International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 07 | July – 2025 DOI: 10.55041/ISJEM04748
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Predicting Prices of Used Cars with Python and ML

[1]M.NAGA KEERTHI, [2]JONNADULA DURGA RAO
1Assistant Professor, Department Of MCA, 2MCA Final Semester,
[1]Master of Computer Applications,
[1]Sanketika Vidya Parishad Engineering College, Vishakhapatnam, Andhra Pradesh, India

Abstract:

This project presents a comprehensive machine learning-based web application designed to predict prices of used cars using Python. The system begins with data preparation, where a cleaned dataset is ingested and structured with key automotive features such as brand, model, year, fuel type, and kilometres driven. The data undergoes preprocessing and feature selection to ensure optimal input for a regression model. A Linear Regression model trained on historical car listings is used to predict prices based on user inputs. The model's performance is evaluated using standard metrics like R-squared and Mean Squared Error. A user-friendly Flask web interface, enhanced with modern CSS styling, allows real-time interaction: users can select car attributes and instantly receive price estimates. This integrated solution bridges data science with interactive deployment, offering valuable insights for buyers, sellers, and automotive analysts.

Index Terms: Used car price prediction, Machine Learning, Python, Linear Regression, Flask Web Application, Data Preprocessing, Feature Engineering, Interactive Prediction, Web Deployment

1.Introduction:

One of the first inquiries someone has when looking to purchase or sell an automobile is, "Exactly how much is this car worth?" There are several variables that can impact a car's price, such as its brand, age, model, fuel type, or special features, so the answer isn't always clear-cut.

To gain a sense of car prices in the past, customers would have perused newspaper ads, ask friends, or visit dealerships. However, there is a more intelligent way to determine this now that we have access to computers' power and data. The project involves predicting car prices using machine learning.

In order to solve this problem, this project aims to predict the Price of a used Car by taking it's Company name, its Model name, Year of Purchase, and other parameters. Particularly, for this analysis use the linear regression method, a powerful tool for predictive modeling, and Python's complex ML environment. And I will also convert it into a full-fledged website using the flask framework.

1.1. Existing system

Right now, many car prices are set by people, especially dealers, based on their own knowledge. There are also some online tools that can guess a car's price using basic details, but they might miss out on important factors. These methods can be a bit general and might not always be up-to-date with current car market trends. So, we have a mix of human judgment and basic tools, but there's room to make it better and more accurate using machine learning.

In short, the existing ways to predict car prices have been a bit basic and might not always be super accurate. With machine learning, we're hoping to make this prediction much smarter and precise, like having a kitchen gadget that tells you exactly how long to cook your meal!

1.1.1.Challenges

Data Quality and Consistency

- The project heavily relies on historical car listing data, which often contains missing, inconsistent, or erroneous records. Variations in naming conventions (e.g., "Hyundai i10" vs "i10") and inaccurate mileage entries can severely impact model accuracy.
- Handling outliers and ensuring uniform encoding of categorical variables demands extensive preprocessing and robust validation procedures.

Feature Engineering and High Dimensionality

- The dataset exhibits high cardinality in categorical features such as model names and fuel types, leading to a proliferation of dummy variables that can dilute predictive power and cause the model to overfit.
- Determining which features truly influence price, and mitigating multicollinearity among variables like year of manufacture and car age, requires careful statistical analysis.

Model Selection and Predictive Performance

- While linear regression offers interpretability, it assumes a linear relationship between features and price, which may not hold true in practice, especially when market dynamics, car conditions, and seller factors introduce complex non-linear interactions.
- Achieving a balance between underfitting (oversimplified model) and overfitting (model too closely tied to training data) remains a significant challenge. Extensive cross-validation and experimentation with more advanced regressors are often necessary.

User Input Validation and Interface Design

- Ensuring that all user-provided inputs are realistic, complete, and properly formatted is critical. Invalid or missing inputs can lead to runtime errors or nonsensical predictions.
- Implementing dynamic, user-friendly interfaces—such as dependent dropdowns (where the model list updates based on the selected company)—requires additional front-end logic and integration with back-end data.

1.2 Proposed system:

We want the computer to learn from lots of car data to predict prices. Instead of just human guesses or basic online tools, this system will use machine learning to look at things like the car's age, brand, and miles driven. By doing this, it can give a more accurate price guess. And the best part? It'll keep learning and getting better as it sees more data, making sure the price suggestions are always upto-date and reliable.

