CAR PRICE PREDICTION

By

Hamim Sheikh,	ID:20215103049
Joy Gupta,	ID:20215103035
Mosharaf Hossain,	ID:20215103022

Submitted in partial fulfillment of the requirements of the degree of

Bachelor of Science in

Computer Science and Engineering

Department of Computer Science and Engineering

Bangladesh University of Business and Technology

November, 2023

Declaration

We herby declare that the project work entitled "CAR PRICE PREDICTION" using Python programmer submitted to BANGLADESH UNIVERSITY OF BUSINESS TECHNOLOGY, is a record of an original work done by us under the guidance of our Lecturer, Shamim Ahmed .We think that this report has not been submitted previously to any other university for any examination.

Hamim Sheikh
ID: 20215103049 _____ Signature

Joy Gupta
ID: 20215103035 _____ Signature

Mosharaf Hossain
ID: 20215103022 _____ Signature

Approval

Firstly, We would like to thank our Course teacher Shamim Ahmed , Assistant Professor, Deparment of Computer Science and Engineering, Bangladesh University of Business and Technology (BUBT) and Our Project Team Member's for their enormous support and encouragement.

Acknowledgement

We would like to express our heartly gratitude to the almighty Allah who offered upon our family and us kind care throughout this journey until the fulfillment of this project. Firstly, We would like to thank our Course Teacher Shamim Ahmed, Assistant Professor, Deparment of Computer Science and Engineering, Bangladesh University of Business and Technology (BUBT) and our Team Member's for their enormous support and encouragement. we should also like to thank my Faculty In charge Nourin Khandokar who helped clear any doubt what so ever we had to encounter while making this project. Lastly, We should like to thank our Family for the tremendous amount of support they offered us while making this Project. we also were the invaluable support of our friends and colleagues which helped us in completing this Project.

Abstract

The sole intention behind the consideration of this Project is to generate and manage a simple database for question. This project is developed considering " CAR PRICE PREDICTION ".

List of Figures

Contents

Chapter-1: Introduction

1.1 Introduction

In this fast world, you don't have your own personal mode of transportation sort of an automobile, life will become even additional agitated. The public choose to obtain their automobile as a result of its convenience to commute between places, permits movement with an out-sized cluster of individuals with fuel potency, and safe mode of transport. The used automobile marketplace is witnessing a boom in India, with the decision for luxurious vehicles sometimes increasing. Till a couple of years, owning a luxury automobile won't be a dream for varied shoppers, as a result of money hurdles, however, this is often bit by bit dynamic as shoppers can simply obtain used luxury vehicles. Machine Learning provides numerous ways through that it's easier to predict the worth of an automobile, by the previous information that is obtainable. We've enforced the model exploitation supervised Learning techniques of Machine Learning, which is outlined by its use of labeled information sets to coach algorithms to classify data or predict outcomes accurately. As the input file is fed into the model, it adjusts its weights till the model has been fitted fittingly, which happens

as a part of the cross-validation method. If there is also further transparency within the marketplace and fewer intermediaries,the seller ought to get the next value for a vehicle and therefore the shopper ought to get one at a lower fee as margins get reduced on every facet.

1.2 Motivation

The motivation behind car price prediction stems from various factors and the practical utility it offers to different stakeholders in the automotive industry and beyond.

Here are some key motivations for car price prediction:

1. Informed Buying and Selling Decisions.
2. Competitive Advantage for Dealerships.
3. Inventory Management.
4. Loan Assessment.
5. Improved Customer Experience.
6. Market Insights.
7. Time Efficiency.

1.3 Existing System

The existing system of car price prediction typically relies on a combination of traditional valuation methods and data-driven analysis. Automotive industry experts and appraisers often use established techniques such as the Kelley Blue Book (KBB) or the National Automobile Dealers Association

(NADA) guides, which provide standardized pricing based on factors like the vehicle's make, model, year, and condition.

Furthermore, some dealerships and online platforms utilize regression analysis and market trend assessments to estimate the value of used cars. This involves comparing the prices of similar vehicles in the market, considering variables such as mileage, maintenance history, geographical location, and demand-supply dynamics. However, these methods have limitations, as they often fail to incorporate the full spectrum of factors that influence car prices in a dynamic market environment.

Additionally, some organizations have started integrating machine learning and artificial intelligence techniques into their pricing models. These models leverage historical data, consumer preferences, macroeconomic indicators, and emerging trends to develop more sophisticated and accurate predictions. They employ algorithms like regression analysis, decision trees, and neural networks to identify patterns and correlations within large datasets, enabling more precise forecasting of car prices.

1.4 Problem with Existing System

For the purposes of car valuation, popular guides tend not to use machine learning. Instead, they source data from local sales and average the prices of many similar cars. This method works well if you have a common car with a common set of features. The condition of the car is judged very roughly, typically on a scale of one to three. Cars that are "unusual" are therefore hard to evaluate. Effectively, no inferences are drawn from similar cars but from a different make and model, whereas with machine learning, the entirety

of the data set and its features are used to train the model predictions. Using machine learning is a solution to the problem of utilization of all the data and will assist in utilizing all the features of a car to make valuations.

1.5 Proposed System

There are two primary phases in the system:

1. Training phase: The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly.

2. Testing phase: the system is provided with the inputs and is tested for its working. The accuracy is checked. And therefore, the data that is used to train the model or test it, has to be appropriate.

The system is designed to detect and predict price of used car and hence appropriate algorithms must be used to do the two different tasks. Before the algorithms are selected for further use, different algorithms were compared for its accuracy. The well-suite done for the task was chosen.

1.6 Project Overview

The car price prediction project aims to develop a robust machine learning model capable of accurately estimating the market value of used cars. Leveraging historical data, the model will be trained to analyze various factors such as make, model, mileage, production year, and additional features to predict the appropriate price range for a given vehicle. Through this project, we aim to provide a reliable tool for both buyers and sellers in the automotive market, facilitating informed decision-making and fair pricing.

Furthermore, the project will contribute to the advancement of the automotive industry by integrating cutting-edge machine learning techniques and enhancing transparency in car transactions.

1.7 Project Scope

The car price prediction project will involve the development of a comprehensive machine learning system that focuses on estimating the market value of used cars. This system will encompass data collection, pre-processing, feature engineering, model selection, and evaluation. The project will incorporate a diverse range of factors including car specifications, historical pricing trends, geographic location, and economic indicators to ensure an accurate prediction of car prices. Additionally, the system will be designed to accommodate scalability and adaptability, allowing for the integration of real-time data and potential expansion to include new car models and markets. The project scope will not extend to the development of a transactional platform, but will concentrate on providing a reliable tool for price estimation in the automotive market.

