

Car purchase Prediction Using ML

Team Members (Team ID : Team-593136)

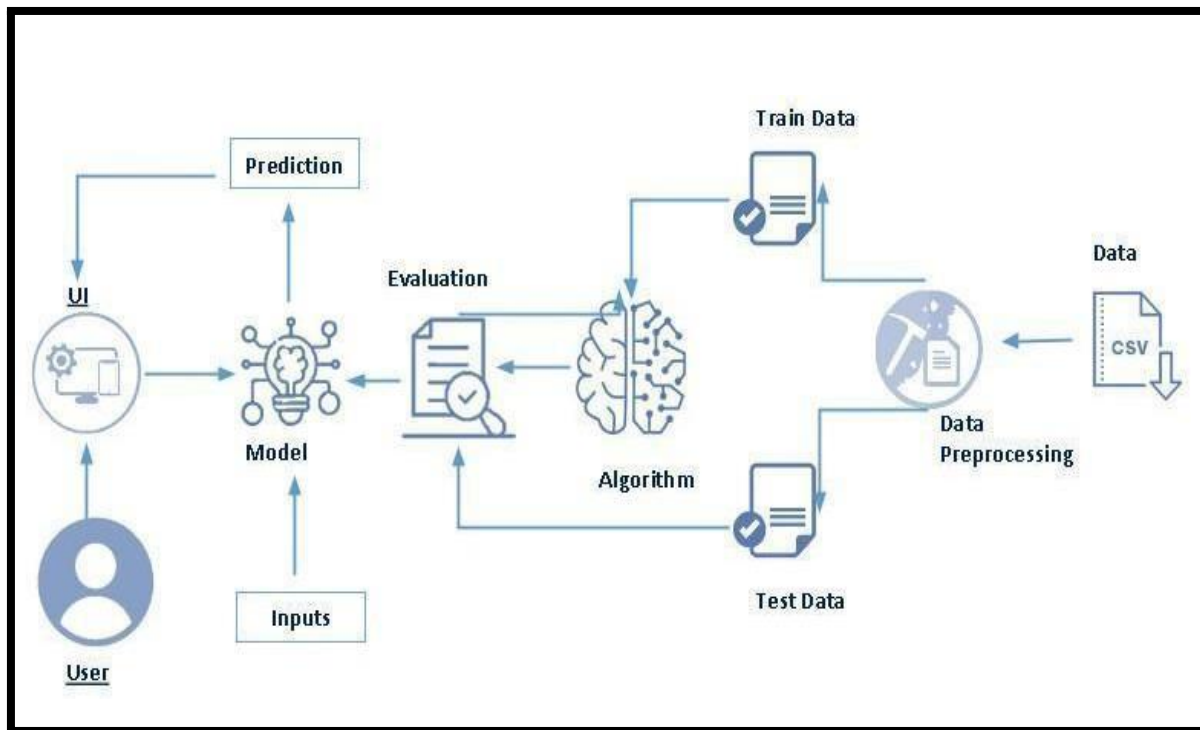
J Threvikram
K Gokul Prasad
P Mithun
R Nevatha Sri

Project Description:

Developed an innovative ML solution to predict car purchases based on customer data. Leveraged features such as age, income, and historical purchase patterns for accurate forecasts. Achieved high predictive accuracy using advanced algorithms and thorough data preprocessing. The model assists potential buyers by estimating their likelihood to make a purchase, guiding decision-making. Seamlessly integrated the model into a user-friendly interface, enabling easy predictions for users. This project revolutionizes the automotive industry by offering insights for tailored marketing strategies. Enhances customer experiences by facilitating informed choices and dealership targeting. A groundbreaking application of ML driving data-powered decisions.

Through meticulous training and feature engineering, the model attains a high accuracy rate, ensuring dependable predictions. Seamlessly integrated into a user-friendly interface, users input their demographics and receive precise purchase likelihoods. This innovation revolutionizes marketing strategies by enabling targeted customer engagement and resource optimization. By harnessing data insights, the project empowers automotive businesses to tailor their approaches, enhancing overall efficiency.

Technical Architecture:



Pre requisites:

To complete this project, you must required following software's, concepts and packages

- **Anaconda navigator and Visual Studio Code:**
- **Python packages:**
 - Open anaconda prompt as administrator
 - Type "pip install numpy" and click enter.
 - Type "pip install pandas" and click enter.
 - Type "pip install scikit-learn" and click enter.
 - Type "pip install matplotlib" and click enter.
 - Type "pip install scipy" and click enter.
 - Type "pip install joblib" and click enter.
 - Type "pip install seaborn" and click enter.
 - Type "pip install Flask" and click enter.

Prior Knowledge:

You must have prior knowledge of following topics to complete this project.

- **ML Concepts**
 - Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
 - Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>

- Decision Tree:
<https://www.analyticsvidhya.com/blog/2022/03/decision-tree-machine-learning-using-python/>
- Random Forest:
<https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/>
- Evaluation metrics:
<https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>
- **Flask Basics** : https://www.youtube.com/watch?v=Ij4I_CvBnt0

Project Objectives:

By the end of this project you will:

- Know fundamental concepts and techniques used for machine learning.
- Gain a broad understanding about data.
- Have knowledge on pre-processing the data/transformation techniques on outlier and some visualization concepts.

Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

- Data collection
 - Collect / create the dataset
- Visualizing and analyzing data
 - Univariate analysis
 - Bivariate analysis
 - Multivariate analysis
- Data pre-processing
 - Checking for null values
 - Handling outlier
 - Handling categorical data
 - Splitting data into train and test
- Model building
 - Import the model building libraries
 - Initializing the model
 - Training and testing the model
 - Evaluating performance of model
 - Save the model

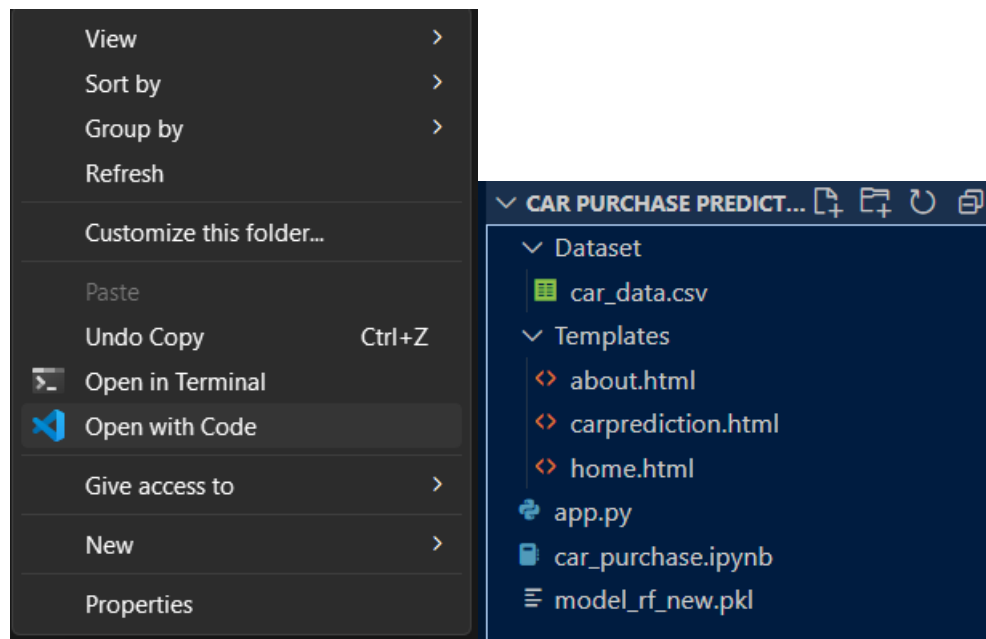
- Application Building
 - Create an HTML file
 - Build python code

Project Structure:

Create a folder with any name choice and create the subfolders as follows.

Name	Date modified	Type	Size
Dataset	05-11-2023 11:46	File folder	
Templates	05-11-2023 11:47	File folder	
app.py	06-11-2023 21:16	Python Source File	2 KB
car_purchase.ipynb	05-11-2023 15:58	IPYNB File	773 KB
model_rf_new.pkl	05-11-2023 16:01	PKL File	1,239 KB

Once you create it, right click and press “open with code”. Now visual studio code will open the folder for you. Once opened, you will see something like this



- The dataset folder contains the dataset which we will be using for our model building.
- “model_rf_new.pkl” is the model we built for the problem.
- We are building a flask application which needs HTML pages stored in the templates folder.
- The app.py file is the python file in which we will build the flask application
- The Templates folder contains the codes for the HTML pages which we will need for the HTML page which will be integrated with the ML model

Milestone 1: Define Problem / Problem Understanding

Activity 1: Specify the business problem

Optimizing automobile firms' marketing and sales strategies is the core business issue that the machine learning model for car purchases attempts to solve. The primary task at hand involves accurately identifying prospective clients who are more likely to acquire an automobile, mostly by means of their age and income demographics. The model's main goal is to provide accurate forecasts about the purchasing habits of customers.

By doing this, the approach gives companies the capacity to more precisely and effectively distribute their marketing budgets. The main goal is to increase marketing efforts' efficiency by making them unique for prospective clients. As a result, this lowers unnecessary marketing expenses while also raising the likelihood of successful conversions. The primary goal of the model is to provide a data-driven solution that improves decision-making in the automobile sector and maximizes the return on marketing expenditures.

Activity 2: Business requirements

These criteria are broken down into a number of smaller requirements:

Data Requirements: We identify the precise data points needed to provide reliable forecasts. This covers demographic information, past purchasing trends, and any other relevant data.

Accuracy and Reliability: In order for our solution to satisfy the demands of the business, we specify the required degree of predicted accuracy and reliability.

Scalability: We ascertain whether the system can accommodate sizable numbers of users and data.

User Interface requirements: We outline the specifications for the user-friendly interface, placing a strong emphasis on accessibility and ease of use.

Integration with Pre-Existing Systems: We determine whether the solution must be able to easily integrate with marketing platforms or for example, Customer Relationship Management (CRM) systems.

Compliance and Security: We take care of all legal and security obligations, paying special attention to the privacy and data of our clients.

Resource Allocation: We determine the resources—human and technological infrastructure included—needed to carry out the project.

Timeline and Milestones: In order to efficiently monitor and track progress, we establish a clear project timeline and specify significant milestones.

Activity 3: Literature Survey (Student Will Write)

Applications of machine learning in the automobile sector are gaining a lot of attention lately. Predictive maintenance, autonomous driving, and consumer behavior analysis are just a few of the many applications that have been thoroughly reviewed and examined, as demonstrated in "Machine Learning Applications in the Automotive Industry: A Comprehensive Review" (Ding et al., 2019). This assessment, while not directly related to our research, offers a basic grasp of the many applications of machine learning in the automotive industry, laying the groundwork for our creative method of using customer data to forecast car purchases.

Research like "Predictive Modeling of Customer Lifetime Value and Purchase Propensity: Application to E-commerce" (Bult and Wansbeek, 1995) has set the foundation for understanding predictive modeling in a variety of scenarios. Predicting consumer purchase behavior is a well-known idea. Although this study isn't specifically related to the car industry, it does highlight how important it is to estimate customers' tendency to buy, which is a crucial component of our initiative. It provides a point of reference for our work in creating a model that helps automakers accurately forecast the buying habits of their customers.

In the automotive business, data-driven decision-making has grown ever more essential for effective marketing initiatives. Studies like Chaffey and Smith's (2013) "Data-Driven Marketing: Leveraging Big Data for Your Business" examine how customer insights and data analytics influence marketing tactics. This study emphasizes how important data is to the sector and how predictive models are necessary to improve marketing campaigns. Our project seeks to comply with these standards and advance data-driven decision-making in the automotive industry by personalizing marketing techniques.

Activity 4: Social and Business Impact.

The Anime Recommendation project can have both social and business impacts.

Social Impact:

Customers gain from a better tailored experience since less irrelevant marketing is sent to them because their likelihood of making a purchase is determined by individual attributes. Positive brand-consumer interactions are fostered by this, as it increases user happiness and trust. Additionally, by matching clients with appropriate automotive selections while taking their financial situation into account, the approach promotes responsible consumerism. Customers are encouraged to acquire cars with knowledge, which fosters financial prudence, as data-driven decisions grow more common.

