Project Report Format

TITLE: CAR PURCHASE PREDICTION USING MACHINE LEARNING

1. INTRODUCTION

1.1 Project Overview

The automotive industry is rapidly evolving, with a plethora of car models and features available to consumers. Choosing the right car that aligns with individual preferences and requirements can be a daunting task. This project aims to develop a machine learning-based system to assist users in making informed decisions when purchasing a car.

Objectives:

Developed a predictive model to recommend cars based on user preferences.

Utilized machine learning algorithms to analyze historical data and user inputs.

Provided a user-friendly interface for seamless interaction.

Methodology:

Data Collection:

Gathered a diverse dataset containing information on various car models, specifications, prices, user reviews, and historical sales data.

Integrated external data sources to enhance the model's accuracy and scope.

Data Preprocessing:

Cleansed and preprocessed the collected data to handle missing values, outliers, and ensure consistency.

Featured engineering to extract relevant information and create meaningful attributes for the model.

Machine Learning Model:

Selected appropriate machine learning algorithms for recommendation systems andTrained the model using the preprocessed data, considering factors such as user preferences, budget, fuel efficiency, etc.

1.2 Purpose

The purpose of car purchase prediction using machine learning is to leverage data-driven insights and algorithms to assist individuals in making informed decisions when buying a car. Here are several key purposes and benefits of implementing car purchase prediction using machine learning:

Data-Driven Decision Making:

Utilizing historical data, user reviews, and specifications to make predictions.

Enabling users to make decisions based on a combination of empirical data and personal preferences.

Financial Considerations:

Helping users make budget-conscious decisions by recommending cars that align with their financial constraints.

Offering insights into long-term costs, including fuel efficiency and maintenance expenses.

Market Trends and Insights:

Analyzing market trends and user behavior to stay updated on the latest preferences and emerging features.
Providing valuable insights to manufacturers and dealerships to adapt their offerings to consumer demands.

Marketing and Sales Optimization:

Assisting dealerships and manufacturers in optimizing their marketing strategies based on user preferences and purchasing patterns.
Improving sales processes by aligning product offerings with customer demand

2. LITERATURE SURVEY

2.1 Existing problem

While car purchase prediction using machine learning holds great promise, there are some challenges and existing problems associated with implementing such systems. Here are several issues that researchers and developers often encounter:

Data Quality and Availability:

Limited availability of high-quality, diverse, and up-to-date data can hinder the performance of machine learning models.
Incomplete or inaccurate data can lead to biased recommendations and reduced accuracy.
Complex Decision Factors:

Car purchase decisions involve a multitude of factors, such as personal preferences, lifestyle, budget, and evolving technology trends. Capturing and weighing these factors accurately is challenging.

Cold Start Problem:

When a user is new to the system, or there is limited historical data for a user, the model may struggle to provide accurate recommendations, leading to the "cold start" problem.
Scalability Issues:

Scalability can be a concern, especially when dealing with a large number of users and an extensive database of car models. Efficiently handling the growing complexity of recommendation systems is crucial.
Privacy Concerns:

Users may be hesitant to share personal information, preferences, and behaviors due to privacy concerns. Striking a balance between personalization and respecting user privacy is a constant challenge.

Bias and Fairness:

Models trained on biased data can perpetuate and amplify existing biases, leading to unfair recommendations. Ensuring fairness and mitigating biases in the recommendation process is an ongoing concern.

Interpretability:

Interpretable models are essential for user trust and understanding. Complex models may lack interpretability, making it challenging to explain recommendations to users.
User Engagement:

Encouraging users to actively engage with the recommendation system and provide feedback is a continual challenge. A lack of user engagement can limit the system's ability to learn and improve.

2.2 References

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d

2.3 Problem Statement Definition

Leading to a time-consuming and sometimes overwhelming decision-making process. Existing methods lack the personalization required to align recommendations with individual preferences, and the dynamic nature of the automotive market introduces further complexities. To address these issues, there is a need for a robust machine learning-based car purchase prediction system that can provide personalized recommendations, streamline decision-making, and adapt to changing user preferences and market trends.

Key Challenges and Opportunities:
Information Overload: The sheer volume of information on various car models, features, and specifications makes it challenging for users to make well-informed decisions.

Subjectivity of Preferences: Individual preferences vary widely, and existing systems struggle to accurately capture and adapt to the subjective nature of user preferences.

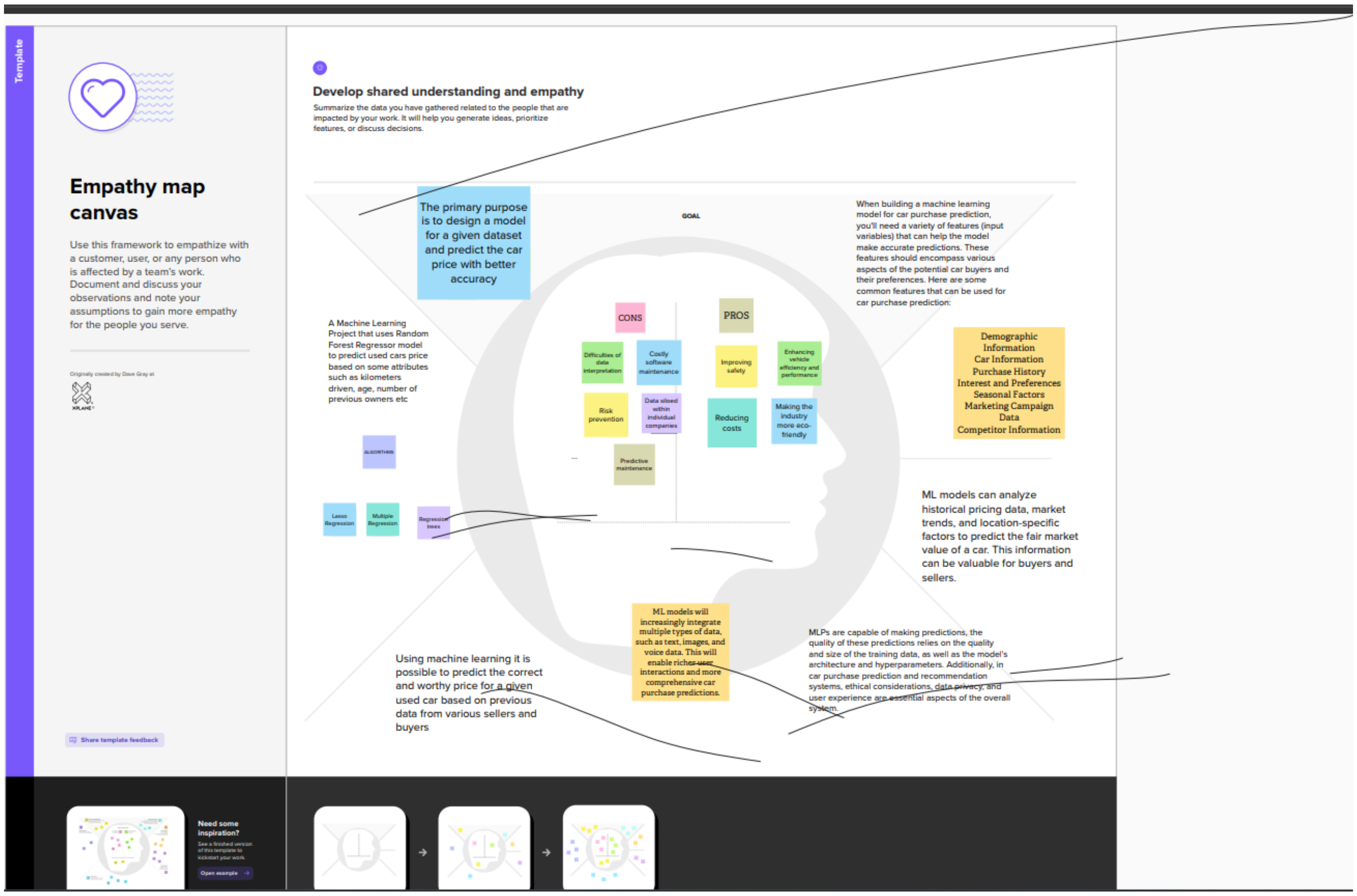
Dynamic Market Trends: The automotive industry is dynamic, with new models and features regularly introduced. Existing systems may fail to quickly adapt to these changes, resulting in outdated recommendations.

