11/22/2023

CAR PURCHASE PREDICTION USING MACHINE LEARNING

Project Documentation

Team ID– 593183
Rishima Chowdhury - 21BCE1097
Aniket Chattopadhyay - 21BEC1564

Index

1. INTRODUCTION

1.1 Project Overview

This project is all about creating a clever computer system using machine learning to predict if someone is likely to buy a car. We're making this system smart by looking at things like age, income, and past car purchases. With advanced algorithms and careful data handling, our model is becoming really good at guessing if someone will actually go ahead and make a car purchase. The idea is to help people decide when it's the right time to buy a car, and at the same time, it's a game-changer for car companies. By giving them insights into customer behaviour, it helps businesses sell cars more effectively and tailor their strategies to match what customers are likely to do. we're making sure the computer is super accurate and trustworthy. We're using all the tricks to ensure that the predictions it makes are reliable. So, it's not just about making car-buying decisions easier for individuals; it's also about making the whole car industry smarter and more efficient.

Our project goes beyond predicting car purchases; it encompasses a user-centric platform with distinct features. The review page encourages a community-driven environment, allowing users to share experiences and insights. Complemented by a blog page offering industry updates, a premium subscription option for exclusive benefits, and a feedback page for continuous improvement, our platform ensures a comprehensive and engaging experience. As users explore predictions, they also find a space for community interaction, valuable information, premium services, and a channel for providing feedback, contributing to a holistic and user-driven automotive platform.

1.2 Purpose

Our project’s aim is to simplify and enhance the car-buying experience. We've developed a clever computer program that analyses important details like age, income, and past purchases to predict whether someone is likely to buy a car. Beyond just predictions, we want to assist individuals in making informed decisions about their car purchases. Picture it as a friendly advisor that considers your unique situation and offers personalized guidance.

To complement this, we're creating a dynamic website with features like a review page for sharing experiences, a blog page for interesting automotive stories, and a premium subscription option for those seeking extra perks. The goal is to not only make the process of buying a car more intuitive for individuals but also to provide valuable insights for car companies to improve their strategies. With a feedback page in place, we're dedicated to evolving and refining our project based on user input, ensuring that the entire experience remains engaging, informative, and enjoyable for everyone involved.

2. LITERATURE SURVEY

2.1 Existing Problem

In the car-selling world, a common issue is that companies struggle to figure out who might actually buy a car. This uncertainty leads to a lot of wasted time and money on ads that might not reach the people who are actually looking to purchase a car. Our project acts as a solution by using a smart computer program to predict, with high accuracy, who is most likely to make a car purchase. By analysing factors like age and income, we help car companies pinpoint their target audience more precisely. This means they can direct their marketing efforts where it matters most, making their strategies more efficient and saving resources. It's like giving them a powerful tool to hit the bullseye and connect with potential buyers in a way that makes sense for everyone involved.

Additionally, our project addresses the lack of personalized assistance in the car-buying process. It offers a user-friendly interface where individuals can input their details and receive customized insights, making the car-buying journey more transparent and less overwhelming. This adds a crucial layer of user-centric support to the industry's challenges.

2.2 References

As our reference of our work, we used the dataset available based on car purchase prediction in Kaggle platform and updated its purchased car column based on various types of customers' car using experience of different size and shapes (e.g., hatchback, sedan or SUV cars) and to prepare the model, almost six types of machine learning algorithms were referred to.

After preparing the model properly we referred to the python flask environment and connected our model with highest accuracy to the HTML and CSS based website to get the proper output as per our requirement.

From Kaggle we also found some basic car purchase prediction and car price prediction models, which we used as reference to build our model. For user interface, we referred to some designs pre-existing in Figma, drew our inspirations from them and created our own UI.

2.3 Problem Statement Definition

The problem statement outlined in the provided text revolves around the need for accurate prediction of car purchases using machine learning based on customer data. The goal is to leverage features such as age, income, and historical purchase patterns to develop a model that provides high predictive accuracy. The intended solution aims to guide potential buyers by estimating their likelihood to make a car purchase and is seamlessly integrated into a user-friendly interface for easy predictions. The overarching objective is to revolutionize the automotive industry by offering insights for tailored marketing strategies, enhancing customer experiences, and empowering businesses to optimize resource allocation through data-powered decision-making.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map canvas is a visual tool used in design thinking to understand and empathize with the experiences and perspectives of a particular user or customer. The canvas typically includes sections for capturing the user's thoughts and feelings, what they see and hear, what they say and do, and their pains and gains. By filling in these sections, teams can gain deeper insights into the user's needs and motivations, fostering a more empathetic approach to designing products or solutions that truly resonate with the user's experience. The empathy map serves as a collaborative and visual aid for teams to align on understanding and addressing user needs effectively.

- Here we have provided the link for empathy map canvas.

https://app.mural.co/t/carpurchase2845/m/carpurchase2845/1698727189249/25bef9b770b4fbea6b9ef96a8b156f19afe49a90?sender=uaee2e51c03015550d05c2852

3.2 Ideation & Brainstorming

An ideation and brainstorming template is a structured tool designed to facilitate creative thinking and idea generation during the early stages of a project or problem-solving process. The template typically includes sections to capture various aspects of ideas, such as the problem statement, potential solutions, key features, and potential challenges. It often provides space for participants to jot down their thoughts, enabling a collaborative and organized approach to generating innovative solutions. This template serves as a guide for individuals or teams to explore a wide range of ideas, encourage open communication, and spark creative thinking, ultimately fostering the development of innovative and effective solutions.

- Here we have provided the link for ideation and brainstorming

https://app.mural.co/t/carpurchase2845/m/carpurchase2845/1698242193399/cfea0253244a0a06090bb16d49175243102e9b17?sender=uaee2e51c03015550d05c2852

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

1. **Data Collection and Integration:**
    - Collect and integrate diverse customer data, including age, income, and historical purchase patterns, to build a comprehensive dataset for analysis.
2. **Machine Learning Model:**
    - Develop and implement machine learning algorithms capable of predicting car purchase behavior based on the collected data.
3. **Prediction Interface:**
    - Create a user-friendly interface that allows users to input their demographic information and receive accurate predictions regarding their likelihood to make a car purchase.
4. **Review Page:**
    - Implement a review page where users can share their experiences and insights, contributing to a community-driven environment.
5. **Blog Page:**
    - Develop a blog page featuring relevant and engaging content about the automotive industry, creating an educational space for users.
6. **Premium Subscription:**
    - Introduce a premium subscription page offering exclusive benefits to users, such as advanced analytics, personalized recommendations, or priority access to new features.
7. **Feedback Mechanism:**
    - Include a feedback page to gather user input and improve the system continuously, ensuring responsiveness to user needs and expectations.
8. **Ethical Considerations:**
    - Incorporate features or mechanisms to address ethical considerations in data usage, ensuring responsible and privacy-conscious practices.

9. **Compatibility:**
    - Ensure compatibility with various devices and platforms, making the solution accessible to a broad user base.
10. **Resource Optimization for Businesses:**
    - Provide insights for businesses to optimize marketing resources and tailor approaches based on the predictions, contributing to efficient and targeted customer engagement.

These functional requirements collectively aim to create a robust and user-centric solution that leverages machine learning to predict car purchases while enhancing the overall experience for both individuals and businesses in the automotive industry.

4.2 Non-Functional Requirements

1. **Performance:** The system should provide predictions in real-time, ensuring quick and responsive user interactions. It should be capable of handling a large volume of user requests concurrently.
2. **Scalability:** The system should be designed to handle an increasing amount of data and users as the user base grows.
3. **Reliability:** The system should be highly reliable, with minimal downtime or disruptions in service.
4. **Security:** User data should be stored securely, and the system should comply with relevant data protection and privacy regulations. Implement authentication and authorization mechanisms to control access to sensitive information.
5. **Usability:** The user interface should be intuitive and user-friendly, catering to users with varying levels of technical expertise. The system should provide clear and easily understandable predictions to users.
6. **Compatibility:** Ensure compatibility with different browsers and devices to reach a broad user audience.
7. **Maintainability:** The system should be designed for ease of maintenance and updates, with modular components that can be modified or replaced without disrupting the entire system.
8. **Ethical Considerations:** The system should adhere to ethical standards in data usage and algorithmic decision-making, avoiding bias and discriminatory outcomes.
9. **Interoperability:** Ensure that the system can seamlessly integrate with other relevant systems and platforms within the automotive industry.
10. **Compliance:** Ensure compliance with industry standards, legal regulations, and ethical guidelines governing data usage and machine learning applications.
11. **Feedback Mechanism:** Implement a feedback mechanism to gather user input on system performance and usability, enabling continuous improvement.
12. **Response Time:** Define acceptable response times for user interactions and predictions, ensuring a positive user experience.

These non-functional requirements are essential for ensuring that the system not only performs its core functions effectively but also meets broader criteria related to performance, security, usability, and ethical considerations.

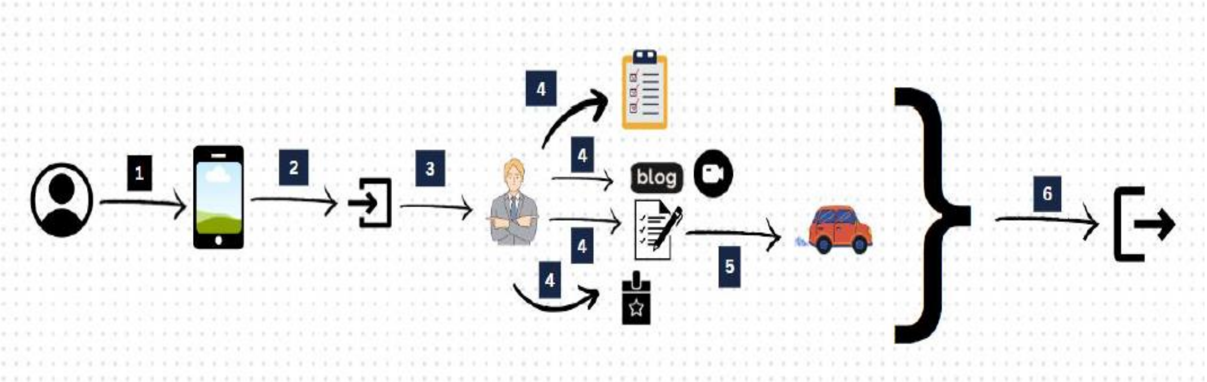
5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

In this project design phase, we used the main two types of diagrams to represent our project plans and to-dos. Those are:

1) Simplified flow diagram,
2) Data flow diagram.

Simplified Flow Diagram:



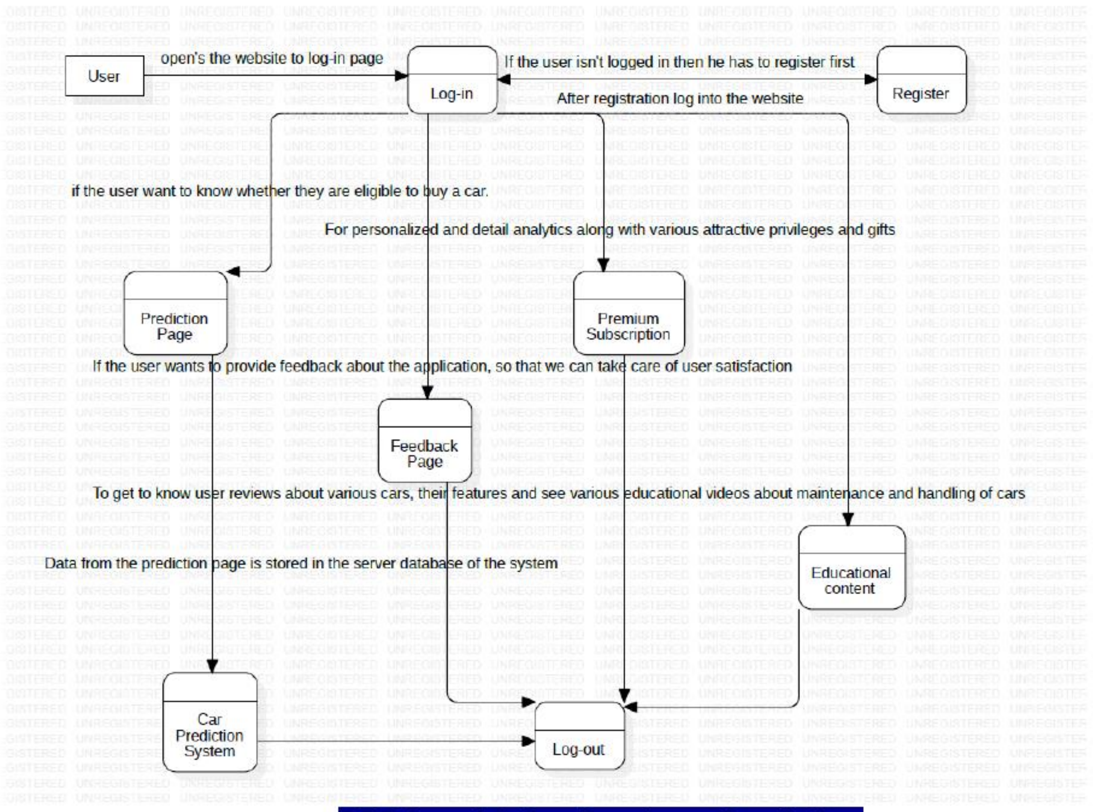
Explanation:

1. User opens the application in mobile or laptop.
2. They register (if the user is new to the site) or log-into (if the user has already registered) the application.
3. The user has to decide:

- Whether he/she wants car prediction analysis, i.e., if he/she is eligible to buy a car, the he has to enter his income and age.
- Whether he wants to write feedback of the application.
- Whether he wants to subscribe for a premium membership.
- Whether he wants to go through other customer reviews about various cars to get a clear idea of what they want to buy.

4. According to the choice the user is guided to that particular web-page.
5. If car prediction analysis is chosen by the user in step 4 then here, he will get the whole predicted analysis.
6. After using the application, the user has to log-out from the website to manage the load.

Data Flow Diagram:


User
open's the website to log-in page
Log-in
If the user isn't logged in then he has to register first
After registration log into the website
Register
if the user want to know whether they are eligible to buy a car.
For personalized and detail analytics along with various attractive privileges and gifts
Prediction Page
Premium Subscription
If the user wants to provide feedback about the application, so that we can take care of user satisfaction
Feedback Page
To get to know user reviews about various cars, their features and see various educational videos about maintenance and handling of cars
Educational content
Data from the prediction page is stored in the server database of the system
Car Prediction System
Log-out

DFD Level-0 Car Purchase Prediction

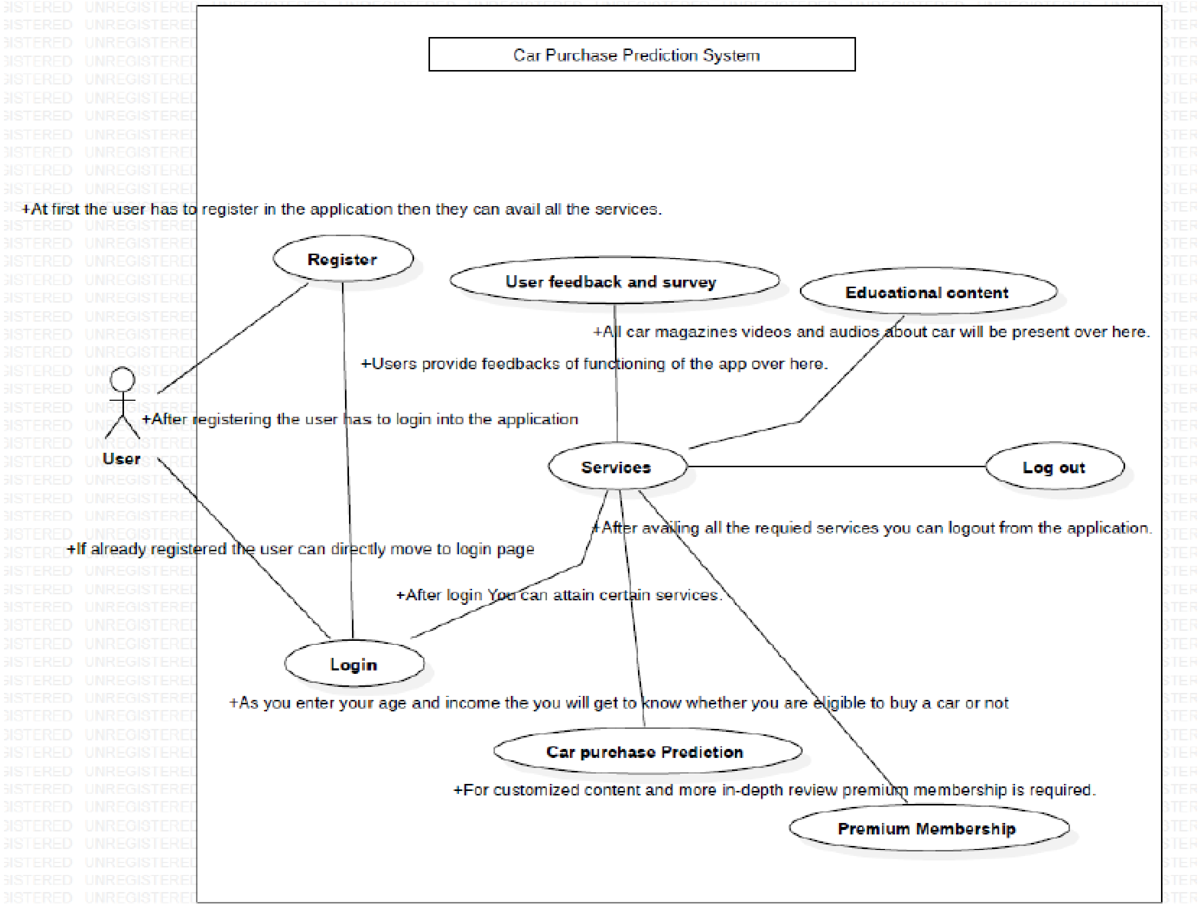
5.2 Solution Architecture

Designing a solution architecture for a car purchase prediction project involves planning, researching and finalizing components of the system to achieve our project goals effectively and efficiently. Overview of the possible solution architecture is:

● **User Interface (UI)-** A user friendly, web-based interface is to be developed that will allow users to interact with the system, where they will input data, view recommendations, read community reviews and access all other contents.
● **Backend Server-** Handles user requests, stores and processes data. This will run core application logic.
● **Database-** Used to store data, create a relational database for structured data, if data is unstructured, we have to create a NoSQL database.
● **Machine Learning Engine-** The backbone of the project is a machine learning engine, a scalable, highly accurate and actively functioning machine learning model has to be chosen, with some of the important frameworks like sci-kit or tensorflow.
● **Data collection and Integration-** The data collected from the users and marketplaces must be integrated into machine learning models.
● **Community Engagement Platform-** A platform where users can share their car ownership experiences, write reviews, and engage in discussions.
● **User Feedback and survey module** - A google forms link can be provided or a form can be created using HTML, CSS, JS.

