Car Price Prediction Using Machine Learning

Mehmet Duman, 17158534936

Department of Computer Engineering, mehmet_duman@ogr.eskisehir.edu.tr

***Abstract*—This project predicts the prices of cars using machine learning techniques to provide a data-driven approach for the resale automotive market. The dataset used is from Kaggle, containing 205 records with attributes such as engine size, curb weight, horsepower, and fuel type. I explored and compared three machine learning models: Linear Regression, Random Forest, and XGBoost. The best performance was exhibited by the Random Forest model, with an RMSE of 1861.35 and an R^2 score of 0.96. Further tuning was done using GridSearchCV on this model. The important features that influence the car prices were identified; from those, engine size and curb weight are the strongest predictors of car prices. The final model does reliable price predictions that could help buyers and sellers in decision-making.**

***Index Terms*—Machine Learning, Car Price Prediction, Regression Models, Random Forest, Hyperparameter Tuning, Feature Importance, Data Preprocessing, GridSearchCV, Model Evaluation, Kaggle Dataset.**

I. Introduction

Setting the correct price for used cars is a very big challenge in today's automotive market. Traditional pricing methods normally depend on subjective assessment or limited data, which mostly results in inconsistencies and inaccuracies. A wrong estimate of the price may lead to financial losses for the seller or overpayment by the buyer; therefore, there is a need to develop a reliable and objective method of pricing.

The project addresses the problem of car price prediction using machine learning techniques, hence providing a data-driven approach. Machine learning models are capable of identifying patterns and relationships that affect car prices from historical data represented with features such as engine size, curb weight, horsepower, fuel type, and car dimensions.

This project is important because it will lead to better price estimates and, hence, fair resale with more transparency. It will also help various stakeholders—be it individual sellers or buyers, dealerships, or online car resale platforms. More precisely, it focuses on developing a model that makes accurate resale price predictions and identifies the key factors contributing to those estimates.

II. Proposed Methodology

Model Selection

For this project, I selected the following machine learning models for car price prediction:

Linear Regression:
Reason: A simple, interpretable baseline model that assumes a linear relationship between features and the target variable. It serves as a foundational benchmark to compare against more complex models.

Random Forest Regressor:
Reason: A robust ensemble method that can handle non-linear relationships and interactions between features.It reduces overfitting by aggregating multiple decision trees, making it suitable for datasets with moderate complexity.

XGBoost (Extreme Gradient Boosting):
Reason: A powerful gradient-boosting model known for its high performance and efficiency. XGBoost handles large datasets well, offers regularization to control overfitting, and is often used in machine learning competitions. These models provide a range of complexity, allowing me to compare simpler linear methods with more advanced ensemble and boosting techniques.

Implementation Strategy Data Preprocessing:

Convert categorical features (e.g., fueltype, carbody, brand) to numerical values using Label Encoding. Replace text values in columns like doornumber and cylindernumber with their corresponding numerical equivalents. Standardize numerical features using StandardScaler to ensure consistent scaling for all models.

Train-Test Split:

Split the dataset into 80% training and 20% testing to train the models and evaluate their performance on unseen data.

Model Training:

Train each model on the training data. For the Random Forest and XGBoost models, perform hyperparameter tuning using GridSearchCV to optimize performance.

Prediction:

Use the trained models to predict car prices on the test dataset.

To assess the performance of each model, I use the following metrics:

Root Mean Squared Error (RMSE):

Measures the average magnitude of prediction errors. Lower values indicate better performance.
Formula:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Mean Absolute Error (MAE):

Measures the average absolute difference between predicted and actual values. It is less sensitive to outliers compared to

RMSE.
Formula:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

R^2 Score (Coefficient of Determination)

Indicates how well the model explains the variability of the target variable. A score closer to 1 is better.
Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Testing Setup:

- **Hardware:**
  - **CPU:** Intel Core i7 7700HQ
  - **RAM:** 32 GB
  - **Storage:** 256 GB SSD
- **Software:**
  - **Operating System:** Windows 10
  - **Programming Language:** Python 3.9
  - **Libraries:**
    * `pandas` (Data Manipulation)
    * `numpy` (Numerical Operations)
    * `matplotlib` and `seaborn` (Data Visualization)
    * `scikit-learn` (Machine Learning Models and Utilities)
    * `xgboost` (XGBoost Model)
- **Environment:**
  - **Development Environment:** Jupyter Notebook in Visual Studio Code
  - **Virtual Environment:** Anaconda

III. Experiments and Results

1. Model Training and Evaluation
The dataset was split into 80% training and 20% testing to ensure that the models were evaluated on unseen data. The following machine learning models were trained:

- Linear Regression
- Random Forest
- XGBoost

The models' performance was evaluated using the following metrics:

- Root Mean Squared Error (RMSE): Measures the average magnitude of prediction errors.
- Mean Absolute Error (MAE): Measures the average absolute difference between predicted and actual values.
- R² Score: Represents how well the model explains the variability of the target variable.