Lack of Personalization: Generic recommendations do not cater to the diverse needs and preferences of individual users, leading to a suboptimal user experience.

User Engagement: Encouraging users to actively engage with the recommendation system and provide feedback is a challenge, hindering the system's ability to learn and improve over time

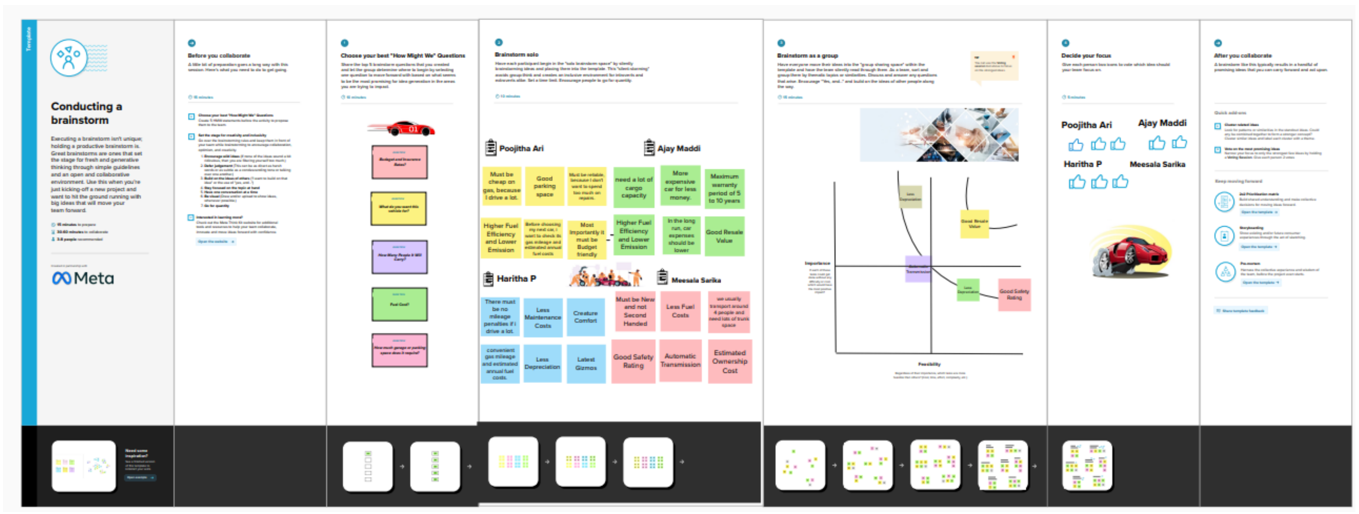
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



Template
Empathy map canvas
Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.
Develop shared understanding and empathy
Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.
GOAL
The primary purpose is to design a model for a given dataset and predict the car price with better accuracy
When building a machine learning model for car purchase prediction, you'll need a variety of features (input variables) that can help the model make accurate predictions. These features should encompass various aspects of the potential car buyers and their preferences. Here are some common features that can be used for car purchase prediction:
A Machine Learning Project that uses Random Forest Regressor model to predict used cars price based on some attributes such as kilometers driven, age, number of previous owners etc
CONS
PROS
Difficulties of data interpretation
Costly software maintenance
Improving safety
Enhancing vehicle efficiency and performance
Risk prevention
Data siloed within individual companies
Reducing costs
Making the industry more eco-friendly
Predictive maintenance
Demographic Information
Car Information
Purchase History
Interest and Preferences
Seasonal Factors
Marketing Campaign Data
Competitor Information
ALGORITHMS
Lasso Regression
Multiple Regression
Regression trees
ML models can analyze historical pricing data, market trends, and location-specific factors to predict the fair market value of a car. This information can be valuable for buyers and sellers.
ML models will increasingly integrate multiple types of data, such as text, images, and voice data. This will enable richer user interactions and more comprehensive car purchase predictions.
Using machine learning it is possible to predict the correct and worthy price for a given used car based on previous data from various sellers and buyers
MLPs are capable of making predictions, the quality of these predictions relies on the quality and size of the training data, as well as the model's architecture and hyperparameters. Additionally, in car purchase prediction and recommendation systems, ethical considerations, data privacy, and user experience are essential aspects of the overall system.
Share template feedback
Need some inspiration?
Open example

3.2 Ideation & Brainstorming



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Functional requirements for a car purchase prediction system using machine learning outline the specific features and capabilities that the system must possess to meet its objectives. These requirements help define the behavior and functionality of the system. Below are some functional requirements for a car purchase prediction system:

User Registration and Profile Management

Users should be able to create accounts and maintain profiles.

The system should store and manage user preferences, purchase history, and feedback.

Input Interface:

Provide an intuitive interface for users to input preferences, such as budget constraints, desired features, and brand preferences.

Data Collection and Integration:

Collect and integrate data from diverse sources, including car specifications, prices, user reviews, and historical sales data.

Ensure that the data is regularly updated to reflect changes in the automotive market.

Preprocessing of Data:

Implement data preprocessing techniques to handle missing values, outliers, and ensure data consistency.

Perform feature engineering to extract relevant information for model training.

Machine Learning Model:

Select and implement appropriate machine learning algorithms for recommendation systems

Train the model using historical data, considering factors such as user preferences, budget, fuel efficiency, and market trends.

Integration with External Systems:

Integrate the recommendation system with external databases and APIs to enhance the range and accuracy of information.

Scalability:

Design the system to handle a large number of users and an extensive database of car models efficiently.

4.2 Non-Functional requirements

Non-functional requirements for a car purchase prediction system using machine learning define the qualities and characteristics that are essential for the system's overall performance, usability, and reliability. Unlike functional requirements that describe what the system should do, non-functional requirements focus on how well the system should perform those functions. Here are some non-functional requirements for the mentioned system:

Performance:

Response Time: The system should provide real-time recommendations with response times of X seconds or less to ensure a seamless user experience.