● **Educational Content Module** - This can contain articles, videos and other tools to educate users.
● **Backup and Disaster Recovery** - implement regular data backups and a disaster recovery plan to ensure data integrity and system availability in case of unforeseen incidents.
● **Accessibility and Voice Interfaces-** accessibility features and voice interfaces are added to ensure that the platform is usable by a diverse range of users, including those with disabilities.

A **Use case Diagram** to explain the full architecture and functionality of the application:


Car Purchase Prediction System
+At first the user has to register in the application then they can avail all the services.
Register
User feedback and survey
Educational content
+All car magazines videos and audios about car will be present over here.
+Users provide feedbacks of functioning of the app over here.
User
+After registering the user has to login into the application
Services
Log out
+After availing all the requied services you can logout from the application.
+If already registered the user can directly move to login page
+After login You can attain certain services.
Login
+As you enter your age and income the you will get to know whether you are eligible to buy a car or not
Car purchase Prediction
+For customized content and more in-depth review premium membership is required.
Premium Membership

6. PROJECT PLANNING & SCHEDULING

In this project planning and scheduling phase, we formed the technical architecture by preparing the technical architecture of the project and listing out the components and technologies along with application characteristics. Up next sprint planning and estimation is calculated via product backlog and sprint schedule calculations and finally sprint delivery schedule is maintained by preparing the project tracker, burndown chart and calculating the project velocity.

6.1 Technical Architecture

Architectural Diagram:

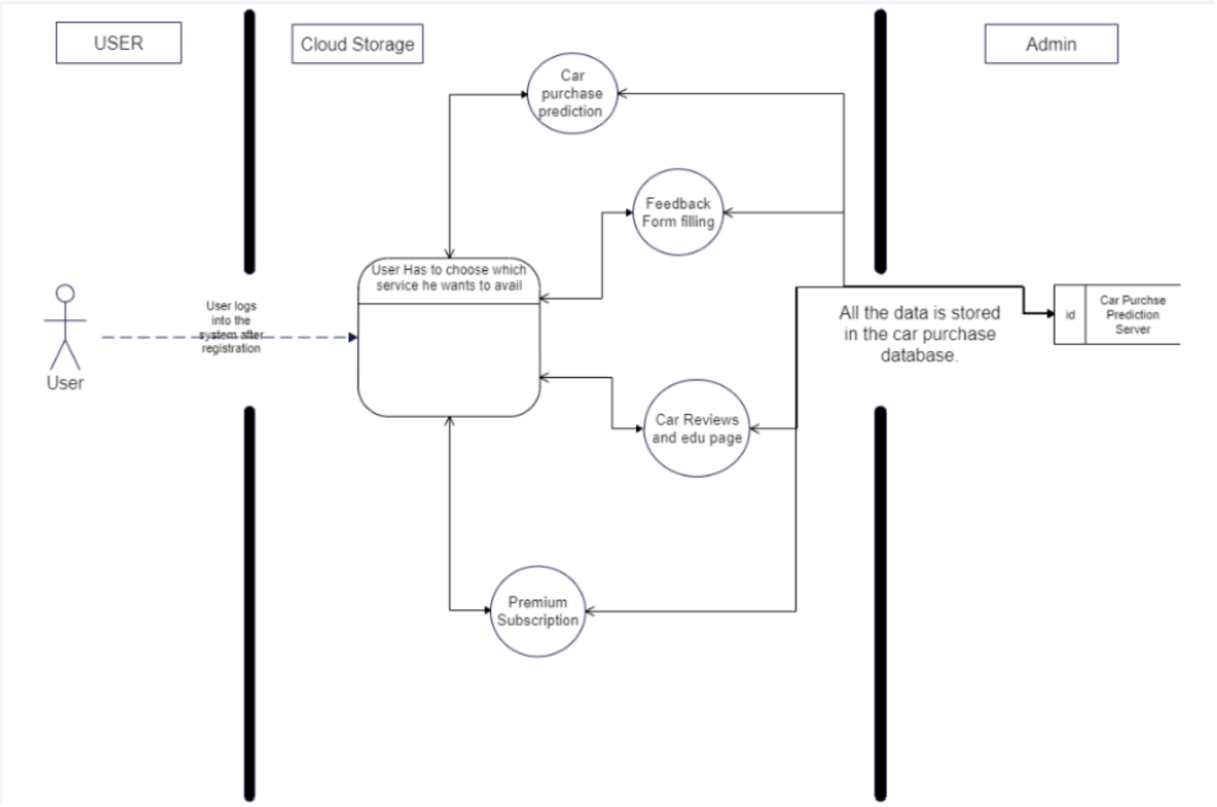

USER
Cloud Storage
Admin
Car purchase prediction
Feedback Form filling
User Has to choose which service he wants to avail
User
User logs into the system after registration
All the data is stored in the car purchase database.
id
Car Purchse Prediction Server
Car Reviews and edu page
Premium Subscription

Table-1 : Components & Technologies:

S.No.	Component	Description	Technology
1.	User Interface	How user interacts with application e.g., Web UI, Mobile App ,etc.	HTML, CSS, JavaScript / React Js, etc.
2.	Application Logic-1	Basic Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for validation of user	JavaScript frameworks and other backend frameworks.
4.	Application logic-3	Logic for creation of educational content	Content management system like WordPress, frontend frameworks, analytics tool like google analytics.
5.	Application Logic-4	Logic for creation of feedback form	Form created using Google or Microsoft forms.
6.	Application Logic-5	Logic for allotting premium subscription	Subscription management like stripe or Braintree, Subscription tiers and plan configurations.
7.	Database	Data Type, Configurations etc.	MySQL, NOSQL, RDBMS,etc
8.	Cloud Database	Database Service on Cloud	Various google cloud databases like, Cloud BigQuery, Cloud Firestore, etc.
9.	File Storage	File storage requirements	Cloud file storage systems and local file storage systems.
10.	External API-1	Purpose of External API used in the application	Automotive data API like Edmunds API, Market Data API like Financial Market Data API, etc
11.	External API-2	Purpose of External API used in the application	Social Media Data API like Facebook Graph API, Environmental Impact Data like EPA Fuel Economy API and Carbon Interface API
12.	Machine Learning Model	Purpose of Machine Learning Model	Regression or Classification model, etc
13.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Google Cloud Platform (GCP), Cloud SQL (PostgreSQL or MySQL), Google Workspace (formerly G Suite).

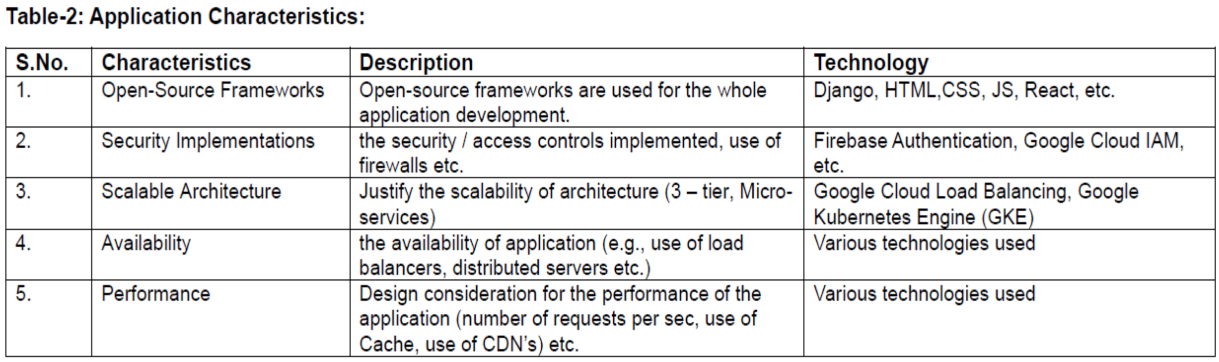


Table-2: Application Characteristics:

S.No.	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks are used for the whole application development.	Django, HTML,CSS, JS, React, etc.
2.	Security Implementations	the security / access controls implemented, use of firewalls etc.	Firebase Authentication, Google Cloud IAM, etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Google Cloud Load Balancing, Google Kubernetes Engine (GKE)
4.	Availability	the availability of application (e.g., use of load balancers, distributed servers etc.)	Various technologies used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Various technologies used

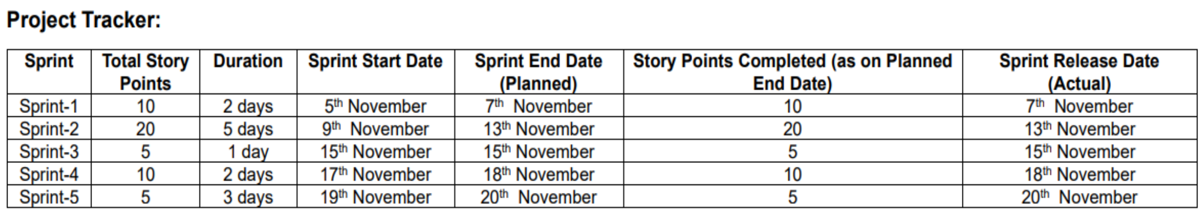
6.2 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	high	Rishima
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email & password	1	medium	Rishima
Sprint-2	Prediction Page	USN-3	Once logged in, I can enter the data and which goes into the database, on which ML model is implemented and prediction is shown in the output screen.	2	high	Aniket
Sprint-3	Feedback Page	USN-4	I can even provide feedback of the website on the page so that the developers can do the necessary changes where they lack.	1	medium	Aniket
Sprint-5	Premium Subscription	USN-5	I can apply for premium membership to avail personalized services and more detailed analysis.	1	medium	Aniket
Sprint-4	Educational Content	USN-6	I can even look into other user's feedback about various car models, and videos about how to maintain my car and all the features of a particular car.	1	low	Rishima

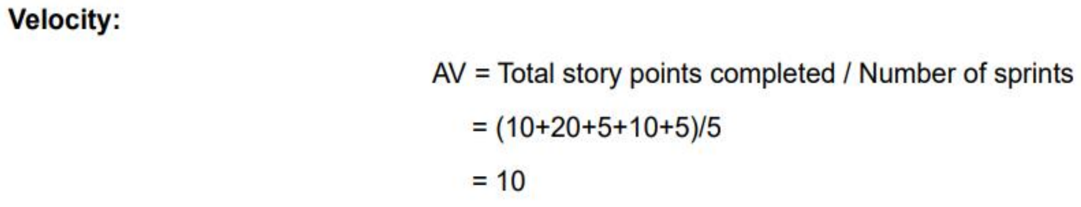
6.3 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart:



Project Tracker:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	2 days	5th November	7th November	10	7th November
Sprint-2	20	5 days	9th November	13th November	20	13th November
Sprint-3	5	1 day	15th November	15th November	5	15th November
Sprint-4	10	2 days	17th November	18th November	10	18th November
Sprint-5	5	3 days	19th November	20th November	5	20th November



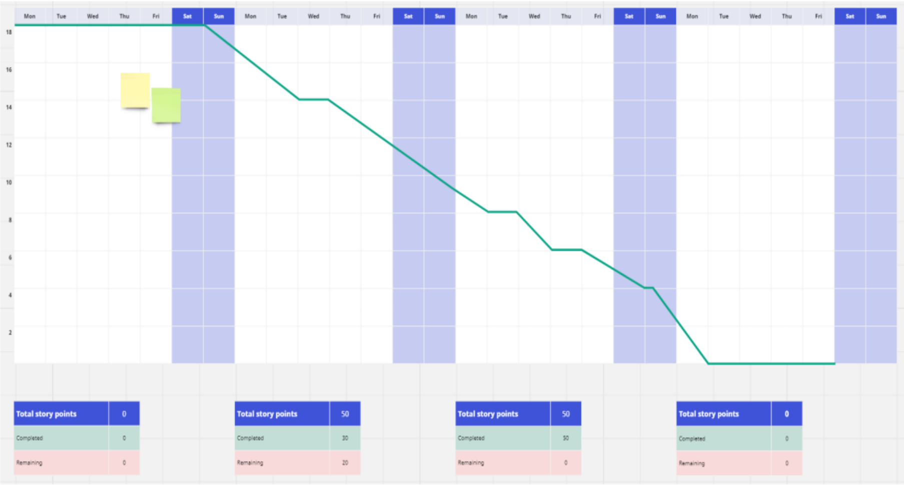
Velocity:

AV = Total story points completed / Number of sprints

= (10+20+5+10+5)/5

= 10

Burndown Chart:



Total story points	0
Completed	0
Remaining	0

Total story points	50
Completed	30
Remaining	20

Total story points	50
Completed	50
Remaining	0

Total story points	0
Completed	0
Remaining	0

7. CODING & SOLUTIONING

7.1 Feature 1 (Car purchase prediction)

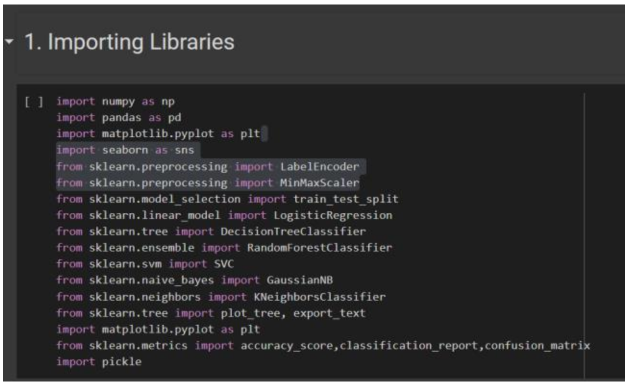
The main Aim of our project was to predict that a person is eligible to buy a car or not based on their age, income and gender.

For this prediction we have downloaded a pre-existing car purchase prediction database from Kaggle and carried out the following steps:

- **Downloading Dataset from Kaggle and making certain changes to fit the problem statement:**

  Llink- https://www.kaggle.com/datasets/rakeshrau/social-network-ads/data
  We have changed the purchased column with the type of cars bought and if not bought then no.

- **Importing Libraries:**
  Necessary libraries are to be included for carrying out all the processes.

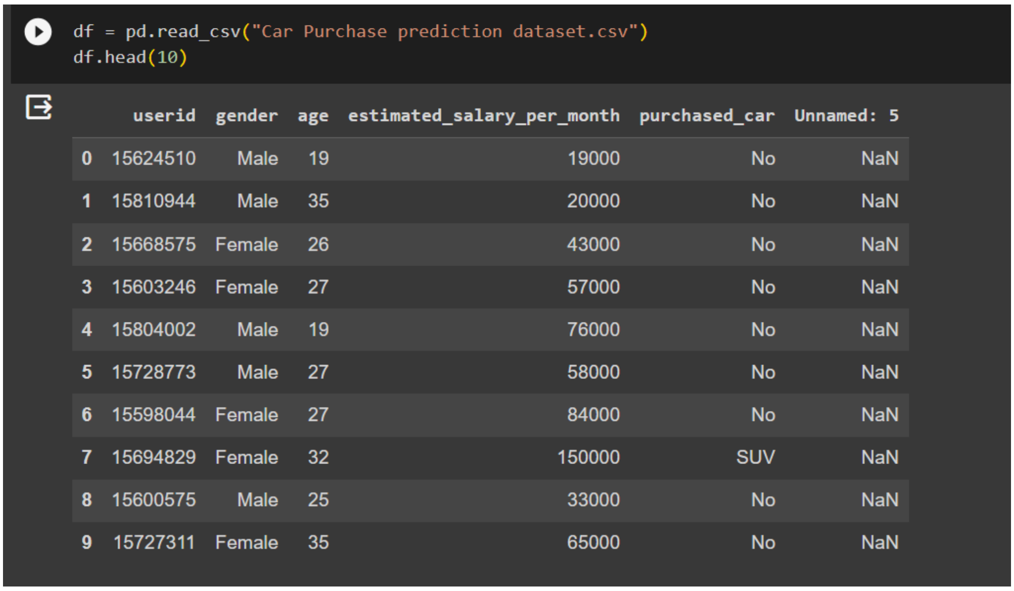


1. Importing Libraries

```
[ ] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.preprocessing import LabelEncoder
    from sklearn.preprocessing import MinMaxScaler
    from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LogisticRegression
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.svm import SVC
    from sklearn.naive_bayes import GaussianNB
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.tree import plot_tree, export_text
    import matplotlib.pyplot as plt
    from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
    import pickle
```

- **Read The Dataset:**

  Our dataset format is in .csv, We can read the dataset with the help of read_csv() of pandas. As a parameter we give the directory of csv file.