1.8 Aims and Objectives

This car price prediction project aims to develop an advanced machine learning solution for accurately estimating the market value of used cars. The key objectives include: Constructing a comprehensive database of historical car sales data to facilitate accurate price prediction. Implementing a robust machine learning algorithm that can effectively analyze various car

features and market trends to generate precise price estimations. Enhancing transparency in the automotive market by providing fair and reliable pricing information for both buyers and sellers. Offering a user-friendly interface that allows users to input specific car details and receive an instant price prediction, thereby facilitating informed decision-making in car transactions. Continuously updating and refining the model based on new data inputs and emerging market trends to ensure the system remains up-to-date and dependable for users. Through these objectives, the project seeks to contribute to a more transparent and efficient automotive market while providing users with a valuable tool for making well-informed car purchasing and selling decisions.

1.9 Organisation of Report

Car price prediction is an active research topic that requires a lot of data and effort. The recent emergence of online marketplaces has strengthened the need for users and sellers to stay informed about the most recent market trends related to the subject. Our goal is to create a reliable and accurate model to forecast the cost of a used car based on a set of characteristics. To this end, we are utilizing the machine learning algorithm linear regression. Our model is based on a combination of linear regression and decision tree algorithms. The model was able to predict car prices with an accuracy of over 90 percentage Random Forest is well-suited for car price prediction because it is a powerful machine-learning algorithm that is capable of handling a high number of input features and modeling complex relationships between these features.The Machine learning model is fit with data consisting of

various variables for training. We have shown how the data is modified to increase accuracy and efficiency. The number of different attributes is measured and also it has been considerable to predict the result in more reliable and accurate. To find the price of used vehicles a well defined model has been developed with the help of three machine learning techniques such as Artificial Neural Network, Support Vector Machine and Random Forest. The model is developed using the forest regressor algorithm, which is a machine learning technique used for regression analysis. The data is collected from various sources, and after pre-processing, the model is trained to estimate the car price based on the input parameters.

Chapter-2: LITERATURE REVIEW

2.1 Introduction

The literature review section provides a comprehensive overview of the existing research and methodologies employed in the domain of car price prediction. This analysis aims to identify the key trends, challenges, and advancements in the field, laying the foundation for the current study. By examining previous studies and relevant literature, this review intends to build upon the existing knowledge and bridge any gaps in understanding. Through an exploration of various approaches and techniques utilized by researchers and practitioners, this literature review serves as a critical reference point for the development and implementation of an effective car price prediction model.

2.2 Security

In the car price prediction project, several security measures are implemented to safeguard sensitive data and ensure the integrity of the system.
These measures include:

Data Encryption: Employing robust encryption techniques to protect confidential data during storage and transmission, thereby preventing unauthorized access.

Access Control: Implementing strict access controls and user authentication protocols to restrict system access only to authorized personnel, thereby preventing data breaches.

Anonymization: Removing any personally identifiable information from the data set to ensure the privacy and anonymity of individuals and organizations involved.

Regular Security Audits: Conducting routine security audits to identify and address potential vulnerabilities or threats, thereby ensuring the system's resilience against cyber attacks.

Secure Data Transfer: Utilizing secure protocols and encrypted channels for data transfer between the user interface and the back-end system to prevent interception and unauthorized access.

Compliance with Data Protection Regulations: Adhering to relevant data protection regulations such as GDPR or other regional laws to ensure the ethical and legal handling of sensitive data.

By implementing these security measures, the car price prediction project aims to uphold the confidentiality, integrity, and availability of data, thereby fostering user trust and confidence in the system.

2.3 Technological components

Python was the major technology used for the implementation of machine learning concepts the reason being that there are numerous inbuilt methods

in the form of packaged libraries present in python. Following are prominent libraries/tools we used in our project.

2.3.1 NUMPY

NumPy is a general-purpose array-processing package. it provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

2.3.2 SCIPY

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas, and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack. The SciPy library is currently distributed under the BSD license, and its development is sponsored and supported by an open community of developers.

2.3.3 SCIKIT-LEARN

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built

2.3.4 JUPYTER

NOTEBOOK The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It includes data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text[3]. It includes data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

2.4 Network Technology/Infrastructure

The car price prediction system relies on a robust network technology infrastructure to ensure seamless data flow, secure communication, and efficient processing. The key components of the network technology and infrastructure include:

High-Speed Internet: Utilizing high-speed internet connectivity to enable fast and reliable communication between different components of the system,

ensuring quick data transfer and analysis.

Secure Network Protocols: Implementing secure network protocols such as Transport Layer Security (TLS) and Secure Shell (SSH) to safeguard data during transmission and prevent unauthorized access or data breaches.

Cloud Computing Services: Leveraging cloud computing services to host and manage the system's data and applications, allowing for scalability, flexibility, and accessibility from multiple locations.

Virtual Private Networks (VPNs): Implementing VPNs to create secure and encrypted connections, particularly when accessing the system remotely, ensuring the protection of sensitive data and maintaining privacy.

Redundant Architecture: Deploying redundant network architecture to minimize the risk of system downtime or data loss, thereby ensuring continuous availability and reliability of the car price prediction system.

Firewall and Intrusion Detection Systems: Installing firewalls and intrusion detection systems to monitor network traffic, detect potential threats, and prevent unauthorized access, thereby bolstering the overall security of the network infrastructure.

By incorporating these network technologies and infrastructure components, the car price prediction system aims to establish a secure, efficient, and resilient network environment, facilitating smooth data processing and seamless communication between various system components and users.