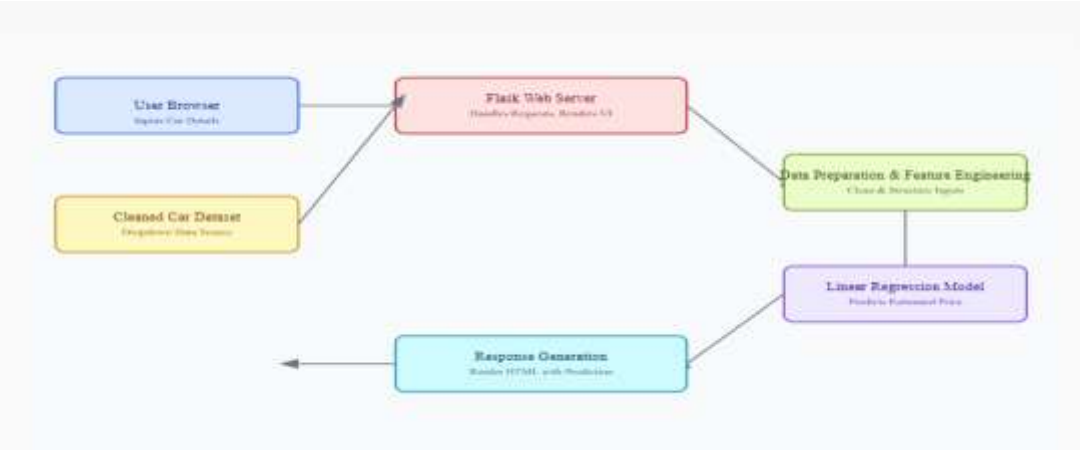

Fig: 1 Proposed Diagram

1.1.1 Advantages:

1. Increased Accuracy

- The Linear Regression model learns from real historical car data.
- Predictions are data-driven, reducing human bias.
- Takes into account multiple features: brand, year, fuel type, kilometres driven, etc.

2. Real-Time Price Estimation

- Users instantly get price estimates via the Flask web interface.

- No need to wait for manual appraisal or search through listings.

3. User-Friendly Interface

- The web app has a clean, modern UI (HTML/CSS).
- Anyone can use it without technical knowledge.
- Dropdowns make it simple to select car attributes.

4. Automates Traditional Methods

- Replaces time-consuming manual pricing by:
  - Dealers' subjective judgment.
  - Scanning ads or contacting multiple sellers.
- Makes valuation faster and more consistent.

5. Learning and Adaptation

- The model can be retrained regularly to reflect:
  - New market trends.
  - Price fluctuations.
  - Emerging car models.
- This keeps predictions up to date.

6. Easy Deployment

- Built using Python and Flask, which are:
  - Lightweight.
  - Easy to deploy on any server or cloud platform (Heroku, AWS, PythonAnywhere).
- Scalable as traffic grows.

2.1 Architecture:

The architecture of the used car price prediction system follows a structured machine learning pipeline, beginning with a cleaned dataset of historical car records that includes essential features such as brand, model, year, fuel type, and kilometres driven. This dataset is split into a training set (80%) and a testing set (20%) to ensure reliable model evaluation. The training set, containing both features and labels (actual prices), is used to train a Linear Regression model that learns the relationship between the input variables and car prices. Meanwhile, the testing set, containing only the features, is reserved for validating the model's predictive performance after training. Once the model is trained, it generates predictions on unseen testing data, and the results are compared to the true prices during the model evaluation phase to measure accuracy and error metrics. This process ensures the model generalizes well and provides accurate, data-driven price estimates when deployed in the Flask web application for real-time user interaction.