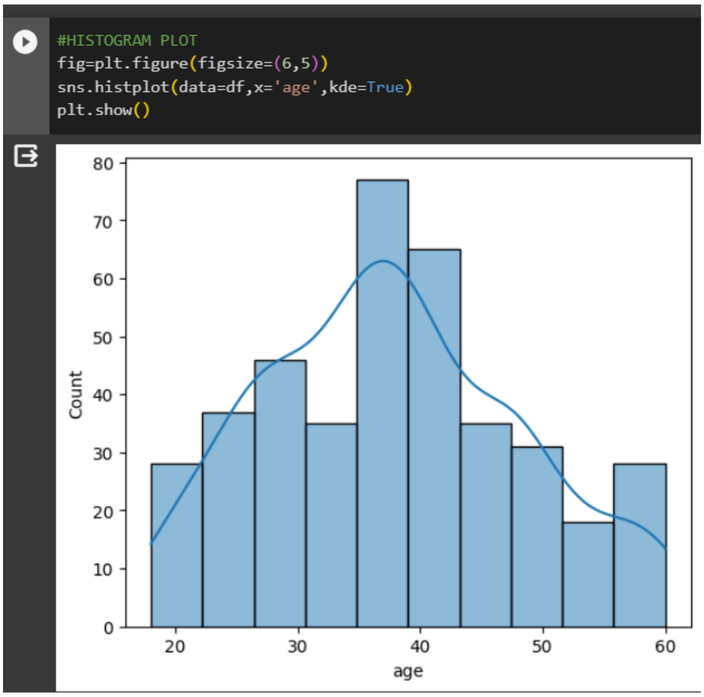


```
df = pd.read_csv("Car Purchase prediction dataset.csv")
df.head(10)
```

	userid	gender	age	estimated_salary_per_month	purchased_car	Unnamed: 5
0	15624510	Male	19	19000	No	NaN
1	15810944	Male	35	20000	No	NaN
2	15668575	Female	26	43000	No	NaN
3	15603246	Female	27	57000	No	NaN
4	15804002	Male	19	76000	No	NaN
5	15728773	Male	27	58000	No	NaN
6	15598044	Female	27	84000	No	NaN
7	15694829	Female	32	150000	SUV	NaN
8	15600575	Male	25	33000	No	NaN
9	15727311	Female	35	65000	No	NaN

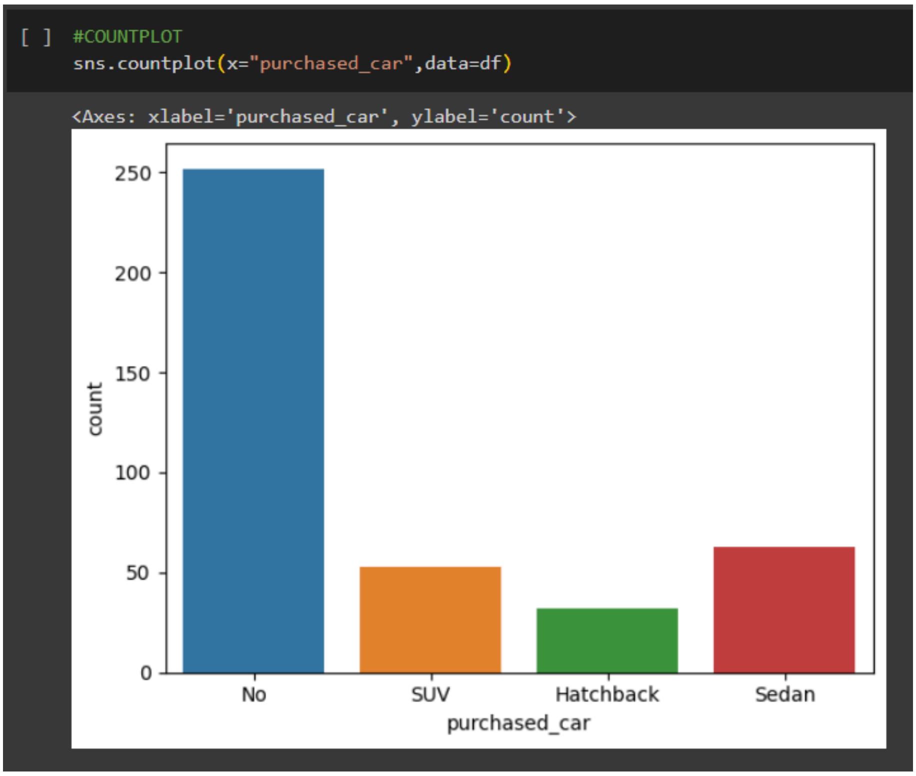
- **Univariate analysis:**

  In simple words, univariate analysis is to find the speciality of a salient feature.



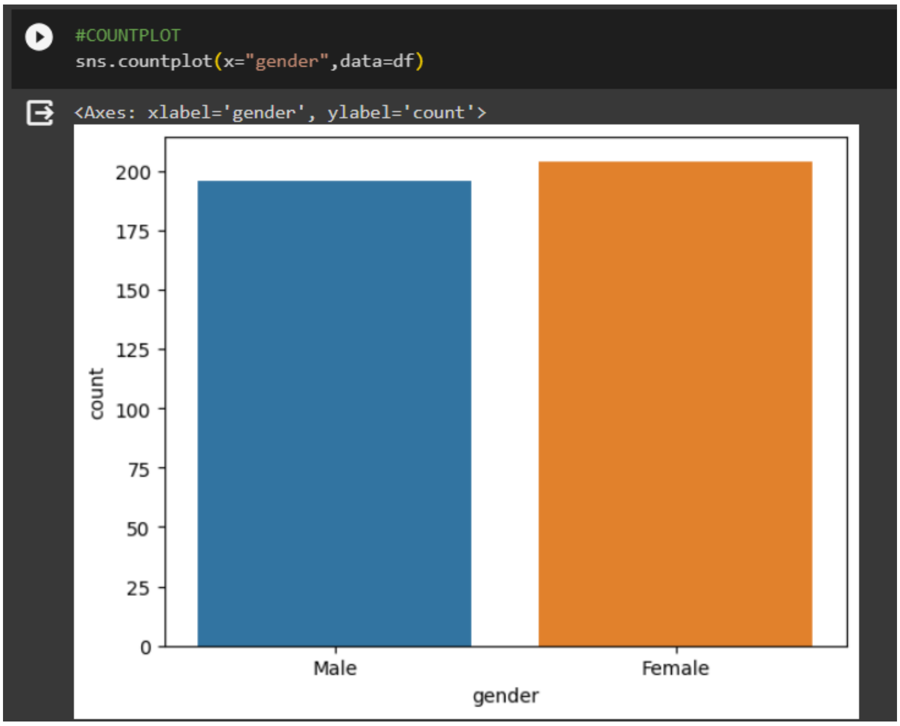
```
#HISTOGRAM PLOT
fig=plt.figure(figsize=(6,5))
sns.histplot(data=df,x='age',kde=True)
plt.show()
```

Inference: In this univariate analysis, a histogram has been plotted for the feature ‘**age**’ based on the user count of different ages using seaborn library present in python.



```
[ ] #COUNTPLOT
    sns.countplot(x="purchased_car",data=df)

    <Axes: xlabel='purchased_car', ylabel='count'>
```

Inference: In this univariate analysis, a countplot has been plotted for the feature ‘**purchased_car**’ based on the count of different types of purchased cars using seaborn library present in python.

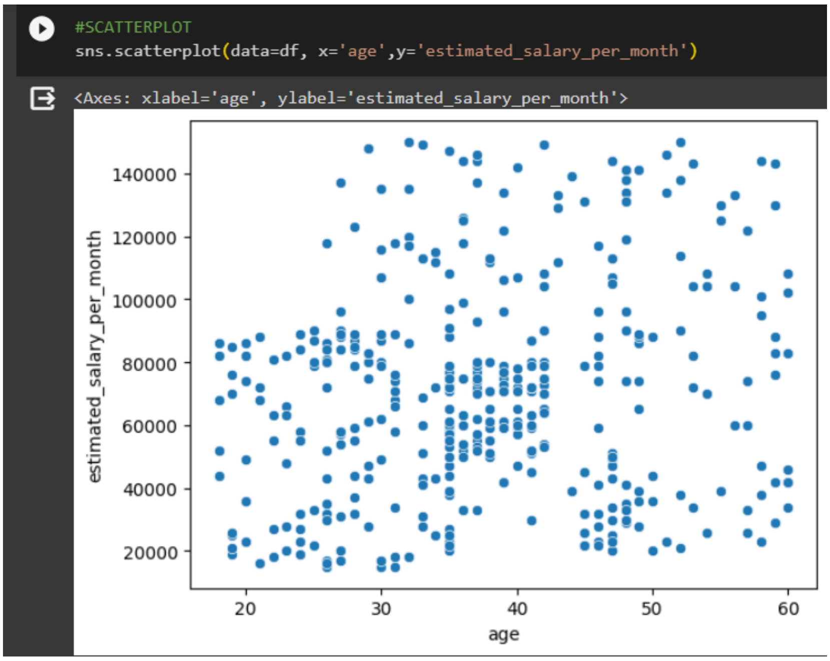


```
#COUNTPLOT
sns.countplot(x="gender",data=df)

<Axes: xlabel='gender', ylabel='count'>
```

Inference: In this univariate analysis, a countplot has been plotted for the feature ‘**gender**’ based on the user count of male and female genders using seaborn library present in python. Here females bought more cars than men.

- **Bivariate Analysis:**

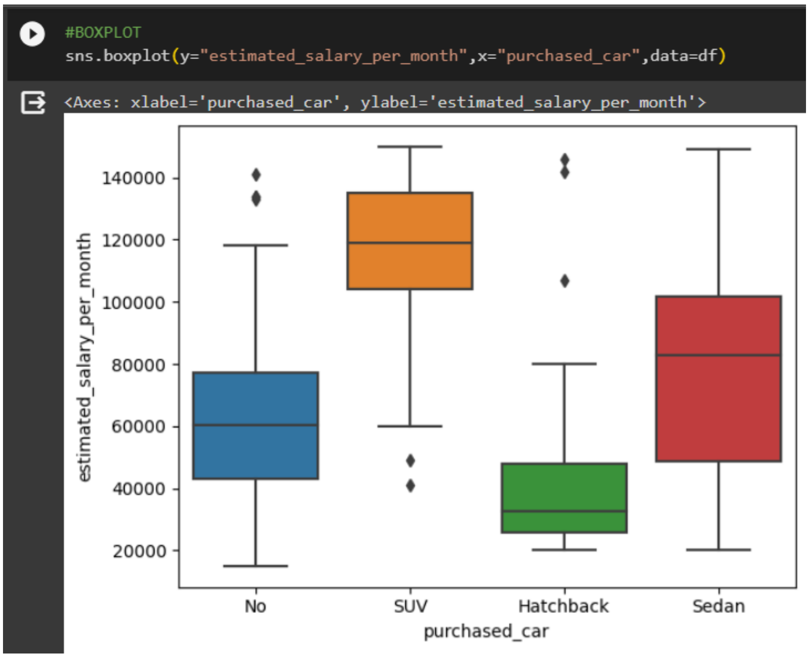
In simple words, bivariate analysis is to find the relation between two features.



```
#SCATTERPLOT
sns.scatterplot(data=df, x='age',y='estimated_salary_per_month')
```

```
<Axes: xlabel='age', ylabel='estimated_salary_per_month'>
```

Inference: In this bivariate analysis, a scatterplot has been plotted between the features '**age**' and '**estimated_salary_per_month**' based on estimated salary of people per month based on their ages using seaborn library present python.



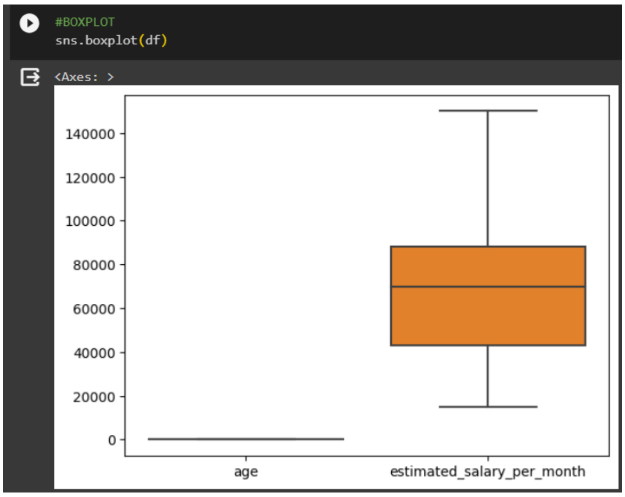
```
#BOXPLOT
sns.boxplot(y="estimated_salary_per_month",x="purchased_car",data=df)
```

```
<Axes: xlabel='purchased_car', ylabel='estimated_salary_per_month'>
```

Inference: In this bivariate analysis, a boxplot has been plotted between the features '**estimated_salary_per_month**' and '**purchased_car**' based on different types of cars bought as per estimated salary of a user per month using seaborn library present in python. Here we can notice that people with minimal salaries tend to go for the hatchback cars whereas people with medium level income up to 1.1 lakhs per month go for the sedans and people with high and stable income go for the SUV cars.

- **Multivariate analysis:**

  In simple words, multivariate analysis is to find the relation between multiple features.

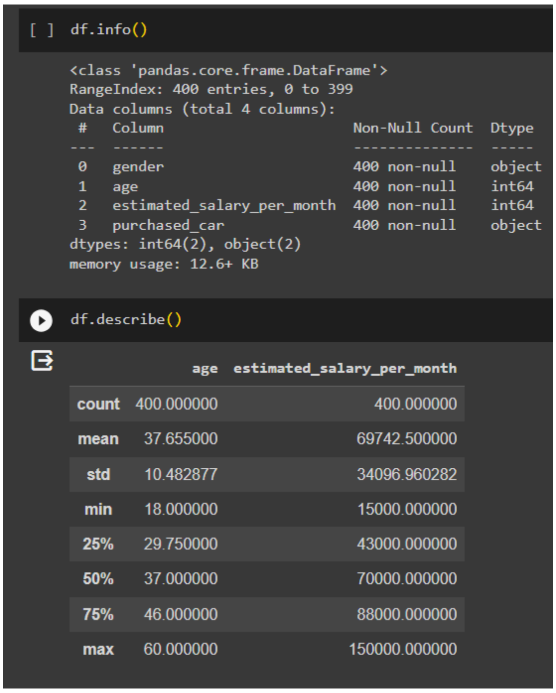


```
#BOXPLOT
sns.boxplot(df)
```

```
<Axes: >
```

Inference: In this multivariate analysis, a boxplot has been plotted among the features '**age**' and '**estimated_salary_per_month**' along with its count using seaborn library present python.

- **Descriptive analysis:**

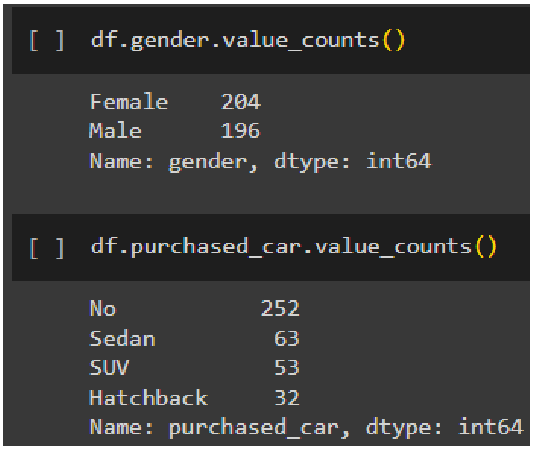


```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 4 columns):
 #   Column                      Non-Null Count  Dtype 
---  ------                      --------------  ----- 
 0   gender                      400 non-null    object
 1   age                         400 non-null    int64 
 2   estimated_salary_per_month  400 non-null    int64 
 3   purchased_car               400 non-null    object
dtypes: int64(2), object(2)
memory usage: 12.6+ KB
```

```
df.describe()
```

	age	estimated_salary_per_month
count	400.000000	400.000000
mean	37.655000	69742.500000
std	10.482877	34096.960282
min	18.000000	15000.000000
25%	29.750000	43000.000000
50%	37.000000	70000.000000
75%	46.000000	88000.000000
max	60.000000	150000.000000

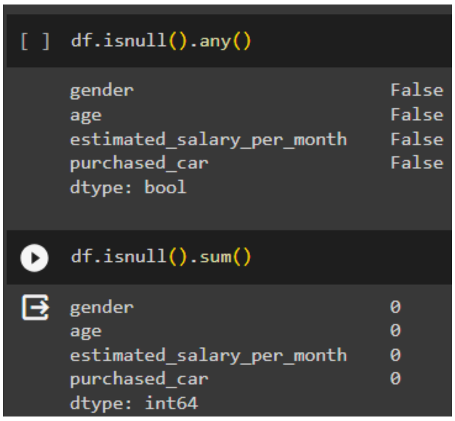


```
[ ]  df.gender.value_counts()

     Female    204
     Male      196
     Name: gender, dtype: int64

[ ]  df.purchased_car.value_counts()

     No           252
     Sedan         63
     SUV           53
     Hatchback     32
     Name: purchased_car, dtype: int64
```

- **Data Pre-processing:**

1. Handling Missing Values:

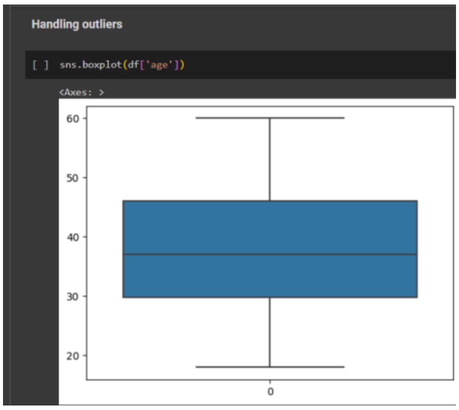
For checking the null values, df.isnull() function is used. To sum those null values we use sum() function to it. From the below image we found that there are no null values present in our dataset. So, we can skip handling of missing values step.



```
[ ]  df.isnull().any()

     gender                        False
     age                           False
     estimated_salary_per_month    False
     purchased_car                 False
     dtype: bool

     df.isnull().sum()

     gender                        0
     age                           0
     estimated_salary_per_month    0
     purchased_car                 0
     dtype: int64
```

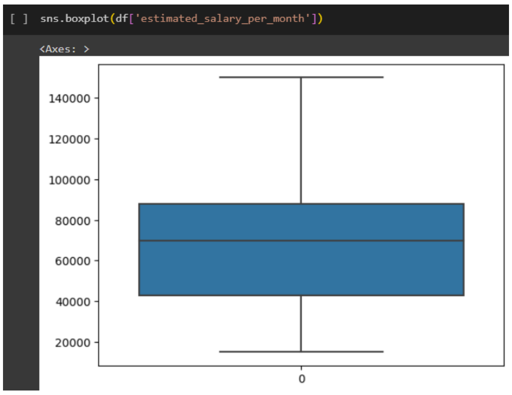
2. Handling outliers:

With the help of boxplot, outliers are visualized.



