



**Delhi Skill and
Entrepreneurship University**
Govt. of NCT of Delhi

MACHINE LEARNING USING PYTHON



Project file



Group Members:

Rishi-41221130
Sapna-41221150
Rashmi-41221125

Submitted to:

Ms. Komal
Dhingra

TITLE OF THE PROJECT

Name of students:

- **Sapna: 41221150**
- **Rishi: 41221130**
- **Rashmi: 41221125**

Name Of Guide: Ms. KOMAL DHINGRA

Designation: PROCTOR

Student's Signature

Guide Signature

Head of Department: Mr. VIRENDER DAGAR

DECLARATION

I hereby declare that the project work entitled “**Title of the project**” submitted to DSEU Dwarka Campus, is a record of an original work done by me under the guidance of “**Name of the supervisor**”. This project work is submitted in the partial fulfilment of the requirements for the award of the Bachelor of Computer Application. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of Candidate:

Name of the students

Roll no:

INDEX

Index		
S. No.	Topic	Page No.
1.	Title of Project	
2.	Declaration	
3.	Acknowledgement	
4.	Introduction	
5.	Literature Review	
6.	Objective	
7.	Project Design	
8.	Work Plan and Methodology	
9.	Implementation / Code etc.	
10.	Testing	
11.	Results and Findings	
12.	Limitations and Future Scope	
13.	Conclusion	
14.	References	

ACKNOWLEDGEMENT

I want to express my gratitude to the exceptional team behind the development of our machine learning model for predicting the selling price of used cars. Each team member, including **Rishi, Rashmi, Sapna** has played a vital role in collecting, analysing, and fine-tuning the data.

I would also like to acknowledge the guidance and support provided by **Ms Komal Dhingra Mam**, whose expertise has been invaluable throughout this journey.

This project marks a significant stride in leveraging data science for practical applications. Thank you all for your hard work, dedication, and collaborative spirit. Together, we have contributed to advancing the landscape of predictive modelling in the used car market.

INTRODUCTION

Welcome to our project! We're working on building a machine learning model that predicts the selling price of used cars. We're doing this in the Jupyter Notebook environment, using Python and the scikit-learn library.

Our process is structured: we start by exploring the data, then move on to preprocessing, selecting the right model, training it, evaluating the results, and finally, visualizing the insights we uncover.

This project isn't just about coding; it's a practical exploration into understanding how we can use machine learning to predict the prices of used cars more accurately. Join us as we navigate through the different steps of this exciting journey!

LITERATURE REVIEW

1. Data Collection:

In the initial phase of our project, the data collection process is fundamental. We utilize the "car data.csv" file obtained from online platforms known for hosting extensive and diverse datasets related to used cars. These platforms often aggregate information from various sources, ensuring a rich and representative dataset for our predictive modelling.

2. Data Preparation:

Data preparation involves several crucial steps to ensure the dataset is clean and ready for analysis. We begin by removing duplicate entries to avoid skewing results. Handling null values is another crucial aspect, where we refer to best practices shared in online data science communities. Exploring the nature of the data through descriptive statistics and visualizations helps us understand its characteristics and inform our data preparation strategies.

3. Data Wrangling:

The data wrangling phase involves organizing and structuring the data for optimal use in our predictive model. Sorting and filtering the data ensure that it is in a proper physical format, while the removal of outliers helps mitigate potential distortions in our analysis. Online data science forums and communities provide insights and discussions on effective data wrangling techniques, influencing our approach to preparing the dataset.

4. Data Analysis:

Choosing between classification or regression is a critical decision in the data analysis phase. We refer to insights shared by practitioners in online data science communities to make informed decisions based on the nature of our prediction task. These discussions often delve into real-world scenarios, providing valuable context for our analytical approach.

5. Train the Model:

The training phase involves utilizing machine learning algorithms and tools like `train_test_split()` to build a predictive model. Online data science platforms and community-driven resources offer

practical guidance on algorithm selection, parameter tuning, and the nuances of the training process. This collective wisdom helps us navigate the complexities of model training effectively.

6. Test the Model:

Model testing is a crucial step to assess accuracy and generalization performance. Drawing from discussions in online data science forums, we experiment with various algorithms and evaluation metrics to identify the best-performing model. Community-driven insights contribute to a robust validation process, ensuring the reliability of our predictive model.