2.5 Algorithm/methodology

The project deals with used cars. Using Parse Hub, the benchmark data set from dubizzle.ae and buyanycar.com was scraped in order to build the effective intelligent model. The project's methodology is as follows:

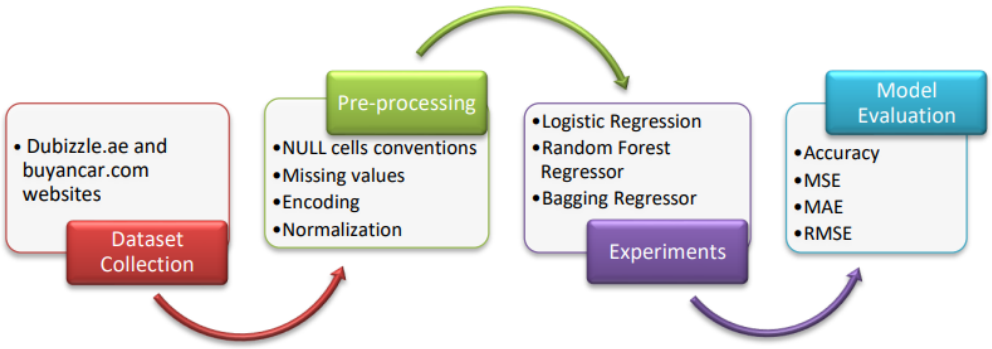


Figure 2.1. Proposed Methodology

After data collection the data-set was pre-processed to remove samples that have missing value, and remove non-numerical part from numerical attributes, converting categorical values into numerical (if needed), fix any discrepancies in the units, as well as removing attributes that does not affect the price evaluations if needed to reduce the complexity of the model. Data Understanding and preparation is an essential part of building a model as it gives the insight into the data and what corrections or modifications shall be done before designing and executing the model, preliminary analysis of the data must be done to have deeper understanding into the quality of the data, in terms of outliers and the skewedness of the figures, descriptive Statistics of categorical and numerical variables was done for that to be achieved. As

well as the ability to understand the main attributes that affect the results of the price. That was done through a correlation matrix for every attribute to understand the relations between the different factors. Afterwards when the data is organized and transformed into a form that could be processed by the data mining technique. Different data mining models were designed to predict prices and values of used cars. In this study three models are proposed to be built using Logistic Regression model technique, Random Forest Regressor and Bagging Regressor. Firstly, the data was portioned into section for training and the other part for testing, portioning percentage can be tested with different ratios to analyse different results. All three models were evaluated on four evaluation matrices known as model score, Mean Square Error (MSE), Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). From all, the Random Forest Regressor outperformed.

2.6 Limitations of the Study

In the past year the world of automobiles has seen a drastic change with the semiconductor shortages after the pandemic, which led to spike in used car prices. Hence, there was fast change in car prices during this study which will affect the actual car pricing prediction future. As the current dataset will undervalue the cars in the market. Therefore, a model that is built on real time data can be best integrated into a mobile app for public use would be the idea solution.

Chapter-3: REQUIREMENT ANALYSIS

3.1 Introduction

The requirement analysis phase is a critical initial step in understanding the needs and objectives of the car price prediction project. It involves examining stakeholders' requirements, system functionalities, and constraints to set the project's direction. This process ensures that the system aligns with user expectations and sets a clear path for development.

3.2 Requirement Analysis

Requirement Analysis for Car Price Prediction:

Functional Requirements:

1.Data collection from reliable sources.

2.Development of a robust machine learning algorithm.

3.User-friendly interface for price estimation.

Non-Functional Requirements:

1.Security measures for data protection.

2.Scalability for accommodating growth.

3.Reliability for consistent predictions.

Technical Requirements:

1.Advanced regression and machine learning algorithms. 2.Integration of cloud computing services.

3.Data visualization capabilities.

3.3 Hardware Requirement Analysis

3.3.1 Computer

Processor : Intel Pentium III or later

Main Memory(RAM) : 256 MB

Cache Memory : 512 KB

Monitor : 14 inch Color Monitor

Keyboard : 108 Keys

Mouse : Optical Mouse

Hard Disk : 160 GB

3.3.2 Server

Front End/Language : Python with Flask or Django Operating System : Windows 7, 8, 9, 10, XP.

3.4 Feasibility Study

3.4.1 Technical Feasibility

Assess the availability of relevant data sources and the feasibility of collecting and processing the required data.
Evaluate the technical expertise and resources required to develop and implement the machine learning algorithms and data processing pipelines.

3.4.2 Financial Feasibility

Determine the costs associated with data acquisition, infrastructure setup, software development, and potential ongoing maintenance.
Conduct a cost-benefit analysis to ensure that the projected benefits of the car price prediction system outweigh the incurred expenses.

Chapter-4: SYSTEM ANALYSIS AND DESIGN

4.1 Introduction

The introduction to system analysis and design for the car price prediction project involves the critical process of understanding requirements and conceptualizing the system architecture. This phase serves as the basis for developing a comprehensive and efficient car price prediction system that meets the needs of users and stakeholders.

4.2 System Architecture

Data Collection Layer: Ingests data from various sources, including historical car sales data, market trends, and economic indicators. Ensures the data is cleaned, processed, and stored in a suitable format for further analysis.
Machine Learning Layer: Utilizes advanced regression models or deep learning algorithms to analyze the collected data and train the car price prediction model.

Incorporates feature engineering techniques to extract relevant insights from the input data.

Application Layer: Provides a user-friendly interface for users to input specific car details and receive accurate price estimations. Facilitates seamless communication between the front-end and back-end components of the system.

Database Layer: Stores the processed data, trained machine learning models, and relevant information required for efficient system operation.

Supports quick retrieval and updating of data to ensure real-time responsiveness and accuracy in predictions. Security Layer: Implements robust security measures to protect sensitive data and prevent unauthorized access or data breaches. Ensures the integrity and confidentiality of the data throughout the system.

Scalability and Integration Layer: Enables the system to handle a growing volume of data and accommodate potential expansion to include new car models and markets.

Facilitates seamless integration with external services or APIs for additional functionalities and data enrichment. By incorporating these architectural components, the car price prediction system aims to provide an efficient, secure, and user-friendly platform for accurate price estimations, contributing to enhanced transparency and informed decision-making in the automotive market.

Chapter-5: IMPLEMENTATION AND TESTING

5.1 Introduction

We present the implementation details of the proposed solution and the comprehensive testing methodology employed to validate its functionality and performance. The implementation phase outlines the steps taken to translate the conceptual design into a practical system, highlighting key technical decisions and considerations. Subsequently, the testing phase discusses the rigorous testing procedures executed to ensure the robustness, reliability, and accuracy of the implemented solution. The section concludes with an assessment of the results obtained from the testing phase, providing insights into the effectiveness and efficiency of the developed system.

5.2 Importing libraries

Here is a concise description for each of the necessary libraries used in Python for machine learning:

NumPy: Essential for numerical computing, providing support for arrays and matrices, and mathematical operations on these arrays.

Pandas: Crucial for data manipulation and analysis, offering data structures and operations for manipulating numerical tables and time series data.

