

Date	23 October 2023
Team ID	Team-592176
Project Name	Project - Car Purchase Prediction

Car purchase Prediction Using ML

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.

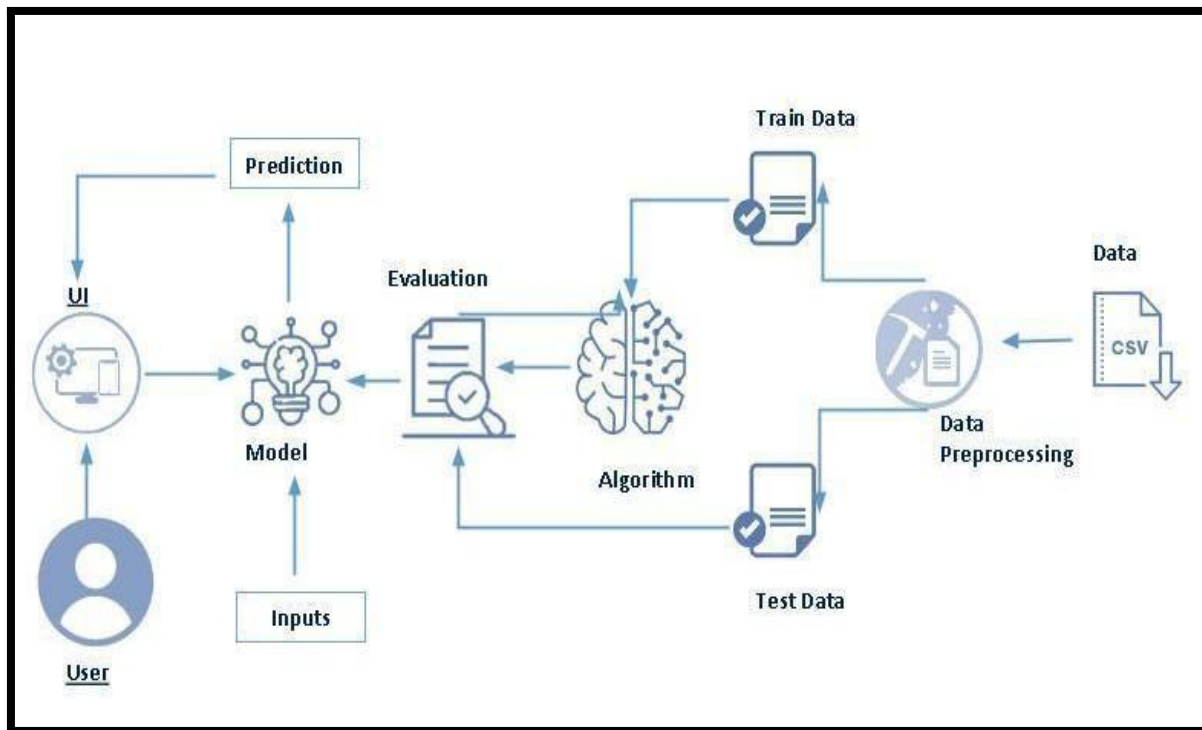
Seamlessly integrated the model into a user-friendly interface, enabling easy predictions for users.

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

- **VS Code and Anaconda:**
 - Refer the link below to download anaconda navigator
- **Python packages:**
 - Pandas
 - Numpy
 - Seaborn & matplotlib
 - Scikit learn

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>
 - GBM & LightGBM : <https://machinelearningmastery.com>

- Evaluation metrics:
<https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>
- **Flask Basics** : https://www.youtube.com/watch?v=li4I_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

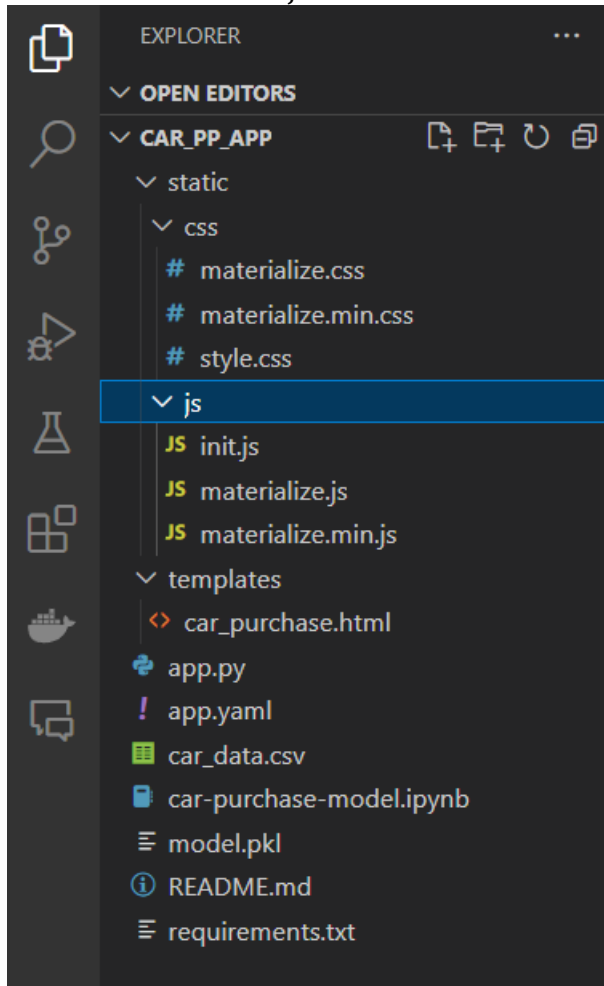
To accomplish this, we have to complete all the activities listed below,

