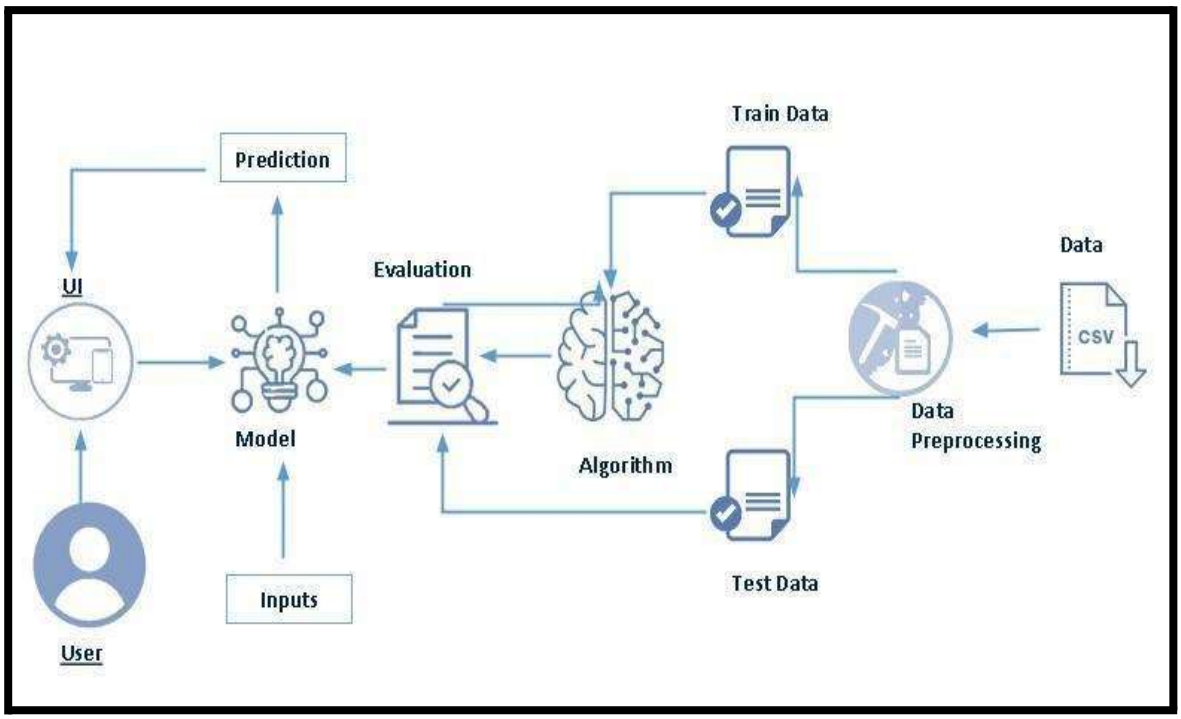
Car purchase Prediction Using ML

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

- **Command Promt navigator and pycharm:**
  - Refer the link below to download anaconda navigator
  - Link : https://youtu.be/1ra4zH2G4o0
- **Python packages:**
  - Open command 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 pickle-mixin" 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

    - RANDOM FOREST : https://www.javatpoint.com/machine-learning-random-forest-algorithm
    - Decision Tree: https://www.analyticsvidhya.com/blog/2022/03/decision-tree-machine-learning-using-python/
    - Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
- **Flask Basics** : https://www.youtube.com/watch?v=lj4I_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

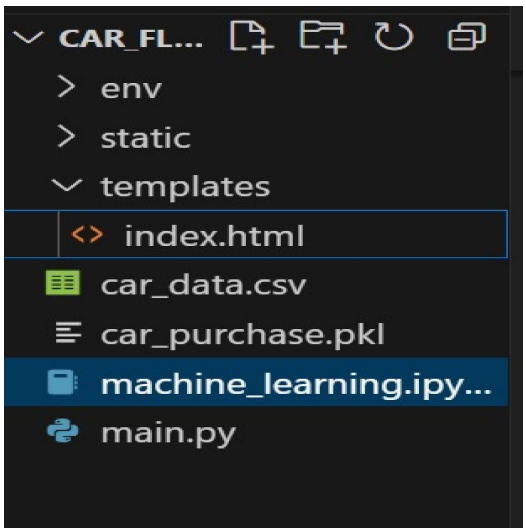
To accomplish this, we have to complete all the activities listed below,

- Data collection
    - Collect the dataset or create the dataset
- 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

Here are some potential business requirements for Share price predictor.