7. Deployment:

The deployment phase involves integrating our model into real-world systems. Here, we leverage platforms like Streamlit, a choice influenced by discussions and experiences shared in online communities focused on data science and machine learning applications. These platforms are recognized for their user-friendly interfaces and seamless integration capabilities.

8. Monitoring:

Post-deployment, ongoing monitoring is essential to identify and address any performance shortfalls. Insights from online data science communities guide our approach to regularly examining the model's performance matrix. These shared experiences contribute to a proactive monitoring strategy, ensuring the sustained effectiveness of our predictive model in dynamic real-world environments.

OBJECTIVE

The objective of this project is to develop a robust machine learning model that accurately predicts the selling price of used cars. Through a structured process of data collection, preparation, wrangling, analysis, training, and testing, we aim to leverage machine learning algorithms to create a predictive model. The ultimate goal is to deploy this model in real-world systems, utilizing platforms like Streamlit, and continuously monitor its performance for any shortcomings. By undertaking these steps, we seek to enhance the understanding and application of predictive analytics in the context of the dynamic used car market.

PROJECT DESIGN

What?

This project aims to develop a machine learning model that predicts the selling price of used cars based on a set of input parameters.

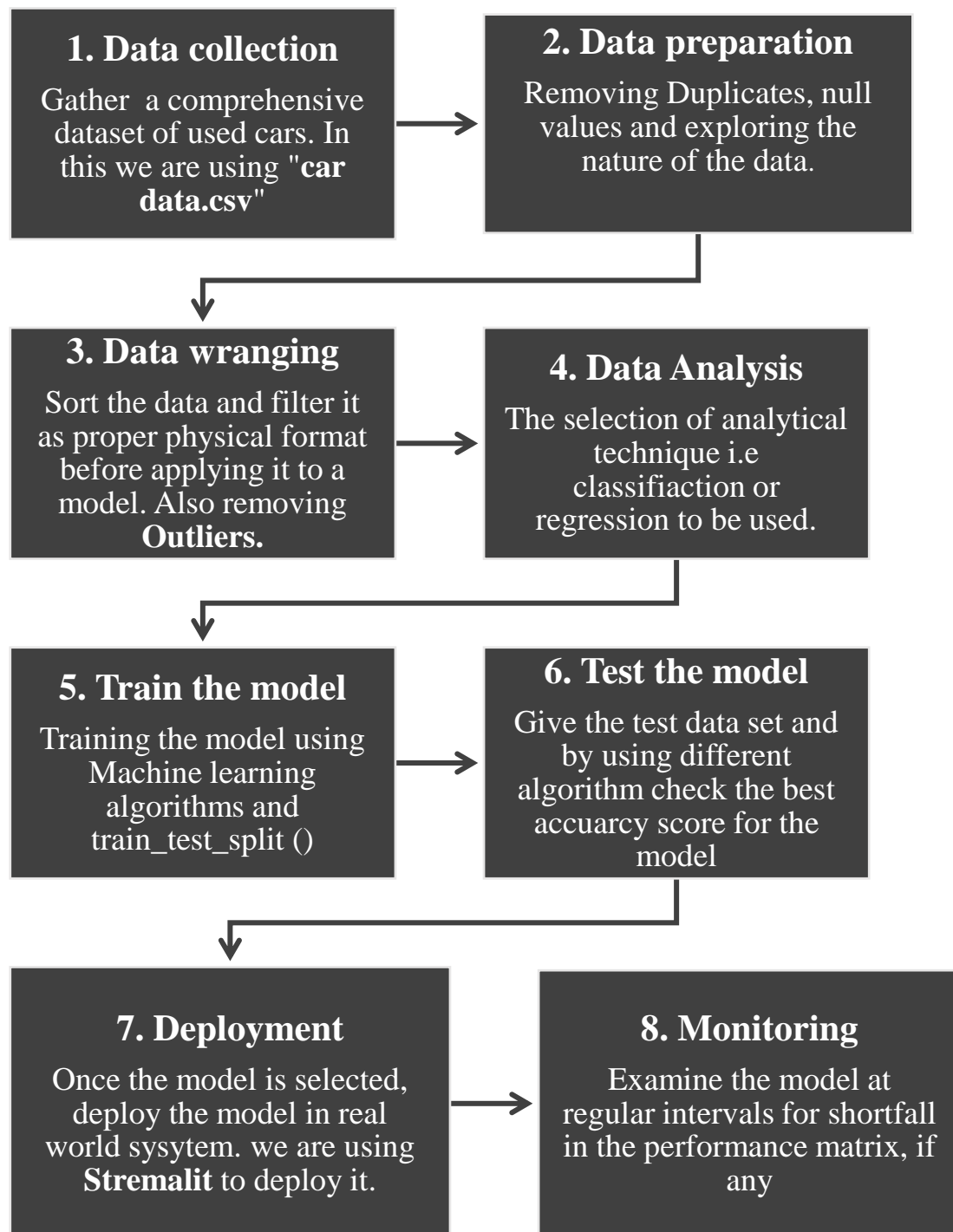
Where?