Matplotlib: Key for creating static, interactive, and animated visualizations in Python, making it easier to represent data visually.

Seaborn: Built on top of Matplotlib, Seaborn is used for creating more attractive and informative statistical graphics.

Scikit-learn: A versatile library for machine learning tasks, providing simple and efficient tools for data mining and analysis.

SciPy: Essential for scientific and technical computing, offering a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python.

Importing these libraries allows you to leverage their functionalities and capabilities for a variety of machine learning tasks and projects in Python.

Code : ! gdown –id 14ClyyyG-In58nThU-LHCyUbMfsm1dfmm

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import plotly.express as px
import seaborn as sns
from sklearn.ensemble import BaggingRegressor
from sklearn.metrics import confusion-matrix
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures
```

5.3 Reading the concerned dataset

To read a dataset into your Python environment, you typically use Pandas, one of the most commonly used libraries for data manipulation and analysis. You can read various types of data files, including CSV, Excel, and more, using the appropriate Pandas functions.

Code :

cs = pd.read-csv('Car-sale-ads.csv') cs.head(3)

5.4 Data Understanding

To gain a better understanding of your dataset, you can perform various operations and analyses using Pandas. Here are some common data understanding techniques: Viewing Data: Use the head() and tail() functions to view the first few and last few rows of the dataset, respectively, and sample() to view a random sample of rows.

Code :

sns.pairplot(data = cs, diag-kind = 'kde')

5.5 Data visualization

Certainly, here's a concise description of data visualization techniques in Python using libraries such as Matplotlib and Seaborn:

Line Plots: Use plt.plot() to create basic line plots for visualizing trends and patterns in data.

Scatter Plots: Utilize plt.scatter() to display the relationship between two variables and identify potential correlations.

Histograms: Use plt.hist() to visualize the distribution of a single variable and understand its frequency distribution.

Bar Plots: Employ plt.bar() or plt.barh() to compare different categories or groups of data.

5.6 Data preparation

Certainly, here is a concise overview of data preparation techniques in Python for machine learning using Pandas and NumPy:

Data Cleaning: Remove or fill missing values using dropna() or fillna(). Use drop duplicates() to eliminate duplicate rows.

Data Transformation: Use apply() or map() to apply functions to the dataset. Employ replace() to replace values and astype() to convert data types.

Feature Scaling: Normalize or standardize numerical features using techniques like Min-Max scaling or Z-score normalization from libraries such as Scikit-learn.

Encoding Categorical Data: Use techniques like one-hot encoding or label encoding from libraries such as Scikit-learn or Pandas to convert categorical data into a format suitable for machine learning models.

Feature Engineering: Create new features from existing ones, such as extracting date features, creating interaction terms, or transforming variables

to improve the performance of machine learning models.

By employing these techniques, you can prepare your dataset to be effectively utilized for machine learning tasks, ensuring that the data is in the appropriate format for model training and evaluation.

5.7 Splitting the Data and feature scaling

Certainly, here's a concise overview of splitting the data and feature scaling in Python for machine learning:

Data Splitting: Use functions from Scikit-learn, such as train-test-split, to split the dataset into training and testing sets for model training and evaluation.

Feature Scaling: Apply techniques such as Min-Max scaling or Standardization from Scikit-learn's MinMaxScaler and StandardScaler to ensure that all features are on a similar scale, preventing any particular feature from dominating the learning process.

5.8 Building a linear regression model

Certainly, here's a brief outline of building a linear regression model in Python using Scikit-learn:

Data Preparation: Organize your feature and target variables as arrays, ensuring that X is 2D and y is 1D.

Data Splitting: Use train-test-split from Scikit-learn to divide the dataset

into training and testing sets.

Model Initialization: Initialize a linear regression model using LinearRegression()from Scikit-learn.

Model Training: Fit the model to the training data using the fit() function.

Making Predictions: Use the predict() function to generate predictions on the test set.

Model Evaluation: Assess the model's performance using metrics such as mean squared error and coefficient of determination (R^2).

5.9 Making Predictions Using the Final Model

5.9.1 Random Forest

Random Forest Regression comes under a Supervised Machine Learning algorithm which uses an ensemble model for regression. An ensemble learning is a technique that clubs the outcomes of different machine learning models for creating more good predictions than one model. Results of Random Forest are:

```
print('RandomForestRegressor Accuracy Evaluation')
print(f'r2 score: {r2_score(y_test, random_forest.predict(x_test))}')
print(f'Mean absolute error: {mean_absolute_error(y_test, random_forest.predict(x_test))}')
print(f'Mean squared error: {mean_squared_error(y_test, random_forest.predict(x_test))}')
```

```
RandomForestRegressor Accuracy Evaluation
r2 score: 0.8670017090356326
```

Figure 5.1. Random Forest

5.9.2 Decision Trees Regressor

Decision Trees are a fundamental supervised machine learning algorithm used for both classification and regression tasks. They operate by partitioning the data into subsets based on various attributes and making decisions based on the partitioned subsets. This process results in a tree-like model where each internal node represents a feature, each branch represents a decision rule, and each leaf node represents the outcome. Decision Trees are highly interpretable and easy to visualize, making them valuable for understanding the reasoning behind the predictions. However, they can suffer from overfitting, especially with complex datasets. Regularization techniques and ensemble methods, such as Random Forest and Extra Trees, are often used to mitigate this issue and improve their performance. Results of Decision Trees are:

```
print('DecisionTreeRegressor Accuracy Evaluation')
print(f'r2 score: {r2_score(y_test, dtr.predict(x_test))}')
print(f'Mean absolute error: {mean_absolute_error(y_test, dtr.predict(x_test))}')
print(f'Mean squared error: {mean_squared_error(y_test, dtr.predict(x_test))}')
```

```
DecisionTreeRegressor Accuracy Evaluation
r2 score: 0.8156117034194138
```

Figure 5.2. Decision Trees Regressor

5.9.3 Extra Trees

Extra Trees is an ensemble learning algorithm, similar to Random Forest, used for supervised machine learning tasks such as regression. It builds

multiple decision trees during training and combines their predictions. What distinguishes Extra Trees is its random selection of split points, which enhances diversity among trees and reduces sensitivity to noise, potentially leading to better generalization and decreased overfitting.

