() to merge the structured datasets from all cities into a single dataframe.
- Ensure that columns are aligned and duplicate entries are handled nto create a unified dataset for model training and save dataframe to csv

2. Handling Missing Values and Data Cleaning:

• Use pandas.dropna() to remove rows or columns containing missing data • Remove symbols and units (₹, Lakh, Crore,kmpl,CC),Eliminate commas and whitespace
• Convert the Values to a Numerical Format suitable for machine learning models..

3. Data Visualization: Exploratory Data Analysis (EDA) was performed to examine the relationships between various features and the target variable (price). This analysis helped uncover key patterns and identify potential outliers. Correlation Matrix: A heatmap of the correlation matrix highlighted significant correlations between dependent and independent features such as modelYear and km with price. Outlier Detection: Outliers in the price column were detected using the Interquartile Range (IQR) method to prevent them from affecting the model's performance

4.Features: Fuel type, Body type, Transmission, Owner, Brand, Model, Model Year, Insurance Validity, Kms Driven, Mileage, Seats, Color, City.Target: Price.Data

5.Preprocessing: One-hot encoding is used for categorical features like Fuel type, Body type, Transmission, etc., to convert them into numeric values for the model.Scaling is performed using StandardScaler to standardize numerical features like Mileage, Seats, etc.

6. Data Preparation: The dataset undergoes several preprocessing steps:Missing Value Handling: Missing values are handled, though not explicitly mentioned in the code, this step should be incorporated in practice.Scaling: The features are scaled using StandardScaler.One-Hot Encoding: Categorical variables are encoded to numerical format using OneHotEncoder.

The features (X) and target (y) are split, and the dataset is divided into training and testing sets (75% training, 25% testing).

7. ModelingMachine Learning Algorithms Used:

- Linear Regression
- Decision Tree Regressor
- Random Forest Regressor
- Gradient Boosting Regressor
- Extra Trees Regressor
- XGBoost Regressor
- LightGBM Regressor
- Ridge Regression
- Lasso Regression

The best_ML_algorithm function is used to fit and evaluate each algorithm using metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2) score. The models are evaluated on both the training and testing datasets.

8. Model Evaluation:

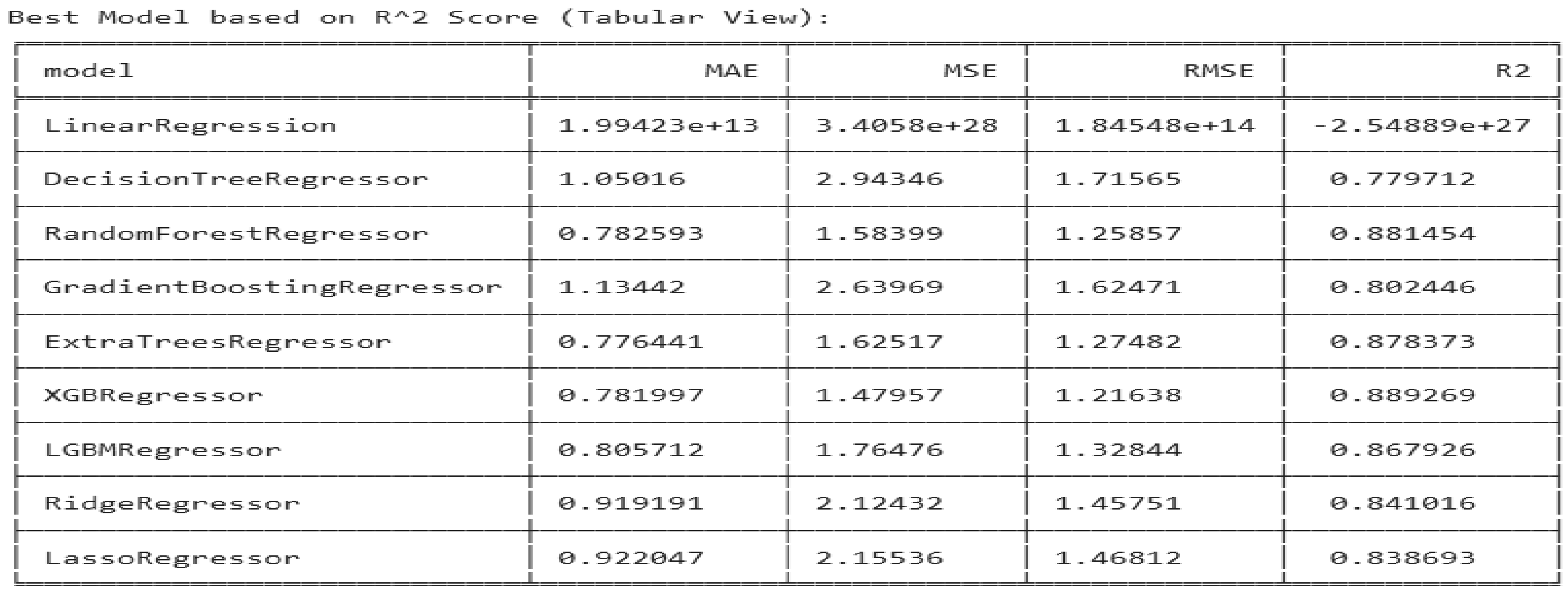
The models were evaluated using the following metrics:

Mean Squared Error (MSE): Measures the average squared difference between actual and predicted values.