Handling outliers

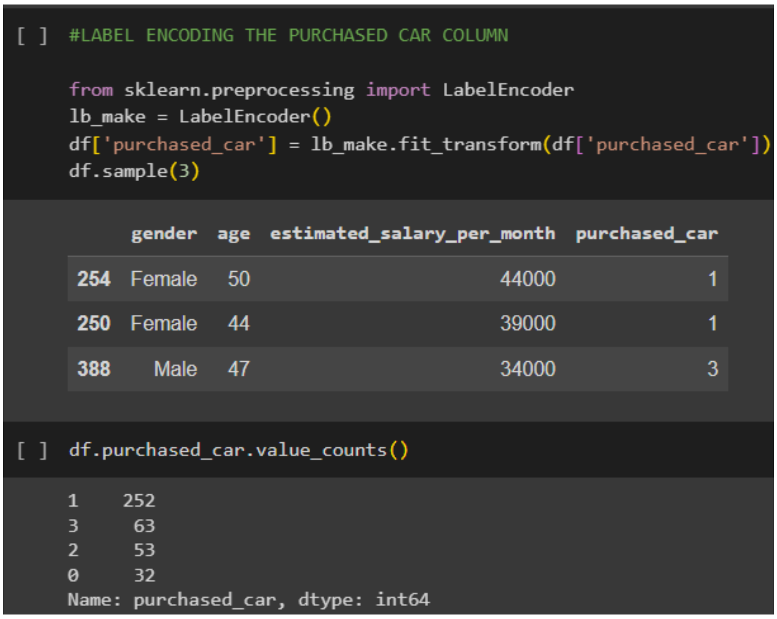
```
[ ]  sns.boxplot(df['age'])

<Axes: >
```



```
[ ] sns.boxplot(df['estimated_salary_per_month'])

<Axes: >
```

3. Label Encoding:

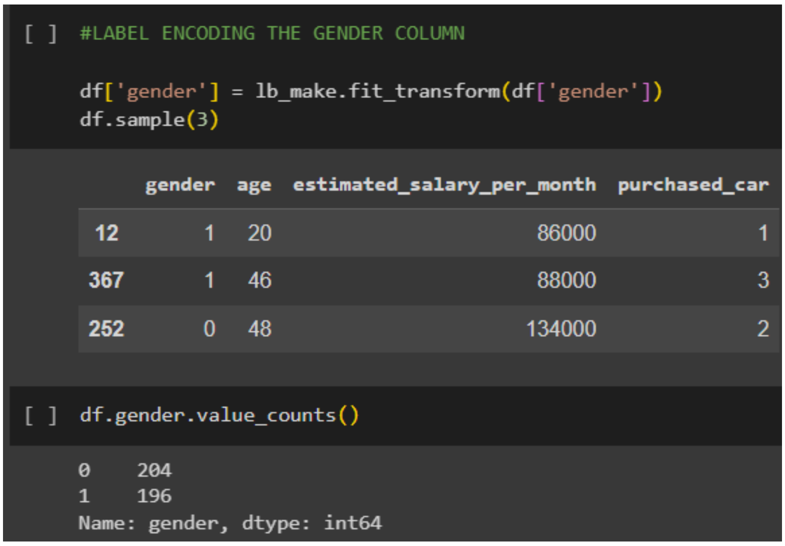
Categorical data columns are encoded for better data training and testing.



```
[ ] #LABEL ENCODING THE PURCHASED CAR COLUMN

    from sklearn.preprocessing import LabelEncoder
    lb_make = LabelEncoder()
    df['purchased_car'] = lb_make.fit_transform(df['purchased_car'])
    df.sample(3)
```

	gender	age	estimated_salary_per_month	purchased_car
254	Female	50	44000	1
250	Female	44	39000	1
388	Male	47	34000	3

```
[ ] df.purchased_car.value_counts()

    1    252
    3     63
    2     53
    0     32
    Name: purchased_car, dtype: int64
```



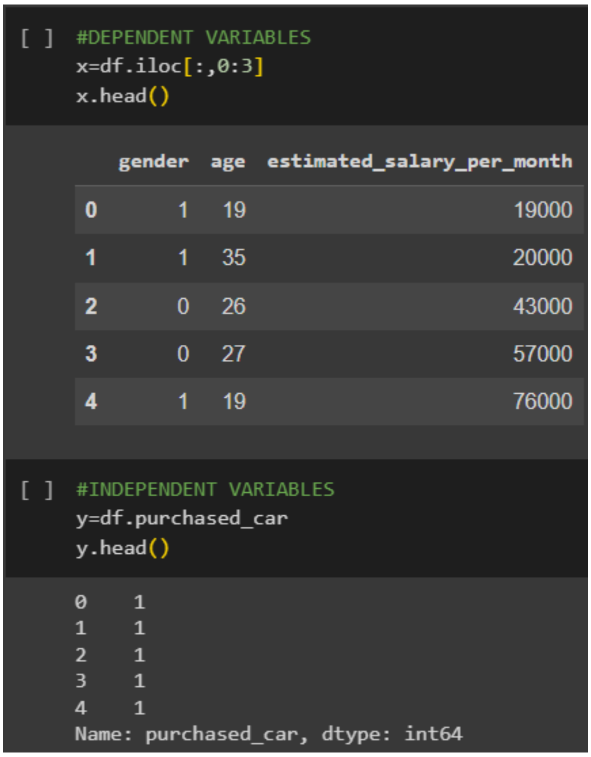
```
[ ] #LABEL ENCODING THE GENDER COLUMN

    df['gender'] = lb_make.fit_transform(df['gender'])
    df.sample(3)
```

	gender	age	estimated_salary_per_month	purchased_car
12	1	20	86000	1
367	1	46	88000	3
252	0	48	134000	2

```
[ ] df.gender.value_counts()

    0    204
    1    196
    Name: gender, dtype: int64
```

4. Splitting independent and dependent variables:

The Dataset is split into train and test sets. First the dataset is split into x and then to y.



```
[ ] #DEPENDENT VARIABLES
    x=df.iloc[:,0:3]
    x.head()
```

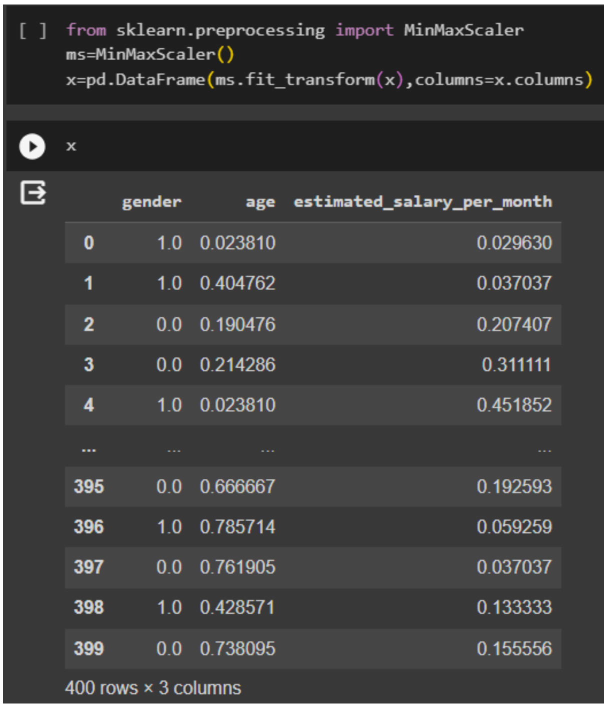
	gender	age	estimated_salary_per_month
0	1	19	19000
1	1	35	20000
2	0	26	43000
3	0	27	57000
4	1	19	76000

```
[ ] #INDEPENDENT VARIABLES
    y=df.purchased_car
    y.head()
```

```
0    1
1    1
2    1
3    1
4    1
Name: purchased_car, dtype: int64
```

5. Feature Scaling:

This is used to scale the data for better data training and testing.

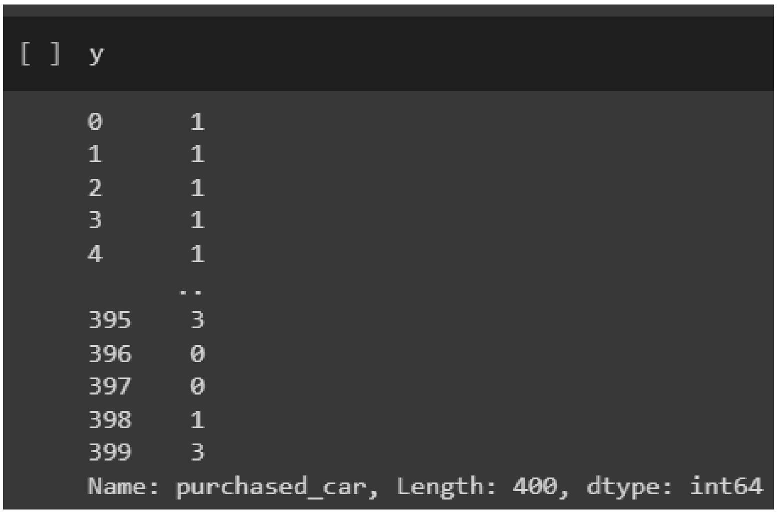


```
[ ] from sklearn.preprocessing import MinMaxScaler
    ms=MinMaxScaler()
    x=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
```

```
x
```

	gender	age	estimated_salary_per_month
0	1.0	0.023810	0.029630
1	1.0	0.404762	0.037037
2	0.0	0.190476	0.207407
3	0.0	0.214286	0.311111
4	1.0	0.023810	0.451852
...	...	...	...
395	0.0	0.666667	0.192593
396	1.0	0.785714	0.059259
397	0.0	0.761905	0.037037
398	1.0	0.428571	0.133333
399	0.0	0.738095	0.155556

400 rows × 3 columns

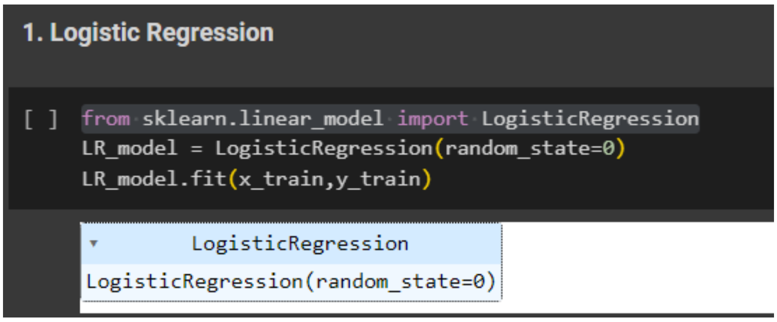


```
[ ] y

    0      1
    1      1
    2      1
    3      1
    4      1
          ..
    395    3
    396    0
    397    0
    398    1
    399    3
    Name: purchased_car, Length: 400, dtype: int64
```

- **Model building:**

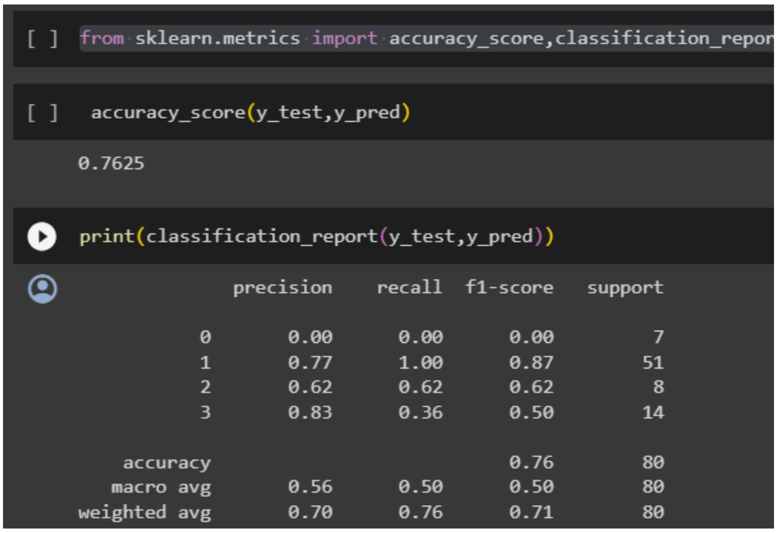
1. Logistic Regression:

Logistic regression is a statistical method used for binary classification, predicting the probability of an event occurring (such as purchase or non-purchase) based on input features, and it employs the logistic function to model the relationship between the features and the binary outcome.



1. Logistic Regression

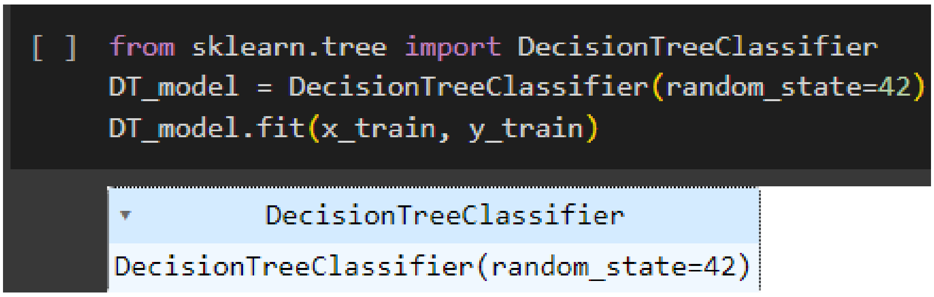
```
[ ] from sklearn.linear_model import LogisticRegression
    LR_model = LogisticRegression(random_state=0)
    LR_model.fit(x_train,y_train)

    ▾       LogisticRegression
    LogisticRegression(random_state=0)
```



```
[ ] from sklearn.metrics import accuracy_score,classification_repor

[ ]  accuracy_score(y_test,y_pred)

    0.7625

    print(classification_report(y_test,y_pred))

                  precision    recall  f1-score   support

               0       0.00      0.00      0.00         7
               1       0.77      1.00      0.87        51
               2       0.62      0.62      0.62         8
               3       0.83      0.36      0.50        14

        accuracy                           0.76        80
       macro avg       0.56      0.50      0.50        80
    weighted avg       0.70      0.76      0.71        80
```

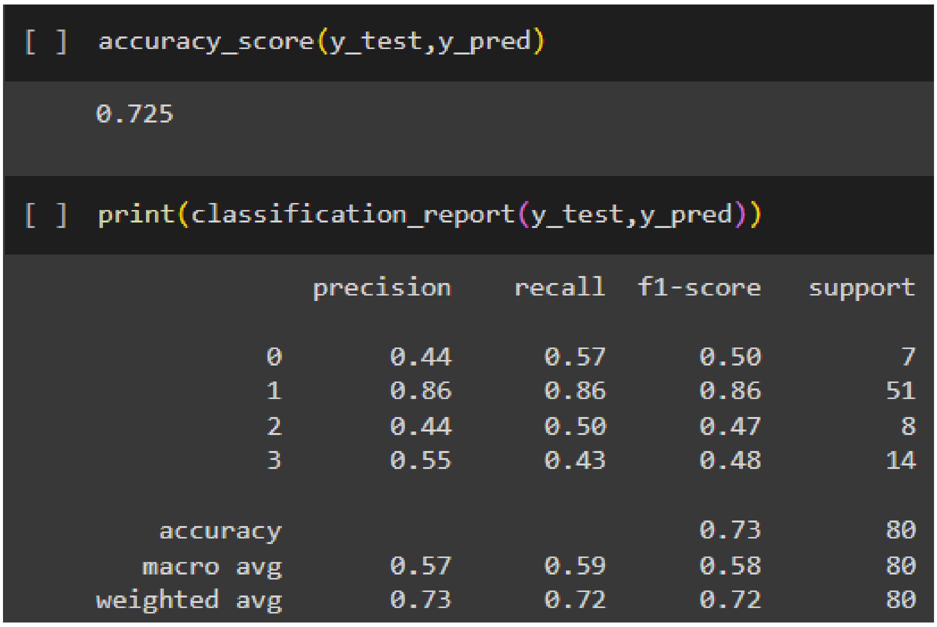
2. Decision Tree:

Decision tree is a machine learning algorithm that uses a tree-like model of decisions to predict outcomes, breaking down a dataset into hierarchical structures based on input features.



```
[ ]  from sklearn.tree import DecisionTreeClassifier
     DT_model = DecisionTreeClassifier(random_state=42)
     DT_model.fit(x_train, y_train)
```

```
▾        DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```



```
[ ]  accuracy_score(y_test,y_pred)
```

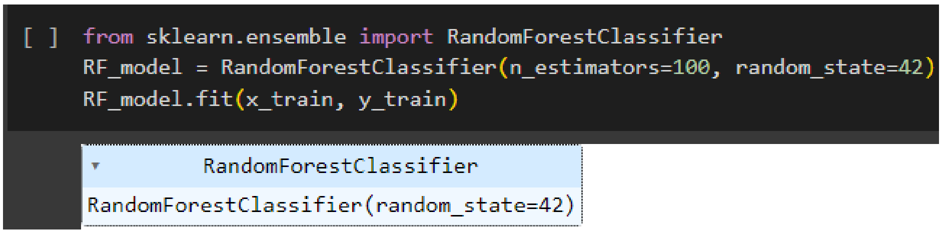
```
0.725
```

```
[ ]  print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.44      0.57      0.50         7
           1       0.86      0.86      0.86        51
           2       0.44      0.50      0.47         8
           3       0.55      0.43      0.48        14

    accuracy                           0.73        80
   macro avg       0.57      0.59      0.58        80
weighted avg       0.73      0.72      0.72        80
```

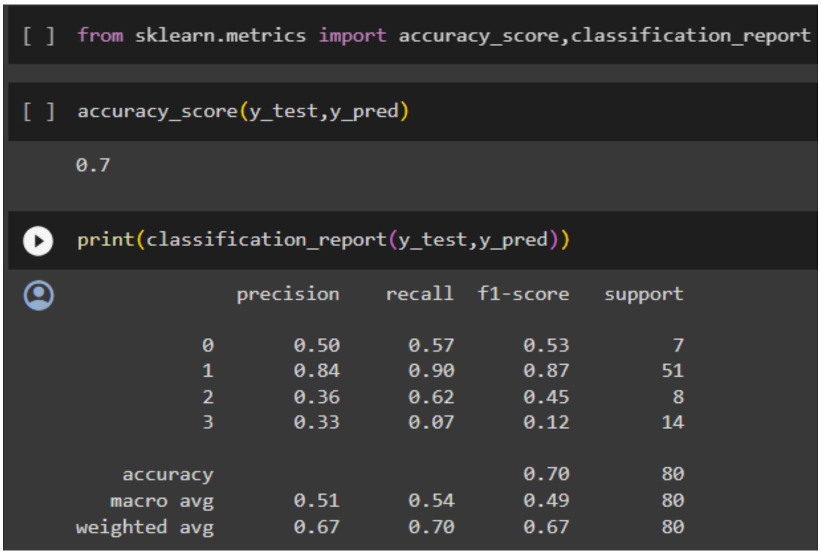
3. Random Forest:

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification tasks or mean prediction for regression tasks.