Business Impact:

Marketing endeavors become intensely concentrated, concentrating on those who have a greater likelihood of converting, leading to increased effectiveness and decreased expenses. The methodology makes it easier to allocate resources more effectively, allocating funds most profitably to initiatives that offer the biggest return on investment. Leads can be prioritized by sales teams, which can speed up the conversion process and possibly increase number of sales. The precise forecasts made by the model eventually lead to increased consumer satisfaction and enhanced brand recognition. As data-driven tactics become more popular, the company maintains its lead in a cutthroat industry and is well-positioned for expansion and innovation.

Milestone 2: Data Collection and Visualizing and analyzing the data

We have obtained the dataset from the following link. Most crucial step in the making of any ML model is the dataset in which the model should be trained on. Without a dataset, there is no ML model.

Link: <https://www.kaggle.com/code/vishesh1412/car-purchase-decision-eda-and-decision-tree/input>

Activity 1: Download the dataset

The dataset is downloaded from kaggle.com. There are many platforms which contain the dataset.

Activity 2: Importing the libraries

These libraries make it possible for python to create an ML model. Hence importing the necessary libraries is a crucial step

```
import pandas as pd
import numpy as np
```

```
from sklearn.model_selection import train_test_split
```

```
import seaborn as sns
import matplotlib.pyplot as plt
```

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

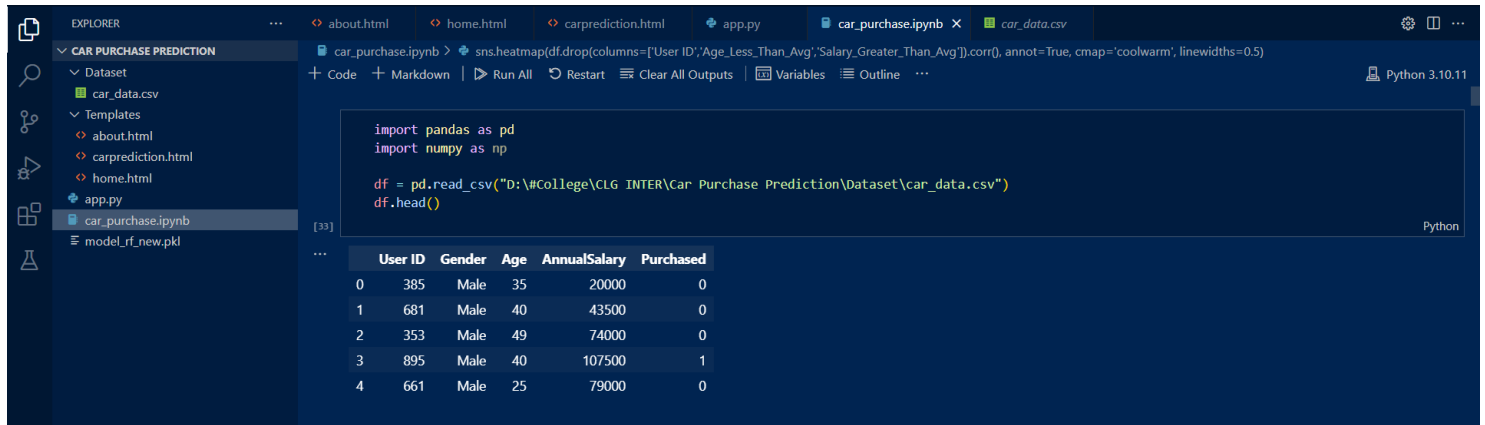
```
from scipy.stats import pointbiserialr
from sklearn.preprocessing import LabelEncoder
```

```
import joblib
```

```
from sklearn.metrics import confusion_matrix
```


Activity 3: Read the Dataset

Our CSV dataset can be used in python with the pandas library. The below code is used to load in the data present in the CSV file.



The screenshot shows a Jupyter Notebook with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'CAR PURCHASE PREDICTION' containing a 'Dataset' folder with 'car_data.csv', and other files like 'about.html', 'home.html', 'carprediction.html', 'app.py', 'car_purchase.ipynb', and 'model_rf_new.pkl'. The code editor shows the following Python code:

```
import pandas as pd
import numpy as np

df = pd.read_csv("D:\#College\CLG INTER\Car Purchase Prediction\Dataset\car_data.csv")
df.head()
```

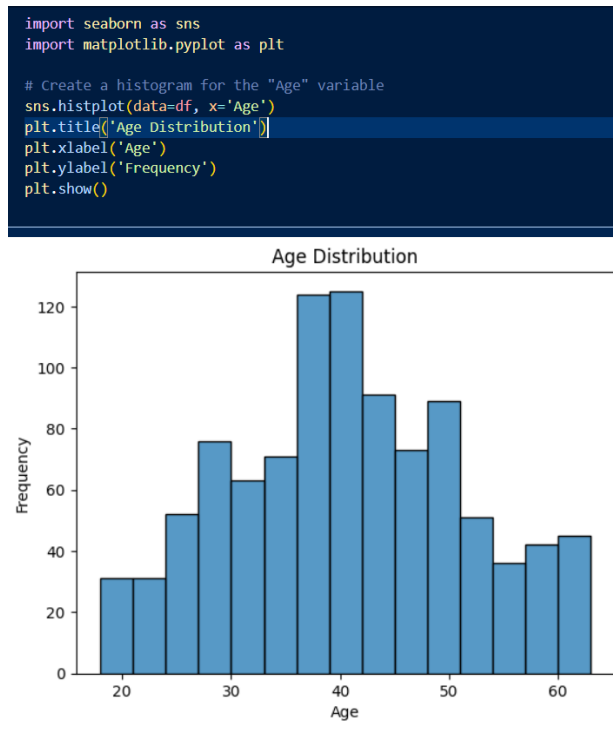
The output of the code is displayed below the code cell, showing the first five rows of the dataset:

	User ID	Gender	Age	AnnualSalary	Purchased
0	385	Male	35	20000	0
1	681	Male	40	43500	0
2	353	Male	49	74000	0
3	895	Male	40	107500	1
4	661	Male	25	79000	0

Activity 4: Univariate analysis

Univariate analysis is used to visualize and understand the dataset with one singular feature.