- Data collection
 - Data collected from Kaggle database
- Visualizing and analyzing data
 - Univariate analysis
 - Bivariate analysis
 - Multivariate analysis
 - Descriptive 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 the Project folder which contains files as shown below



- We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
- Model.pkl is our saved model. Further we will use this model for flask integration.
- Data folder contains dataset and Templates folder contains html files.

Milestone 1: Define Problem / Problem Understanding

Activity 1: Specify the business problem

The business problem addressed by the car purchase prediction ML model is to enhance the efficiency of marketing and sales strategies for automotive companies. The primary challenge is to identify potential customers who are more likely to make a car purchase based on their age and income demographics. By accurately predicting customer purchase behaviors, the model assists businesses in

allocating their marketing resources more effectively. This optimization leads to personalized targeting of potential customers, reducing marketing costs while increasing the chances of successful conversions. Overall, the model aims to provide a data-driven solution that enhances the decision-making process in the automotive industry and maximizes the return on marketing investments.

Activity 2: Business requirements

Data Collection and Integration: Data collected from KAGGLE.COM

Integration with Marketing Strategies: The model integrate with the company's marketing strategies, enabling targeted campaigns towards potential car buyers.

Scalability: Design the model to handle a growing number of customer data points while maintaining prediction accuracy and response times.

Resource Optimization: Assist in optimizing marketing resources by directing efforts towards customers with a higher probability of purchasing a car, leading to reduced costs and improved ROI.

Customization: Consider the ability to customize the model for specific market segments or product lines, enabling fine-tuned predictions and targeted strategies.

Activity 3: Literature Survey

A literature survey would involve researching and reviewing existing studies, articles, and other publications on the topic of project. The survey would aim to gather information on current systems, their strengths and weaknesses, and any gaps in knowledge that the project could address. The literature survey would also look at the methods and techniques used in previous projects, and any relevant data or findings that could inform the design and implementation of the current project.

Activity 4: Social and Business Impact.

Social Impact:

The introduction of the car purchase prediction ML model brings notable social implications. Customers benefit from a more personalized experience as their purchasing likelihood is assessed based on individual characteristics, reducing irrelevant marketing outreach. This enhances user satisfaction and trust, fostering positive brand-consumer relationships. Moreover, the model encourages responsible consumption by aligning customers with suitable car options, considering their

financial circumstances. As data-driven decisions become more prevalent, it encourages an informed approach to car purchases, promoting financial prudence among consumers.

Business Impact:

On the business front, the car purchase prediction ML model yields substantial impact. Marketing efforts become laser-focused, targeting individuals with a higher chance of conversion, resulting in enhanced efficiency and cost reduction. The model facilitates improved resource allocation, optimizing budget allocation for campaigns that promise the highest return on investment. Sales teams can prioritize leads, streamlining the conversion process and potentially increasing sales volume. Over time, the model's accurate predictions contribute to higher customer satisfaction rates and improved brand reputation. As data-driven strategies gain prominence, the business stays ahead in a competitive market, poised for growth and innovation.

Milestone 2: Data Collection and Visualizing and analyzing the data

ML depends heavily on data, It is most crucial aspect that makes algorithm training possible. So this section allows you to download the required dataset.

Activity 1: Download the dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used car_data.csv data. This data is downloaded from kaggle.com. Please refer the link given below to download the dataset.

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

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques.

Note: There is n number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Activity 2: Importing the libraries

Import the necessary libraries as shown in the image. Here we have used visualization style as five thirty eight.

```

import pandas as pd # basic imports

import matplotlib.pyplot as plt # visualization of dataset
plt.style.use('ggplot')
import seaborn as sns # visualization of dataset

import plotly.express as px # for interactive visualisation

import lightgbm as lgb # I will use lightgbm algorithm as it is good model binary predictions

dataset = pd.read_csv(r"C:\Users\abhis\OneDrive\Desktop\CAR_PP_APP\car_data.csv")

dataset.head()

```

Activity 3: Read the Dataset

Our dataset is in .csv

In pandas we have a function called `read_csv()` to read the dataset. As a parameter we have to give the directory of csv file.