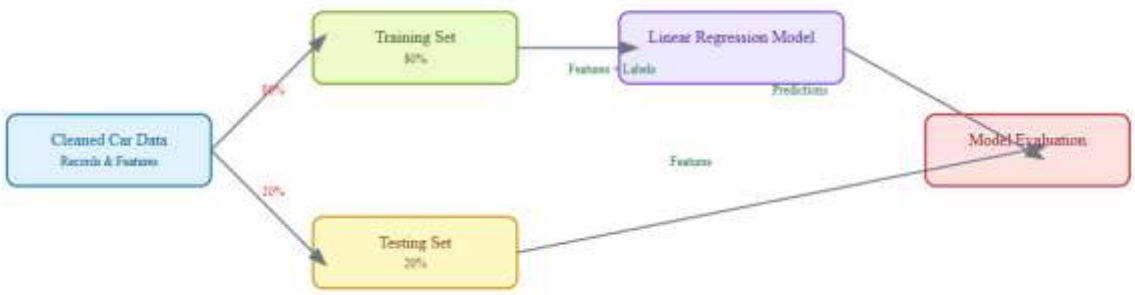

Fig:2 Architecture

2.2 Algorithm:

For predicting used car prices, a variety of supervised machine learning algorithms can be applied to achieve accurate estimates. Linear Regression is the most common and interpretable method, modelling the relationship between features such as brand, model, year, fuel type, and kilometres driven, and the car's price. This approach assumes a linear relationship between inputs and outputs, making it computationally efficient and easy to explain. However, real-world car pricing often involves non-linear patterns that linear regression may not fully capture. To address this, Decision Tree Regression can be used to split the data into branches based on feature values and create decision rules that predict prices more flexibly. Although decision trees are intuitive, they can overfit the data if not properly pruned.

Random Forest Regression overcomes this issue by combining multiple decision trees and averaging their predictions to improve generalization and reduce variance. Another highly effective technique is Gradient Boosting Regression, including frameworks like XGBoost, which build ensembles of trees sequentially, correcting previous errors in each iteration to achieve high predictive accuracy. Additionally, Support Vector Regression (SVR**)** leverages kernel functions to model complex, non-linear relationships in the data. Each of these algorithms offers distinct advantages: linear regression provides simplicity and speed, tree-based methods handle non-linearities and outliers well, and boosting methods are powerful for capturing intricate patterns in large datasets. In this project, you can begin by establishing a baseline model using linear regression and then experiment with decision trees, random forests, and gradient boosting to compare their performance. All of these algorithms are supported by Python's scikit-learn library[15], making them easy to integrate, train, and evaluate within your Flask web application[7].

2.3 Techniques:

The project begins by collecting and cleaning a dataset containing historical records of used cars, including details like brand, model, year, fuel type, and kilometres driven. Data cleaning techniques are applied to remove duplicates, handle missing values, and ensure consistent formatting. Next, preprocessing steps such as encoding categorical variables and scaling numeric features help prepare the data for machine learning algorithms. Splitting the dataset into training and testing sets ensures that the model can be evaluated fairly on unseen data**.**

To build the prediction system, several regression algorithms are considered, with Linear Regression selected as the primary model due to its simplicity and interpretability. The model is trained on the prepared training data to learn relationships between car features and prices. Evaluation metrics like R-squared and Mean Squared Error are calculated on the testing data to measure the model's accuracy and detect any overfitting. Additional algorithms, such as Decision Trees, Random Forests, and Gradient Boosting, can also be explored to compare performance and improve prediction quality.

After training, the final model is saved as a Pickle file, allowing it to be reused without retraining each time. The project uses the Flask web framework to build an interactive web application where users can enter car details through a simple HTML form. When a user submits the form, the application processes the input, generates a predicted price in real time, and displays it on the website. This combination of data science techniques and web deployment ensures the system is accessible, easy to use, and provides valuable insights for anyone looking to estimate the price of a used car

2.4 Tools:

This project uses Python 3 as the primary programming language, leveraging Pandas [18] and NumPy [17] for data analysis, cleaning, and manipulation of the used car dataset stored in a CSV file. The scikit-learn [15] library is employed to implement and train machine learning models, particularly Linear Regression, along with evaluation metrics to assess performance. Pickle is used to serialize the trained model for reuse without retraining. The web application is built using the Flask framework, with Flask-CORS enabling cross-origin access, while the front-end relies on HTML [9], CSS [10], and optionally Bootstrap for modern, responsive styling. Development and testing are conducted in Jupyter Notebook [2] or VS Code, with Git providing version control. The system can be deployed locally or hosted on cloud platforms such as Heroku, PythonAnywhere, or AWS Elastic Beanstalk, and ppi or virtualenv are used to manage project dependencies efficiently.