```
[ ]
    print('ExtraTreesRegressor Accuracy Evaluation')
    print(f'r2 score: {r2_score(y_test, etr.predict(x_test))}')
    print(f'Mean absolute error: {mean_absolute_error(y_test, etr.predict(x_test))}')
    print(f'Mean squared error: {mean_squared_error(y_test, etr.predict(x_test))}')

    ExtraTreesRegressor Accuracy Evaluation
    r2 score: 0.8707240410469842
```

Figure 5.3. Extra Trees

5.10 Model Evaluation

Certainly, here's a concise description of model evaluation techniques for Decision Trees, Random Forest, and Extra Trees:

Decision Trees: Evaluate using metrics like accuracy, precision, recall, and F1-score for classification, and mean squared error for regression. Cross-validation helps assess generalizability.

Random Forest: Assess ensemble performance with metrics such as accuracy, precision, recall, F1-score for classification, and mean squared error for regression. Cross-validation and out-of-bag error estimation aid in understanding generalization.

Extra Trees: Employ similar metrics for evaluation, including accuracy, precision, recall, F1-score, and mean squared error. Cross-validation helps assess robustness and generalization.

5.11 Conclusion

In this discussion, we covered key aspects of machine learning in Python, including data handling, preparation, visualization, and modeling. We explored the process of building both linear regression and random forest models, essential techniques for predictive modeling tasks. By understanding these fundamental concepts and employing the appropriate libraries and methods, you can effectively analyze data, create models, and make accurate predictions for various machine learning applications.

Chapter-6: USER MANUAL

6.1 Introduction

This user manual provides essential instructions and guidance for effectively using our product, offering clear and comprehensive information for users of all levels of expertise.This guide is tailored to accommodate users at every level of familiarity, from beginners to seasoned users, and is structured to provide detailed insights into the setup, operation, maintenance, and troubleshooting of the product. With a focus on clarity and accessibility, this manual aims to empower users with the knowledge and confidence needed to navigate the product's features and functionalities seamlessly. Dive into this guide to discover step-by-step instructions, expert tips, and in-depth explanations that will enable you to make the most of your experience with our product.

6.2 Importing libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import plotly.express as px
import seaborn as sns
from sklearn.ensemble import BaggingRegressor
from sklearn.metrics import confusion_matrix
```

```
import lightgbm as lgb
```

```
[ ] from sklearn import preprocessing
    from sklearn.preprocessing import StandardScaler
    from sklearn.preprocessing import PolynomialFeatures
```

```
[ ] from sklearn.linear_model import LinearRegression, Ridge
    from sklearn.naive_bayes import GaussianNB
    from sklearn.neighbors import KNeighborsRegressor
    from sklearn.tree import DecisionTreeRegressor
    from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor, ExtraTreesRegressor, BaggingRegressor
```

```
[ ] from sklearn.linear_model import Ridge,Lasso,LinearRegression
    from sklearn.preprocessing import StandardScaler
```

```
[ ] from sklearn.model_selection import train_test_split, cross_val_score
    from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
```

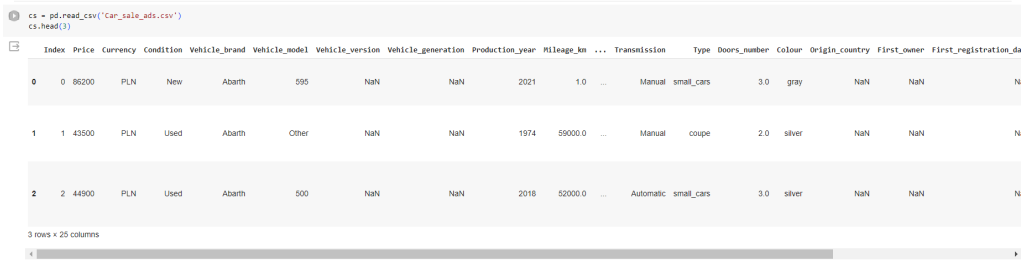
```
[ ] from sklearn.metrics import f1_score
```

```
[ ] from sklearn.model_selection import train_test_split
    from sklearn.metrics import mean_squared_error
    from sklearn.metrics import mean_absolute_error
    from sklearn.metrics import r2_score
```

Figure 6.1. Importing libraries

First,Import all the necessary libaries.

6.3 Reading the concerned dataset



```
cs = pd.read_csv('car_sale_ads.csv')
cs.head(3)
```

	Index	Price	Currency	Condition	Vehicle_brand	Vehicle_model	Vehicle_version	Vehicle_generation	Production_year	Mileage_km	...	Transmission	Type	Doors_number	Colour	Origin_country	First_owner	First_registration_da
0	0	86200	PLN	New	Abarth	595	NaN	NaN	2021	1.0	...	Manual	small_cars	3.0	gray	NaN	NaN	N
1	1	43500	PLN	Used	Abarth	Other	NaN	NaN	1974	59000.0	...	Manual	coupe	2.0	silver	NaN	NaN	N
2	2	44900	PLN	Used	Abarth	500	NaN	NaN	2018	52000.0	...	Automatic	small_cars	3.0	silver	NaN	NaN	N

3 rows × 25 columns

Figure 6.2. Reading the concerned dataset

After,import the all the necessary libraries then read the concerned dataset.

6.4 Data Understanding

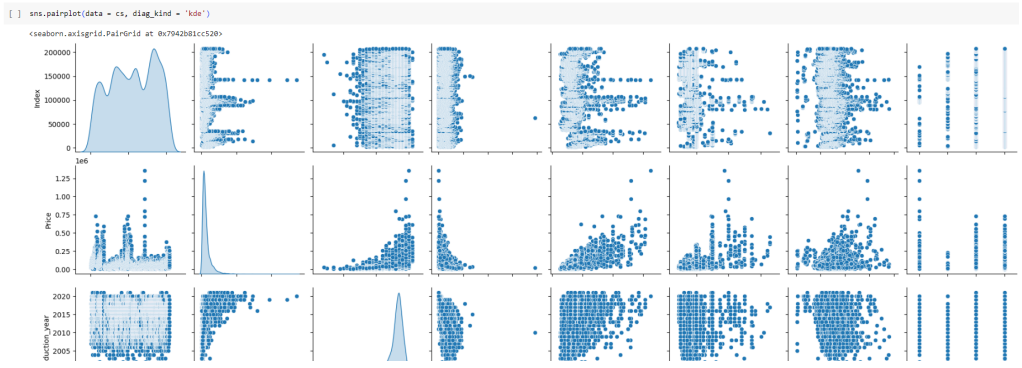
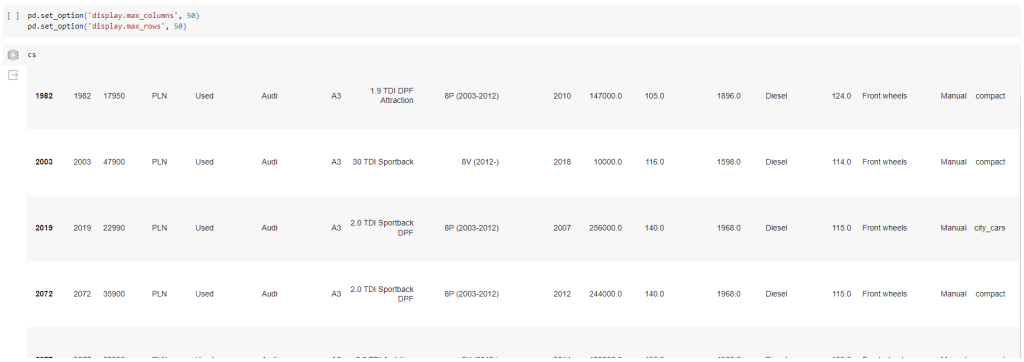