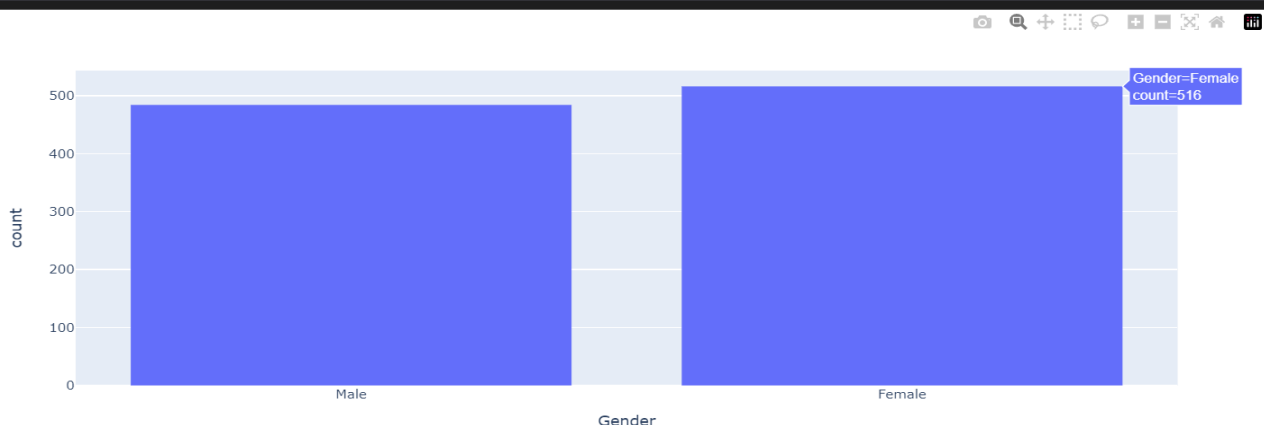
Activity 4: Univariate analysis

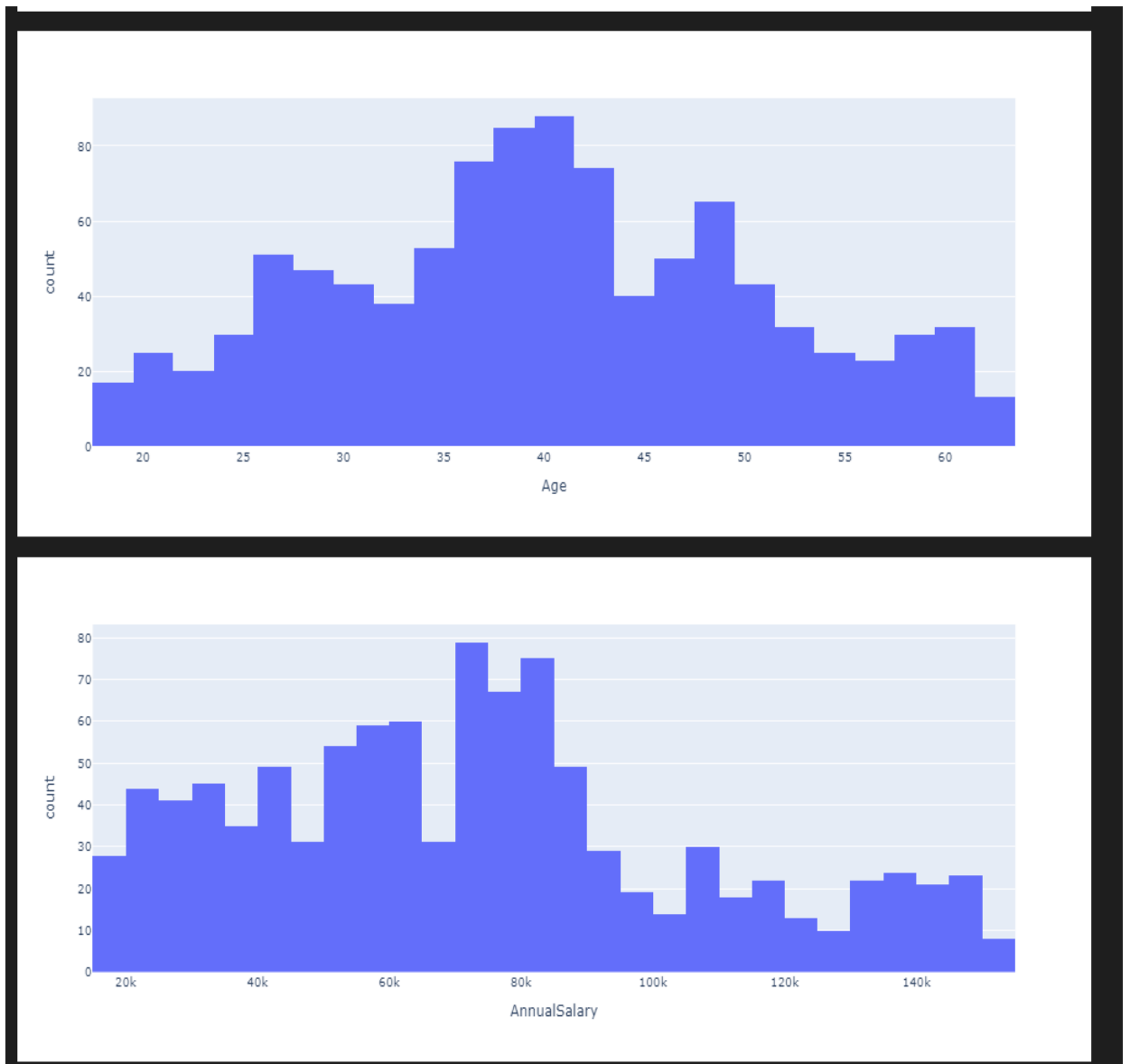
In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as Histplot and countplot.

```

fig = px.histogram(dataset, x='Gender')
fig.show()
fig = px.histogram(dataset, x='Age')
fig.show()
fig = px.histogram(dataset, x='AnnualSalary')
fig.show()