The proposed machine learning model for predicting used car prices will be developed within the Jupyter Notebook environment, leveraging the capabilities of the Python programming language. The core libraries to be utilized include scikit-learn, a comprehensive machine learning library in Python.

How?

The implementation of the machine learning model for predicting used car prices will follow a systematic and structured approach within the Jupyter Notebook environment. The key steps involve data exploration, preprocessing, model selection, training, evaluation, and visualization.

WORK PLAN AND METHEDOLOGY



IMPLEMENTATION/CODE

1) Display the first 5 rows of the dataset:

```
1 data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

2) Display the first 5 rows of the dataset:

```
1 data.tail()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
296	city	2016	9.50	11.6	33988	Diesel	Dealer	Manual	0
297	brio	2015	4.00	5.9	60000	Petrol	Dealer	Manual	0
298	city	2009	3.35	11.0	87934	Petrol	Dealer	Manual	0
299	city	2017	11.50	12.5	9000	Diesel	Dealer	Manual	0
300	brio	2016	5.30	5.9	5464	Petrol	Dealer	Manual	0

3) Find the shape of the dataset (no. of rows and columns)

```
data.shape
```

(301, 9)

```
print("number of rows",data.shape[0])  
print("number of columns",data.shape[1])
```

number of rows 301
number of columns 9

4) Get info about the dataset.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 301 entries, 0 to 300  
Data columns (total 9 columns):  
#   Column          Non-Null Count  Dtype  
---  ---  
0   Car_Name        301 non-null    object  
1   Year            301 non-null    int64  
2   Selling_Price   301 non-null    float64  
3   Present_Price   301 non-null    float64  
4   Kms_Driven      301 non-null    int64  
5   Fuel_Type       301 non-null    object  
6   Seller_Type     301 non-null    object  
7   Transmission     301 non-null    object  
8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)  
memory usage: 21.3+ KB
```

5) Data preprocessing:

```
data.head(1)
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0

```
date_time = datetime.datetime.now()
```

```
print(date_time)
```

```
2023-11-26 15:35:38.193635
```

```
data['Age']=date_time.year - data['Year']
```

```
data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	9
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	10
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	6
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	12

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	9
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	10
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	6
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	12
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	9

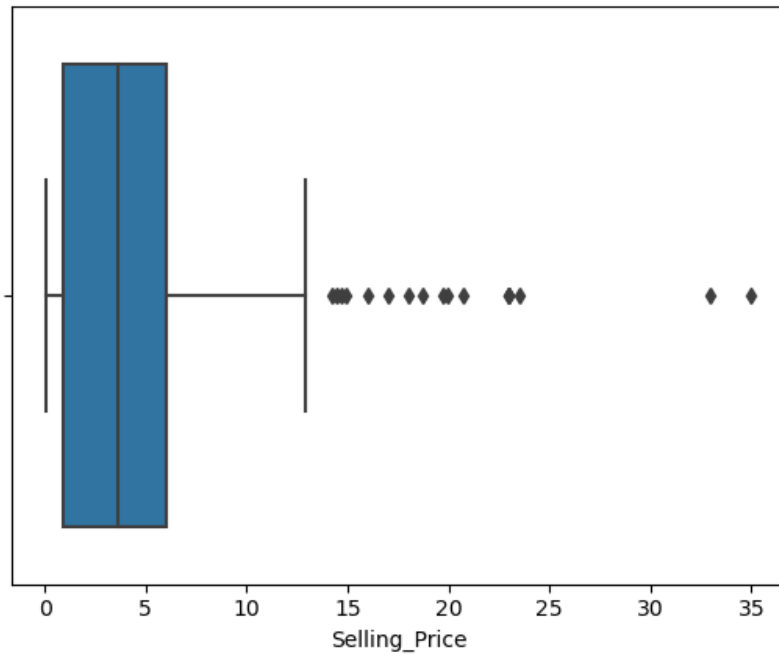
```
data.drop('Year',axis=1,inplace=True)
```