```
[ ]  from sklearn.ensemble import RandomForestClassifier
     RF_model = RandomForestClassifier(n_estimators=100, random_state=42)
     RF_model.fit(x_train, y_train)
```

```
▾        RandomForestClassifier
RandomForestClassifier(random_state=42)
```



```
from sklearn.metrics import accuracy_score,classification_report
```

```
accuracy_score(y_test,y_pred)
```

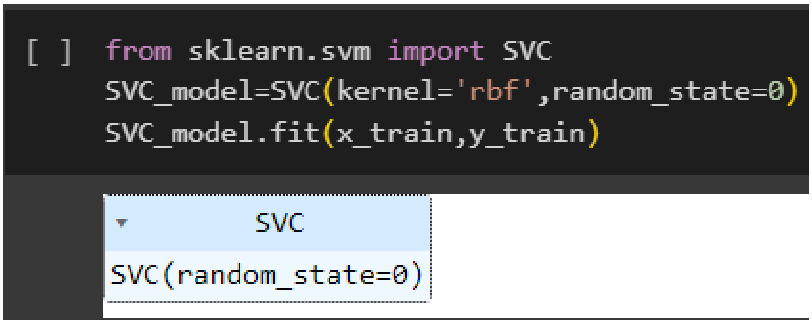
```
0.7
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.50      0.57      0.53         7
           1       0.84      0.90      0.87        51
           2       0.36      0.62      0.45         8
           3       0.33      0.07      0.12        14

    accuracy                           0.70        80
   macro avg       0.51      0.54      0.49        80
weighted avg       0.67      0.70      0.67        80
```

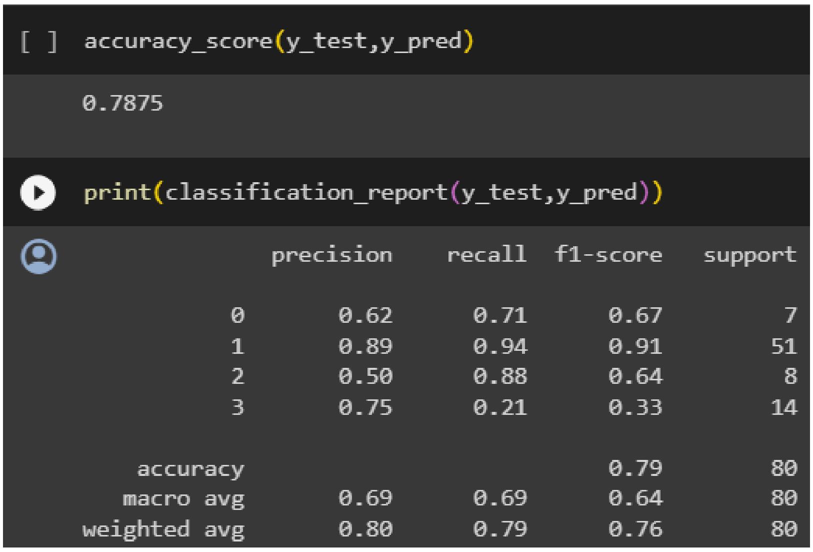
4. Support Vector Machine:

Support Vector Classification (SVC) is a supervised machine learning algorithm that finds a hyperplane in an N-dimensional space, which distinctly classifies data into different categories.



```
from sklearn.svm import SVC
SVC_model=SVC(kernel='rbf',random_state=0)
SVC_model.fit(x_train,y_train)
```

```
SVC
SVC(random_state=0)
```



```
accuracy_score(y_test,y_pred)
```

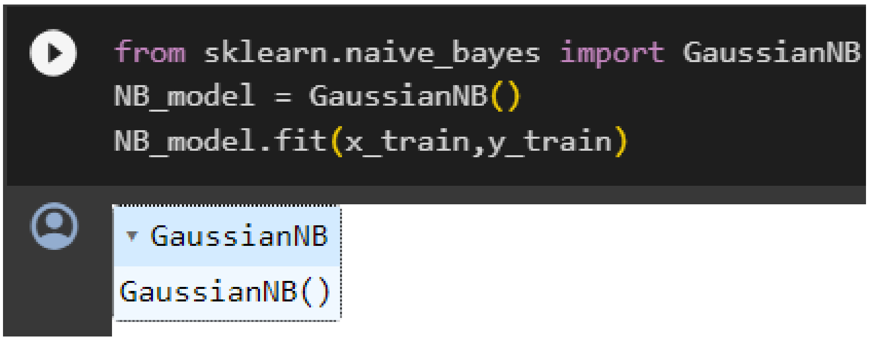
```
0.7875
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.62      0.71      0.67         7
           1       0.89      0.94      0.91        51
           2       0.50      0.88      0.64         8
           3       0.75      0.21      0.33        14

    accuracy                           0.79        80
   macro avg       0.69      0.69      0.64        80
weighted avg       0.80      0.79      0.76        80
```

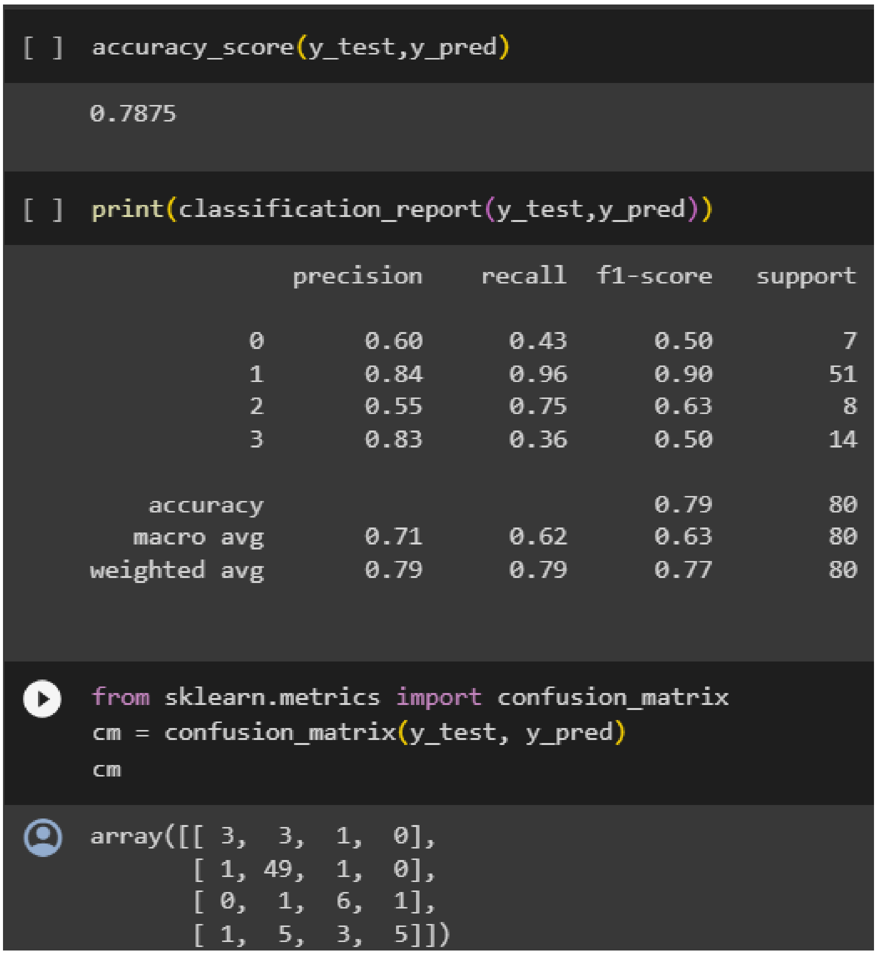
5. Naive Bayes:

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem, assuming independence among features, often used for text classification and spam filtering.



```
from sklearn.naive_bayes import GaussianNB
NB_model = GaussianNB()
NB_model.fit(x_train,y_train)
```

```
▾ GaussianNB
GaussianNB()
```



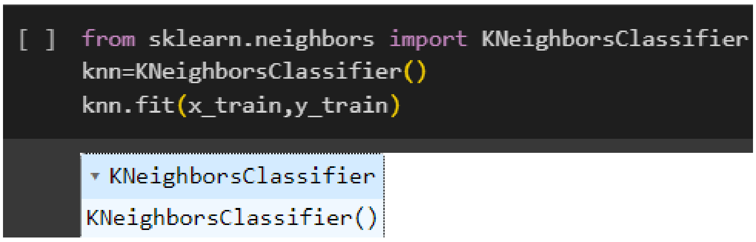
```
[ ] accuracy_score(y_test,y_pred)

    0.7875

[ ] print(classification_report(y_test,y_pred))

                  precision    recall  f1-score   support

               0       0.60      0.43      0.50         7
               1       0.84      0.96      0.90        51
               2       0.55      0.75      0.63         8
               3       0.83      0.36      0.50        14

        accuracy                           0.79        80
       macro avg       0.71      0.62      0.63        80
    weighted avg       0.79      0.79      0.77        80
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
cm
```

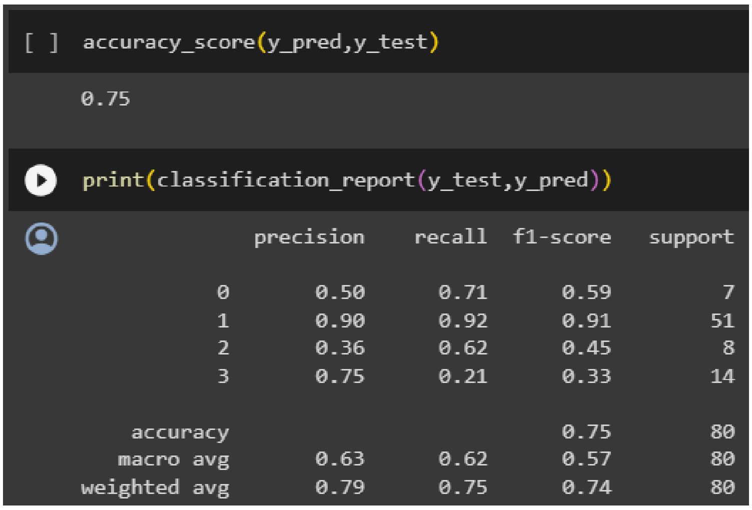
```
array([[ 3,  3,  1,  0],
       [ 1, 49,  1,  0],
       [ 0,  1,  6,  1],
       [ 1,  5,  3,  5]])
```

6. KNN

K-Nearest Neighbors (KNN) is a simple, instance-based learning algorithm that classifies a new data point based on the majority class of its k nearest neighbors in the feature space.



```
[ ] from sklearn.neighbors import KNeighborsClassifier
    knn=KNeighborsClassifier()
    knn.fit(x_train,y_train)

    ▾ KNeighborsClassifier
    KNeighborsClassifier()
```



```
[ ] accuracy_score(y_pred,y_test)

    0.75

    print(classification_report(y_test,y_pred))

                  precision    recall  f1-score   support

               0       0.50      0.71      0.59         7
               1       0.90      0.92      0.91        51
               2       0.36      0.62      0.45         8
               3       0.75      0.21      0.33        14

        accuracy                           0.75        80
       macro avg       0.63      0.62      0.57        80
    weighted avg       0.79      0.75      0.74        80
```

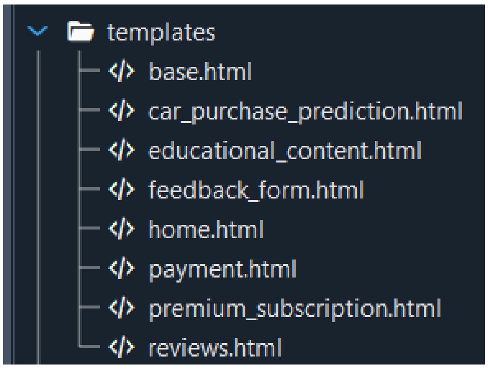
- **Application Building**

  - **Building HTML pages**

For this project create eight HTML files namely:

- home.html
- base.html
- car_purchase_prediction.html
- educational_content.html
- feedback_form.html
- payment.html
- reviews.html
- premium_subscription.html

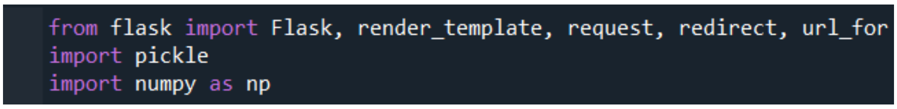
and saved in Templates folder.



```
templates
├─ base.html
├─ car_purchase_prediction.html
├─ educational_content.html
├─ feedback_form.html
├─ home.html
├─ payment.html
├─ premium_subscription.html
└─ reviews.html
```

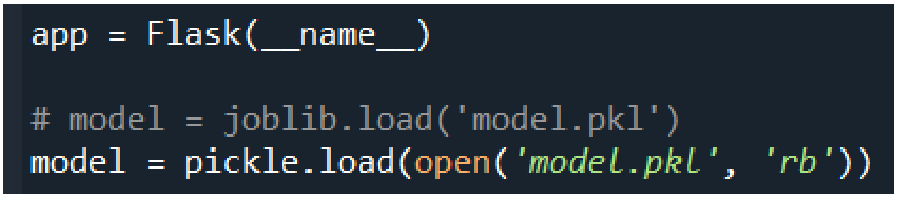
- **Build python code:**

Import the libraries:



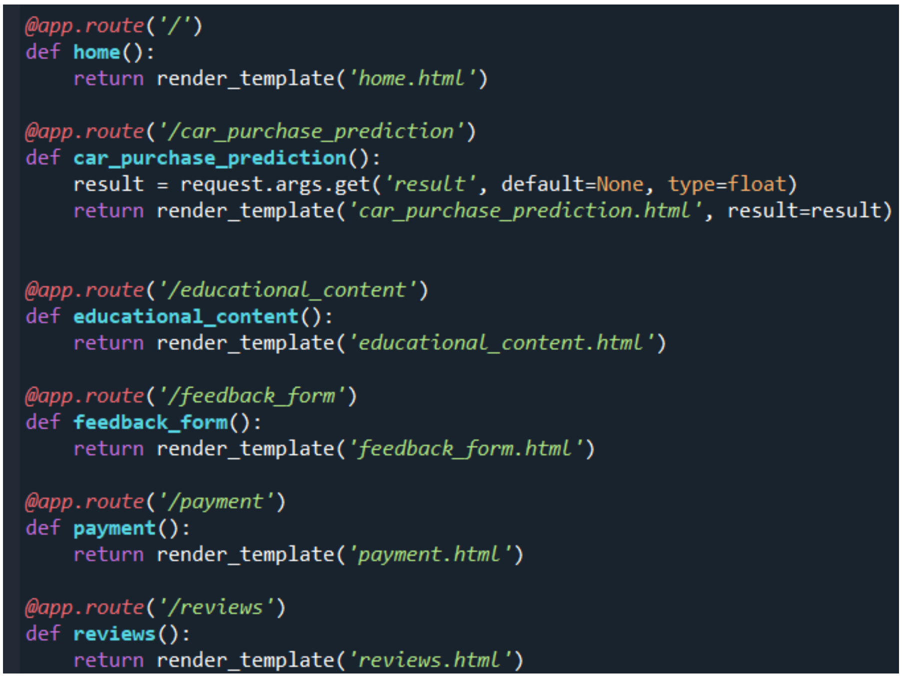
```python
from flask import Flask, render_template, request, redirect, url_for
import pickle
import numpy as np
```

Load the saved model. Importing flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (__name__) as argument.

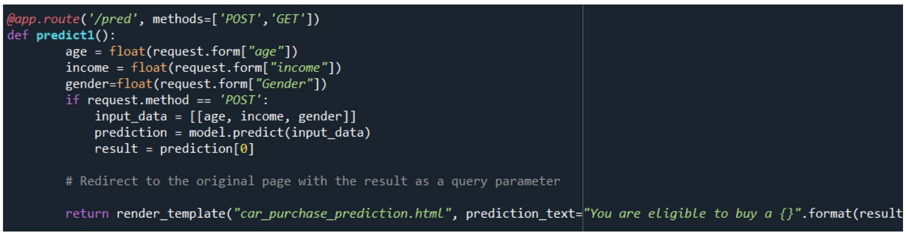


```python
app = Flask(__name__)

# model = joblib.load('model.pkl')
model = pickle.load(open('model.pkl', 'rb'))
```

Render HTML page:

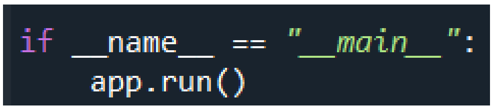


```python
@app.route('/')
def home():
    return render_template('home.html')

@app.route('/car_purchase_prediction')
def car_purchase_prediction():
    result = request.args.get('result', default=None, type=float)
    return render_template('car_purchase_prediction.html', result=result)


@app.route('/educational_content')
def educational_content():
    return render_template('educational_content.html')

@app.route('/feedback_form')
def feedback_form():
    return render_template('feedback_form.html')

@app.route('/payment')
def payment():
    return render_template('payment.html')

@app.route('/reviews')
def reviews():
    return render_template('reviews.html')
```



```python
@app.route('/pred', methods=['POST','GET'])
def predict1():
        age = float(request.form["age"])
        income = float(request.form["income"])
        gender=float(request.form["Gender"])
        if request.method == 'POST':
            input_data = [[age, income, gender]]
            prediction = model.predict(input_data)
            result = prediction[0]

        # Redirect to the original page with the result as a query parameter

        return render_template("car_purchase_prediction.html", prediction_text="You are eligible to buy a {}".format(result
```

Main Function:

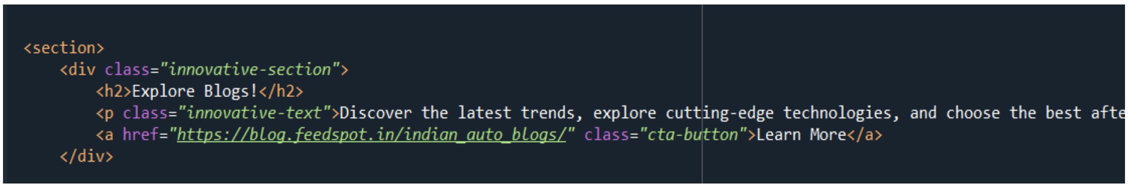


```
if __name__ == "__main__":
    app.run()
```

7.2 Feature 2 (Educational content)

- Blogs:

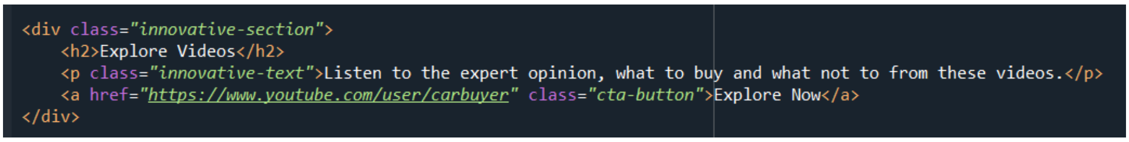
  From here you will be redirected to a site where you'll get multiple cars related blogs to read and acquire knowledge from.