```

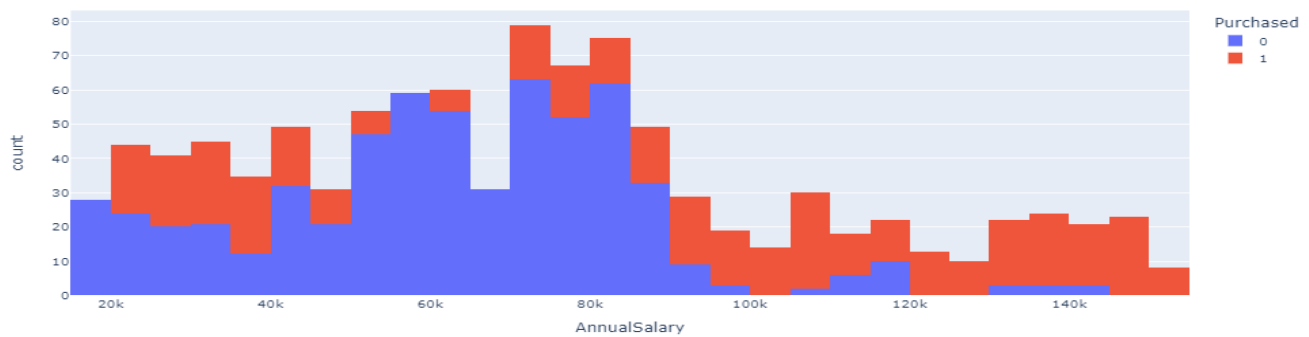
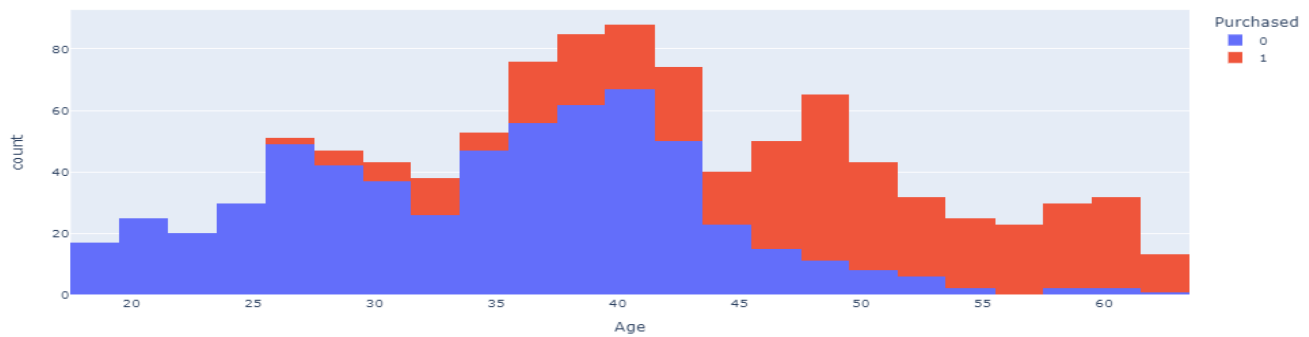
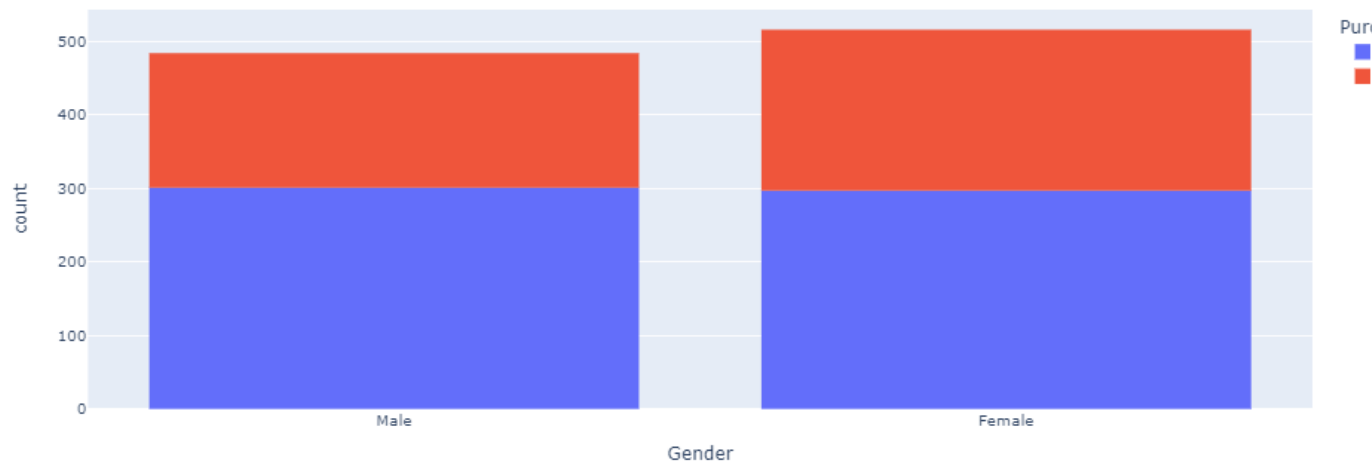