Data Collection and Integration: Gather and integrate a comprehensive dataset that includes customer age and income information, ensuring data accuracy and consistency.

Integration with Marketing Strategies: The model should seamlessly 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 (Student Will Write)

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.

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

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
import numpy as np
import scipy.stats as stats

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.metrics import accuracy_score, roc_auc_score, classification_report, confusion_matrix
from sklearn.model_selection import GridSearchCV
```
✓ 0.0s

Activity 3: Read the Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

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.

```
dataset = pd.read_csv('C:/Users/UdayA/Documents/Projects_SB/Car-Purchase_Decision/Data/car_data.csv')
print(dataset.shape)

dataset.head()
```
✓ 0.0s

```
(1000, 5)
```

	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

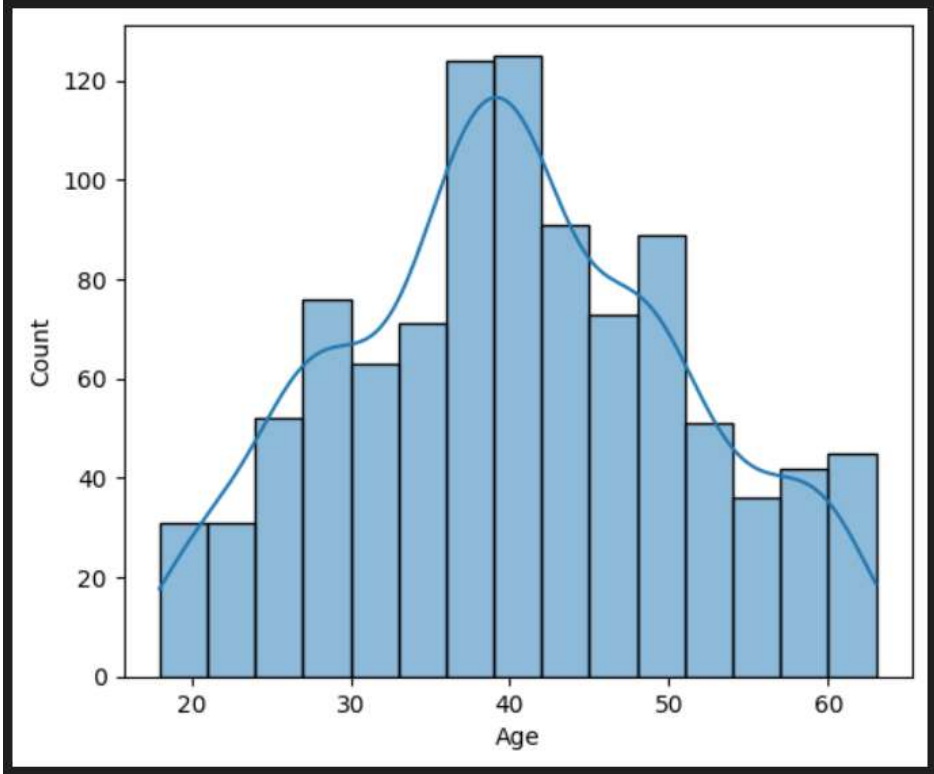
Activity 4: Univariate analysis

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

- Seaborn package provides a wonderful function distplot. With the help of distplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.

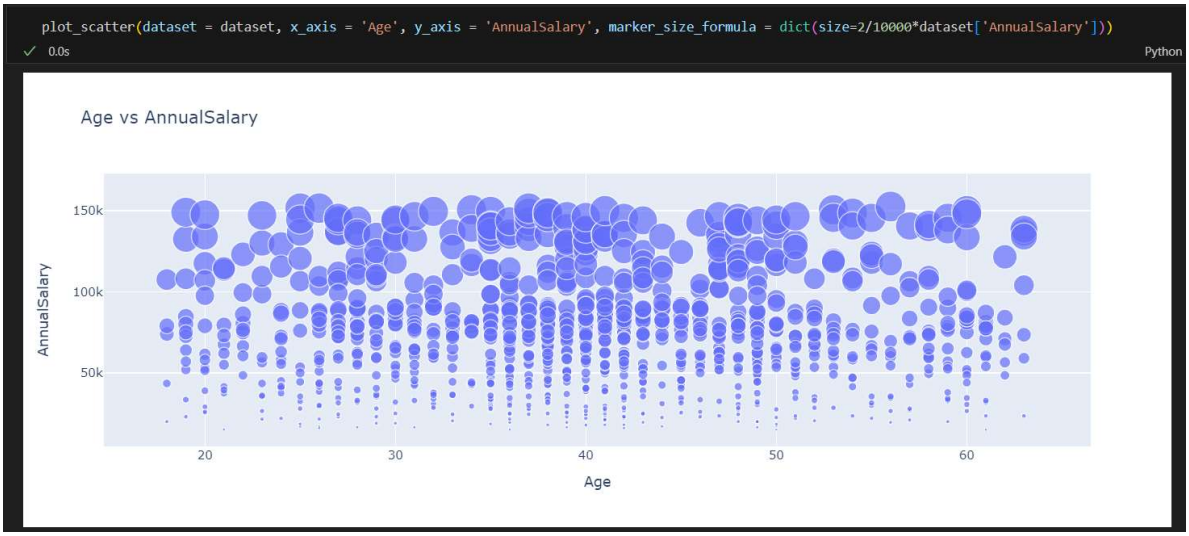
```
fig = plt.figure(figsize = (6, 5))
sns.histplot(data = dataset, x = 'Age', kde = True)