Figure 6.3. Data Understanding

After,Read the dataset then need to data understanding.

6.5 Data handling



```
[ ] pd.set_option('display.max_columns', 50)
    pd.set_option('display.max_rows', 50)

    cs
```

1982	1982	17950	PLN	Used	Audi	A3	1.9 TDI DPF Attraction	8P (2003-2012)	2010	147000.0	105.0	1896.0	Diesel	124.0	Front wheels	Manual	compact
2003	2003	47900	PLN	Used	Audi	A3	30 TDI Sportback	8V (2012-)	2018	10000.0	116.0	1598.0	Diesel	114.0	Front wheels	Manual	compact
2019	2019	22990	PLN	Used	Audi	A3	2.0 TDI Sportback DPF	8P (2003-2012)	2007	256000.0	140.0	1968.0	Diesel	115.0	Front wheels	Manual	city_cars
2072	2072	35900	PLN	Used	Audi	A3	2.0 TDI Sportback DPF	8P (2003-2012)	2012	244000.0	140.0	1968.0	Diesel	115.0	Front wheels	Manual	compact

Figure 6.4. Data handling

6.6 Data visualization

```
df1['Mileage_km']=df1['Mileage_km'].fillna(df1['Mileage_km'].mean())
df1['Power_HP']=df1['Power_HP'].fillna(df1['Power_HP'].mean())
df1['Displacement_cm3']=df1['Displacement_cm3'].fillna(df1['Displacement_cm3'].mean())
```

```
df1.corr()
```

```
<ipython-input-21-49b3fcfeb4d1>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the valu
  df1.corr()
```

	Index	Price	Production_year	Mileage_km	Power_HP	Displacement_cm3	Doors_number
Index	1.000000	-0.207358	-0.001348	0.032977	-0.256300	-0.237150	0.092527
Price	-0.207358	1.000000	0.467099	-0.385035	0.749099	0.576739	-0.190794
Production_year	-0.001348	0.467099	1.000000	-0.599922	0.170316	-0.025895	0.038158
Mileage_km	0.032977	-0.385035	-0.599922	1.000000	-0.085244	0.119282	0.019035
Power_HP	-0.256300	0.749099	0.170316	-0.085244	1.000000	0.850327	-0.195331
Displacement_cm3	-0.237150	0.576739	-0.025895	0.119282	0.850327	1.000000	-0.183976
Doors_number	0.092527	-0.190794	0.038158	0.019035	-0.195331	-0.183976	1.000000