Activity 5: Bivariate analysis

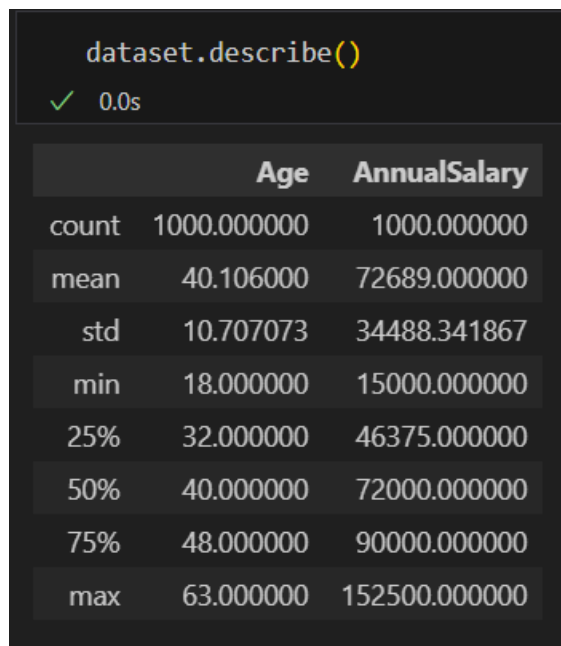
To find the relation between two features we use bivariate analysis. Here we are visualizing the relationships

```
fig = px.histogram(dataset, x='Gender', color='Purchased')
fig.show()
fig = px.histogram(dataset, x='Age', color='Purchased')
fig.show()
fig = px.histogram(dataset, x='AnnualSalary', color='Purchased')
fig.show()
```



Activity 6: Descriptive analysis

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.



```
dataset.describe()
```

✓ 0.0s

	Age	AnnualSalary
count	1000.000000	1000.000000
mean	40.106000	72689.000000
std	10.707073	34488.341867
min	18.000000	15000.000000
25%	32.000000	46375.000000
50%	40.000000	72000.000000
75%	48.000000	90000.000000
max	63.000000	152500.000000

Milestone 3: Data Pre-processing

As we have understood how the data is let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much of randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

- 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

- Let's find the shape of our dataset first, To find the shape of our data, df.shape method is used. To find the data type, df.info() function is used.

```
dataset.info()
✓ 0.0s

<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   object
dtypes: int64(3), object(2)
memory usage: 39.2+ KB
```

- 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.

```
dataset.isnull().any()
✓ 0.0s

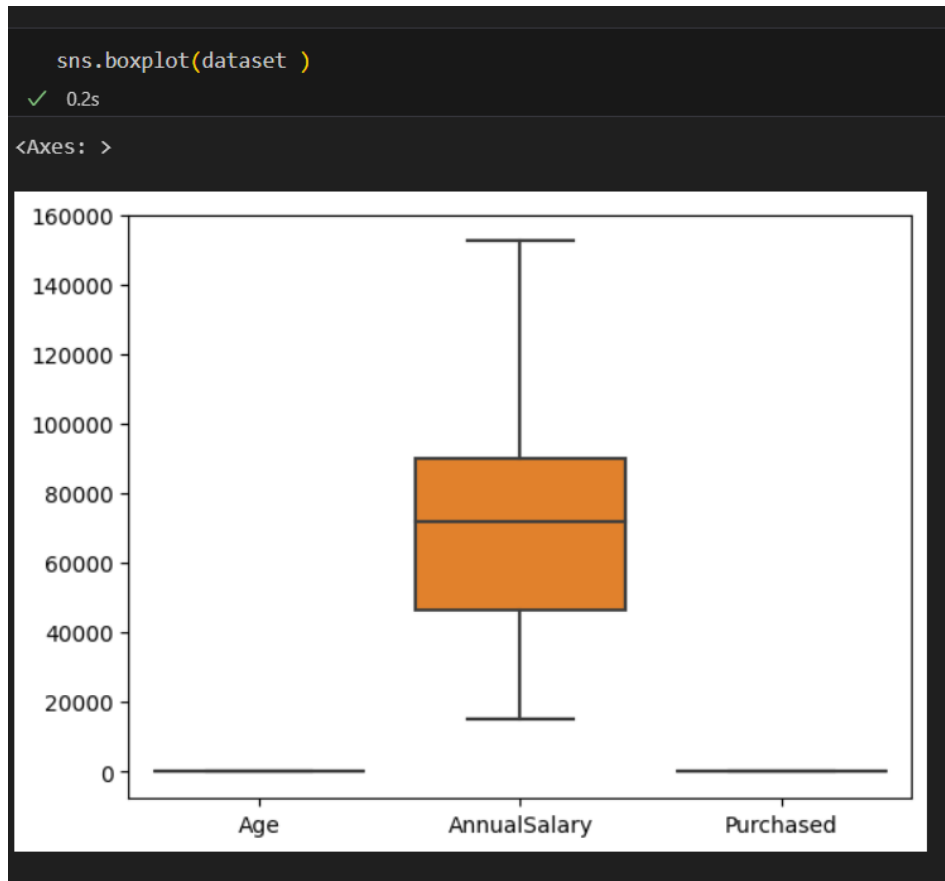
User ID      False
Gender       False
Age          False
AnnualSalary False
Purchased    False
dtype: bool
```