```
<section>
    <div class="innovative-section">
        <h2>Explore Blogs!</h2>
        <p class="innovative-text">Discover the latest trends, explore cutting-edge technologies, and choose the best afte
        <a href="https://blog.feedspot.in/indian_auto_blogs/" class="cta-button">Learn More</a>
    </div>
```

- Videos:

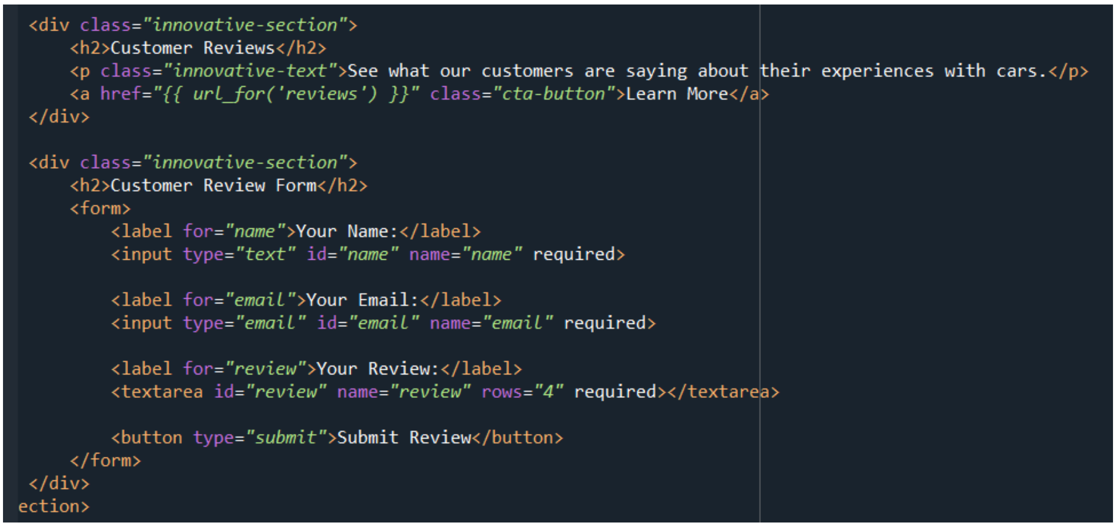
  You'll be redirected to various YouTube videos page, where you'll get visual explanation of various topics related to cars.



```
<div class="innovative-section">
    <h2>Explore Videos</h2>
    <p class="innovative-text">Listen to the expert opinion, what to buy and what not to from these videos.</p>
    <a href="https://www.youtube.com/user/carbuyer" class="cta-button">Explore Now</a>
</div>
```

- Reviews:

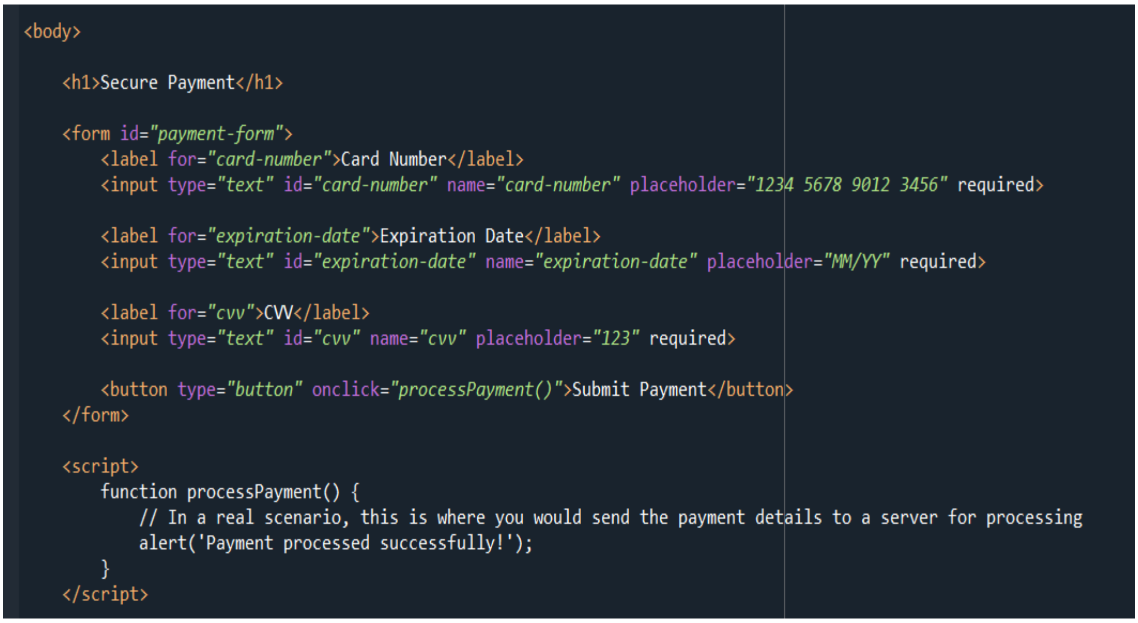
  You can see various customer reviews on various cars and you can add your own review too.



```
 <div class="innovative-section">
     <h2>Customer Reviews</h2>
     <p class="innovative-text">See what our customers are saying about their experiences with cars.</p>
     <a href="{{ url_for('reviews') }}" class="cta-button">Learn More</a>
 </div>

 <div class="innovative-section">
     <h2>Customer Review Form</h2>
     <form>
         <label for="name">Your Name:</label>
         <input type="text" id="name" name="name" required>

         <label for="email">Your Email:</label>
         <input type="email" id="email" name="email" required>

         <label for="review">Your Review:</label>
         <textarea id="review" name="review" rows="4" required></textarea>

         <button type="submit">Submit Review</button>
     </form>
 </div>
ection>
```

7.3 Feature 3 (Premium Subscription)

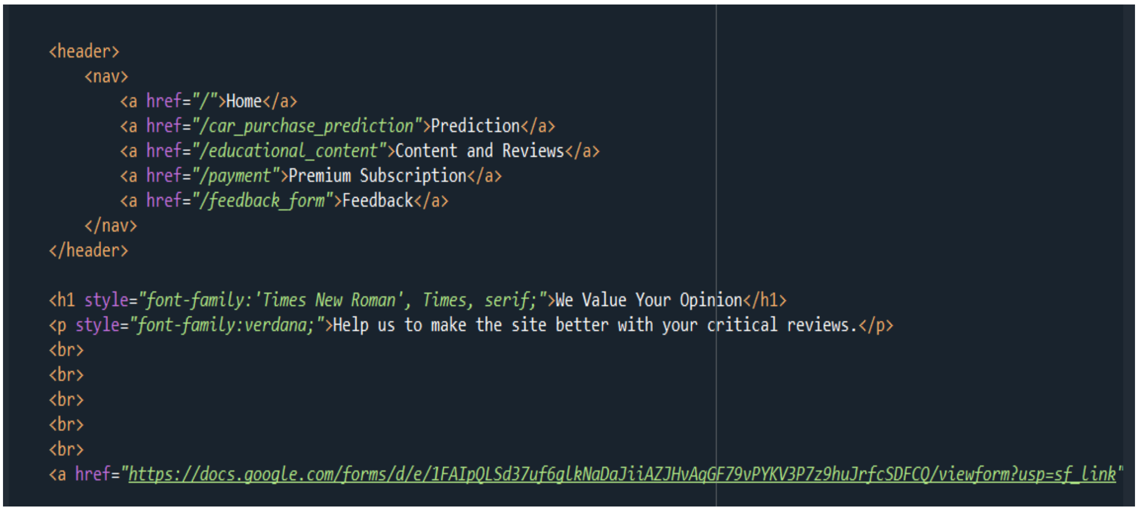
You can gain premium subscription to get personalized analysis.



```
<body>

    <h1>Secure Payment</h1>

    <form id="payment-form">
        <label for="card-number">Card Number</label>
        <input type="text" id="card-number" name="card-number" placeholder="1234 5678 9012 3456" required>

        <label for="expiration-date">Expiration Date</label>
        <input type="text" id="expiration-date" name="expiration-date" placeholder="MM/YY" required>

        <label for="cvv">CVV</label>
        <input type="text" id="cvv" name="cvv" placeholder="123" required>

        <button type="button" onclick="processPayment()">Submit Payment</button>
    </form>

    <script>
        function processPayment() {
            // In a real scenario, this is where you would send the payment details to a server for processing
            alert('Payment processed successfully!');
        }
    </script>
```

7.4 Feature 4(Feedback)

We Have added a feedback form for the customers to give their critical review on how we can make our website better.



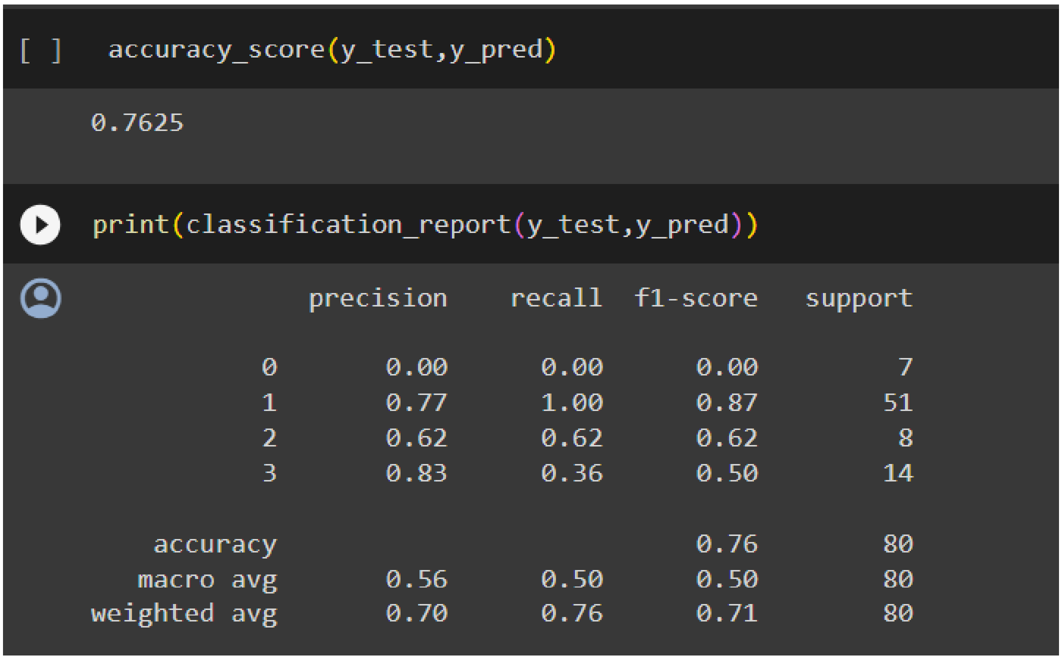
```
<header>
    <nav>
        <a href="/">Home</a>
        <a href="/car_purchase_prediction">Prediction</a>
        <a href="/educational_content">Content and Reviews</a>
        <a href="/payment">Premium Subscription</a>
        <a href="/feedback_form">Feedback</a>
    </nav>
</header>

<h1 style="font-family:'Times New Roman', Times, serif;">We Value Your Opinion</h1>
<p style="font-family:verdana;">Help us to make the site better with your critical reviews.</p>
<br>
<br>
<br>
<br>
<br>
<a href="https://docs.google.com/forms/d/e/1FAIpQLSd37uf6glkNaDaJiiAZJHvAqGF79vPYKV3P7z9huJrfcSDFCQ/viewform?usp=sf_link"
```

8. PERFORMANCE METRICES

8.1 Performance Metrics

In total we have used 6 models to compare and find the best model that gives the highest accuracy. The models which we have used and their accuracies are:

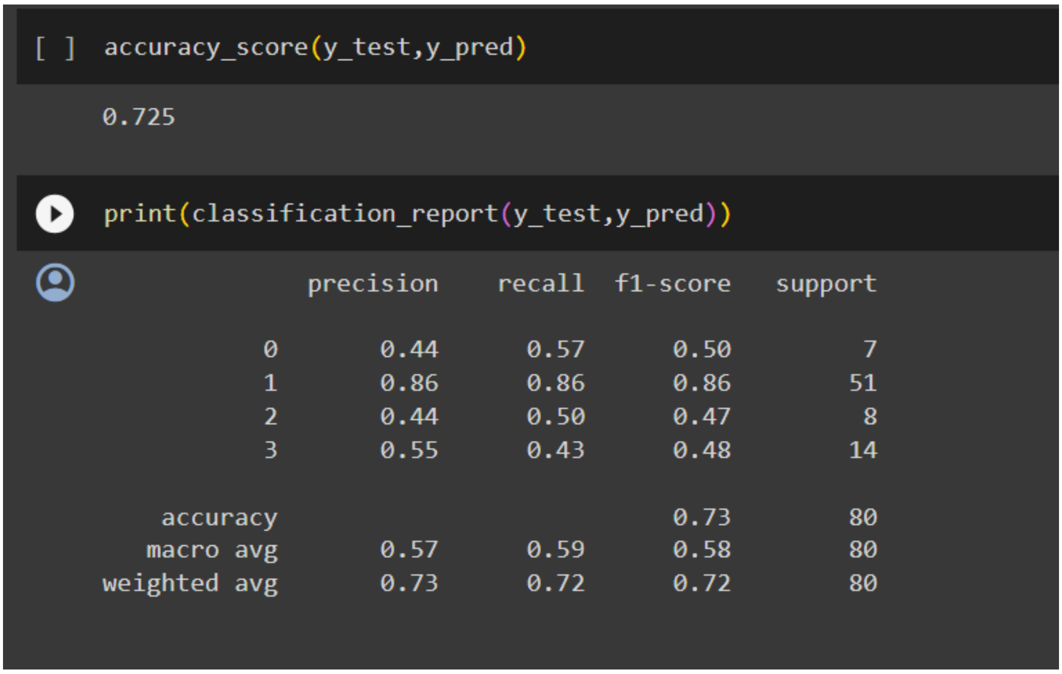
1. **Logistic Regression:**
   Accuracy: 76.25%



```
[ ]  accuracy_score(y_test,y_pred)