plt.show()
```

✓ 0.2s



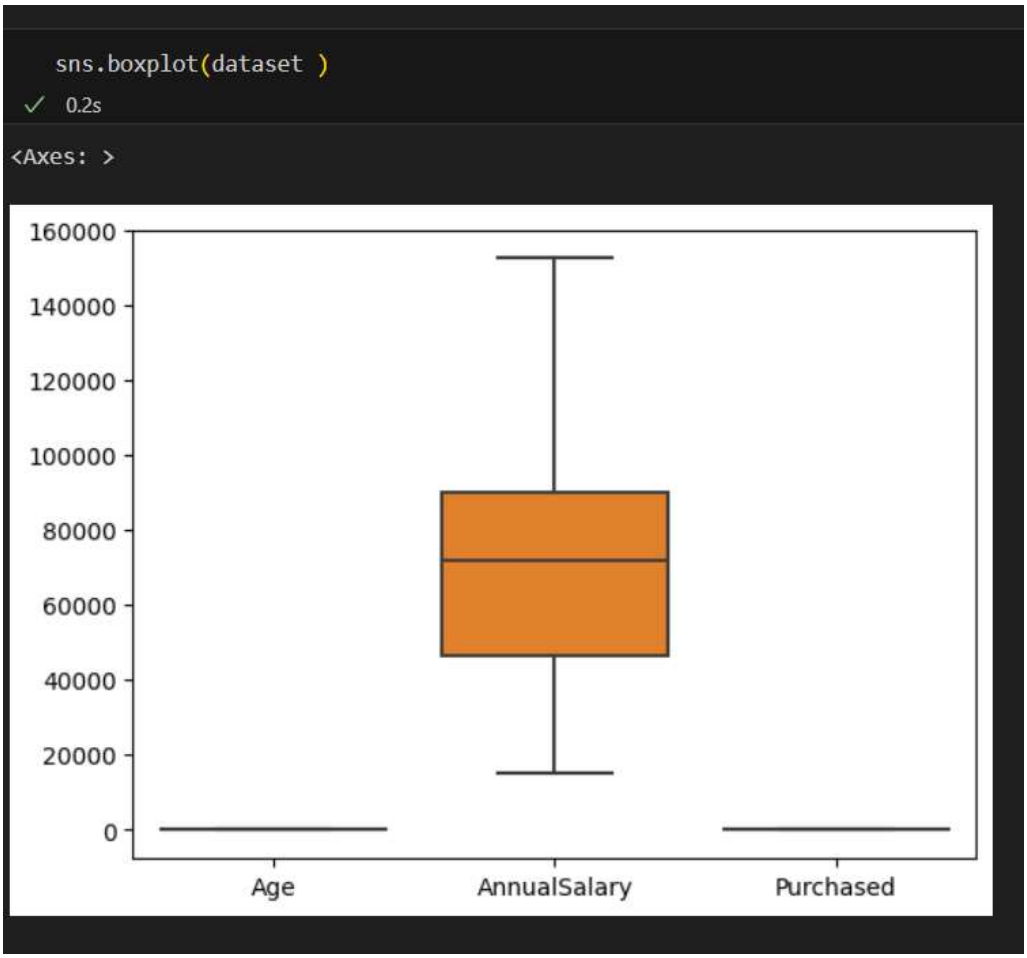
Activity 5: Bivariate analysis

To find the relation between two features we use bivariate analysis. Here we are visualizing the relationship between ‘Age’ and ‘Annualsalary’ variables using scatterplot.



Activity 6: Multivariate analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used boxplot from seaborn package.