```
data.head()
```

	Car_Name	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	3.35	5.59	27000	Petrol	Dealer	Manual	0	9
1	sx4	4.75	9.54	43000	Diesel	Dealer	Manual	0	10
2	ciaz	7.25	9.85	6900	Petrol	Dealer	Manual	0	6
3	wagon r	2.85	4.15	5200	Petrol	Dealer	Manual	0	12
4	swift	4.60	6.87	42450	Diesel	Dealer	Manual	0	9

6) Outlier Removal:

```
sns.boxplot(x=data['Selling_Price'])  
figzi
```



```
In [27]: sorted(data['Selling_Price'],reverse=True)
```

```
Out[27]: [35.0,  
          33.0,  
          23.5,  
          23.0,  
          23.0,  
          23.0,  
          20.75,  
          19.99,  
          19.75,  
          18.75,  
          18.0,  
          17.0,  
          16.0,  
          14.9,  
          14.73,  
          14.5,  
          14.25,  
          12.9,  
          12.5,  
          11.75]
```

```
data = data[~(data['Selling_Price']>=33.0) &(data['Selling_Price']<=35.0)]
```

```
data.shape
```

```
(299, 9)
```

7) Encoding the categorical value:

```
data.head(1)
```

	Car_Name	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	3.35	5.59	27000	Petrol	Dealer	Manual	0	9

```
data['Fuel_Type'].unique()
```

```
array(['Petrol', 'Diesel', 'CNG'], dtype=object)
```

```
data['Fuel_Type'] = data['Fuel_Type'].map({'Petrol':0, 'Diesel':1, 'CNG':2})
```

```
data['Fuel_Type'].unique()
```

```
array([0, 1, 2], dtype=int64)
```

```
data['Seller_Type'].unique()
```

```
array(['Dealer', 'Individual'], dtype=object)
```

```
data['Seller_Type'] = data['Seller_Type'].map({'Dealer':0, 'Individual':1})
```

```
data['Seller_Type'].unique()
```

```
data['Transmission'].unique()
```

```
array(['Manual', 'Automatic'], dtype=object)
```

```
data['Transmission'] = data['Transmission'].map({'Manual':0, 'Automatic':1})
```

```
data['Transmission'].unique()
```

```
array([0, 1], dtype=int64)
```

```
data.head()
```

	Car_Name	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	3.35	5.59	27000	0	0	0	0	9
1	sx4	4.75	9.54	43000	1	0	0	0	10
2	ciaz	7.25	9.85	6900	0	0	0	0	6
3	wagon r	2.85	4.15	5200	0	0	0	0	12
4	swift	4.60	6.87	42450	1	0	0	0	9

8) Store Feature Matrix in X and Response(Target) in Vector Y

```
X = data.drop(['Car_Name', 'Selling_Price'], axis=1)
Y = data['Selling_Price']
```

```
Y
```

```
0      3.35
```

```
1      4.75
```

```
2      7.25
```

```
3      2.85
```

```
4      4.60
```

```
...
```

```
296     9.50
```

```
297     4.00
```

```
298     3.35
```

```
299    11.50
```

```
300     5.30
```

```
Name: Selling_Price, Length: 299, dtype: float64
```

9)

9. Splitting The Dataset Into the Training Set and Test Set

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.20,random_state=42)
```

10) Model training:

```
lr = LinearRegression()
lr.fit(X_train,Y_train)

rf = RandomForestRegressor()
rf.fit(X_train,Y_train)

xgb = GradientBoostingRegressor()
xgb.fit(X_train,Y_train)

xg = XGBRegressor()
xg.fit(X_train,Y_train)
```

▼ XGBRegressor

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
```

11) Predicting on Test Data