2. Performance Comparison of Models
The Table I* summarizes the performance of each model:

From the results, the Random Forest model outperformed the other models, achieving the lowest RMSE and MAE, along with the highest R² Score. This indicated that the Random Forest model was the most suitable for predicting car prices.

Model	RMSE	MAE	R^2
Linear Regression	3475.90	2102.01	0.85
Random Forest	1855.76	1304.89	0.96
XGBoost	2385.04	1682.92	0.93

TABLE I

Metric	Before Tuning	After Tuning
RMSE	1855.76	1861.35
MAE	1304.89	1297.17
R^2 Score	0.96	0.96

TABLE II

3. Hyper-parameter Tuning To further optimize the Random Forest model, GridSearchCV was employed for hyper-parameter tuning. The following hyper-parameters were tuned:

- max_depth: Maximum depth of each tree
- min_samples_leaf: Minimum number of samples required at a leaf node
- min_samples_split: Minimum number of samples required to split a node
- n_estimators: Number of trees in the forest

After the hyper-parameter tuning process, the best hyper-parameters identified were:

- max_depth: 10
- min_samples_leaf: 1
- min_samples_split: 2
- n_estimators: 300

The performance of the Random Forest model after hyper-parameter tuning is shown in Table II**

IV. Code Explanation

A. 1. Importing Libraries

- **Pandas** and **NumPy**: For data manipulation and numerical operations.
- **Matplotlib** and **Seaborn**: For data visualization.
- **Scikit-learn**: For preprocessing, model training, evaluation, and hyperparameter tuning.
- **XGBoost**: For the XGBoost regression model.

B. 2. Data Loading and Initial Exploration

- **Dataset Loading:** The dataset is loaded using the `pandas read_csv()` function.
- **Initial Data Exploration:**
  - `df.head()`: Displays the first few rows of the dataset.
  - `df.shape`: Returns the number of rows and columns.
  - `df.info()`: Provides information about the dataset, including column names and data types.
  - `df.describe()`: Generates summary statistics for numerical columns.
  - `df.isnull().sum()`: Checks for missing values.

C. 3. Exploratory Data Analysis (EDA)

- **Distribution of Car Prices:** Visualized using a histogram with `sns.histplot()` to understand the distribution of the target variable (`price`).
- **Correlation Matrix Heatmap:** Visualizes correlations between numerical features using `sns.heatmap()`.
- **Distribution of Numerical Features:** Plots histograms for various numerical features to analyze their distributions.

D. 4. Data Preprocessing

- **Feature Engineering:**
  - Dropping `car_ID` as it is not relevant for prediction.
  - Extracting car brand from the `CarName` column.
  - Mapping `doornumber` and `cylindernumber` to integers.
- **Categorical Encoding:**
  - Identifies categorical and numerical columns.
  - Uses `LabelEncoder` to convert categorical columns into numerical format.

E. 5. Train-Test Splitting

- Splits the dataset into **80% training** and **20% testing** sets using `train_test_split()`.

F. 6. Feature Scaling

- Uses `StandardScaler` to scale the numerical features, ensuring they are on the same scale, which improves model performance.

G. 7. Model Training and Evaluation

- **Linear Regression:** Trained using the `LinearRegression` class.
- **Random Forest:** Trained using the `RandomForestRegressor` class with default parameters.
- **XGBoost:** Trained using the `XGBRegressor` class.
- **Evaluation:** The `evaluate_model()` function calculates:
  - **Root Mean Squared Error (RMSE)**
  - **Mean Absolute Error (MAE)**
  - **R^2 Score**

H. 8. Hyperparameter Tuning

- Uses **GridSearchCV** to optimize the hyperparameters of the `RandomForestRegressor`.
- Tuned hyperparameters include:
  - `n_estimators`: Number of trees in the forest.
  - `max_depth`: Maximum depth of each tree.
  - `min_samples_split`: Minimum number of samples required to split a node.
  - `min_samples_leaf`: Minimum number of samples required at each leaf node.

I. 9. Feature Importance Analysis

- Uses the `feature_importances_` attribute of the Random Forest model to identify the most influential features.
- Visualizes feature importance using a bar chart with `sns.barplot()`.

J. 10. Code Organization

- **Data Loading and Preprocessing:** Functions such as `read_csv()`, `drop()`, and `map()`.
- **EDA:** Visualization functions like `sns.histplot()` and `sns.heatmap()`.
- **Model Training:** Classes like `LinearRegression`, `RandomForestRegressor`, and `XGBRegressor`.
- **Evaluation:** The `evaluate_model()` function for metrics and plotting.
- **Hyperparameter Tuning:** Using `GridSearchCV`.