Activity 7: 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 lets 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.

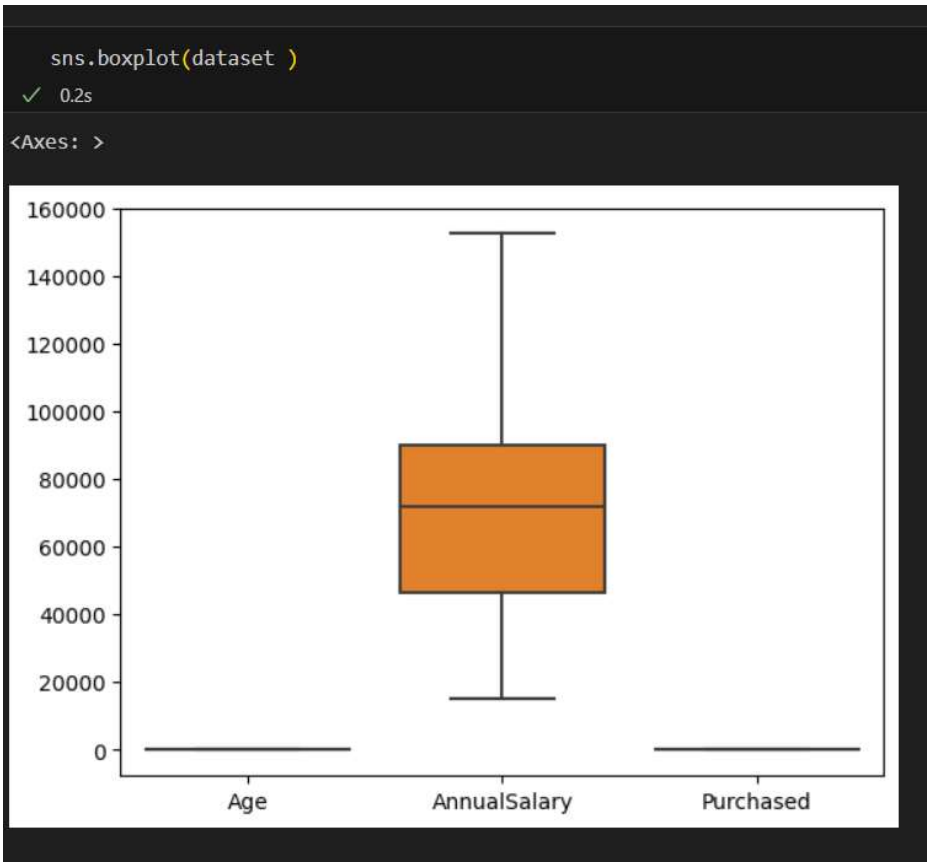
```
dataset.isnull().any()
✓ 0.0s
User ID         False
Gender          False
Age             False
AnnualSalary    False
Purchased       False
dtype: bool
```

Let's look for any outliers in the dataset

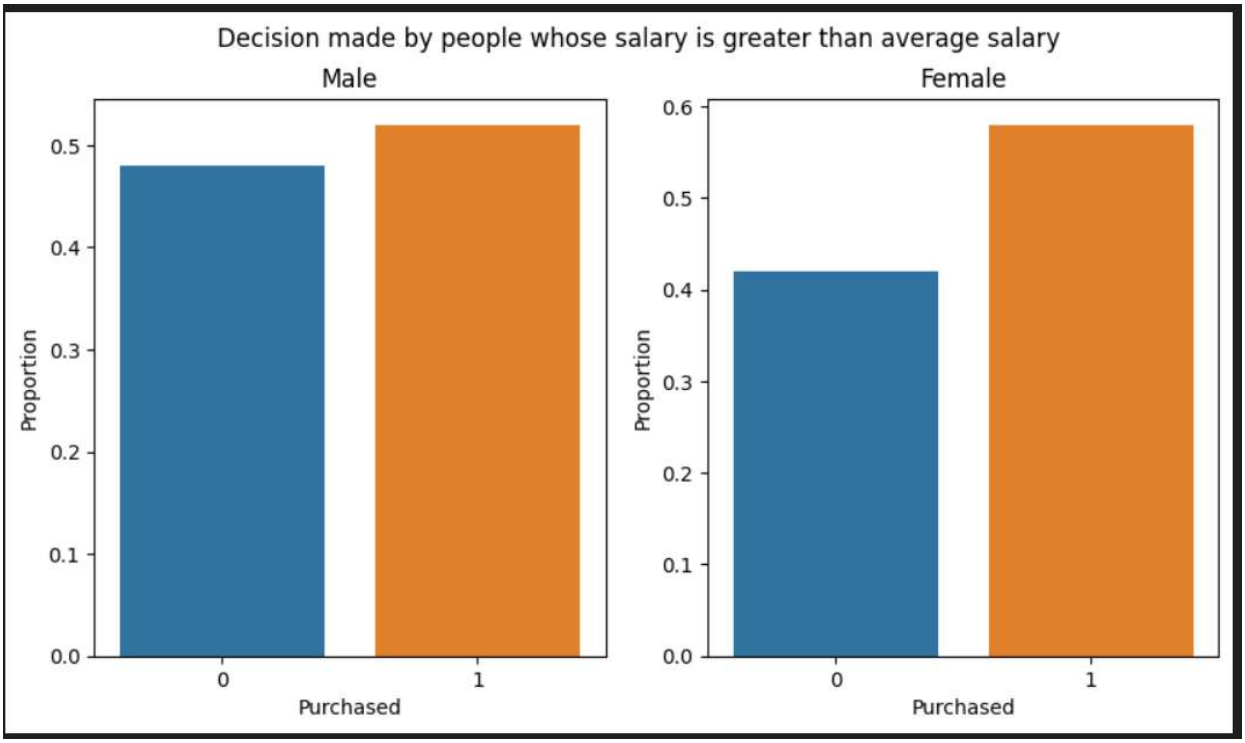
Activity 2: Handling outliers

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