Scalability: The system should be designed to handle a scalable number of users and a growing database of car models without significant degradation in performance.

Reliability:

Availability: The system should have a high availability rate (e.g., 99.9%) to ensure users can access it whenever needed.

Fault Tolerance: The system should be resilient to failures and able to recover gracefully from any unexpected issues without significant downtime.

Security:

Data Encryption: User data, especially personal and sensitive information, should be encrypted during transmission and storage.

Access Control: Implement strict access controls to ensure that only authorized users can access and modify certain system functionalities and data.

Usability:

User Interface Consistency: Maintain a consistent and intuitive user interface across all platforms to enhance user experience and ease of use.

Accessibility: Ensure the system is accessible to users with disabilities, adhering to relevant accessibility standards.

Compatibility:

Cross-Browser Compatibility: The user interface should be compatible with major web browsers (e.g., Chrome, Firefox, Safari) to ensure a consistent experience.

Device Compatibility: The system should be compatible with various devices, including desktops, laptops, tablets, and smartphones.

Modularity: Design the system with a modular architecture to facilitate easy updates, enhancements, and maintenance.

Documentation: Provide comprehensive documentation for system components, APIs, and data structures to assist future development and troubleshooting.

Flexibility: The system should be flexible enough to accommodate changes in business rules, data sources, and machine learning algorithms.

Adaptability to Market Changes: Ensure the system can quickly adapt to changes in the automotive market, such as new car models and features.

Performance Monitoring and Logging:

Logging: Implement logging mechanisms to capture system events, errors, and user activities for troubleshooting and analysis.

Performance Monitoring: Set up tools to monitor system performance continuously, identifying and addressing bottlenecks proactively.

Data Privacy and Compliance:

Compliance with Data Protection Regulations: Ensure the system complies with relevant data protection regulations (e.g., GDPR, HIPAA) to safeguard user privacy.

Data Retention Policies: Define and adhere to data retention policies to manage the storage and disposal of user data.

Interoperability:

API Compatibility: If applicable, ensure that the system's APIs are compatible with industry standards, facilitating integration with external systems and services.

User Training and Support:

Training Materials: Provide user training materials to assist users in understanding how to use the system effectively.

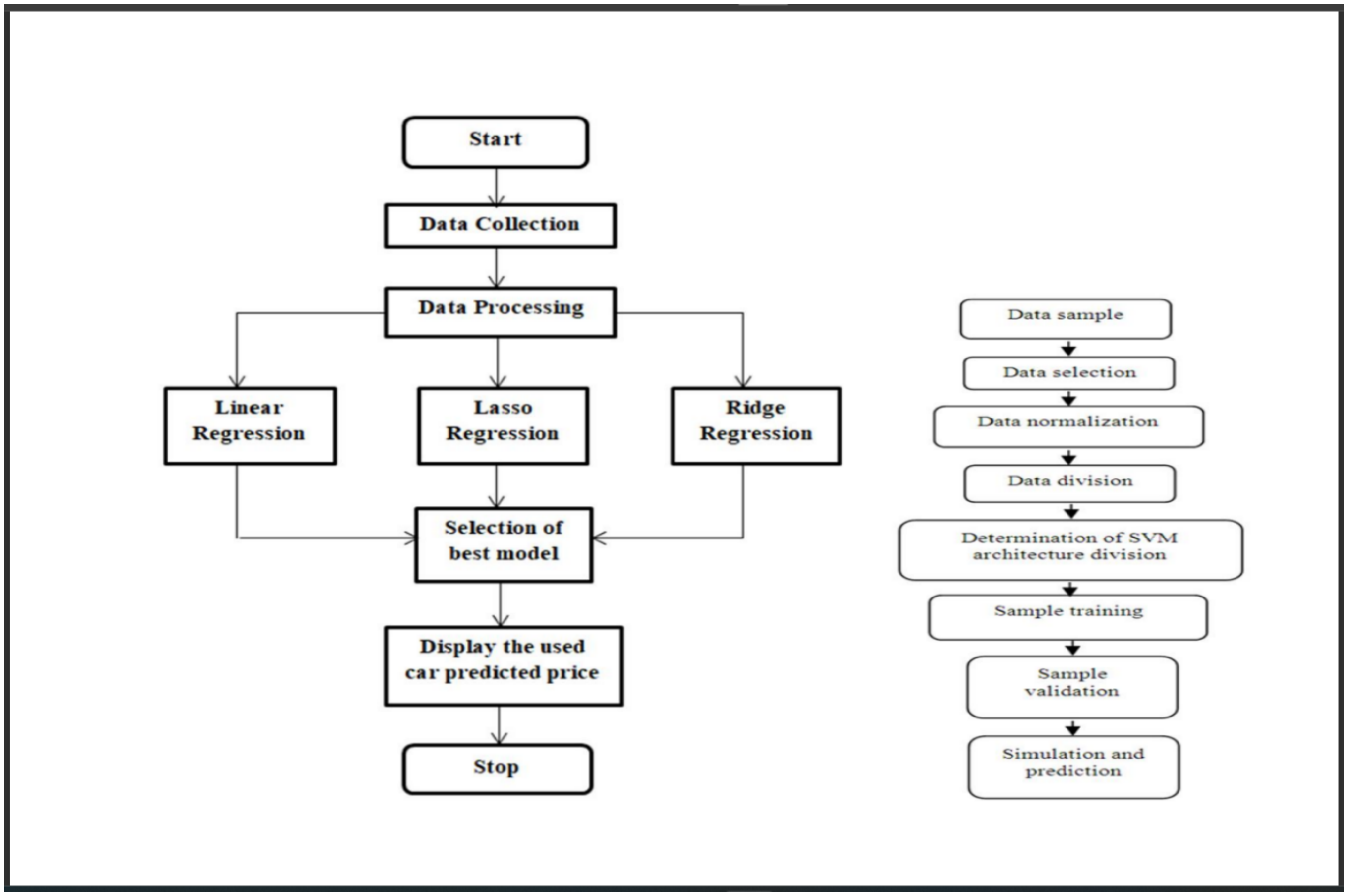
Customer Support: Establish a system for addressing user queries and issues promptly, potentially incorporating chat support or a helpdesk.