Distributions

2-d distributions

Figure 6.5. Data visualization

6.7 Data preparation

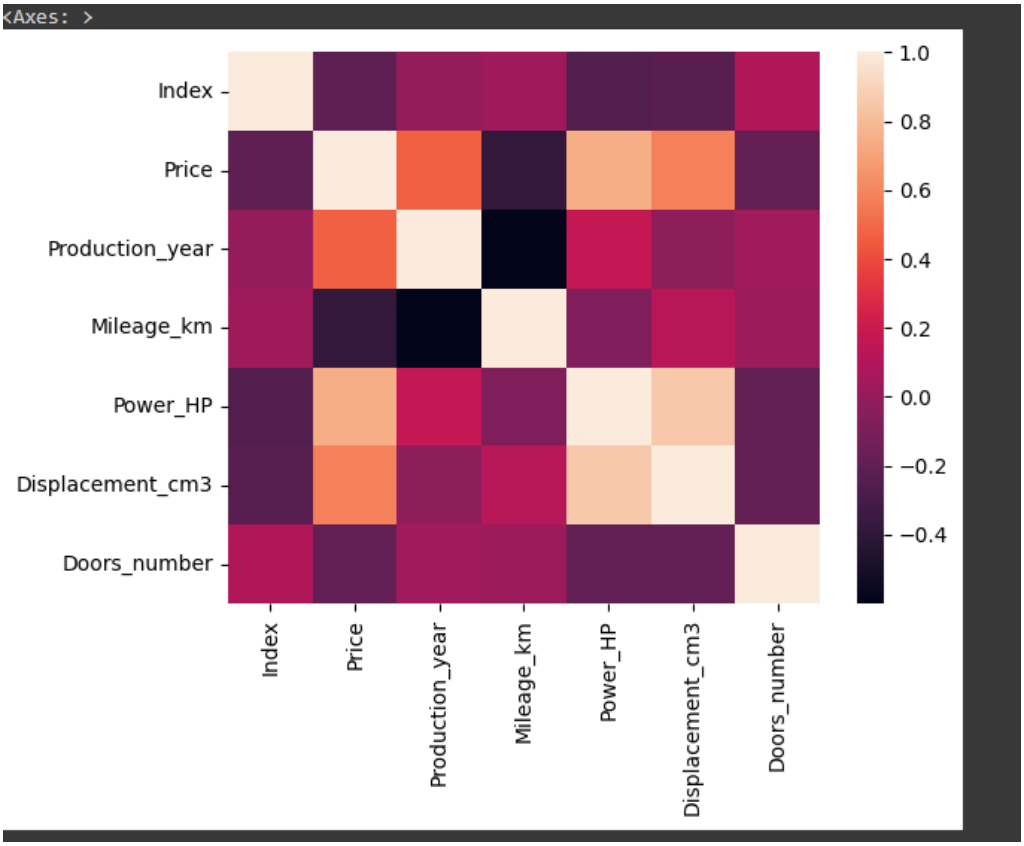


Figure 6.6. Data preparation

6.8 Splitting the Data and feature scaling

```
[ ] x_train,x_test,y_train,y_test=train_test_split(x, y,train_size=0.8,random_state=123)

[ ] lgb_train = lgb.Dataset(x_train, y_train)
    lgb_eval = lgb.Dataset(x_test, y_test)

[ ] params={'metric': 'rmse'}

[ ] scaler=StandardScaler()
    x_train_sc=pd.DataFrame(scaler.fit_transform(x_train))
    x_test_sc=pd.DataFrame(scaler.transform(x_test))

[ ] model1=LinearRegression()
    model2=Lasso(alpha=1.0)
    model3=Ridge(alpha=0.1)

[ ] scores=[]
    kf=KFold(n_splits=4,shuffle=True,random_state=71)
    for tr_idx,va_idx in kf.split(x_train_sc):
        tr_x,va_x=x_train_sc.iloc[tr_idx],x_train_sc.iloc[va_idx]
        tr_y,va_y=y_train.iloc[tr_idx],y_train.iloc[va_idx]

        model1.fit(tr_x,tr_y)
        va_pred1=model1.predict(va_x)
        score_rmse1=np.sqrt(mean_squared_error(va_y,va_pred1)).mean()
        score_mae1=mean_absolute_error(va_y,va_pred1).mean()
        score_r21=r2_score(va_y,va_pred1).mean()

    print('rmse1:',score_rmse1)
    print('mae1:',score_mae1)
    print('R21:',score_r21)

    rmse1: 36691.51005841801
    mae1: 19921.198825036256
    R21: 0.7314754813242481
```

Figure 6.7. Data preparation

6.9 Making Predictions Using the Final Model

```
# Plot the results
plt.figure(figsize=(10, 6))
plt.scatter(x_values, y_test, color='black', label='Actual values', s=2)
plt.plot(x_values, y_pred, color='blue', linewidth=1, label='Predicted values')
plt.title('Extra Trees Regressor')
plt.xlabel('Data points')
plt.ylabel('Target values')
plt.legend()
plt.show()
```

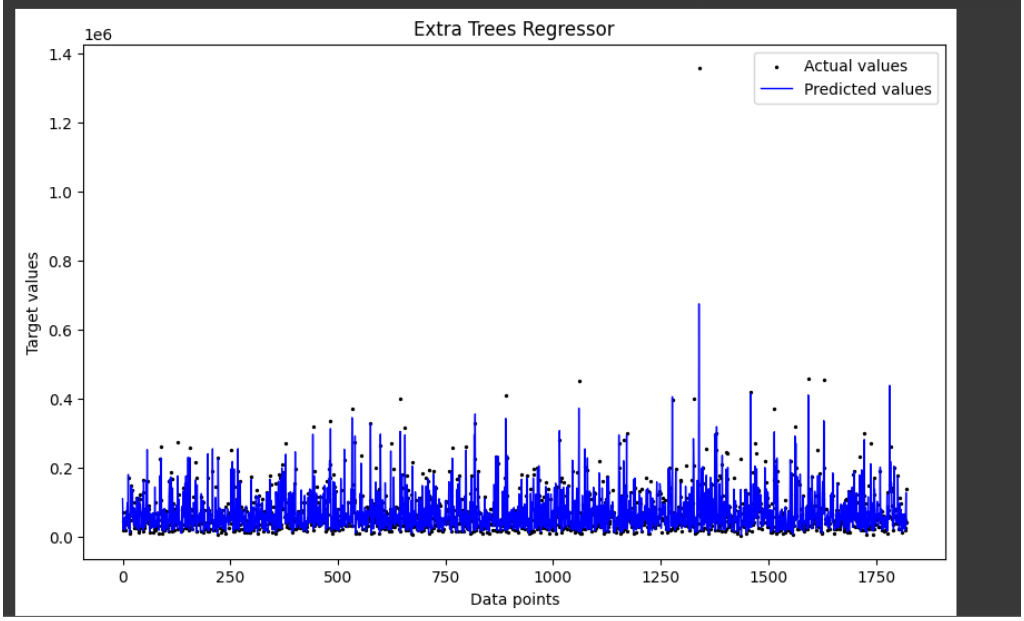


Figure 6.8. Extra Trees Regressor

```
plt.figure(figsize=(10, 6))
plt.scatter(x_values, y_test, color='black', label='Actual values', s=2)
plt.plot(x_values, y_pred, color='blue', linewidth=1, label='Predicted values')
plt.title('Random Forest Regression')
plt.xlabel('Data points')
plt.ylabel('Target values')
plt.legend()
plt.show()
```

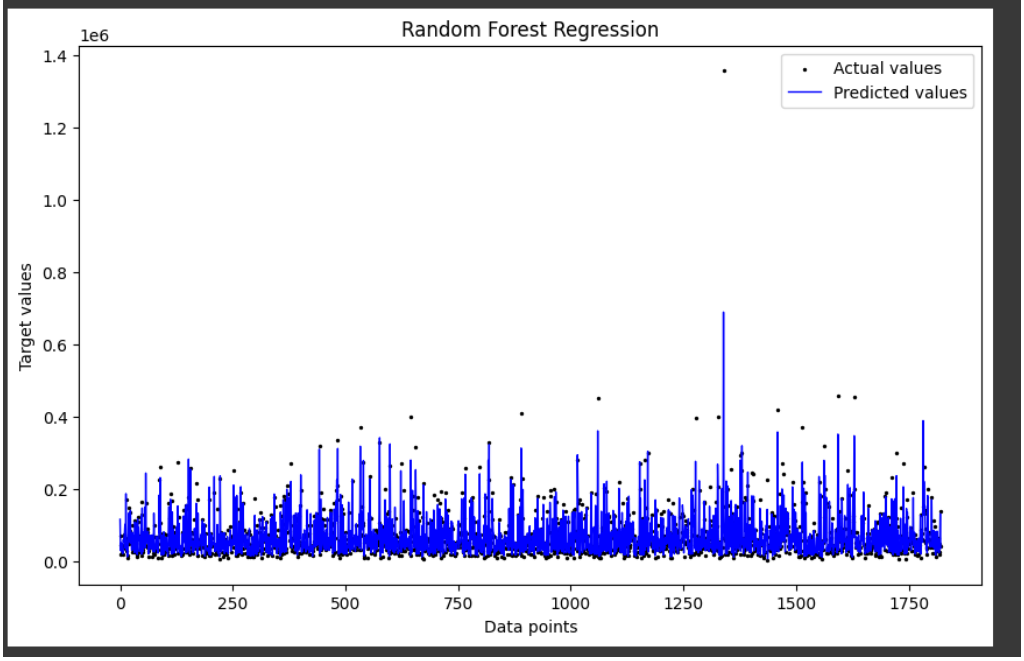


Figure 6.9. Random Forest Regressor

```
x_values = np.arange(len(y_test))
y_pred = dtr.predict(x_test)

# Plot the results
plt.figure(figsize=(10, 6))
plt.scatter(x_values, y_test, color='black', label='Actual values', s=2)
plt.plot(x_values, y_pred, color='blue', linewidth=1, label='Predicted values')
plt.title('Decision Trees Regressor')
plt.xlabel('Data points')
plt.ylabel('Target values')
plt.legend()
plt.show()
```