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 07 | July – 2025 DOI: 10.55041/ISJEM04748
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

.2.5 Methods:

The project begins by collecting and cleaning historical car data[5], removing duplicates, handling missing values, and standardizing formats to ensure data consistency. Preprocessing techniques[20] such as encoding categorical variables and scaling numeric features are applied to prepare the data for modeling. The cleaned dataset is then split into training and testing subsets to enable fair evaluation of model performance. A Linear Regression model is trained on the training data to learn the relationship between car features and their prices, and its accuracy is assessed using metrics like R-squared and Mean Squared Error on the test set. The trained model is saved as a Pickle file for efficient reuse. A Flask web application is developed to provide an interactive interface where users can input car details through a web form. Upon submission, the application validates the inputs, processes them into the required format, uses the model to predict the car price in real time, and displays the estimated value back to the user, combining data science with web deployment for an accessible prediction tool.

III. METHODOLOGY

3.1 Input:

The inputs for this project consist of key attributes of a used car that significantly influence its price. These include the Company name (brand), the Model name, the Year of Purchase, the Fuel Type (such as petrol, diesel, CNG, or LPG), and the Kilometres Driven, which reflects how much the car has been used. In the web application, users provide these inputs through a form with dropdown menus for selecting the company, model, year, and fuel type, along with a numeric field to enter the kilometres driven. These inputs are then collected, validated to ensure correctness and completeness, encoded or transformed as needed to match the model's expected format, and finally used by the trained machine learning model to generate an accurate price prediction.

Extracting Training Data

```
X=car[['name','company','year','kms_driven','fuel_type']]
y=car['Price']
```

```
X
```

	name	company	year	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	36000	Diesel
4	Ford Figo	Ford	2012	41000	Diesel
...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	50000	Petrol
812	Tata Indica V2	Tata	2009	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	132000	Petrol
814	Tata Zest XM	Tata	2018	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	40000	Diesel

815 rows × 5 columns

Fig 1: Extracting training data

Cleaned Data

car

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

Fig 2: Cleaned data

3.2 Method of Process:

The process for this project starts with collecting and importing the historical dataset of used cars[5], which includes key features such as the car's name, company, year of manufacture, kilometres driven, fuel type, and price. The data is then cleaned to remove duplicates, fix inconsistencies, and handle missing values, ensuring it is suitable for modeling. Next, the dataset is split into input features (X) and the target variable (y), where X contains the car attributes and y contains the prices. Categorical features like company, model name, and fuel type are encoded into numeric representations, while numeric columns are transformed if necessary to standardize the scale. The prepared data is split into training and testing sets to allow the model to be evaluated on unseen data. A Linear Regression model is trained on the training set to learn relationships between car attributes and price. The model's accuracy is then assessed using evaluation metrics such as R-squared and Mean Squared Error. Once validated, the trained model is saved using Pickle[19] for reuse during prediction. Finally, a Flask web application is developed, which loads the saved model, presents an input form for users to enter car details, processes and validates the inputs, and generates a real-time estimated price, displaying it back to the user in a clear, user-friendly interface.

3.3 Output:

The output of this project is a dynamic web-based prediction interface where users can estimate the price of a used car in real time. After launching the Flask application, the system displays a visually appealing web page with a car-themed background and an interactive form. The user begins by selecting the car company from a dropdown list containing all available brands (e.g., BMW, Hyundai, Maruti). Once the company is chosen, the model dropdown dynamically updates to show only the models corresponding to that company. The user then selects the year of purchase, chooses the fuel type (Petrol, Diesel, or LPG), and enters the number of kilometres driven. After providing all the details, the user clicks the prediction button, and the trained machine learning model processes the inputs. Within seconds, the application displays the estimated price of the car prominently below the form, formatted in Indian Rupees (₹). This output is clear, immediate, and easy to interpret, offering an accessible and reliable valuation tool for buyers and sellers of used cars. The real-time feedback helps users make informed decisions without needing any technical knowledge.