Cost:

Operational Costs: Keep operational costs within a defined budget, including hosting, maintenance, and potential scaling expenses

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories



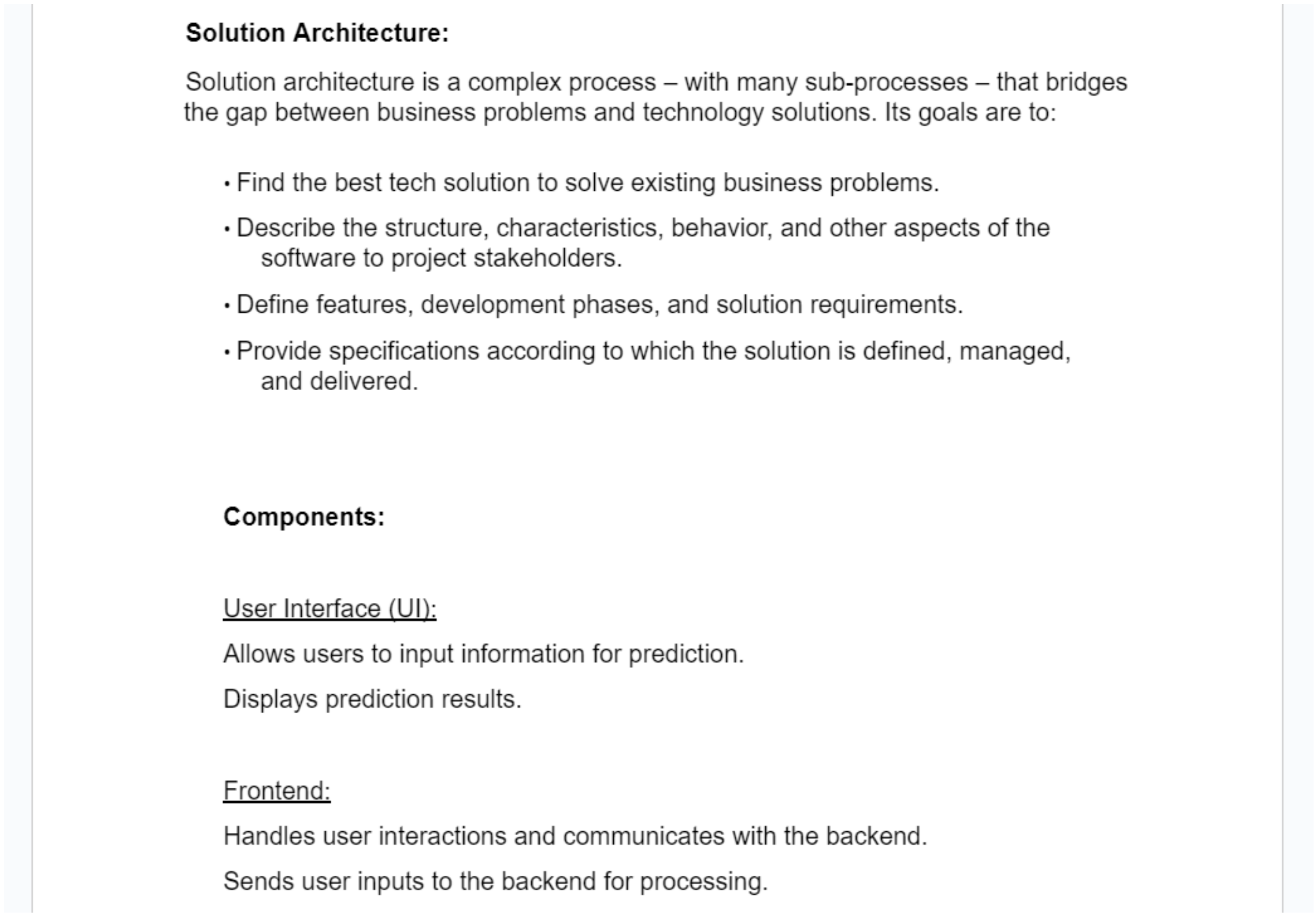
User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	Create user registration form	User can enter their name,email,and password -form includes validation for email format and password strength-user account is created and stored in database	High	Release-1.0
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Release-1.0
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Release-1
		USN-4	As a user, I can register for the application through Gmail		Medium	Release-1.0
	Login	USN-5	As a registered user i want to log into my account(create login page)	User can enter their email and password-successful login redirects to the prediction dashboard-failed login shows error message	High	Release-1.0
	Car data input		As a user i want to input data about car i'm interested in car purchasing(implement car data input)	Users can enter details like make, model, year, price range, etc. - Form includes validation for required fields. - Data is stored for prediction.	High	Release-1.0
Prediction Request	As a user, I want to request a prediction for the car I entered.		Add a "Get Prediction" button.	Users can enter details like make, model, year, price range, etc. - Form includes validation for required fields. - Data is stored for prediction.	High	Release-1.0
Feedback Submission	As a user, I want to provide feedback on the predicted price.		Create a feedback form.	Users can enter comments and a rating on the prediction. - Feedback is stored for analysis.	Medium	Release-1.1
User Profile	As a user, I want to view and edit my profile information.		Develop a user profile page.	Users can view their profile details. - Users can edit and update their profile information	Medium	Release-1.1
Admin Dashboard	As an admin, I want to have access to an admin dashboard.		Create an admin dashboard.	Admins can log in with specific credentials. - Dashboard displays user statistics and feedback.	High	Release-1.1
Data Analysis	As a data analyst, I want to analyze user feedback for improvements.		Implement data analysis tools.	Access to user feedback data for analysis. - Generate insights and reports on user feedback.	High	Release-1.2

Model Improvement	As a developer, I want to improve the machine learning model.		Develop an enhancement plan.	Model improvements implemented based on analysis results. - Performance and accuracy of predictions increase.	High	Release-1.2

5.2 Solution Architecture



Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Components:

User Interface (UI):

Allows users to input information for prediction.

Displays prediction results.

Frontend:

Handles user interactions and communicates with the backend.

Sends user inputs to the backend for processing.

Backend:

Receives input from the frontend.

Processes the input and prepares it for the machine learning model.

Interfaces with the machine learning model for prediction.

Sends the prediction results back to the frontend.

Machine Learning Model:

Trained model for predicting the likelihood of a user purchasing a car.

Takes features such as customer demographics, browsing history, and financial information as input.

Outputs a prediction score or probability.

Data Storage:

Stores historical data used for training and validating the machine learning model.

May include databases or data warehouses.

APIs:

Connects different components of the system.

Exposes endpoints for communication between the frontend, backend, and machine learning model.

Data Preprocessing:

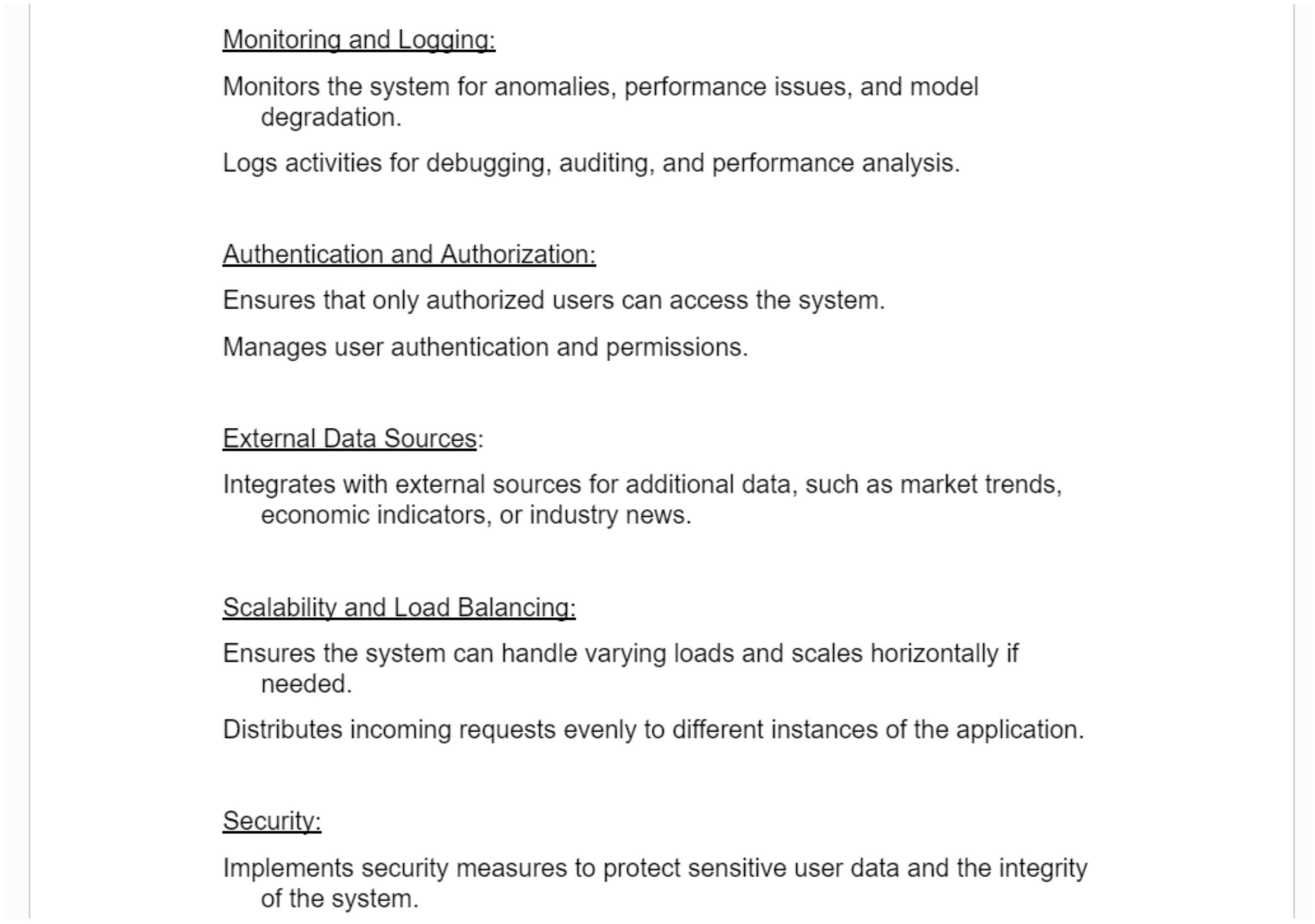
Cleans and preprocesses incoming data before it is fed into the machine learning model.

Handles missing values, scales features, and ensures data quality.

Training Pipeline:

In a separate environment, this pipeline periodically retrains the machine learning model using new data.

The updated model is then deployed to the production environment.


Monitoring and Logging:

Monitors the system for anomalies, performance issues, and model degradation.

Logs activities for debugging, auditing, and performance analysis.

Authentication and Authorization:

Ensures that only authorized users can access the system.

Manages user authentication and permissions.

External Data Sources:

Integrates with external sources for additional data, such as market trends, economic indicators, or industry news.

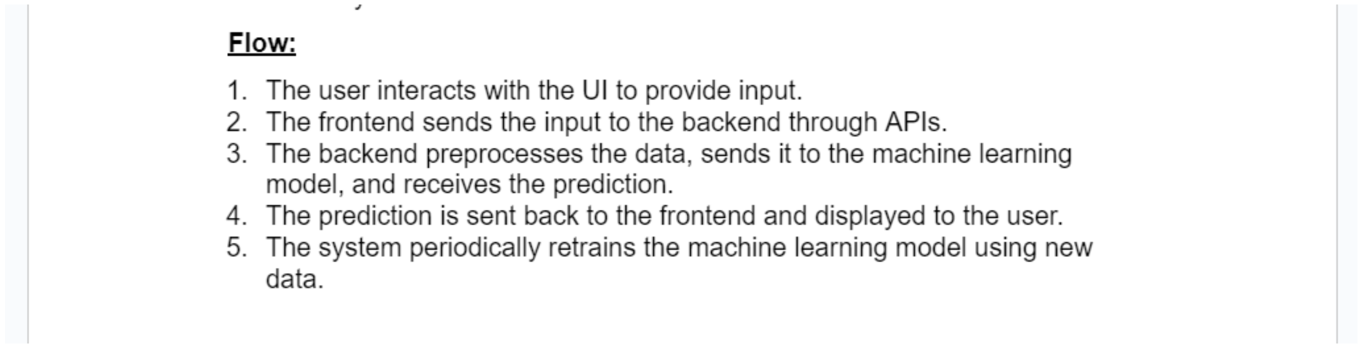
Scalability and Load Balancing:

Ensures the system can handle varying loads and scales horizontally if needed.

Distributes incoming requests evenly to different instances of the application.

Security:

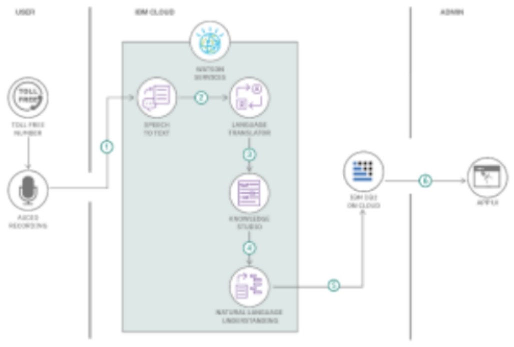
Implements security measures to protect sensitive user data and the integrity of the system.


Flow:

1. The user interacts with the UI to provide input.
2. The frontend sends the input to the backend through APIs.
3. The backend preprocesses the data, sends it to the machine learning model, and receives the prediction.
4. The prediction is sent back to the frontend and displayed to the user.
5. The system periodically retrains the machine learning model using new data.