     0.7625
```

```
print(classification_report(y_test,y_pred))

              precision    recall  f1-score   support

           0       0.00      0.00      0.00         7
           1       0.77      1.00      0.87        51
           2       0.62      0.62      0.62         8
           3       0.83      0.36      0.50        14

    accuracy                           0.76        80
   macro avg       0.56      0.50      0.50        80
weighted avg       0.70      0.76      0.71        80
```

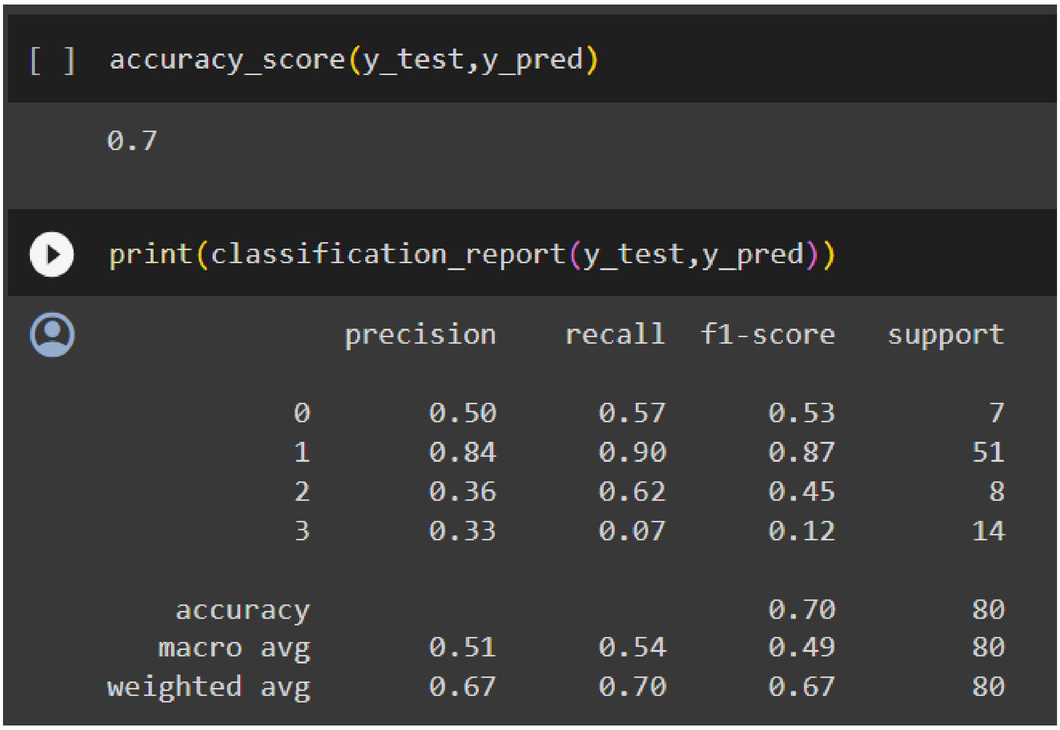
2. **Decision Tree:**
   Accuracy: 72.5%



```
[ ]  accuracy_score(y_test,y_pred)

     0.725
```

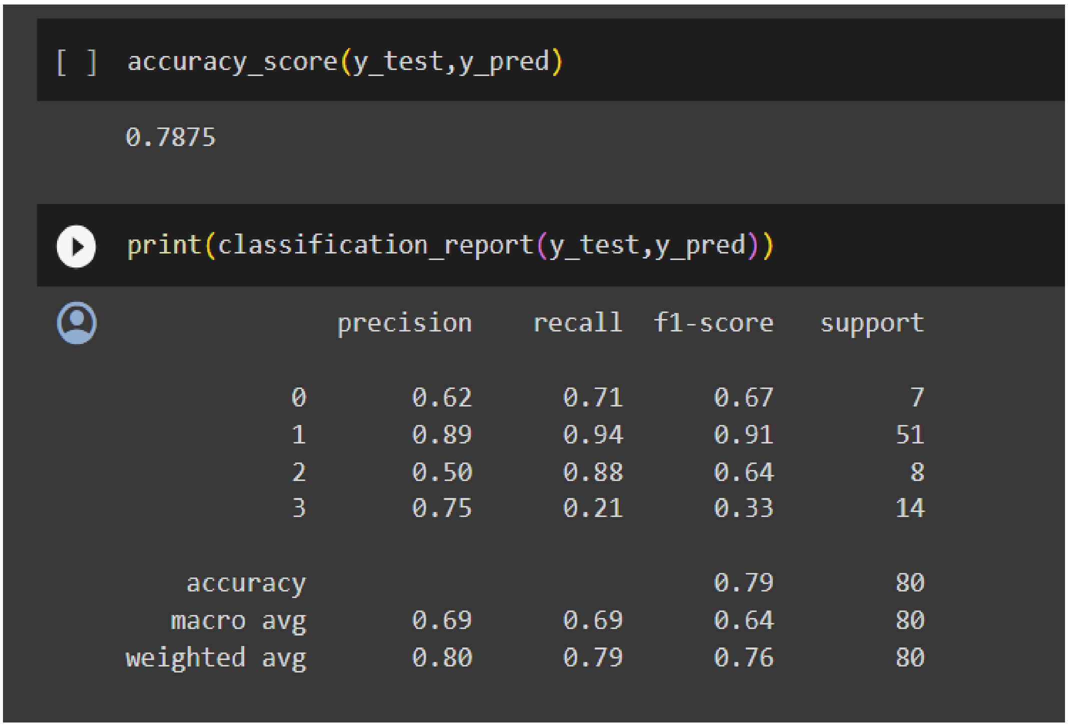
```
print(classification_report(y_test,y_pred))

              precision    recall  f1-score   support

           0       0.44      0.57      0.50         7
           1       0.86      0.86      0.86        51
           2       0.44      0.50      0.47         8
           3       0.55      0.43      0.48        14

    accuracy                           0.73        80
   macro avg       0.57      0.59      0.58        80
weighted avg       0.73      0.72      0.72        80
```

3. **Random Forest:**
   Accuracy: 70%



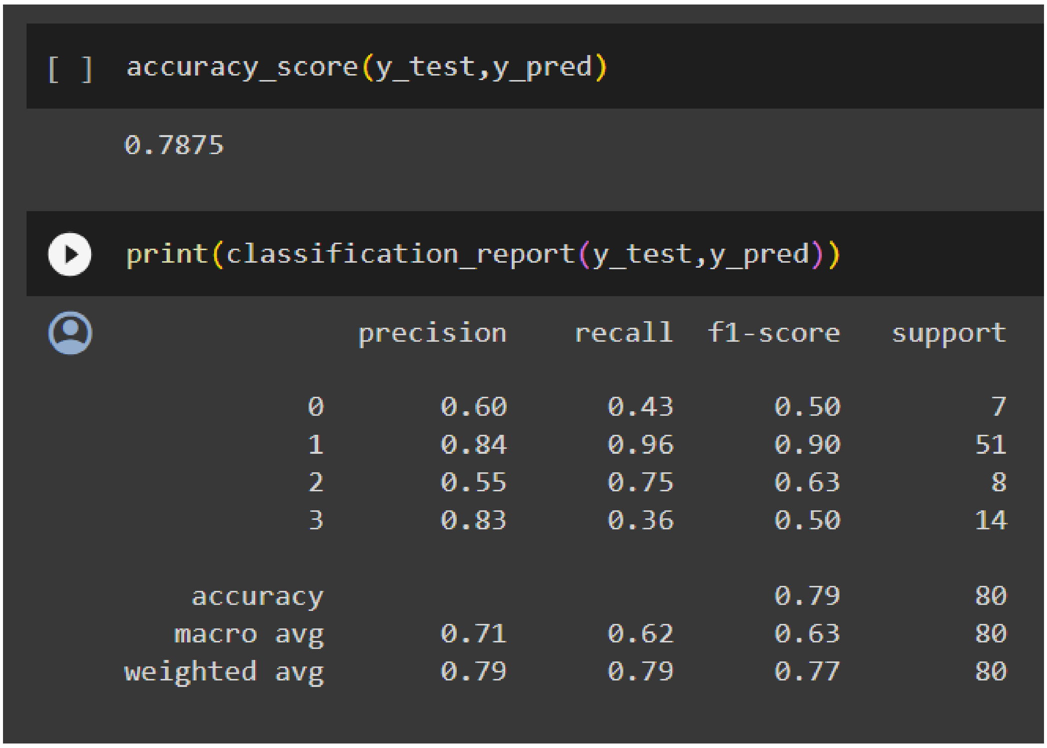
```
[ ]  accuracy_score(y_test,y_pred)

     0.7

     print(classification_report(y_test,y_pred))

                   precision    recall  f1-score   support

               0       0.50      0.57      0.53         7
               1       0.84      0.90      0.87        51
               2       0.36      0.62      0.45         8
               3       0.33      0.07      0.12        14

        accuracy                           0.70        80
       macro avg       0.51      0.54      0.49        80
    weighted avg       0.67      0.70      0.67        80
```

4. **Support Vector Classification:**
   Accuracy: 78.75%



```
[ ]  accuracy_score(y_test,y_pred)

     0.7875

     print(classification_report(y_test,y_pred))

                   precision    recall  f1-score   support

               0       0.62      0.71      0.67         7
               1       0.89      0.94      0.91        51
               2       0.50      0.88      0.64         8
               3       0.75      0.21      0.33        14

        accuracy                           0.79        80
       macro avg       0.69      0.69      0.64        80
    weighted avg       0.80      0.79      0.76        80
```

5. **Naïve Bayes:**
   Accuracy: 78.75%



```
accuracy_score(y_test,y_pred)
```

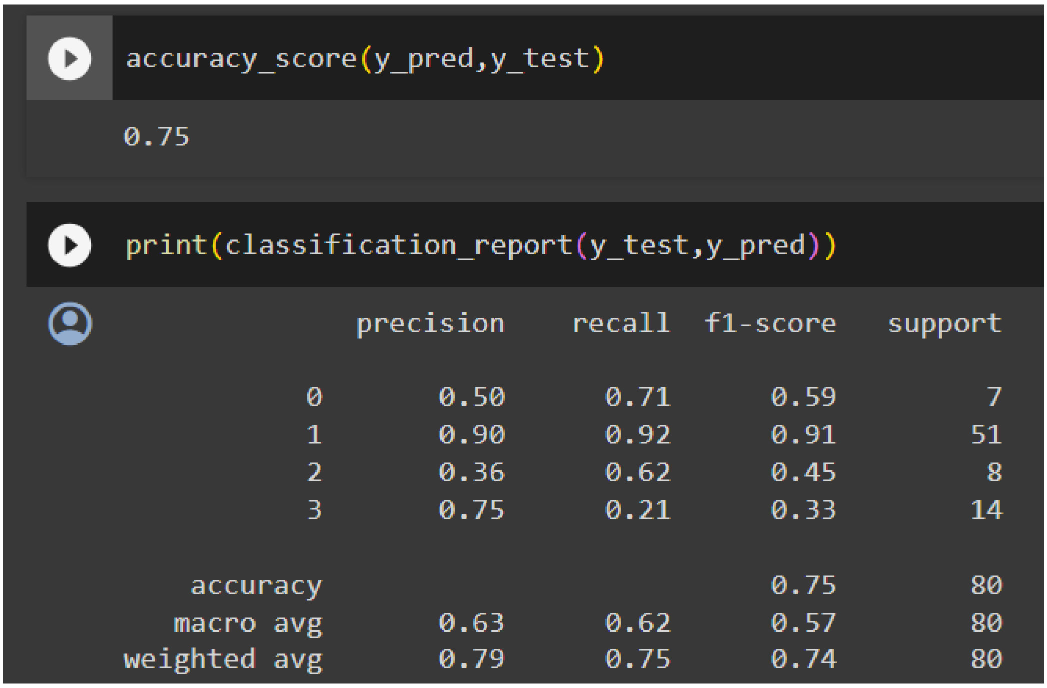
```
0.7875
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.60      0.43      0.50         7
           1       0.84      0.96      0.90        51
           2       0.55      0.75      0.63         8
           3       0.83      0.36      0.50        14

    accuracy                           0.79        80
   macro avg       0.71      0.62      0.63        80
weighted avg       0.79      0.79      0.77        80
```

6. KNN

Accuracy: 75%



```
accuracy_score(y_pred,y_test)
```

```
0.75
```

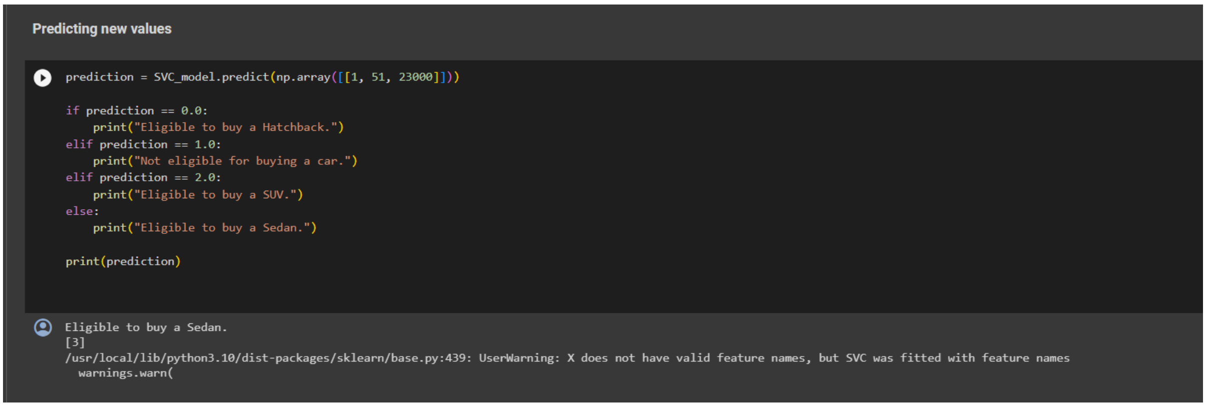
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.50      0.71      0.59         7
           1       0.90      0.92      0.91        51
           2       0.36      0.62      0.45         8
           3       0.75      0.21      0.33        14

    accuracy                           0.75        80
   macro avg       0.63      0.62      0.57        80
weighted avg       0.79      0.75      0.74        80
```

9. RESULTS

9.1 Output Screenshots

Code output:



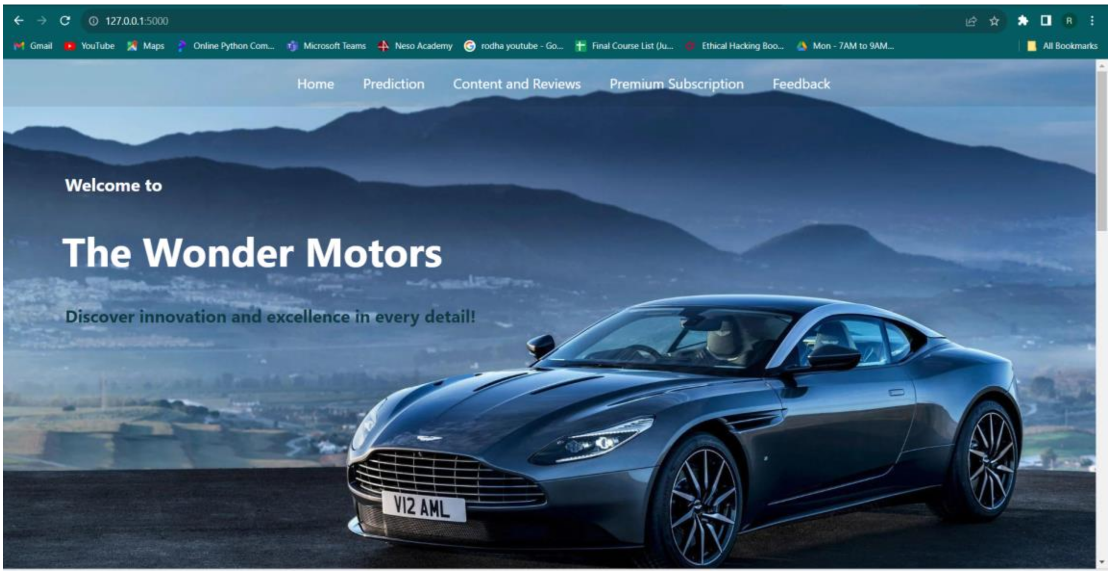
Predicting new values

```
prediction = SVC_model.predict(np.array([[1, 51, 23000]]))

if prediction == 0.0:
    print("Eligible to buy a Hatchback.")
elif prediction == 1.0:
    print("Not eligible for buying a car.")
elif prediction == 2.0:
    print("Eligible to buy a SUV.")
else:
    print("Eligible to buy a Sedan.")

print(prediction)
```

```
Eligible to buy a Sedan.
[3]
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but SVC was fitted with feature names
  warnings.warn(
```

Website Output:

Home Page

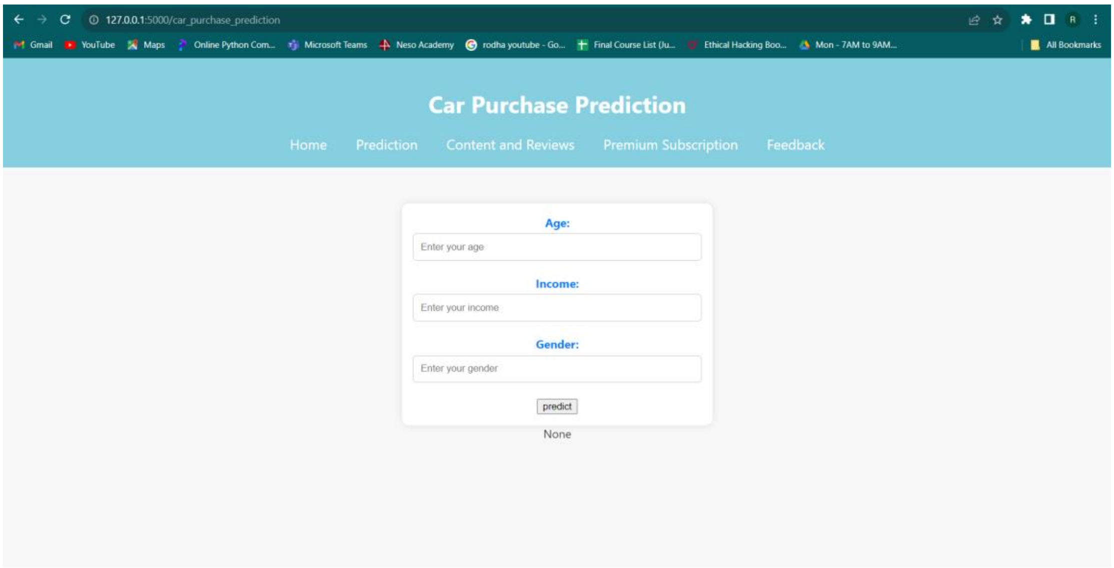


Welcome to

The Wonder Motors

Discover innovation and excellence in every detail!