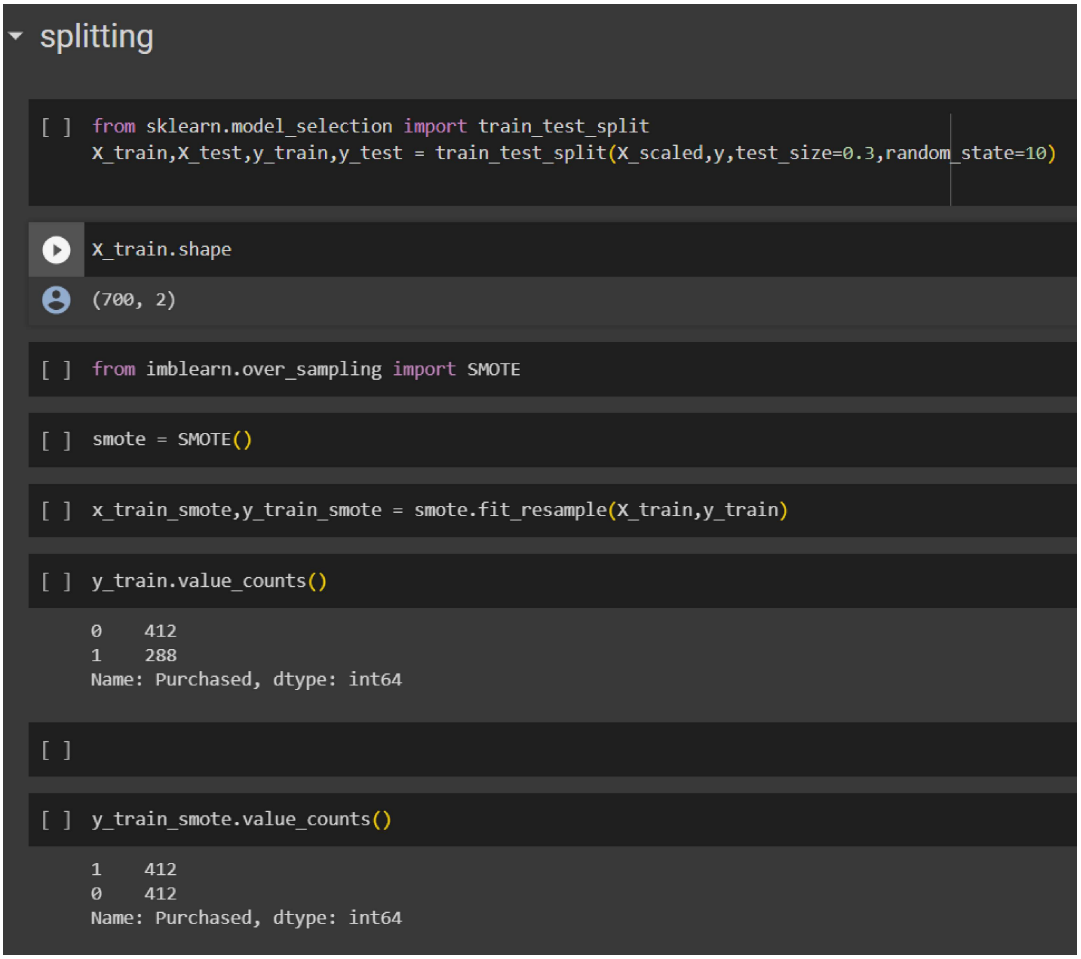


Activity 4: Splitting data into train and test

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

Here x and y variables are created. On x variable, df is passed with dropping the target variable. Andon 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.



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 saved based on its performance.

Activity : Random Forest

model building