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture



Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2 **Project:Car Purchase Prediction Using ML**

Reference:

https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

Table-1 : Components & Technologies:

S.No	**Component**	**Description**	**Technology**
1.	User Interface	Developing a user interface to present predictions and relevant information to end-users.	• Web development frameworks (React, Angular, Vue.js) • Visualization libraries (D3.js, Chart.js)
2.	Data Collection and Preprocessing	This component involves gathering data from various sources and preparing it for analysis. This can include cleaning data, handling missing values, and converting it into a suitable format.	• Data Collection: APIs, databases, data warehouses • Data Preprocessing: Python libraries (Pandas, NumPy), SQL for data cleaning, and transformat.
3.	Feature Engineering	Feature engineering is the process of selecting and transforming raw data into features that can be used by machine learning models for better predictions.	• Python libraries (Pandas, NumPy) for data manipulation • Scikit-learn for feature selection and extraction
4.	Machine Learning Model Selection:	Choosing the appropriate machine learning models based on the nature of the prediction task.	• Scikit-learn for traditional machine learning models • TensorFlow or PyTorch for deep learning models

5.	Training and Testing:	Splitting the dataset into training and testing sets to train the model and evaluate its performance	Scikit-learn for model training and evaluation
6.	Model Deployment:	Deploying the trained machine learning model to a production environment.	• Cloud services (AWS, Azure, Google Cloud) • Containerization tools like Docker • Serverless computing (AWS Lambda, Azure Functions)
7.	Real-time Data Processing:	Setting up a pipeline to handle real-time data from car sensors or other sources.	• Apache Kafka or Apache Flink for stream processing • Python (Flask, FastAPI) for building APIs to handle real-time data
8.	Scalability and Performance Optimization:	Ensuring the architecture can scale to handle a larger volume of data and optimizing performance.	• Kubernetes for container orchestration • Load balancing technologies
9.	Monitoring and Logging:	Implementing tools to monitor the system's performance and log relevant information for troubleshooting.	• ELK Stack (Elasticsearch, Logstash, Kibana) for log management • Prometheus and Grafana for monitoring
10.	Security and Privacy:	Implementing security measures to protect sensitive data and ensure user privacy.	• Encryption algorithms for data protection • Identity and access management (IAM) solutions
11.	Documentation and Maintenance:	Documenting the architecture, code, and processes for future reference and establishing a maintenance plan.	• Documentation tools (Swagger, Sphinx) • Version control systems (Git) for code management

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Data Collection and Preprocessing: • Involves acquiring data from various sources. • Cleans and preprocesses data to make it suitable for analysis.	Data collection and preprocessing ensure that the data used for training and testing machine learning models is clean, accurate, and in a format that can be effectively utilized.	• Data Collection: APIs (e.g., RESTful APIs), databases (e.g., MySQL, MongoDB), data warehouses (e.g., Amazon Redshift) • Data Preprocessing: Python libraries (Pandas, NumPy), SQL for data cleaning and transformation
2.	Feature Engineering: • Selects and transforms raw data into relevant features. • Enhances the predictive power of machine learning models.	Feature engineering is crucial for creating input features that capture the essential patterns and relationships within the data, improving the model's ability to make accurate predictions.	• Python libraries (Pandas, NumPy) • Scikit-learn for feature selection and extraction

3.	Machine Learning Model Selection: • Involves choosing appropriate algorithms for the prediction task. • Selection based on the nature of the data and the problem at hand.	Model selection is a critical decision that impacts the accuracy and efficiency of predictions. It involves choosing the best-suited machine learning algorithm or a combination of algorithms for the specific task.	• Scikit-learn for traditional machine learning models • TensorFlow or PyTorch for deep learning models

S.No	Characteristics	Description	Technology
4.	Training and Testing: • Splits the dataset into training and testing sets. • Trains the machine learning model on the training set and evaluates its performance on the testing set.	Training and testing ensure that the machine learning model learns from the data and can generalize well to make accurate predictions on new, unseen data.	Scikit-learn for model training and evaluation
5.	Model Deployment: • Involves deploying the trained model to a production environment. • Enables the model to make predictions on new data.	Model deployment takes the trained model from the development environment and makes it available for use in real-world scenarios, allowing it to make predictions on live data.	• Cloud services (AWS, Azure, Google Cloud) • Containerization tools like Docker • Serverless computing (AWS Lambda, Azure Functions)

6.2 Sprint Planning & Estimation

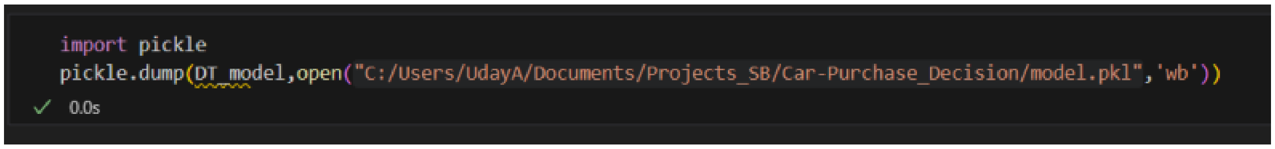
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection and Preparation	USN-1	As a user, I want to collect historical car sales data, including features such as car model, year, mileage, price, and customer preferences. The Task is to Gather and clean historical car sales data.	5	High	Poojitha, Ajay
Sprint-2	Feature Selection and Engineering	USN-2	As a user, I want to identify relevant features for car price prediction, including brand reputation, mileage, age, and additional features. The Task is to select and engineer relevant features for the machine learning model.	8	High	Haritha, Sarika
Sprint-3	Model Development and Training	USN-3	As a user, I want to develop and train a machine learning model (e.g., regression) based on the selected features and historical data. The task is to Develop and train the car price prediction model.	13	High	Poojitha, Ajay, Haritha
Sprint-4	Model Evaluation and Optimization	USN-4	As a user,I want to evaluate the model's performance, analyze the results, and optimize the model for accuracy. The task is to Evaluate and optimize the car price prediction model.	8	High	Ajay, Sarika
Sprint-5	Prototype Deployment	USN-5	As a user, I want to deploy the prototype, allowing users to input car features and receive a predicted price. The task is to Deploy the car price prediction prototype.	5	High	Haritha, Poojitha

6.3 Sprint Delivery Schedule

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	5	4 Days	20 Oct 2022	23 Oct 2022	5	23 Oct 2022
Sprint-2	8	4 Days	24 Oct 2022	27 Oct 2022	8	27 Oct 2022
Sprint-3	13	5 Days	28 Oct 2022	01 Nov 2022	13	01 Nov 2022
Sprint-4	8	11 Days	2 Nov 2022	12 Nov 2022		
Sprint-5	5	8 Days	13 Nov 2022	20 Nov 2022		

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

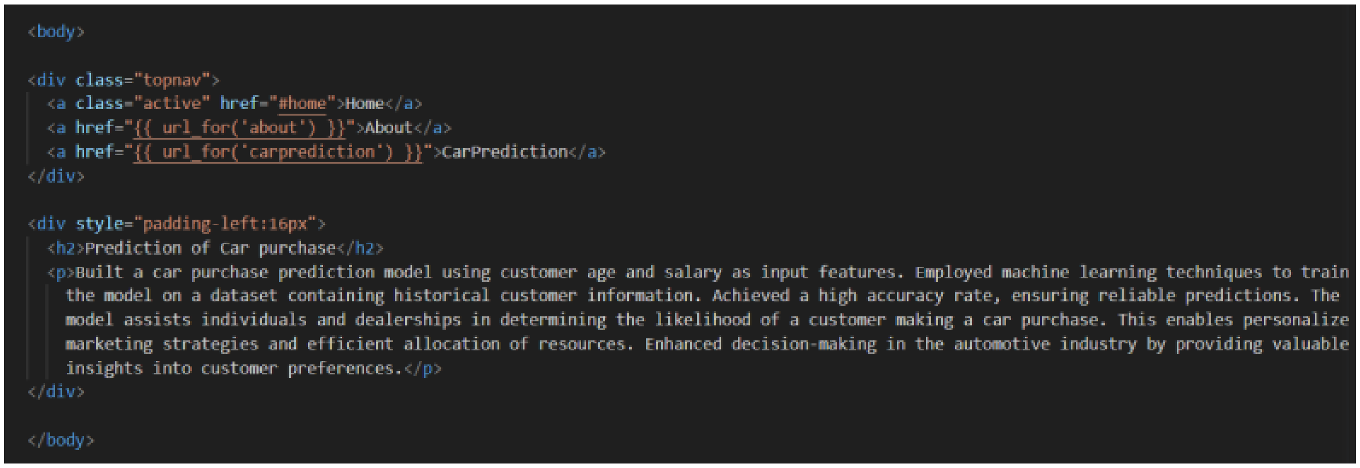


```
import pickle
pickle.dump(DT_model,open("C:/Users/UdayA/Documents/Projects_SB/Car-Purchase_Decision/model.pkl",'wb'))
```