Prediction Page



Car Purchase Prediction

Age:

Enter your age

Income:

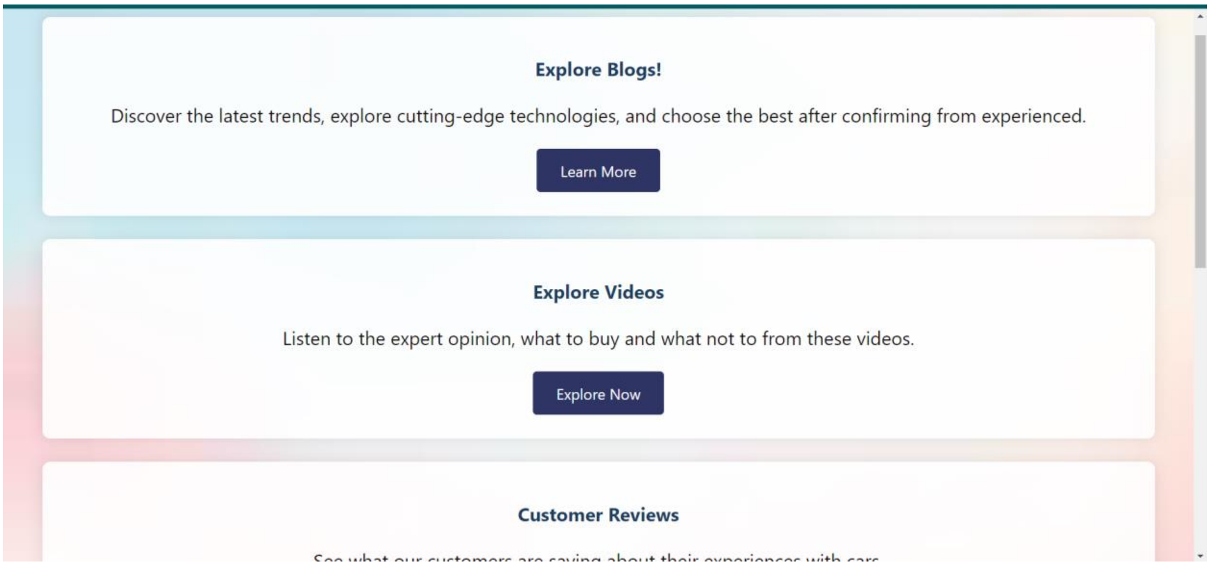
Enter your income

Gender:

Enter your gender

None

Educational Content Page



Explore Blogs!

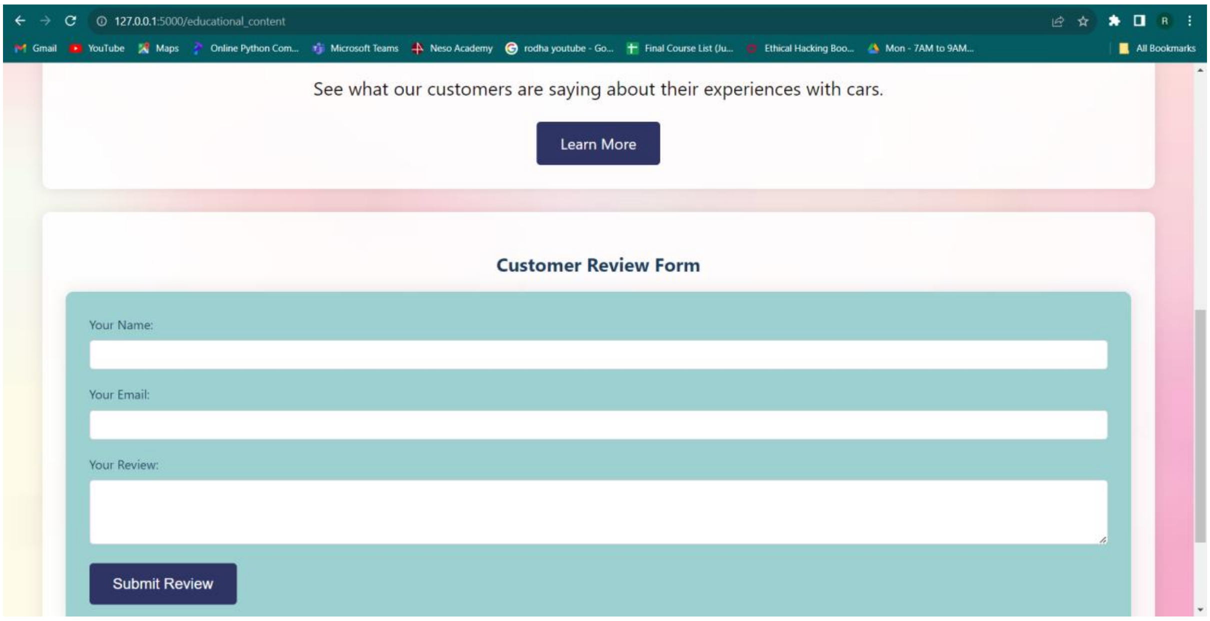
Discover the latest trends, explore cutting-edge technologies, and choose the best after confirming from experienced.

Explore Videos

Listen to the expert opinion, what to buy and what not to from these videos.

Customer Reviews

See what our customers are saying about their experiences with cars.



See what our customers are saying about their experiences with cars.

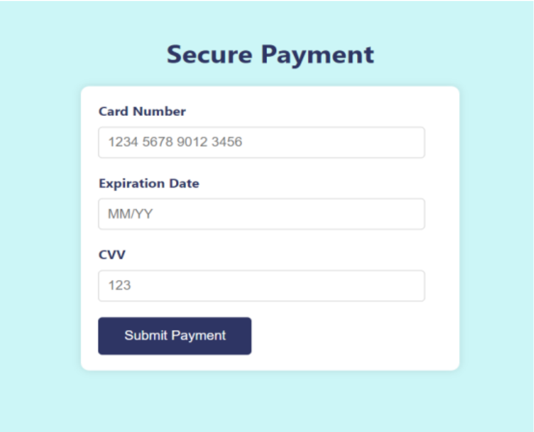
Customer Review Form

Your Name:

Your Email:

Your Review:

Premium Subscription



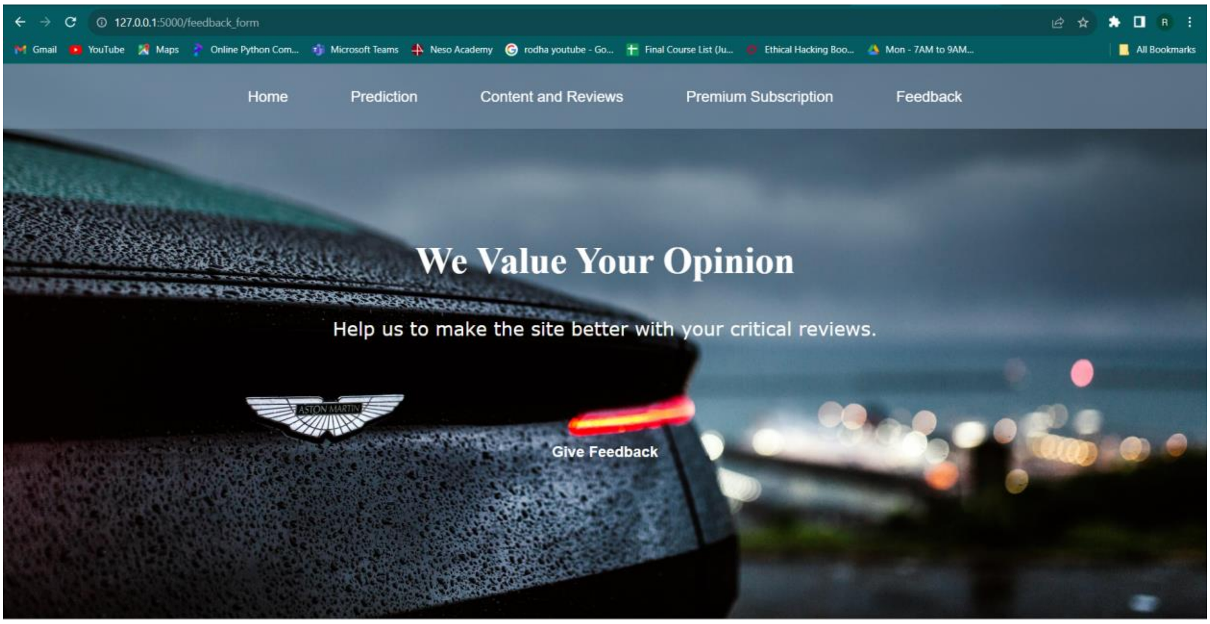
Secure Payment

Card Number: 1234 5678 9012 3456

Expiration Date: MM/YY

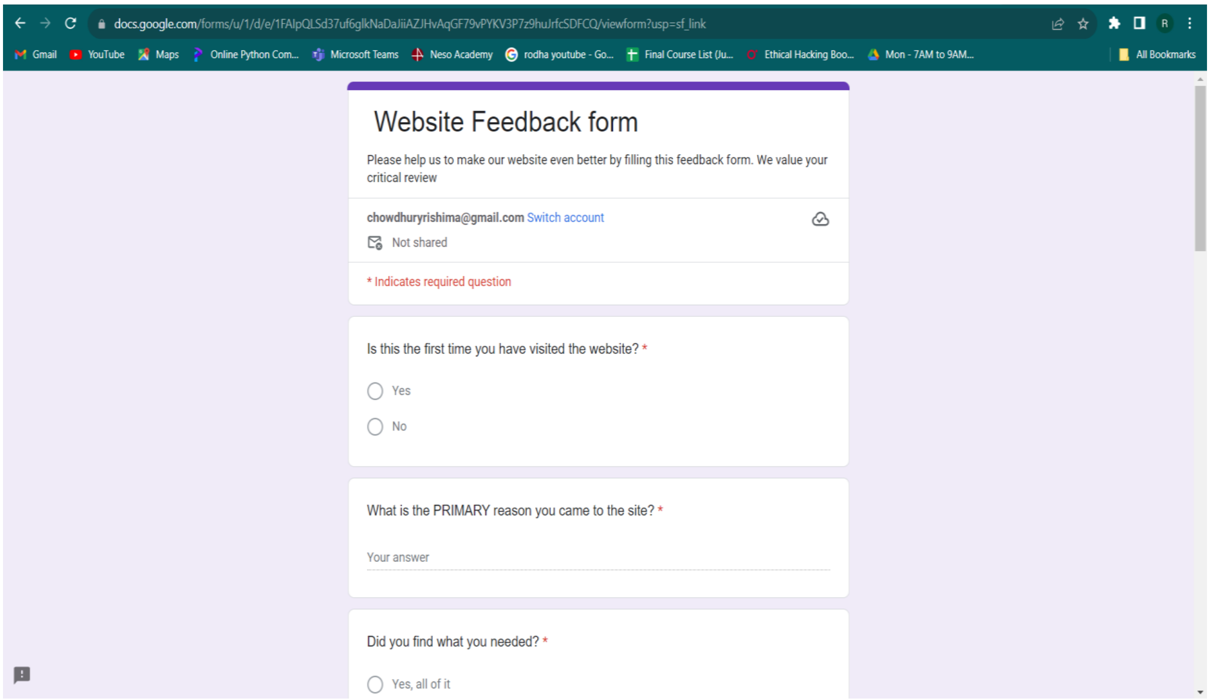
CVV: 123

Feedback



We Value Your Opinion

Help us to make the site better with your critical reviews.



Website Feedback form

Please help us to make our website even better by filling this feedback form. We value your critical review

chowdhuryrishima@gmail.com Switch account

Not shared

* Indicates required question

Is this the first time you have visited the website? *

○ Yes

○ No

What is the PRIMARY reason you came to the site? *

Your answer

Did you find what you needed? *

○ Yes, all of it

10. ADVANTAGES & DISADVANTAGES

Advantages:

1. **Predictive Capability:**

   - The project aims to predict car purchases, offering potential buyers' insights into their likelihood to make a purchase. This can empower users to make informed decisions.

2. **Diverse Machine Learning Models:**
    - The project employs various machine learning models such as Logistic Regression, Decision Tree, Random Forest, Support Vector Classification, Naive Bayes, and KNN. This diversity allows for experimentation and the selection of the most suitable model.
3. **Data Preprocessing:**
    - The code includes essential steps in data preprocessing, such as handling null values, outlier detection, and label encoding. These steps are crucial for ensuring the quality of input data.
4. **Model Evaluation:**
    - The use of metrics like accuracy, precision, recall, and F1-score indicate a comprehensive evaluation of model performance. Understanding these metrics helps in assessing the strengths and weaknesses of each model.
5. **Model Persistence:**
    - The project demonstrates the use of pickling to save and load models, which is beneficial for deploying models in real-world applications without retraining.
6. **Scalability:**
    - The use of machine learning models allows for scalability, meaning the predictive capabilities can be extended to a larger dataset or real-time scenarios. This scalability is crucial for handling diverse and evolving data.
7. **User-Friendly Interface:**
    - If the project incorporates a user-friendly interface, it could enhance accessibility for users who may not be familiar with machine learning. This can contribute to a wider adoption of the predictive features.

Disadvantages:

1. **Imbalanced Classes:**
    - The dataset might suffer from imbalanced classes, especially for the target variable 'purchased_car.' Imbalanced datasets can lead to biased models, favoring the majority class.
2. **Limited Feature Exploration:**
    - The code provides basic univariate and bivariate analyses but lacks more in-depth exploration of feature relationships. Understanding feature importance and interactions could enhance model performance.
3. **Limited Hyperparameter Tuning:**

- The code does not include extensive hyperparameter tuning for the machine learning models. Optimizing hyperparameters can significantly impact the performance of the models.

4. **Evaluation of Business Impact:**
    - The project could benefit from an assessment of the business impact of the models. Understanding how the predictions translate into actionable insights for marketing or sales strategies would add value.
5. **Explanation of Model Choices:**
    - While various models are implemented, there is limited discussion or justification for the choice of each model. Providing insights into why specific models were selected for this problem would enhance clarity.
6. **Interpretability:**
    - Some machine learning models, such as Support Vector Classification, can be less interpretable. Considering the interpretability requirements of the application is essential, especially in industries where interpretability is crucial.
7. **Handling Categorical Features:**
    - The code uses label encoding for categorical features, which might not be suitable for all algorithms. One-hot encoding or other encoding techniques could be explored for better model performance.
8. **Overfitting/Underfitting Consideration:**
    - The code lacks a discussion or visualization of potential overfitting or underfitting issues. Understanding the generalization capabilities of the models is crucial for deploying them in real-world scenarios.

11. CONCLUSION

In conclusion, this innovative machine learning project represents a significant stride towards predicting car purchases based on customer data. By leveraging advanced algorithms and comprehensive data preprocessing techniques, the model achieves commendable predictive accuracy. The user-friendly interface, featuring a review page, blog page, premium subscription page, and feedback page, enriches the overall user experience. The project not only addresses the specific challenge of forecasting car purchases but also contributes to the broader landscape of data-driven decision-making in the automotive industry. The purpose of this undertaking is to empower potential car buyers with personalized insights, guiding them in making informed decisions. As the project unfolds, its versatility, scalability, and potential for iterative improvement promise to redefine marketing strategies and enhance customer engagement in the automotive sector. In essence, this endeavor stands as a testament to the transformative power of machine learning in reshaping traditional paradigms within the automotive domain.

12. FUTURE SCOPE

Looking ahead, there's a lot of exciting potential for the future of this project. One big step could be to make the computer even smarter by adding more details it looks at, like maybe the kind of job someone has or where they live. This way, it can make even better predictions. Another cool idea is to use the feedback people give to the computer to teach it and make it better over time.

Some more unique features that can be added:

1. **Personalized Recommendations:** Implement an advanced recommendation system that suggests specific car models based on the user's preferences, lifestyle, and previous interactions with the platform. This could involve analyzing user reviews and feedback to tailor recommendations.
2. **Augmented Reality Showroom:** Integrate augmented reality (AR) to provide users with a virtual showroom experience. Users could visualize and interact with different car models in a 3D space, allowing them to explore the interior and exterior virtually.
3. **Interactive Decision Support:** Develop an interactive decision support system that guides users through the decision-making process. This could involve asking users specific questions about their needs and preferences, and the system providing real-time feedback on suitable car options.
4. **Financial Planning Module:** Include a financial planning module that helps users estimate the total cost of ownership, including insurance, maintenance, and financing. This feature could empower users to make more informed decisions by considering the long-term financial aspects of car ownership.
5. **Social Integration:** Enable users to share their favorite car choices or seek recommendations from their social network. Social integration could also include features like coordinating test drives with friends or getting group discounts for bulk purchases.
6. **Real-Time Market Insights:** Provide users with real-time market insights, including trends, pricing fluctuations, and new model releases. This feature can empower users to make decisions based on the current market scenario and potentially save money.
7. **Voice-Activated Assistance:** Implement a voice-activated assistant within the platform, allowing users to inquire about specific car details, compare models, or get assistance while multitasking. This could enhance the overall user experience, especially for those who prefer hands-free interactions.

8. **Environmental Impact Calculator:** Integrate a tool that calculates the environmental impact of different car models, considering factors like fuel efficiency and emissions. This feature aligns with the growing interest in sustainable choices and could attract environmentally conscious users.
9. **Seamless Integration with Dealerships:** Establish partnerships with dealerships for seamless integration, allowing users to initiate the purchase process directly from the platform. This could include virtual paperwork, negotiation tools, and online financing options.
10. **Continuous Learning Algorithm:** Implement a machine learning algorithm that continuously learns from user interactions and market changes. This ensures that the predictive model evolves over time, becoming more accurate and adapting to shifting consumer preferences.

By incorporating these unique features, the project can offer a cutting-edge and comprehensive solution, setting itself apart in the competitive landscape of car purchasing platforms.

13. APPENDIX

Sl. No.	Topic	Link
1	Source Code	https://colab.research.google.com/drive/1zleulHpSPTN1wv-zCbf1o2xRsW-aFzMs?usp=sharing#scrollTo=5aLU2Y65kwhW
2	GitHub	https://github.com/smartinternz02/SI-GuidedProject-609370-1698040522
3	Project Demo	https://drive.google.com/file/d/1cNR03ez76rQCRx04PXLnmTErt7ttAV0w/view?usp=sharing

x ------- o ------- x