Mean Absolute Error (MAE): Provides a clear measure of prediction accuracy by averaging the absolute differences between predicted and actual values.

R² Score: Indicates how well the independent variables explain the variance in the dependent

Model Comparision



```
Best Model based on R^2 Score (Tabular View):
╒═══════════════════════════╤═════════════╤════════════╤═════════════╤══════════════╕
│ model                     │         MAE │        MSE │        RMSE │           R2 │
╞═══════════════════════════╪═════════════╪════════════╪═════════════╪══════════════╡
│ LinearRegression          │ 1.99423e+13 │ 3.4058e+28 │ 1.84548e+14 │ -2.54889e+27 │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ DecisionTreeRegressor     │ 1.05016     │ 2.94346    │ 1.71565     │  0.779712    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ RandomForestRegressor     │ 0.782593    │ 1.58399    │ 1.25857     │  0.881454    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ GradientBoostingRegressor │ 1.13442     │ 2.63969    │ 1.62471     │  0.802446    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ ExtraTreesRegressor       │ 0.776441    │ 1.62517    │ 1.27482     │  0.878373    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ XGBRegressor              │ 0.781997    │ 1.47957    │ 1.21638     │  0.889269    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ LGBMRegressor             │ 0.805712    │ 1.76476    │ 1.32844     │  0.867926    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ RidgeRegressor            │ 0.919191    │ 2.12432    │ 1.45751     │  0.841016    │
├───────────────────────────┼─────────────┼────────────┼─────────────┼──────────────┤
│ LassoRegressor            │ 0.922047    │ 2.15536    │ 1.46812     │  0.838693    │
╘═══════════════════════════╧═════════════╧════════════╧═════════════╧══════════════╛
```

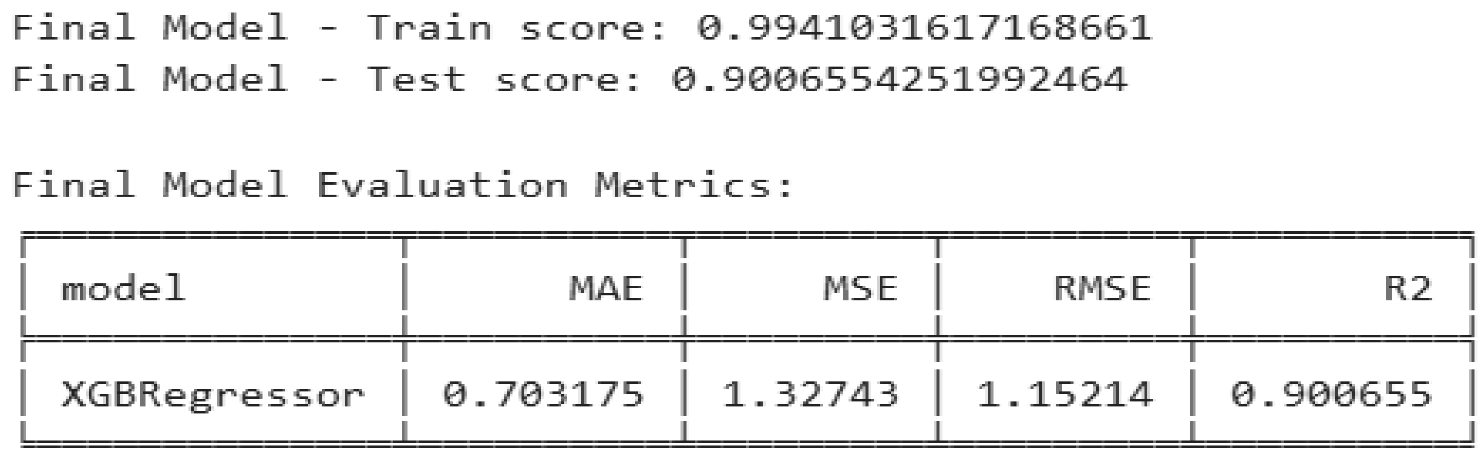
XGBoost:

(Extreme Gradient Boosting) is a powerful machine learning algorithm that is widely used for supervised learning tasks such as regression and classification. It is known for its performance, speed, and scalability in handling large datasets.In the context of regression tasks, XGBRegressor is an implementation of the gradient boosting machine algorithm, specifically designed for regression problems. It builds an ensemble of decision trees, where each subsequent tree attempts to correct the errors of the previous one.Key Features of XGBoost Regressor:

- Gradient Boosting: XGBoost improves upon traditional boosting by using gradient descent to minimize the error in predictions.
- Regularization: XGBoost includes L1 (Lasso) and L2 (Ridge) regularization, which helps in reducing overfitting.
- Handling Missing Data: XGBoost can automatically handle missing values, making it robust when working with incomplete datasets.
- Feature Importance: It can provide insights into which features have the most significant impact on the predictions.