Let's look for any outliers in the dataset

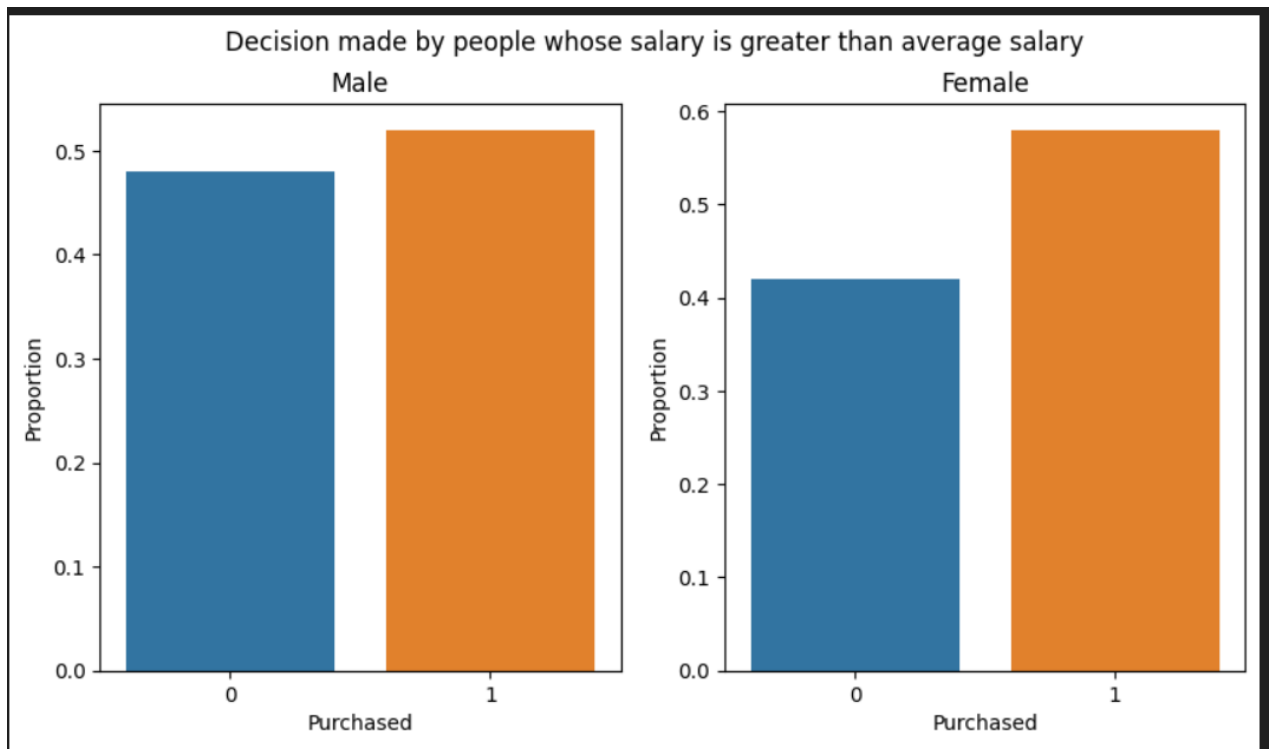
Activity 2: Handling outliers

With the help of boxplot, outliers are visualized. And here we are going to find upper bound and lower bound of all features with some mathematical formula.

- From the below diagram, we could visualize that Monetary feature has outliers. Boxplot from seaborn library is used here.