V. Discussion and Conclusion

In this project, I aimed to predict car prices using machine learning techniques, focusing on three models: Linear Regression, Random Forest, and XGBoost. Following data preprocessing, exploratory data analysis (EDA), and model evaluation, the Random Forest model stood out as the best performer, achieving an RMSE of 1855.76 and an R^2 score of 0.96. Hyperparameter tuning using GridSearchCV further improved the Random Forest model's performance, reducing the Mean Absolute Error (MAE) to 1297.17. This strong performance suggests that the Random Forest model effectively captures the non-linear relationships and interactions within the data, making it ideal for car price prediction.

The analysis of feature importance highlighted engine size, curb weight, and horsepower as the most influential factors in determining car prices. This implies that vehicles with larger engines, higher curb weight, and greater horsepower are typically priced higher. Additionally, features like highway miles per gallon (MPG) offered secondary insights into vehicle efficiency and value. EDA revealed that car prices are right-skewed, meaning lower-priced cars are more common, and correlation analysis confirmed strong relationships between car prices and key numerical features.

These findings have practical implications for various stakeholders. For buyers and sellers, the model offers a data-driven approach to pricing, helping sellers set competitive prices and enabling buyers to make informed purchasing decisions. Car dealerships and online resale platforms can incorporate this predictive model into their pricing strategies, enhancing transparency and trust. Understanding the primary factors influencing car prices helps prioritize key aspects during evaluations and negotiations.

Despite its strong performance, the model has some limitations. The dataset does not account for subjective factors like vehicle condition, mileage, market trends, and regional preferences, which can also impact car prices. Including these variables in future iterations could further enhance prediction accuracy.

In conclusion, this study demonstrates the effectiveness of machine learning techniques in predicting car prices, showcasing the robustness of the Random Forest model for this task. Leveraging data-driven approaches can lead to more accurate, consistent, and objective pricing in the car resale market. Future work can focus on integrating more comprehensive datasets to improve the model's real-world applicability.

VI. Ethical Statement

I confirm that all experimental work in this project was carried out by me with integrity. All sources and methods used in the study have been properly cited and included in the references. No plagiarism has been committed in any part of this study.

VII. AI Tools Usage

In this project, AI tools were utilized in various stages to enhance the quality of the report and the implementation of the code. The specific areas where AI tools were applied include:

- **Formatting and LaTeX Code Generation:** AI tools were used to generate the necessary LaTeX formatting codes and mathematical formulas to ensure the documentation is presented professionally and clearly.
- **Language Support:** In sections where my proficiency in English was insufficient, AI tools were consulted to improve the clarity and correctness of the language used in the text.
- **Debugging and Guidance:** During the coding process, AI tools provided support in identifying and resolving errors, as well as suggesting appropriate steps to take when facing issues with code implementation and model training.

These AI tools significantly contributed to the overall efficiency and quality of the project, ensuring that both the documentation and code were well-structured and accurate.

VIII. References

1) H. Buoy, "Car Price Prediction Dataset," Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/hellbuoy/car-price-prediction/data. [Accessed: June 18, 2024].
2) F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
3) T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, 2016, pp. 785–794.
4) W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference (SciPy 2010)*, 2010, pp. 56–61.
5) J. Hunter, "Matplotlib: A 2D Graphics Environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
6) M. L. Waskom, "Seaborn: Statistical Data Visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021.
7) Stack Overflow User, "How to map values in a DataFrame column," Stack Overflow, 2016. [Online]. Available: https://stackoverflow.com/questions/19798153/how-to-map-values-in-a-dataframe-column. [Accessed: June 18, 2024].
8) Stack Overflow User, "How to encode categorical features using LabelEncoder," Stack Overflow, 2017. [Online]. Available: https://stackoverflow.com/questions/24458645/label-encoding-across-multiple-columns-in-scikit-learn. [Accessed: June 18, 2024].
9) Stack Overflow User, "How to create a heatmap in Seaborn," Stack Overflow, 2015. [Online]. Available: https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-seaborn. [Accessed: June 18, 2024].
10) Stack Overflow User, "How to split data into train and test sets," Stack Overflow, 2013. [Online]. Available: https://stackoverflow.com/questions/38250710/how-to-split-data-into-train-and-test-sets-in-sklearn. [Accessed: June 18, 2024].
11) Stack Overflow User, "Hyperparameter tuning with GridSearchCV," Stack Overflow, 2018. [Online]. Available: https://stackoverflow.com/questions/42222325/how-to-use-gridsearchcv-in-sklearn. [Accessed: June 18, 2024].