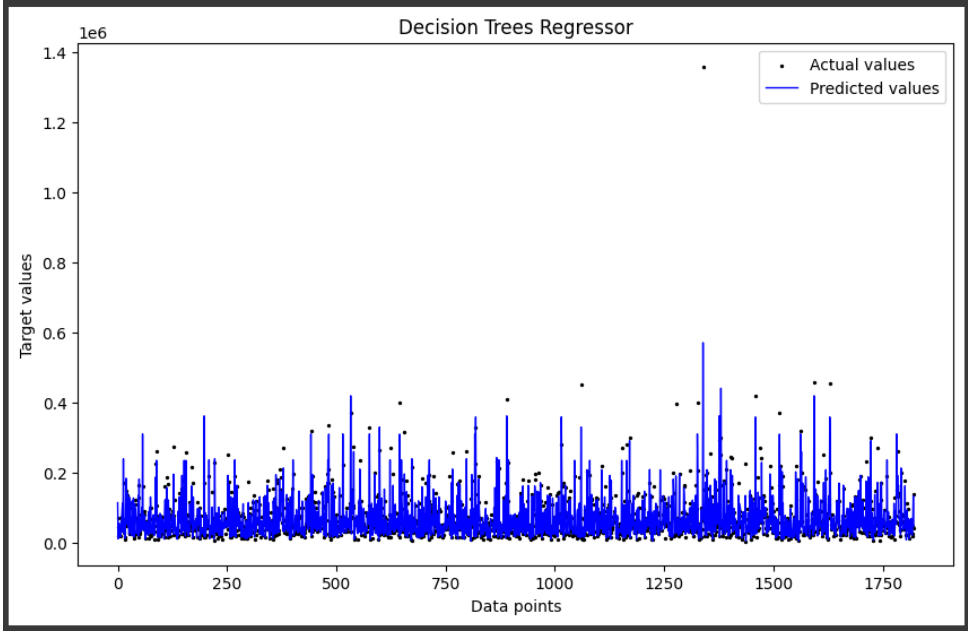


Figure 6.10. Decision Trees Regressor

6.10 Model Evaluation

```
fig, ax = plt.subplots()
rects1 = ax.bar(x - width / 2, train_scores, width, label='Actual Value')
rects2 = ax.bar(x + width / 2, test_scores, width, label='Predicted Value')
ax.set_xlabel('Models')
ax.set_ylabel('Scores')
ax.set_title('Scores by model and dataset')
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()
plt.savefig('scores_comparison.png')
```

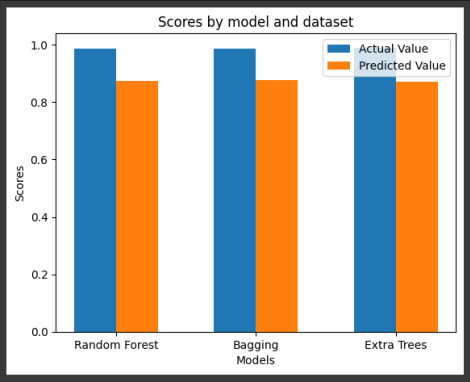


Figure 6.11. Model Evaluation

6.11 Conclusion

The conclusion of a user manual typically summarizes the key points covered in the document and emphasizes the importance of following the instructions for optimal product use and safety. It often includes a call to action, encouraging users to reach out for further assistance or support if needed. Additionally, it may highlight any essential maintenance tips or additional resources available for users to consult. By providing a comprehensive overview and reiterating the significance of adhering to the guidelines, the conclusion aims to ensure a positive user experience and the successful utilization of the product.

Chapter-7: Conclusion And Further Work

7.1 Conclusion

Using data mining and machine learning approaches, this project proposed a scalable framework for Dubai based used cars price prediction. Buy any car.com website was scraped using the Parse Hub scraping tool to collect the benchmark data. An efficient machine learning model is built by training, testing, and evaluating three machine learning regressors named Random Forest Regressor, Linear Regression, and Bagging Regressor. As a result of pre-processing and transformation, Random Forest Regressor came out on top with 95within the Google Colab environment. In comparison to the system's integrated Jupyter notebook and Anaconda's platform, algorithms took less training time in Google Colab.

7.2 Future Plan

In the future, more data will be collected using different web-scraping techniques, and deep learning classifiers will be tested. Algorithms like Quantile Regression, ANN and SVM will be tested. Afterwards, the intelligent model

will be integrated with web and mobile-based applications for public use. Moreover, after the data collection phase Semiconductor shortages have incurred after the pandemic which led to an increase in car prices, and greatly affected the secondhand market. Hence having a regular Data collection and analysis is required periodically, ideally, we would be having a real time processing program.

Refrences

[1].M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in Proc. of USENIX Security Symposium, 2017.

[2].Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, —Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization‖, 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018

[3].Hossein Hadian Jazi, Hugo Gonzalez, Natalia Stakhanova, and Ali A. Ghorbani. "Detecting HTTP-based Application Layer DoS attacks on Web Servers in the presence of sampling." Computer Networks, 2017

[4]. A. Shiravi, H. Shiravi, M. Tavallaee, A.A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Comput. Security 31 (3) (2012) 357–374

[5].R. Doshi, N. Apthorpe and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, 2018, pp. 29-35.

[6].Jerome H. Friedman, (2002), Stochastic gradient boosting, Computational Statistics and Data Analysis, 38, (4), 367-378

[7].Friedman, Jerome H. Greedy function approximation: A gradient boosting machine. Ann. Statist. 29 (2001), no. 5, 1189–1232.