Activity 3: Decision made by people whose salary is greater than average salary



Activity 4: Splitting data into train and test

Now let's split the Dataset into train and test sets. First split the dataset into x and y and then split the data set

Here x and y variables are created. On x variable, df is passed with dropping the target variable. And on y target variable is passed. For splitting training and testing data we are using `train_test_split()` function from sklearn. As parameters, we are passing x, y, test_size, random_state.

```
from sklearn.model_selection import train_test_split

# Splitting your dataset into features (X) and labels (y)
x = dataset.drop('Purchased', axis=1) # Features
y = dataset['Purchased'] # Labels

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=101)
```

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 four classification algorithms. The best model is GBM & LIGHT GBM

Gradient boosting is a [machine learning](#) technique used in [regression](#) and [classification](#) tasks, among others. It gives a prediction model in the form of an [ensemble](#) of weak prediction models, i.e., models that make very few assumptions about the data, which are typically simple [decision trees](#).^{[1][2]} When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms [random forest](#)

Light GBM, short for light gradient-boosting machine, is a [free and open-source](#) distributed [gradient-boosting](#) framework for [machine learning](#), originally developed by [Microsoft](#).^{[4][5]} It is based on [decision tree](#) algorithms and used for [ranking](#), [classification](#) and other machine learning tasks. The development focus is on performance and scalability.

```
# Create our model
carp = lgb.LGBMRegressor(n_estimators=50,
                          learning_rate=0.05,
                          objective='binary',
                          metric=['binary_logloss'],
                          importance_type='gain'
                          )
```

```
# Fit our model
carp.fit(x_train, y_train,
        eval_set=(x_test, y_test))
```

LightGBM [Info] Number of positive: 330, number of negative: 470

```
LGBMRegressor
LGBMRegressor(importance_type='gain', learning_rate=0.05,
              metric=['binary_logloss'], n_estimators=50, objective='binary')
```

```
y_pred = carp.predict(x_test) ❗
```

7]

```
# MAKE Y_PRED AS 0 OR 1
threshold = 0.5
y_pred = (y_pred >= threshold).astype(int)
print(y_pred)
```

```
• from sklearn.metrics import accuracy_score

# Calculate the accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print the accuracy
print(f"Accuracy: {accuracy:.3f}")
```

Accuracy: 0.920

Our model is performing well. So, we are saving the model by `pickle.dump()`.

```
import pickle  
  
pickle.dump(carp,open('model.pkl','wb'))  
model=pickle.load(open('model.pkl','rb'))
```

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

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 uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

- Building HTML Pages
- serverside script

Activity1: Building Html Pages:

For this project create HTML files and save them in Templates folder.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1.0"/>
  <title>Starter Template - Materialize</title>

  <!-- CSS -->
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
  <link href="/static/css/materialize.css" type="text/css" rel="stylesheet" media="screen,projection"/>
  <link href="/static/css2/style.css" type="text/css" rel="stylesheet" media="screen,projection"/>
</head>

<body>
  <nav class="light-blue lighten-1" role="navigation">
    <div class="nav-wrapper container"><a id="logo-container" href="#" class="brand-logo">Logo</a>
    <ul class="right hide-on-med-and-down">
      <li><a href="#">Navbar Link</a></li>
    </ul>

    <ul id="nav-mobile" class="sidenav">
      <li><a href="#">Navbar Link</a></li>
    </ul>
    <a href="#" data-target="nav-mobile" class="sidenav-trigger"><i class="material-icons">menu</i></a>
    </div>
  </nav>

  <div class="section no-pad-bot" id="index-banner">
    <div class="container">
      <br><br>
      <h1 class="header center orange-text">Car Purchase Prediction</h1>
      <div class="row center">
        <h5 class="header col s12 light">Predict the probability of car purchase based on the age , annnual salary and gender of the pote
        <br>
        </h5>
      </div>
    </div>

    <div class="row">
      <div class="col s12">
        <form action="/predict" method="POST" class="col s12">
          <div class="row">
            <div class="input-field col s4">
              <label for="first_name"><b>Age</b></label>
              <br>
              <input id="first_name" name="Age" type="text" class="validate">
            </div>
            <div class="input-field col s4">
              <label for="last_name"><b>AnnualSalary </b></label>
              <br>
              <input id="last_name" name="AnnualSalary" type="text" class="validate">
            </div>
            <div class="input-field col s4">
              <label for="_name"><b>Gender_Female</b></label>
              <br>
              <input id="_name" name="Gender_Female" type="text" class="validate">
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
```

```

        <br>
        <input id="_name" name="Gender_Male" type="text" class="validate">
    </div>

</div>

<div class="row center">
    <button type="submit" class="btn-large waves-effect waves-light orange">Predict Probability</button>
</div>
</form>
</div>

<br><h4 align="center"><b>{{pred}}</b></h4>
    <br>
</div>
</div>

<div class="footer-copyright">
    <div class="container">
        Made by <a class="orange-text text-lighten-3" href="http://materializecss.com">Materialize</a>
    </div>
</div>
</footer>