```
[ ] from sklearn.ensemble import RandomForestClassifier
    model2 =RandomForestClassifier(criterion='entropy')
```

```
[ ] model2.fit(x_train_smote,y_train_smote)
```

```
RandomForestClassifier
RandomForestClassifier(criterion='entropy')
```

```
n_estimators = [int(x) for x in np.linspace(start = 10, stop = 80, num = 10)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [2,4]
# Minimum number of samples required to split a node
min_samples_split = [2, 5]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2]
# Method of selecting samples for training each tree
bootstrap = [True, False]
```

```
param_grid = {'n_estimators': n_estimators,
              'max_features': max_features,
              'max_depth': max_depth,
              'min_samples_split': min_samples_split,
              'min_samples_leaf': min_samples_leaf,
              'bootstrap': bootstrap}
print(param_grid)
```

```
{'n_estimators': [10, 17, 25, 33, 41, 48, 56, 64, 72, 80], 'max_features': ['auto', 'sqrt'], 'max_depth': [2, 4], 'min_samples_split': [2, 5], 'min_samples_leaf': [1, 2], 'bootstrap':
```

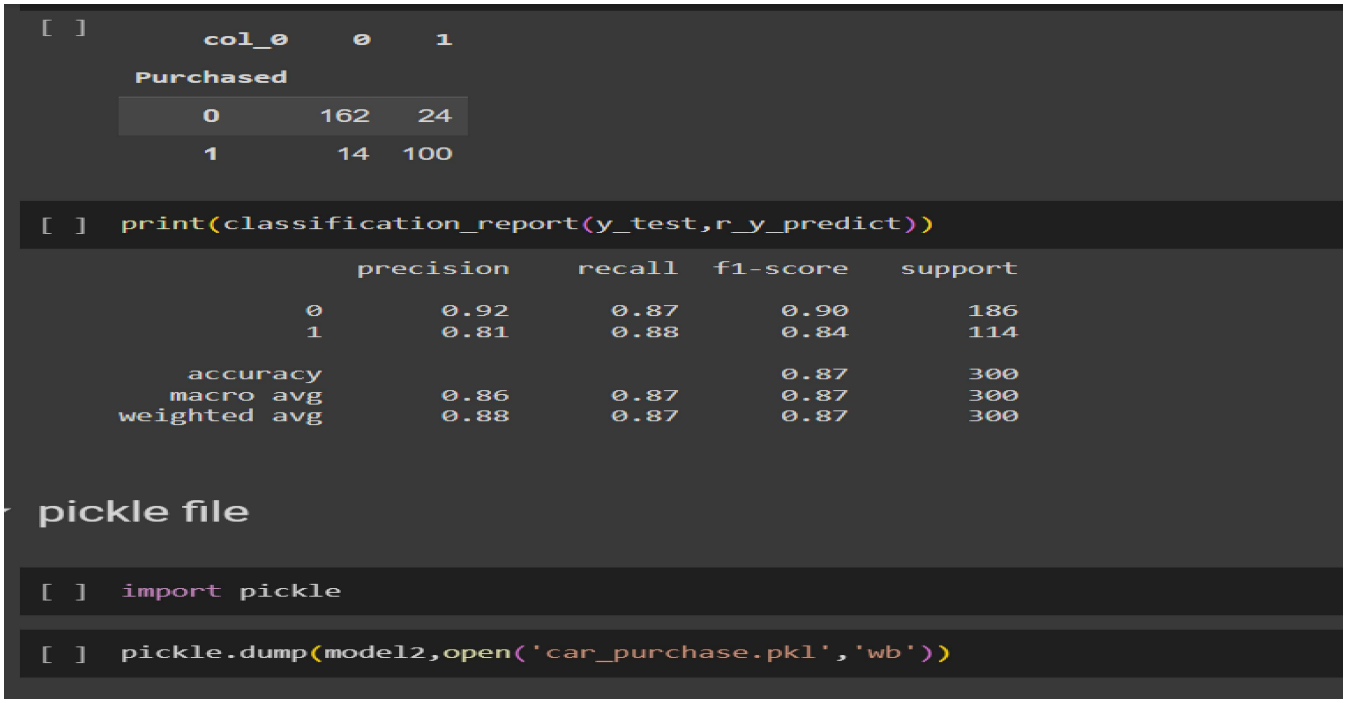
```
[ ] from sklearn.model_selection import GridSearchCV
    rf_Grid = GridSearchCV(estimator = model2, param_grid = param_grid, cv = 3, verbose=2, n_jobs = 4)
```