```
y_pred1 = lr.predict(X_test)
y_pred2 = rf.predict(X_test)
y_pred3 = xgb.predict(X_test)
y_pred4 = xg.predict(X_test)
```

12) Saving the model


```
xg = XGBRegressor()  
xg_final = xg.fit(X,Y)
```

```
xg_final.save_model('xgb_model.json')
```

```
joblib.dump(xg_final, 'Car_Price_Predictor')  
['Car_Price_Predictor']
```

```
model = joblib.load('Car_Price_Predictor')
```

GUI:

```
from tkinter import *
import joblib
import pandas as pd

def show_entry_fields():
    p1 = float(e1.get())
    p2 = float(e2.get())
    p3 = float(e3.get())
    p4 = float(e4.get())
    p5 = float(e5.get())
    p6 = float(e6.get())
    p7 = float(e7.get())

    model = joblib.load('car_price_predictor')
    data_new = pd.DataFrame({
        'Present_Price': [p1], # Wrap p1 in a list to create a DataFrame
        'Kms_Driven': [p2],
        'Fuel_Type': [p3],
        'Seller_Type': [p4],
        'Transmission': [p5],
        'Owner': [p6],
        'Age': [p7]
    })

    result = model.predict(data_new)
    Label(master, text=f"Car Purchase amount: {result[0]:.2f}").grid(row=8, columnspan=2)
```

```
master = Tk()
master.title("Car Price Prediction Using Machine Learning")

Label(master, text="Car Price Prediction Using Machine Learning", bg="black", fg="white").grid(row=0, columnspan=2)

Label(master, text="Present_Price").grid(row=1)
Label(master, text="Kms_Driven").grid(row=2)
Label(master, text="Fuel_Type").grid(row=3)
Label(master, text="Seller_Type").grid(row=4)
Label(master, text="Transmission").grid(row=5)
Label(master, text="Owner").grid(row=6)
Label(master, text="Age").grid(row=7)

e1 = Entry(master)
e2 = Entry(master)
e3 = Entry(master)
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)

e1.grid(row=1, column=1)
e2.grid(row=2, column=1)
e3.grid(row=3, column=1)
e4.grid(row=4, column=1)
```

```
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)

e1.grid(row=1, column=1)
e2.grid(row=2, column=1)
e3.grid(row=3, column=1)
e4.grid(row=4, column=1)
e5.grid(row=5, column=1)
e6.grid(row=6, column=1)
e7.grid(row=7, column=1)

Button(master, text='Predict', command=show_entry_fields).grid(row=8, column=1, columnspan=2)

mainloop()
```

TESTING

Evaluating the Algorithm

```
score1 = metrics.r2_score(Y_test,y_pred1)
score2 = metrics.r2_score(Y_test,y_pred2)
score3 = metrics.r2_score(Y_test,y_pred3)
score4 = metrics.r2_score(Y_test,y_pred4)
```

```
print(score1,score2,score3,score4)
```

```
0.6790884983129402 0.751777828416083 0.8842357609786246 0.8887471822279068
```

```
Final_Data = pd.DataFrame({'Models':['LR','RF','GBR','XG'],
                           'R2_SCORE':[score1,score2,score3,score4]})
```

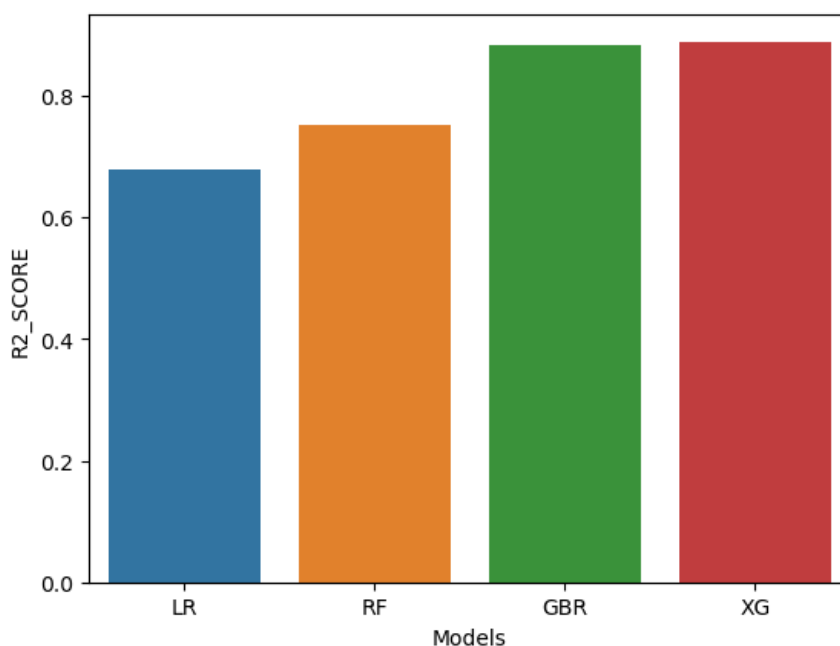
```
Final_Data
```

	Models	R2_SCORE
0	LR	0.679088
1	RF	0.751778
2	GBR	0.884236
3	XG	0.888747