International Scientific Journal of Engineering and Management (ISJEM)
Volume: 04 Issue: 07 | July – 2025
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI: 10.55041/ISJEM04748

Fig1: Home Page of web application

Fig2: Displaying available car companies

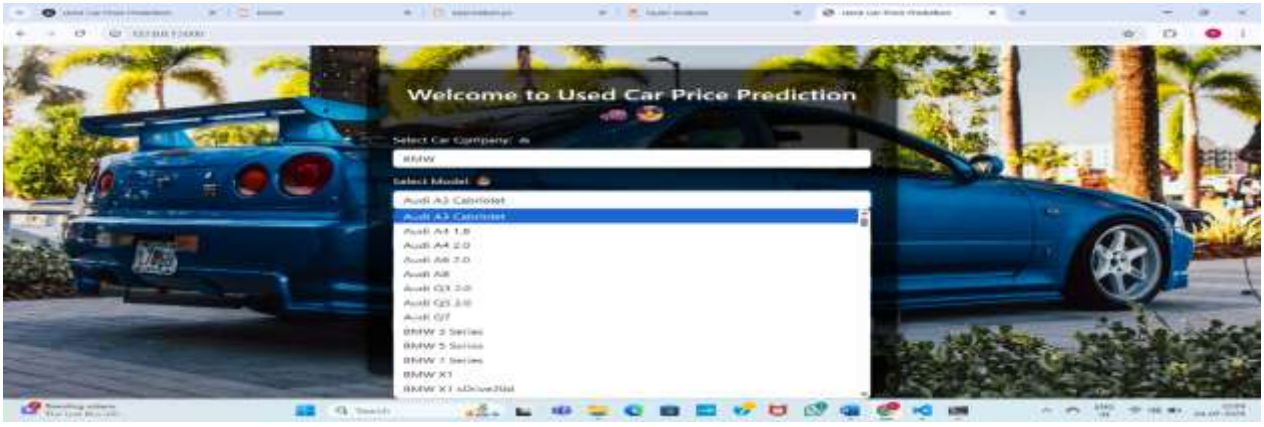

Fig3: Displaying available car companies

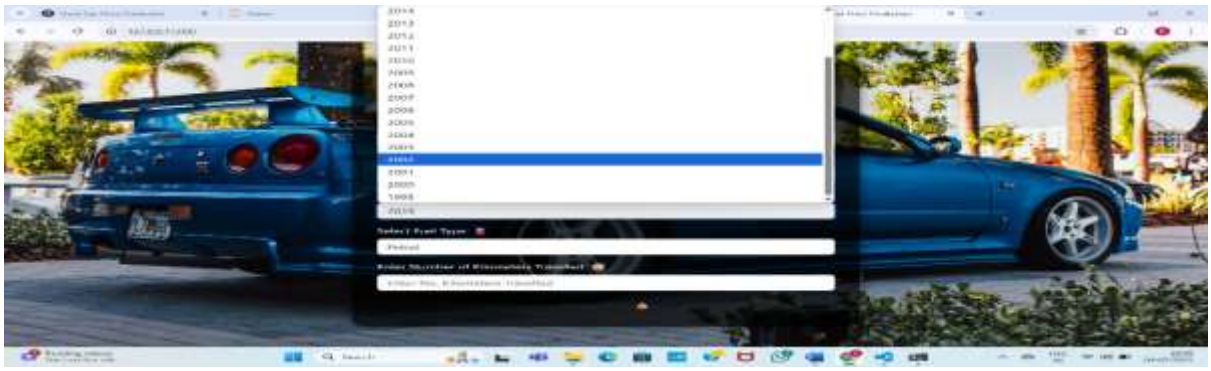

Fig4: Displaying available years

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 07 | July – 2025 DOI: 10.55041/ISJEM04748
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Fig5: Displaying available Fuel types

Fig6: Enter number of kilometres Travelled by the car

Fig7: Final Output

IV. RESULTS:

The project successfully implemented a machine learning–based system to predict used car prices by training a Linear Regression model[8] on historical car data with features such as brand, model, year, fuel type, and kilometres driven. The model demonstrated good predictive accuracy, as evaluated through R-squared and Mean Squared Error metrics. A user-friendly Flask web application was developed to make the predictions accessible, allowing users to enter car details through interactive dropdowns and input fields. Upon submission, the system processes the inputs and instantly displays the estimated price in Indian Rupees. Overall, the project delivers a reliable, real-time valuation tool that simplifies price estimation for buyers and sellers of used cars.

V. DISCUSSIONS:

The project showed that machine learning can accurately predict used car prices when trained on clean, structured data. Linear Regression [8] worked well as a baseline model but may not capture all complex pricing factors. Data preprocessing was essential to handle inconsistent labels and missing values, ensuring the model performed reliably. Integrating the trained model with a Flask web application made the tool easy and practical for users to access real-time price estimates.