Packages such as seaborn are used to achieve this. Seaborn and matplotlib package can be used to achieve the same.

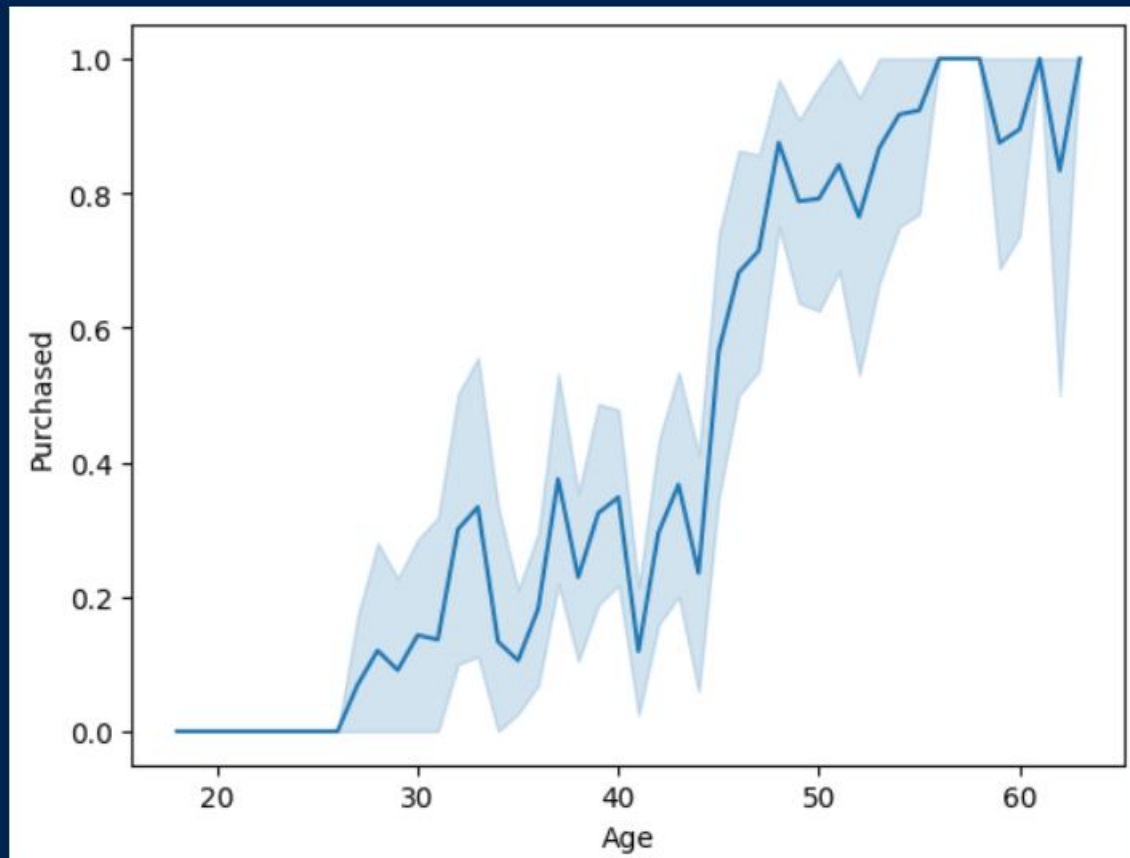


Activity 5: Bivariate analysis

Bivariate analysis is used to visualize and understand the dataset with two feature.

```
sns.lineplot(x = df.Age,y=df.Purchased)
```

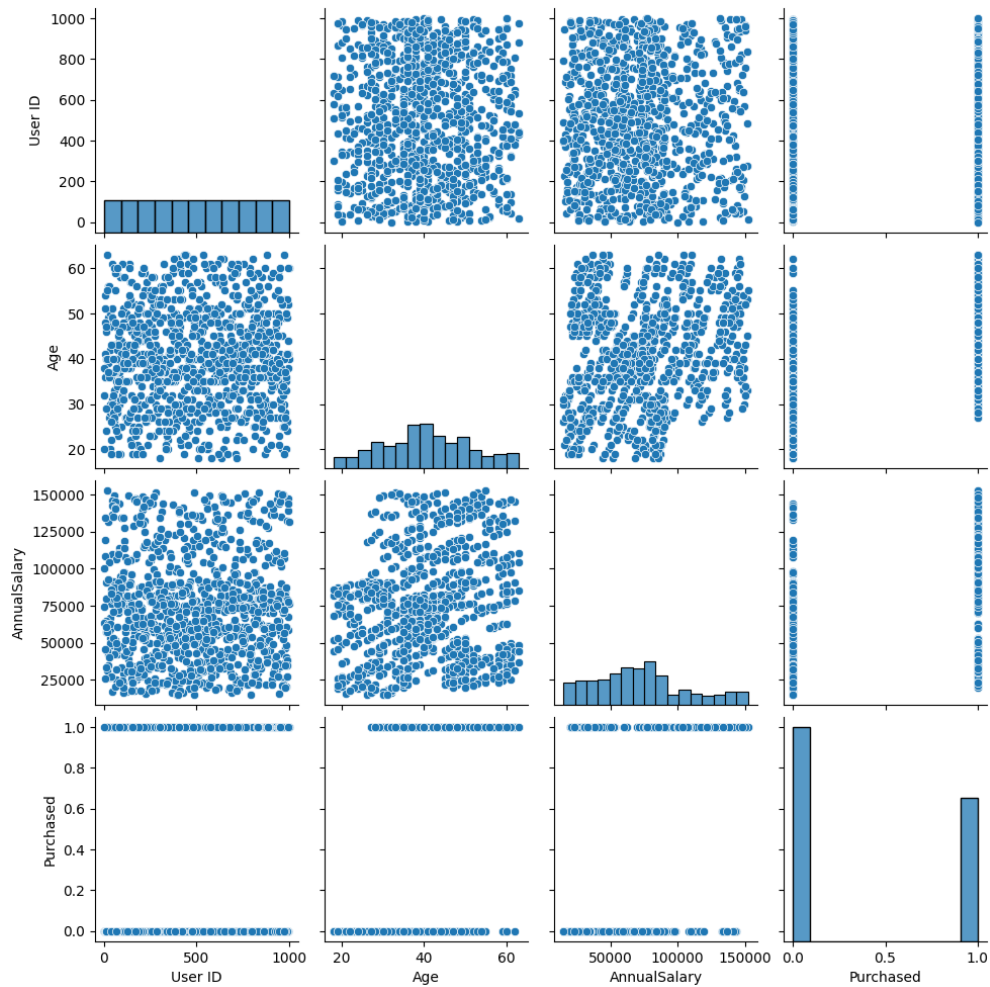
```
<Axes: xlabel='Age', ylabel='Purchased'>
```