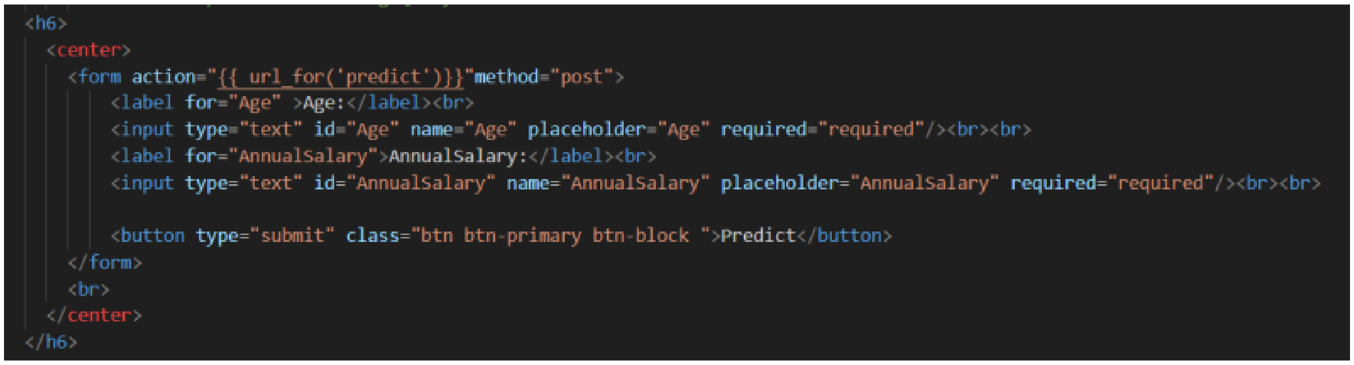
✓ 0.0s

7.2 Feature 2

Building Html Pages:

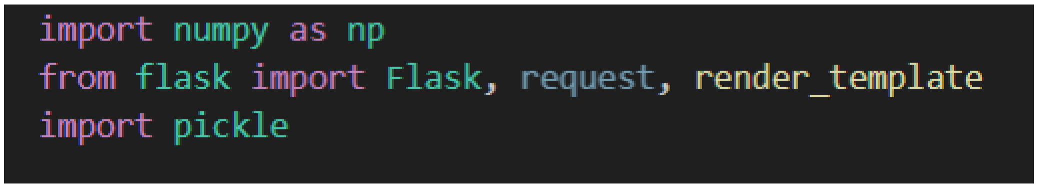


```
<body>

<div class="topnav">
  <a class="active" href="#home">Home</a>
  <a href="{{ url_for('about') }}">About</a>
  <a href="{{ url_for('carprediction') }}">CarPrediction</a>
</div>

<div style="padding-left:16px">
  <h2>Prediction of Car purchase</h2>
  <p>Built a car purchase prediction model using customer age and salary as input features. Employed machine learning techniques to train
    the model on a dataset containing historical customer information. Achieved a high accuracy rate, ensuring reliable predictions. The
    model assists individuals and dealerships in determining the likelihood of a customer making a car purchase. This enables personalize
    marketing strategies and efficient allocation of resources. Enhanced decision-making in the automotive industry by providing valuable
    insights into customer preferences.</p>
</div>

</body>
```

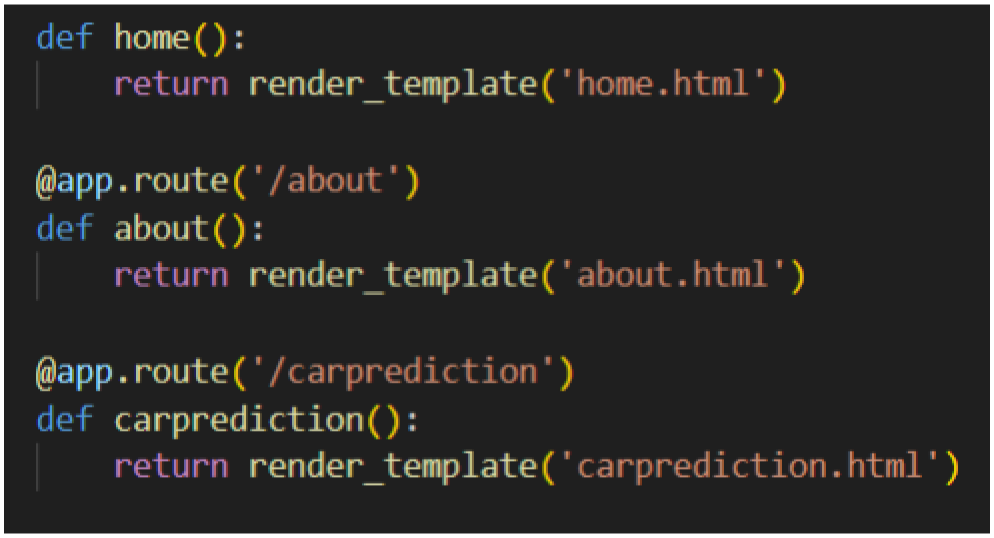


```
<h6>
  <center>
    <form action="{{ url_for('predict')}}"method="post">
        <label for="Age" >Age:</label><br>
        <input type="text" id="Age" name="Age" placeholder="Age" required="required"/><br><br>
        <label for="AnnualSalary">AnnualSalary:</label><br>
        <input type="text" id="AnnualSalary" name="AnnualSalary" placeholder="AnnualSalary" required="required"/><br><br>

        <button type="submit" class="btn btn-primary btn-block ">Predict</button>
    </form>
    <br>
  </center>
</h6>
```

