Final Predictive Model with User Guide

# Data Analysis and Model Training Report

This document provides a detailed overview of the data analysis, preprocessing, and predictive modeling steps used in the project. The final model is based on Gradient Boosting, with the best performance metrics for predicting car prices across various cities.

## Data Cleaning and Preprocessing

The dataset from multiple cities was combined, and unnecessary columns were dropped. The data cleaning process involved handling missing values, extracting relevant details from nested columns, and scaling numerical features. Additionally, one-hot encoding and label encoding were applied to categorical features.

## Exploratory Data Analysis (EDA)

Several visualizations were created to understand the relationships between features and target variables. The scatter plot shows the relationship between kilometers driven and price, while the heatmap visualizes the correlation between numerical features. The box plot provides insights into price distribution by ownership status.

## Model Selection and Hyperparameter Tuning

Various models were tested, including Linear Regression, Lasso, Ridge, Decision Tree, Random Forest, and Gradient Boosting. The best model was selected based on the lowest Mean Squared Error (MSE) and highest R² score after hyperparameter tuning. Gradient Boosting performed best, with optimized hyperparameters achieving a high R² score and low error.

## Final Model Performance

The final Gradient Boosting model achieved the following performance on the test dataset:  
• Test MSE: 0.0000  
• Test MAE: 0.0021  
• Test R²: 0.9601  
These metrics indicate that the model has a high level of accuracy in predicting car prices.

## Conclusion

The Gradient Boosting model has been saved as a trained model using joblib, and the model columns have been stored. The model is ready for deployment and further analysis.

## User Guide

To use the model for predictions, follow these steps:  
1. Load the model: `best\_model = joblib.load("best\_model7.pkl")`  
2. Load the columns: `model\_columns = joblib.load("model\_columns7.pkl")`  
3. Prepare your input data following the format of the model columns.  
4. Use the model to predict car prices using the `predict` method: `predictions = best\_model.predict(new\_data)`  
5. Ensure that the new data has the same preprocessing steps applied, including scaling and encoding as done during model training.  
6. If you are working with multiple inputs, consider using batch prediction techniques to speed up the process.

**Example Prediction Code:**  
```python  
import joblib  
import pandas as pd  
best\_model = joblib.load("best\_model7.pkl")  
model\_columns = joblib.load("model\_columns7.pkl")  
new\_data = pd.DataFrame({  
 "kilometers\_driven": [45000],  
 "engine\_displacement": [1198],  
 "Seats": [5],  
 # Add other features as per model\_columns...  
})  
predictions = best\_model.predict(new\_data)  
print(predictions)  
```

## Model Comparison Visualizations