Activity 6: Multivariate analysis

Multivariate analysis means finding the relationship between multiple variables. Seaborn package is used for this pairplot.

```
sns.pairplot(df)
```



Milestone 3: Data Pre-processing

Data preprocessing is an important step towards building an ML model. The dataset downloaded will not always be suitable for model training. This is because the dataset may contain null values or high randomness or contain outliers. Without taking necessary steps to clear these, we will not find good result for our ML model

- Handling missing values
- Handling categorical data
- Handling outliers
- Scaling Techniques
- Splitting dataset into training and test set

Note: These are the general steps of pre-processing the data before using it for machine learning.

Depending on the condition of your dataset, you may or may not have to go through all these steps.

Activity 1: Checking for null values

- The `{ }.info()` and `{ }.isnull().any()` function can provide us the information regarding null values present in our dataset.

```
df.isnull().any()
```

User ID	False
Gender	False
Age	False
AnnualSalary	False
Purchased	False

```
dtype: bool
```



```
df.info()
```

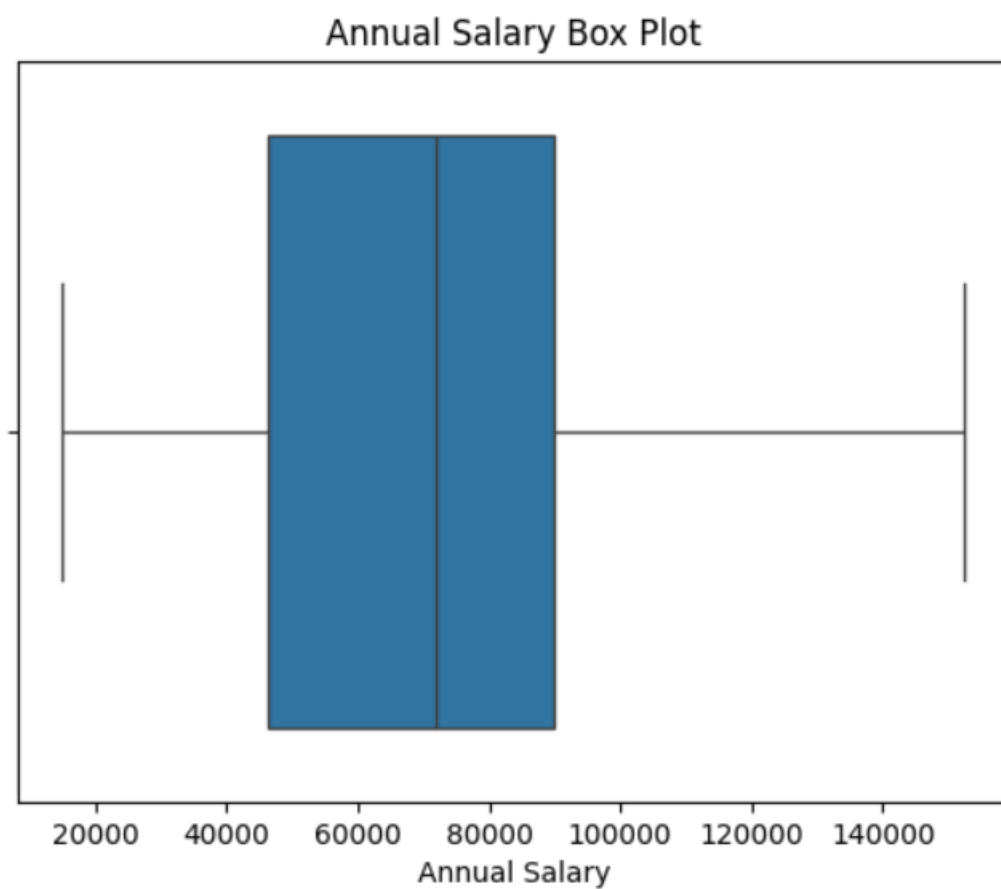
```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1000 entries, 0 to 999  
Data columns (total 5 columns):  
#   Column          Non-Null Count  Dtype  
---  ---            -  
0   User ID         1000 non-null   int64  
1   Gender          1000 non-null   object  
2   Age             1000 non-null   int64  
3   AnnualSalary    1000 non-null   int64  
4   Purchased       1000 non-null   int64  
dtypes: int64(4), object(1)  
memory usage: 39.2+ KB
```

Activity 2: Handling outliers

We can visualize any present outliers using the `sns.boxplot({})`.