```
sns.barplot(x = Final_Data['Models'],y = Final_Data['R2_SCORE'])
```

```
<Axes: xlabel='Models', ylabel='R2_SCORE'>
```

```
<Axes: xlabel='Models', ylabel='R2_SCORE'>
```



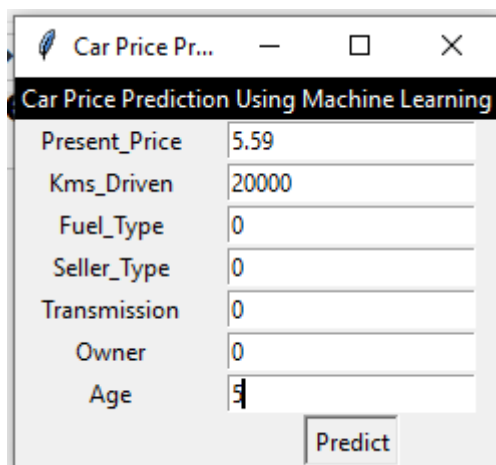
Predicting on new data

```
import pandas as pd
data_new = pd.DataFrame({
    'Present_Price':5.59,
    'Kms_Driven':20000,
    'Fuel_Type': 0,
    'Seller_Type': 0,
    'Transmission': 0,
    'Owner': 0,
    'Age': 5
},index=[0])
```

```
model.predict(data_new)
```

```
array([3.830781], dtype=float32)
```

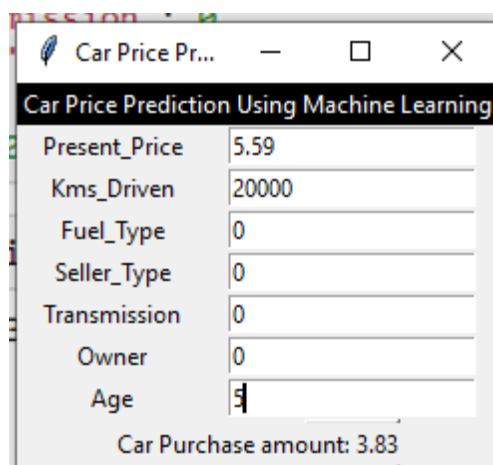
Tkinter:



Car Price Prediction Using Machine Learning

Present_Price	5.59
Kms_Driven	20000
Fuel_Type	0
Seller_Type	0
Transmission	0
Owner	0
Age	5

Predict



Car Price Prediction Using Machine Learning

Present_Price	5.59
Kms_Driven	20000
Fuel_Type	0
Seller_Type	0
Transmission	0
Owner	0
Age	5

Car Purchase amount: 3.83

DEPLOYMENT

- We are employing the machine learning model based on the XGBoost algorithm for the deployment of a web application using the Streamlit framework.
- The deployment process involves integrating the trained XGBoost model into the Streamlit framework, which is a popular Python library for creating interactive and data-driven web applications.
- Throughout the deployment process, we ensure seamless integration and functionality, providing a user-friendly interface for interacting with the XGBoost model's predictions and results.
- In summary, the utilization of the XGBoost model in conjunction with the Streamlit framework enables us to create a dynamic and interactive web application, offering users an intuitive experience while exploring the outputs and predictions of our machine learning model.

Code we used to deploy web application:

```
import streamlit as st
import numpy as np
import pandas as pd
import datetime
import xgboost as xgb

import joblib
date_time = datetime.datetime.now()
model = xgb.XGBRegressor()
model.load_model('xgb_model.json')

def main():
    html_temp="""
    <div style = "background-color:lightblue;padding:16px">
    <h2 style="color:black;text-align:center;"> Car Price Prediction Using ML</h2>
    </div>
    """
    st.markdown(html_temp,unsafe_allow_html=True)

    st.markdown("##### Are you planning to sell your car !?\n##### So let's try evaluating the price..")

    st.write('')
    st.write('')
    p1 = st.number_input('What is the current ex-showroom price of the car ? (In Lakhs)',2.5,25.0,step=1.0)
```

```

    p2 = st.number_input('What is distance completed by the car in Kilometers
?',100,50000000,step=100)

    s1 = st.selectbox('What is the fuel type of the car ?',('Petrol','Diesel',
'CNG'))
    if s1=="Petrol":
        p3=0
    elif s1=="Diesel":
        p3=1
    elif s1=="CNG":
        p3=2

    s2 = st.selectbox('Are you a dealer or an individual ?',
('Dealer','Individual'))

    if s2=="Dealer":
        p4=0
    elif s2=="Individual":
        p4=1

    s3 = st.selectbox('What is the Transmission Type ?', ('Manual','Automatic'))
    if s3=="Manual":
        p5=0
    elif s3=="Automatic":
        p5=1

    p6 = st.slider("Number of Owners the car previously had",0,3)

    years = st.number_input('In which year car was purchased
?',1990,date_time.year,step=1)
    p7 = date_time.year-years

    data_new = pd.DataFrame({
        'Present_Price':p1,
        'Kms_Driven':p2,
        'Fuel_Type':p3,
        'Seller_Type':p4,
        'Transmission':p5,
        'Owner':p6,
        'Age':p7
    },index=[0])
    try:
        if st.button('Predict'):
            prediction = model.predict(data_new)
            if prediction>0:
                st.balloons()

```

```

        st.success('You can sell the car for {:.2f}
lakhs'.format(prediction[0]))
    else:
        st.warning("You will be not able to sell this car !!")
    except:
        st.warning("Oops!! Something went wrong\nTry again")