```
[ ] rf_Grid.fit(x_train_smote, y_train_smote)
```

```
Fitting 3 folds for each of 320 candidates, totalling 960 fits
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

```
rf_Grid.best_params_
```

```
{'bootstrap': True,
 'max_depth': 2,
 'max_features': 'sqrt',
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'n_estimators': 10}
```



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

Activity: Building Html Pages:

For this project create three HTML files namely

- home.html
- about.html
- carprediction.html

and save them in Templates folder.

main.py | machine_learning.ipynb | index.html

templates > index.html > html > h1

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Background Image</title>
    <style type="text/css">
    body{
        Background-image:url('carphoto1.jpeg');

    }
</style>
</head>

    <h1>

        <center>
            <form action="/predict"method="post">
                <label for="Age">Age:</label><br>
                <input type="text"id="Age"name="Age"placeholder="Age"required="required"/><br><br>
                <label for="Annualsalary">Annualsalary:</label><br>
                <input type="text"id="Annualsalary"name="Annualsalary"placeholder="Annualsalary"required="required"/><br><br>

                <button type="submit" class="btn btn-primary btn-block">predict</button>
                </form>
                <br>
        </center>
            </h1>
    <h2><b>{{y}}</b></h2>

</body>
</html>
```

Activity 2: Build Python code:

Import the libraries

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

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.

```
app=Flask(__name__)

model=pickle.load(open("C:/Users/UdayA/Documents/Projects_SB/Car-Purchase_Decision/model.pkl",'rb'))
```

Render HTML page:

```
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.

In the above example, '/' URL is bound with home.html function. Hence, when the home page of the web server is opened in browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

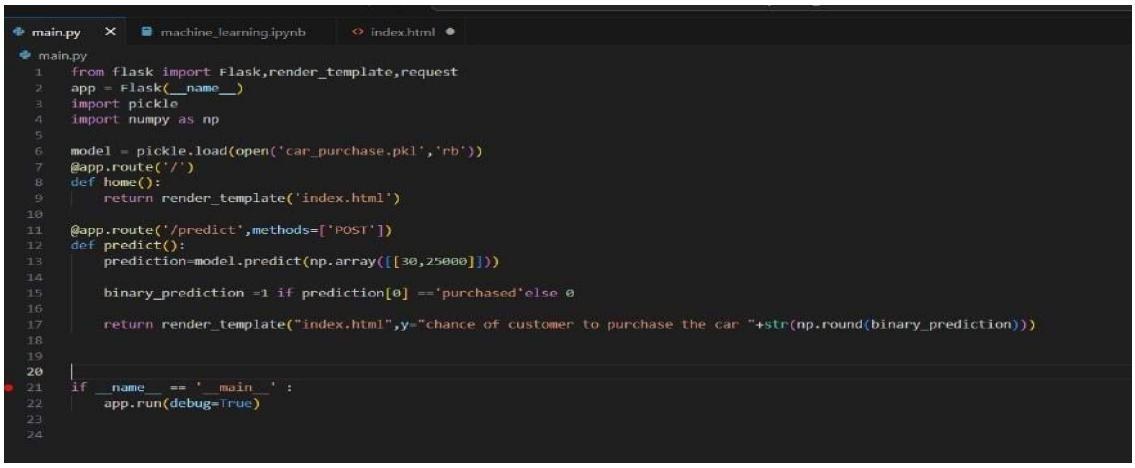
Retrieves the value from UI:

```
@app.route('/predict',methods=['POST'])
def predict():
    # Make a prediction
    prediction = model.predict(np.array([[30, 25000]]))

    # Convert class label to numeric value
    binary_prediction = 1 if prediction[0] == 'Purchased' else 0

    return render_template('carprediction.html', prediction_text='Chance of customer to purchase the car is {}'.format(binary_prediction)
```