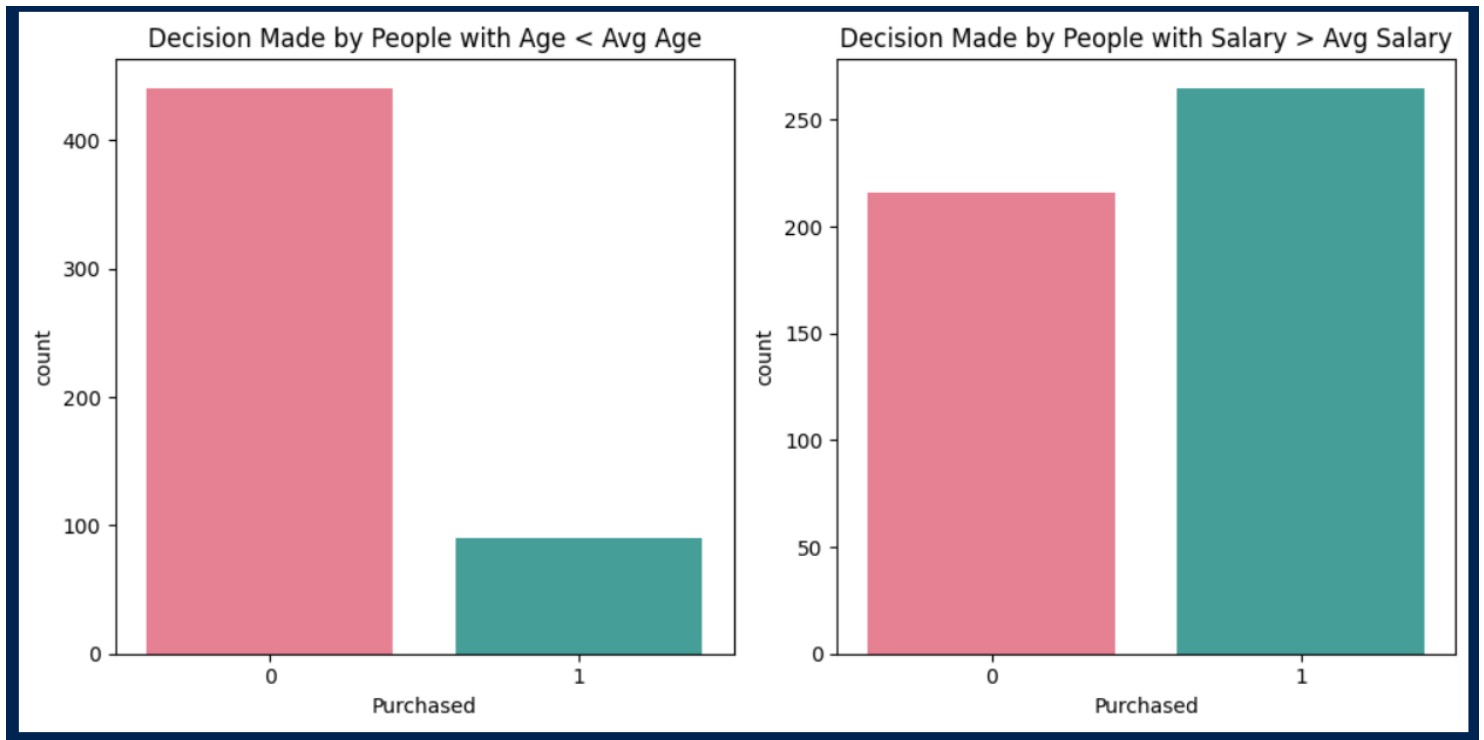
The below diagram will be the result of our boxplot for the column “AnnualSalary”.

```
sns.boxplot(data=df, x='AnnualSalary')  
plt.title('Annual Salary Box Plot')  
plt.xlabel('Annual Salary')  
plt.show()
```

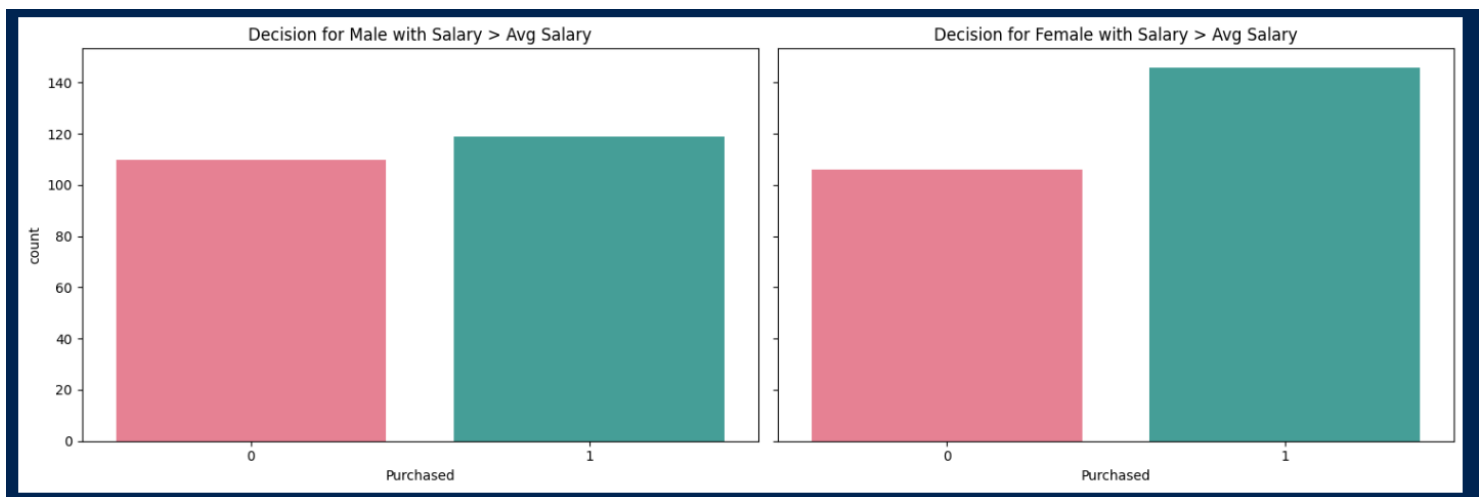


Activity 3: General Visualization

Decision made by people whose age is lesser than average age



Decision made w.r.t Gender whose salary is greater than average salary



Activity 4: Splitting data into train and test

Splitting the dataset into Training and testing is a good approach towards building an ML model.

This is because the model has to be tested on data that it has not been trained with yet, or

Unknown Data. This unknown data will be our test data. Only when the model is tested on this “test” data, we will come to know about its accuracy and error.