if __name__ == '__main__':
    main()

```

Output:

carpriceprediction-kjmf5ka5nuoajwkpnz4j.streamlit.app

Car Price Prediction Using ML

Are you planning to sell your car !?

So let's try evaluating the price..

What is the current ex-showroom price of the car ? (In Lakhs)

2.50

What is distance completed by the car in Kilometers ?

100

What is the fuel type of the car ?

Petrol

Are you a dealer or an individual ?

Dealer

What is the Transmission Type ?

Manual

Number of Owners the car previously had

0 3

In which year car was purchased ?

1990

Predict

You can sell the car for 1.21 lakhs

Website:

[Car Price Prediction](#)

RESULTS AND FINDINGS

Results:

Model Accuracy: The accuracy of the machine learning model in predicting used car prices is a key result. This metric, measured through evaluation techniques such as Mean Absolute Error (MAE) or other relevant metrics, quantifies how well the model aligns with actual market prices.

Algorithm Comparison: Comparative analysis of different machine learning algorithms applied during the testing phase provides insights into which algorithms perform most effectively for the given prediction task. This result informs the selection of the algorithm with the best predictive capabilities.

Deployment Success: The successful deployment of the model using platforms like Streamlit represents a tangible outcome. It reflects the project's practical utility, allowing users to interact with and benefit from the predictive capabilities of the model in real-world scenarios.

Findings:

Key Features: Identification of key features influencing used car prices is a crucial finding. The model's analysis reveals which variables have the most significant impact on pricing, offering valuable insights into the factors driving market dynamics.

Pattern Recognition: The model's ability to recognize and leverage patterns in the data is a notable finding. Understanding how well the model captures and utilizes patterns in the used car market contributes to insights into the market's predictability.

User Feedback: If implemented, user feedback on predicted prices provides qualitative findings. User interactions and feedback contribute to ongoing model improvement, addressing specific challenges or scenarios not initially considered.

Market Dynamics: The model's performance sheds light on the dynamics of the used car market. The findings may include information on the stability of pricing patterns, responsiveness to external factors, and the model's adaptability to changing conditions.

Error Analysis: An analysis of instances where the model deviates significantly from actual prices offers insights into potential limitations or challenges. Understanding when and why the model may struggle provides valuable information for future enhancements.

Interpretability: Findings related to the interpretability of the model contribute to understanding how well stakeholders, including non-experts, can grasp the rationale behind the model's predictions. This is crucial for building trust and confidence in the model's use.

LIMITATION AND FUTURE SCOPE

Limitations:

- **Data Quality:** The accuracy of the predictive model heavily depends on the quality of the input data. If the dataset contains inaccuracies, outliers, or missing values, it can adversely impact the model's performance.
- **Model Generalization:** The model's ability to generalize to new, unseen data may be limited. If the training dataset is not representative of all possible scenarios, the model may struggle with unfamiliar patterns.
- **Assumption of Stationarity:** The project assumes that the underlying patterns in used car pricing remain relatively constant over time. Economic, seasonal, or market changes may challenge this assumption.
- **Feature Relevance:** The model's performance relies on the relevance of chosen features. If important factors affecting car prices are not included or new influential features emerge, the model may become less effective.
- **Interpretability:** Complex machine learning models, while powerful, can lack interpretability. Understanding the factors driving predictions may be challenging, especially for stakeholders who are not data scientists.

Future Scope:

- **Dynamic Data Updates:** Implement a mechanism to dynamically update the model as new data becomes available. Continuous learning and adaptation to changing market trends could enhance the model's accuracy over time.
- **Feature Engineering:** Explore advanced feature engineering techniques to enhance the model's understanding of complex relationships within the data. This could involve creating new features or transforming existing ones to better capture nuances.
- **Ensemble Methods:** Investigate the use of ensemble methods to combine predictions from multiple models. This approach may improve overall model performance by leveraging the strengths of different algorithms.

- **Integration of External Data:** Incorporate external data sources, such as economic indicators, fuel prices, or demographic information, to enrich the dataset and capture additional factors influencing used car prices.
- **User Feedback Integration:** Implement a feedback loop where users can provide feedback on predicted prices. This information can be used to refine the model, addressing specific challenges or anomalies not captured during the initial training.
- **Explainable AI (XAI):** Integrate explainable AI techniques to enhance model interpretability. This is particularly important for gaining the trust of end-users and stakeholders, allowing them to understand how and why specific predictions are made.
- **Market Segmentation:** Explore the possibility of creating specialized models for different market segments or geographical regions. This approach recognizes that factors influencing used car prices may vary across different contexts.
- **Real-time Deployment:** Move towards real-time deployment to provide users with up-to-the-minute predictions. This could involve using technologies like Apache Kafka for stream processing and real-time analytics.

CONCLUSION

In conclusion, this project represents a comprehensive exploration of predictive analytics in the domain of used car pricing. From the initial stages of data collection to the deployment and monitoring of a machine learning model, each phase contributes to the overarching goal of accurate and reliable predictions.

The data-centric approach, leveraging the "car data.csv" dataset, ensures a robust foundation for the model. The thorough data preparation and wrangling processes address issues such as duplicates, null values, and outliers, paving the way for a clean and well-structured dataset.

The choice between classification and regression, informed by data analysis, reflects a strategic decision-making process tailored to the nature of the prediction task. The subsequent model training and testing phases, employing machine learning algorithms and rigorous evaluation metrics, aim to achieve a high level of predictive accuracy.

The deployment phase, facilitated by platforms like Streamlit, extends the project's impact to real-world systems. It provides a user-friendly interface for leveraging the predictive model in practical scenarios, contributing to informed decision-making in the used car market.

Lastly, the incorporation of monitoring mechanisms ensures the continued effectiveness of the deployed model. Regular evaluations and adjustments based on performance metrics enable the adaptation of the model to evolving conditions, maintaining its relevance in dynamic real-world environments.

In essence, this project not only delves into the intricacies of machine learning and data science but also addresses the practical application of predictive analytics in an industry with tangible economic implications. Through each phase, the project aims to advance the understanding and implementation of data-driven solutions, laying the groundwork for future developments in the field.

REFERENCES

- **Kaggle dataset:**

<https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho?select=car+data.csv>



Car details v3.csv



CAR DETAILS FROM
CAR DEKHO.csv



car data.csv



car details v4.csv

-