CarDheko - Used Car Price Prediction (Regression Model) Project Report

By: SAMUELSON G

Introduction:

In this project, we aim to enhance the customer experience and optimize the pricing process for used cars by developing a machine learning model. This model utilizes historical car price data and various features such as the car's make, model, year, fuel type, and transmission type to accurately predict used car prices. The model will be integrated into an interactive web application using Streamlit, allowing users to input car details and receive real-time price predictions.

A. Approach:

1. Import files and Data wrangling:

- Load the datasets from multiple cities, which are in unstructured format (like containing text, JSON-like entries, different data types, missing value 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. Feature Selection:

Categorical Features: Attributes such as fuel type, body type, and Brand, Insurance Validity, Color, city, transmission.

Numerical Features: Variables are Ownerno, Model year, kms driven, Milage, Engine, Seats were cleaned and converted to integers.

5. Encoding and Scaling:

• OneHot Encoding is used to convert categorical data into numerical form, which is often required by machine learning models.

• Standard Scaler ensures that all numerical features are on the same scale, preventing certain features from dominating others.

B. Model Development

1. **Train-Test Split:** Split the dataset into training and testing sets to evaluate model performance.
    - Common split ratios are 70-25

2. **Model Selection:**

    i) Linear Regression:
    - Overview: Linear Regression was chosen as the baseline model due to its simplicity and ease of interpretation.
    - Regularization: Ridge and Lasso regression were applied to prevent overfitting.

    ii) Gradient Boosting Regressor (GBR):
    - GradientBoostingRegressor is a machine learning technique used for regression tasks. It is part of the ensemble learning family
    - Gradient Boosting is based on the idea of fitting new models to the residuals (errors) of the previous models.

    iii) Decision Tree Regressor:
    - Decision Trees were chosen for their interpretability and capability to model nonlinear relationships.
    - Pruning was applied to prevent overfitting by limiting the tree depth.

    iv) Random Forest Regressor
    - A Random Forest is an ensemble of decision trees. In the case of regression, each decision tree predicts a value, and the final prediction is the average of all the trees' outputs.
    - Each tree is trained on a different random subset of the data (using bootstrapping).
    - For each node in a tree, only a random subset of features is considered for splitting.

3. **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.

R^2 Score: Indicates how well the independent variables explain the variance in the dependent.

4. **Model Justification**

After evaluating multiple regression models, the Random Forest Regressor was selected as the final model for deployment. The justification for this choice includes:

- **Handling Non-Linearity**: Random Forest can capture complex relationships in the data, making it suitable for predicting prices based on various features.
- **Reduced Overfitting**: As an ensemble method, it mitigates overfitting by averaging the predictions of multiple decision trees.
- **Performance Metrics**: It achieved the best performance metrics among all models tested, including the lowest Mean Absolute Error (MAE) and Mean Squared Error (MSE).

Model Comparison:

Model	MAE	MSE	RMSE	R2
LinearRegression	9.440573e+12	8.305860e+27	9.113649e+13	-6.216080e+26
DecisionTreeRegressor	1.035340e+00	2.902112e+00	1.703559e+00	7.828069e-01
RandomForestRegressor	7.787296e-01	1.582703e+00	1.258055e+00	8.815510e-01
GradientBoostingRegressor	1.134603e+00	2.642666e+00	1.625628e+00	8.022237e-01
RidgeRegressor	2.124322e+00	9.191905e-01	1.457505e+00	8.410164e-01
LassoRegressor	2.155362e+00	9.220470e-01	1.468115e+00	8.386934e-01

5. Results:

Random Forest:

• Achieved the best performance with the highest R^2 and the lowest MSE/MAE, making it the chosen model for deployment.

• **Hyperparameter Tuning:** Grid Search was employed to identify the optimal parameters, such as n_estimators and max_depth. By systematically testing a range of values for these parameters, Grid Search helped in determining the best combination that enhances the Random forest model's performance.

6. 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 Column Transformer, 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 Random Forest regression algorithm, is integrated into the pipeline. This means that the model is directly trained on the pre-processed 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, kilometres 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 Random Forest model to predict the car's price.
• The predicted price is displayed instantly, enhancing the user experience.
. Backend Implementation

iii) Model Loading:

• The trained Random Forest model, Standardscaler, Labelencoder are loaded into the application using the pickle 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. D. 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.

Limitations

While the Random Forest model performed well, there are certain limitations to consider:

- **Data Quality**: The accuracy of the predictions is highly dependent on the quality and completeness of the input data. Missing or erroneous data can lead to inaccurate predictions.
- **Feature Selection**: The model's performance is influenced by the features included. Some relevant features may have been omitted, which could enhance predictive power.
- **Market Variability**: The used car market can be volatile, and external factors (e.g., economic conditions, changes in consumer preferences) may impact car prices in ways not captured by the model.

Future Work

To further enhance the model's accuracy and usability, the following areas for future work are proposed:

- **Feature Expansion**: Incorporating additional features such as car condition, service history, and market trends could improve prediction accuracy.
- **Advanced Techniques**: Exploring more complex algorithms, such as deep learning methods, may yield better results.
- **Real-Time Data Integration**: Integrating real-time market data could allow for dynamic pricing adjustments based on current market conditions.

Conclusion

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 informed decisions when buying or selling vehicles. For sales representatives, it simplifies the valuation process, saving time and ensuring consistency across pricing.