We can use `train_test_split` for splitting our data into training and testing.

```
# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

```
x.head()
```

	Gender	Age	AnnualSalary	Age_Salary_Interact
0	1	35	20000	700000
1	1	40	43500	1740000
2	1	49	74000	3626000
3	1	40	107500	4300000
4	1	25	79000	1975000

```
y.head()
```

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

Note: (This step is not necessary for every dataset)

I have added another column in our dataset named “Age_Salary_Interact”. This new column is a function of both “Age” column and “AnnualSalary” column of our dataset.

This was added as a measure to increase the accuracy of our model. These columns were selected based on their correlation with the parameter that has to be predicted. Hence, it will change for different datasets.

Milestone 4: Model Building

Now our data is cleaned and it's time to build the model. We can train our data on different algorithms. For this project we are applying the RandomForest Classifier method. This is an advanced method for Decision Tree models, as a random forest classifier is built on multiple decision trees.

Activity 1: Random Forest Classifier

Decision Tree Classifier algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

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

model_rf_new = RandomForestClassifier(n_estimators=100, random_state=0)

# Training model with train data
model_rf_new.fit(x_train, y_train)
```

▼ RandomForestClassifier
RandomForestClassifier(random_state=0)

Activity 3: Evaluating performance of the model.

```
# Pred on training set
y_train_pred = model_rf_new.predict(X_train)

# Pred on test set
y_test_pred = model_rf_new.predict(X_test)
y_pred = y_test_pred
#accuracy for both sets
train_accuracy = accuracy_score(y_train, y_train_pred)
test_accuracy = accuracy_score(y_test, y_test_pred)

# accuracy scores
print("Train Set Accuracy:", train_accuracy)
print("Test Set Accuracy:", test_accuracy)
```

Train Set Accuracy: 0.9975
Test Set Accuracy: 0.94

```
# Display a classification report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.95	0.95	0.95	121
1	0.92	0.92	0.92	79
accuracy			0.94	200
macro avg	0.94	0.94	0.94	200
weighted avg	0.94	0.94	0.94	200

Activity 4: Saving the model

Since our model is performing well with an accuracy score of about 94% on unseen data, we can go ahead and save the created “model”.

Save the model

```
import joblib

model_file_path = "D:\#College\CLG INTER\Car Purchase Prediction\model_rf_new.pkl"

# Save the model
joblib.dump(model_rf_new, model_file_path)

print(f"Random Forest model saved to {model_file_path}")
```

[61]

```
.. Random Forest model saved to D:\#College\CLG INTER\Car Purchase Prediction\model_rf_new.pkl
```

Milestone 5: Application Building

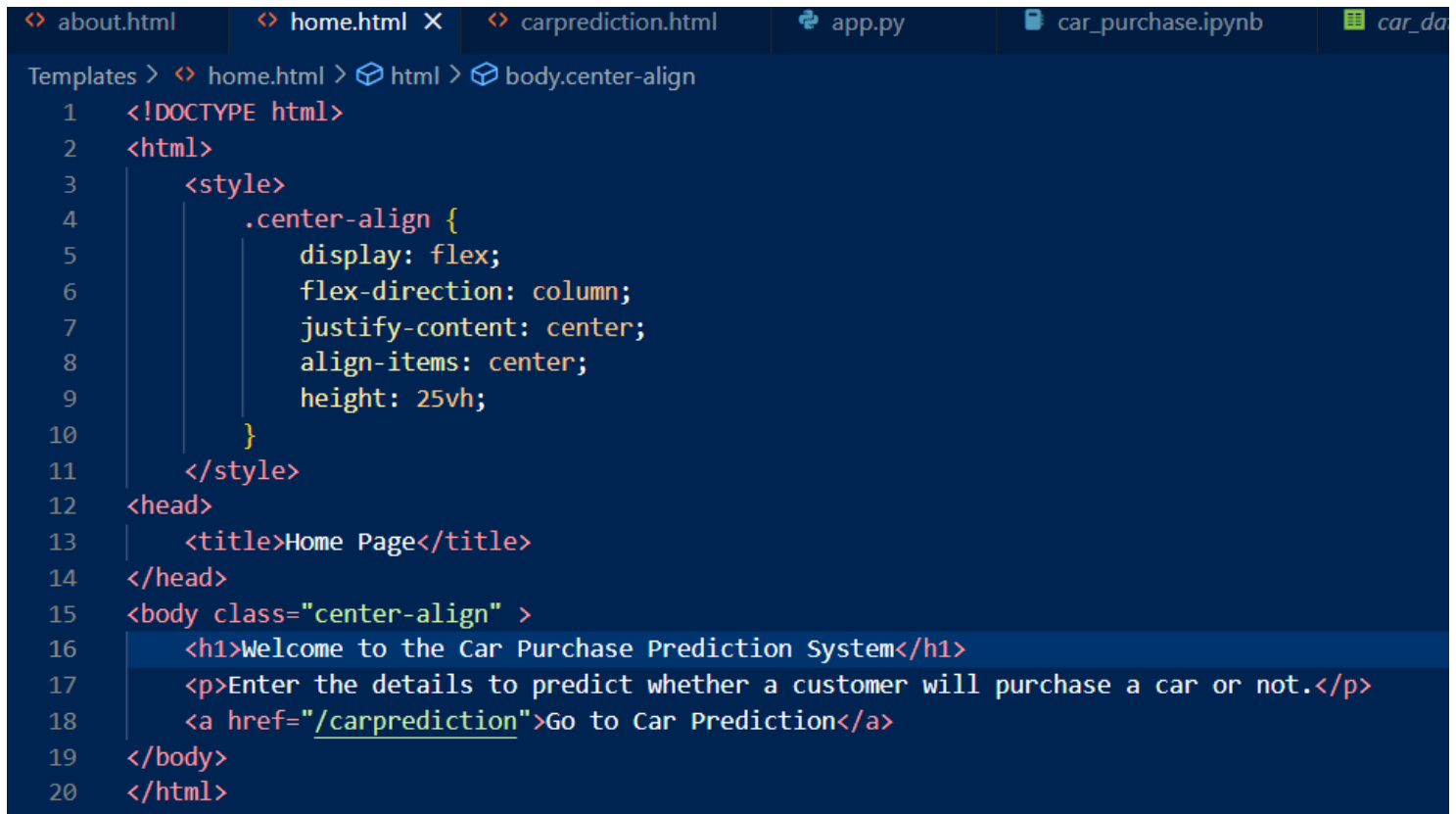
In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the users where the user should give in the required details (Age, Gender and AnnualSalary) in our case.

Our task is to build HTML applications for the webpage that we will access through our python Flask code.

Activity1: Building Html Pages:

For this project create three HTML files namely

- home.html