Future improvements could include testing advanced models like Random Forests or Gradient Boosting to further boost prediction accuracy and better reflect real market trends.

VI. CONCLUSION:

This project successfully demonstrated how machine learning can be applied to predict used car prices accurately. By using Linear Regression [8], the system provided reliable estimates based on key features like brand, model, year, fuel type, and kilometres driven. Careful data cleaning and preprocessing were critical to achieving consistent results. The Flask web application [7] made the predictions easily accessible through a simple, user-friendly interface. The approach saves time compared to manual pricing methods and reduces human bias. In the future, incorporating more advanced algorithms and additional data could further improve performance and precision. machine learning algorithms for solving complex problems. The project also demonstrates the potential of machine learning algorithms in the financial industry for detecting fraud and improving security.

VII. FUTURE SCOPE:

In the future, this project can be improved by using more advanced machine learning algorithms like Random Forests or Gradient Boosting to increase prediction accuracy. Adding more features, such as car condition, location, and ownership details, could make the estimates even more precise. The system can be upgraded to retrain itself automatically with new market data to stay current. Enhancing the web application with interactive charts and downloadable reports would make it more user-friendly. Deploying the app on a cloud platform will help it handle more users easily. These improvements will make the tool smarter, faster, and more valuable to buyers and sellers.

VIII. ACKNOWLEDGEMENT:

Muppala Naga Keerthi working as an Assistant Professor in Master of Computer Applications in Sanketika Vidya Parishad Engineering College, Visakhapatnam, Andhra Pradesh, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE with 14 years of experience in computer science, and member in IAENG. Her areas of interests in C, Java, Data Structures, DBMS, Web Technologies, Software Engineering and Data Science.

Jonnadula Durga Rao is pursuing his final semester MCA in Sanketika Vidya Parishad Engineering College, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE With interest in Machine learning J Durga Rao has taken up his PG project on PREDICTING PRICES OF USED CARS WITH PYTHON AND ML and published the paper in connection to the project under the guidance of M Naga Keerthi, Assistant Professor, Master of Computer Applications, SVPEC.

REFERENCES

1. PyCharm – Python IDE: https://www.jetbrains.com/pycharm/download/

ISSN: 2583-6129
DOI: 10.55041/ISJEM04748

2. Jupyter Notebook:
https://jupyter.org/

3. GitHub – Car Price Prediction Projects:
https://github.com/topics/car-price-prediction

4. Car Images – IStockPhoto:
https://www.istockphoto.com

5. Car Dataset – Kaggle:
https://www.kaggle.com/datasets/arijitshingharoy/quikr-carcsv

6. YouTube – Car Price Prediction Tutorials:
https://www.youtube.com/results?search_query=car+price+prediction+using+machine+learning

7. Flask Web App Tutorial – DigitalOcean:
https://www.digitalocean.com/community/tutorials/how-to-make-a-web-application-using-flask-in-python-3

8. Linear Regression – GeeksforGeeks:
https://www.geeksforgeeks.org/ml-linear-regression/

9. HTML Guide – W3Schools:
https://www.w3schools.com/html/

10. CSS Guide – W3Schools:
https://www.w3schools.com/css/

11. Bootstrap Documentation:
https://getbootstrap.com/docs/3.4/css/

12. ER Diagram – Simplilearn:
https://www.simplilearn.com/tutorials/sql-tutorial/er-diagram-in-dbms

13. DFD Diagram Tool – SmartDraw:
https://www.smartdraw.com/data-flow-diagram/

14. Test Case Format with Example – Studocu:
https://www.studocu.com/in/document/srm-institute-of-science-and-technology/design-and-analysis-of-algorithms/test-case-format-with-example/85359390

15. Scikit-Learn Documentation – Official:
https://scikit-learn.org/stable/documentation.html

16. Flask Documentation – Official:
https://flask.palletsprojects.com/

17. NumPy Documentation:
https://numpy.org/doc/

18. Pandas Documentation:
https://pandas.pydata.org/docs/

19. Pickle Module – Python Docs:
https://docs.python.org/3/library/pickle.html

20. Data Preprocessing in ML – Analytics Vidhya:
https://www.analyticsvidhya.com/blog/2021/06/data-preprocessing-techniques-you-should-know/