9.Hyperparameter Tuning Using Optuna: Optuna is used to tune the hyperparameters of the XGBoost model:Hyperparameters such as n_estimators, max_depth, learning_rate, subsample, and others are tuned.The best hyperparameters are identified using cross-validation and stored in best_params.Final Model: The best model is trained using the tuned hyperparameters and evaluated on the test set.

10.Model Prediction and Saving: final trained model is used for making predictions on new data.The predicted car prices are displayed along with the actual values.The final model is saved using Joblib for later use in deployment.



```
Final Model - Train score: 0.9941031617168661
Final Model - Test score: 0.9006554251992464

Final Model Evaluation Metrics:
╒══════════════╤══════════╤═════════╤═════════╤══════════╕
│ model        │      MAE │     MSE │    RMSE │       R2 │
╞══════════════╪══════════╪═════════╪═════════╪══════════╡
│ XGBRegressor │ 0.703175 │ 1.32743 │ 1.15214 │ 0.900655 │
╘══════════════╧══════════╧═════════╧═════════╧══════════╛
```

11.Results:

• Achieved the best performance with the highest R^2 and the lowest MSE/MAE, making it the chosen model for deployment.
The XGBoost Regressor performed the best based on evaluation metrics (R^2 score).The model's performance was improved with hyperparameter tuning using Optuna.Regularization techniques (Ridge and Lasso) were applied to improve model generalization.The model is ready for deployment for real-time car price predictions.

12.Pipeline:

• Modular Structure: The pipeline is structured in a modular fashion, allowing for clear separation between data preprocessing and model training. This enhances the readability and maintainability of the code, making it easier to modify individual components without affecting the overall workflow.
• Data Preprocessing: The pipeline incorporates two distinct preprocessing steps for handling different types of data: numerical and categorical. Numerical features are scaled using the StandardScaler, while categorical features are transformed using the specified encoder. This ensures that each type of data is processed appropriately to improve model performance.

• ColumnTransformer Usage: By utilizing ColumnTransformer, the pipeline efficiently applies different preprocessing techniques to specified columns in the dataset. This allows for simultaneous handling of multiple feature types, optimizing the data preparation process and ensuring that the model receives data in the correct format.
• Integration of Model: The final model, trained using the xgboost regression algorithm, is integrated into the pipeline. This means that the model is directly trained on the preprocessed data, streamlining the workflow from data input to predictions and making it straightforward to apply the same preprocessing steps to new data when making predictions.
• Reproducibility and Consistency: By encapsulating the entire workflow (from preprocessing to model training) in a single pipeline, the approach ensures reproducibility. It guarantees that the same preprocessing steps are applied consistently during both training and inference, which is crucial for achieving reliable predictions and validating model performance.

C. Model Deployement- Streamlit

• Streamlit is an open-source Python library designed for quickly building custom web applications tailored for data science and machine learning tasks. Its ease of use and adaptability make it an excellent choice for deploying machine learning models in interactive applications

1. Features of the Application :

i) User Input Interface:

• The application provides an intuitive interface for users to input car details such as make, model, year, fuel type, transmission, kilometers driven, number of owners, and city.

• Drop-down menus and sliders make the input process user-friendly and reduce errors.

ii) Price Prediction:

• Upon receiving user inputs, the application leverages the trained Xgboost model to predict the car's price.

• The predicted price is displayed instantly, enhancing the user experience.

. Backend Implementation

iii) Model Loading:

• The trained xgboost model, Standardscaler, Labelencoder are loaded into the application using the joblib library, ensuring it is ready for predictions.

• Data Preprocessing: User inputs are preprocessed in the same way as the training data, ensuring consistency and accuracy in predictions.

Conclusion -Project Impact

• Deploying the predictive model through the Streamlit application revolutionizes the user experience at CarDekho by delivering swift and reliable price estimates for used cars.

• This real-time pricing tool empowers customers with data-driven insights, enabling them to make informed decisions when buying or selling vehicles. For sales representatives, it simplifies the valuation process, saving time and ensuring consistency across pricing.

• Moreover, this deployment lays the groundwork for future innovations, where advanced features like personalized recommendations or integration with real-time market data can further enhance the accuracy and sophistication of the predictions