```
<? about.html | <? home.html X | <? carprediction.html | app.py | car_purchase.ipynb | car_da
Templates > <? home.html > html > body.center-align
1  <!DOCTYPE html>
2  <html>
3      <style>
4          .center-align {
5              display: flex;
6              flex-direction: column;
7              justify-content: center;
8              align-items: center;
9              height: 25vh;
10         }
11     </style>
12 <head>
13     <title>Home Page</title>
14 </head>
15 <body class="center-align" >
16     <h1>Welcome to the Car Purchase Prediction System</h1>
17     <p>Enter the details to predict whether a customer will purchase a car or not.</p>
18     <a href="/carprediction">Go to Car Prediction</a>
19 </body>
20 </html>
```

- about.html

```

about.html x home.html carprediction.html app.py car_purchase.ipynb car_data.csv
Templates > about.html > html > body.center-align
1  <!DOCTYPE html>
2  <html>
3  <style>
4      .center-align {
5          display: flex;
6          flex-direction: column;
7          justify-content: center;
8          align-items: center;
9          height: 25vh;
10     }
11 </style>
12 <head>
13     <title>About Us</title>
14 </head>
15 <body class="center-align" >
16     <h1>About Us</h1>
17     <p>We are a team dedicated to developing innovative machine learning solutions for car purchase prediction.</p>
18     <a href="/">Go to Home</a>
19 </body>
20 </html>
21

```

- carprediction.html

```

about.html home.html carprediction.html x app.py car_purchase.ipynb car_data.csv
Templates > carprediction.html > html > body > form.center-align
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Car Purchase Prediction</title>
5      <style>
6          /* CSS for center alignment */
7          .center-align {
8              display: flex;
9              flex-direction: column;
10             justify-content: center;
11             align-items: center;
12             height: 0vh;
13             margin-top: 200px;
14             color: rgb(0, 0, 0);
15
16             max-height: 100%
17         }
18         .display1 {
19             position: fixed;
20             bottom: 0;
21             width: 100%;
22             text-align: center;
23             background-color: white;
24         }
25     }
26
27     /* CSS for the heading */
28     .heading {
29         font-size: 35px;
30         margin-bottom: 20px;
31         display: flex;

```

and save them in Templates folder.

```
▼ Templates
  <> about.html
  <> carprediction.html
  <> home.html
```

Activity 2: Build Python code:

Import the required libraries/Modules

```
app.py > predict
1  from flask import Flask, render_template, request
2  import joblib
3  import webbrowser
4  import keyboard
```

Flask constructor takes the name of the current module (`_name_`) as argument.

```
6  app = Flask(__name__)
7
8  # Load the model
9  rf_model = joblib.load("model_rf_new.pkl")
10
```

Rendering the created HTML page:

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

@app.route("/about")
def about():
    return render_template("about.html")

@app.route("/carprediction")
def carprediction():
    return render_template("carprediction.html")
```

Here we will be using declared constructor to route to the HTML page which we have created earlier.

Whenever you enter the values from the html page the values can be retrieved using POST Method.

```
@app.route("/predict", methods=["POST"])
def predict():
    # ... (omitted code) ...

    return render_template("carprediction.html", prediction=prediction)
```

The POST method is used to retrieve the values that are entered in the HTML page and used as inputs to be fed to our model, once the prediction is generated, this function returns the prediction and the associated message will be displayed on the HTML page as coded.

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request.

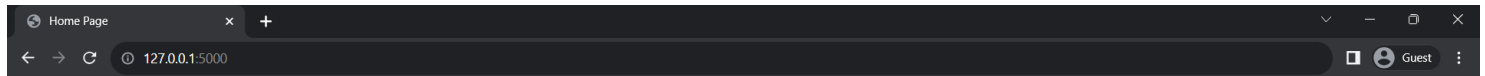
Main Function:

```
0  if __name__ == "__main__":
1      |   open_browser()
2      |   app.run(host='127.0.0.1', port=5000, debug=True)
3
```

Activity 3: Run the application

Open Visual studio code in the folder with all the project folders. When the user runs “app.py” file, a new webpage will be opened in the default browser (if not opened, click on the server in the VS Code terminal)

The Home Page



Welcome to the Car Purchase Prediction System

Enter the details to predict whether a customer will purchase a car or not.

[Go to Car Prediction](#)

If you click on “Go to Car Prediction” option, you will redirected to carprediction.html page . In this page the user will give the required inputs and the prediction made by our model will be generated.

Car Purchase Prediction

Gender:
Male ▼

Age:

Annual Salary:

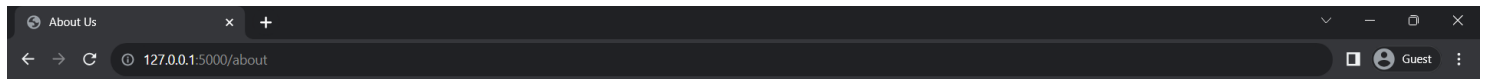
Predict

[Go to Home](#) [Go to About Us](#)

Based on the inputs, the result will look like these

Car Purchase Prediction	Car Purchase Prediction
<p>Gender: Male ▼</p> <p>Age: <input type="text"/></p> <p>Annual Salary: <input type="text"/></p> <p>Predict</p>	<p>Gender: Male ▼</p> <p>Age: <input type="text"/></p> <p>Annual Salary: <input type="text"/></p> <p>Predict</p>
Prediction: The user is likely to make a purchase	Prediction: The user is likely to make a purchase

When the user clicks on “Go to About us” page, a small description of the same is displayed



About Us

We are a team dedicated to developing innovative machine learning solutions for car purchase prediction.

[Go to Home](#)

Conclusion:

In conclusion, the developed customer car purchase prediction model proves its effectiveness in aiding decision-making within the automotive industry. The model provides useful insights into buying behaviors by leveraging client age and salary as predictive factors. Its precise predictions enable organizations to tailor marketing strategies to specific client categories, hence maximizing resource allocation. The concept improves user experience and engagement by providing tailored interactions.