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()

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

Main Function:



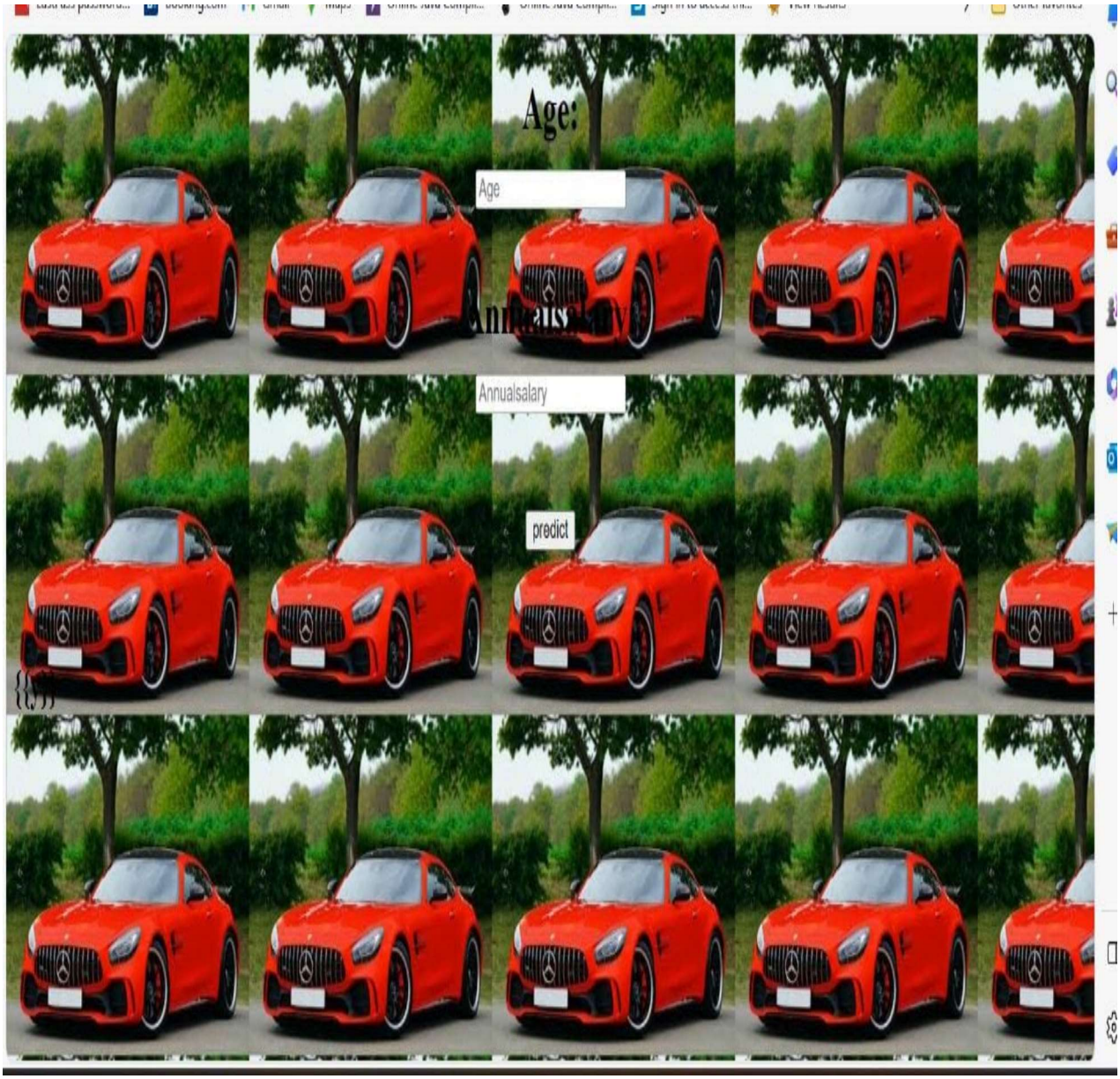
Activity 3: Run the application

Open Visual studio code and Import all the project folders.

When you run the app.py file and click on the server url in terminal, you will redirected to home page. The home page will looks like:

If you click on About option, you will redirected to about.html page and it looks like:

Then click on 'CarPrediction' option from top left corner, you will redirected to carprediction. html page. Enter the customer details and click on predict button, the output looks like:



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.