<!-- Scripts-->
<script src="https://code.jquery.com/jquery-2.1.1.min.js"></script>
<script src=".js/materialize.js"></script>
<script src="js/init.js"></script>

</body>
</html>

```

Activity 2: Build Python code:

Import the libraries

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.

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

app = Flask(__name__)

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

Render HTML

```
@app.route('/')
def hello_world():
    return render_template("car_purchase.html")

@app.route('/predict', methods=['POST', 'GET'])
def predict():
    # Extracting the input features from the form
    Age = int(request.form['Age'])
    AnnualSalary = int(request.form['AnnualSalary'])
    Gender_Male = int(request.form['Gender_Male'])
    Gender_Female = int(request.form['Gender_Female'])

    # Create a numpy array for prediction
    input_features = np.array([[Age, AnnualSalary, Gender_Male, Gender_Female]])
```

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict()

```
# Make a prediction using the loaded model
prediction = model.predict(input_features)

# Assuming prediction is a 1D array, you can access the first element directly
probability_of_purchase = prediction[0]

if probability_of_purchase > 0.5:
    return render_template('car_purchase.html', pred=f'Probability of this person purchasing a car is {probability_of_purchase:.2f}',
else:
    return render_template('car_purchase.html', pred=f'Probability of this person purchasing a car is {probability_of_purchase:.2f}',

if __name__ == '__main__':
    app.run(debug=True)
```

This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the carprediction.html page earlier.

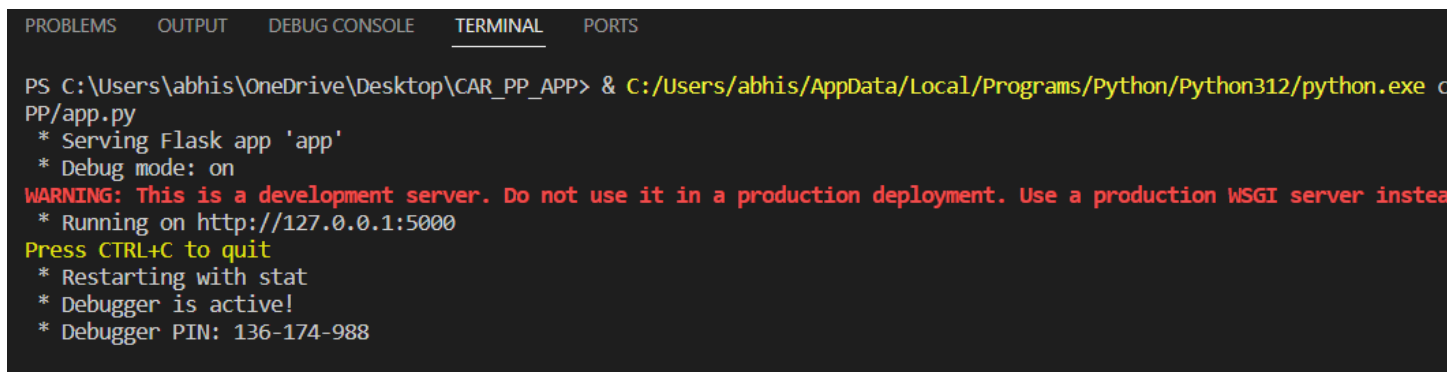
Main Function:

```
if __name__ == '__main__':  
    app.run(debug=True)
```

Activity 3: Run the application

Open Visual studio code and Import all the project folders.

When you run the app.py file



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  
  
PS C:\Users\abhis\OneDrive\Desktop\CAR_PP_APP> & C:/Users/abhis/AppData/Local/Programs/Python/Python312/python.exe c  
PP/app.py  
* Serving Flask app 'app'  
* Debug mode: on  
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead  
* Running on http://127.0.0.1:5000  
Press CTRL+C to quit  
* Restarting with stat  
* Debugger is active!  
* Debugger PIN: 136-174-988
```

and click on the server url in terminal, you will be redirected to homepage. The home page will look like:

Enter the customer details and click on predict button, the output looks like:

Car Purchase Prediction

Predict the probability of car purchase based on the age , annnual salary and gender of the potentil customer

Age

AnnualSalary

Gender_Female

Gender_Male

PREDICT PROBABILITY

PREDICT PROBABILITY

Probability of this person purchasing a car is 0.94

Made by [Materialize](#)

Conclusion:

In conclusion, the developed customer car purchase prediction model proves its effectiveness in aiding decision-making within the automotive industry. Leveraging customer age and salary as predictive features, the model offers valuable insights into purchase behaviors. Its accurate predictions empower

businesses to tailor marketing strategies for targeted customer segments, optimizing resource allocation. By enabling personalized interactions, the model enhances user experience and engagement. Its integration into a user-friendly web interface simplifies access, making it an invaluable tool for both customers and dealerships. The model not only streamlines sales efforts but also drives customer satisfaction through informed choices. As a pivotal advancement in the automotive landscape, this model marks a significant step toward data-driven success.