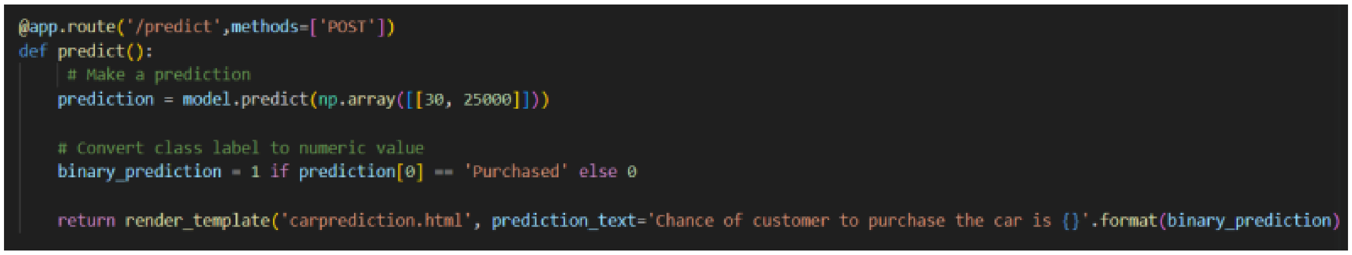
Build Python code:



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



```
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')
```

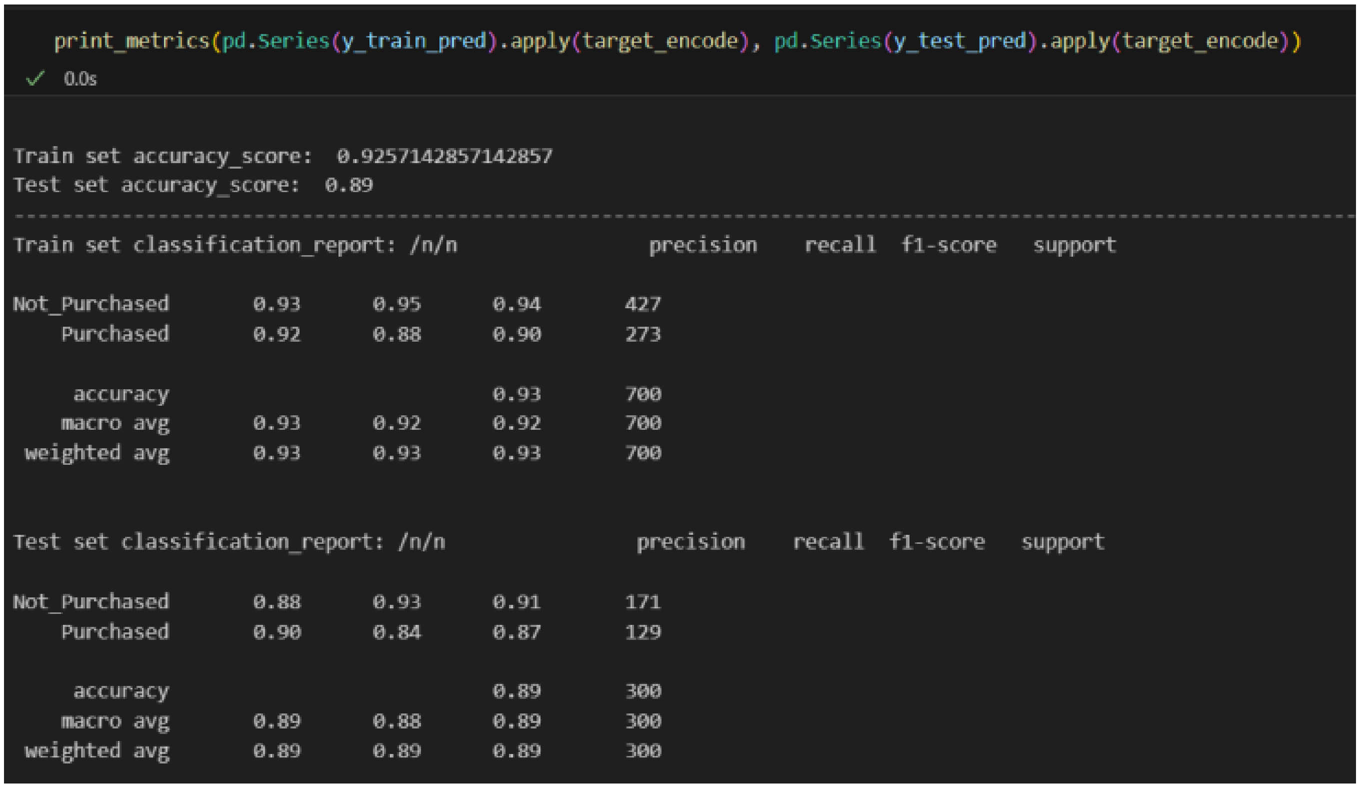


```
@app.route('/predict',methods=['POST'])
def predict():
     # Make a prediction
    prediction = model.predict(np.array([[30, 25000]]))

    # 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)
```

7.3 Database Schema (if Applicable)

8. PERFORMANCE TESTING

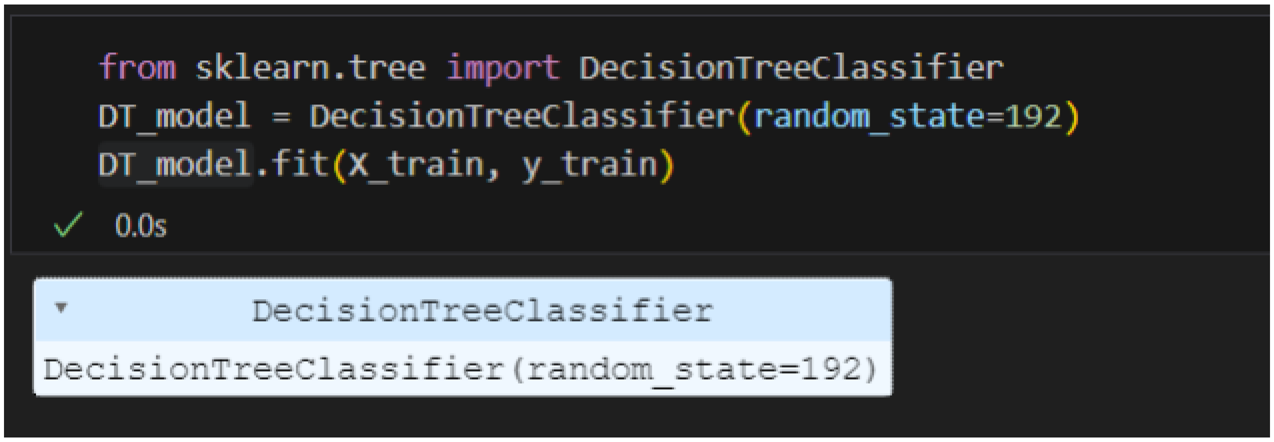
8.1 Performance Metrics



```
print_metrics(pd.Series(y_train_pred).apply(target_encode), pd.Series(y_test_pred).apply(target_encode))
✓ 0.0s

Train set accuracy_score:  0.9257142857142857
Test set accuracy_score:  0.89
---------------------------------------------------------------------------------------------------------
Train set classification_report: /n/n               precision    recall  f1-score   support

Not_Purchased       0.93      0.95      0.94       427
    Purchased       0.92      0.88      0.90       273

     accuracy                           0.93       700
    macro avg       0.93      0.92      0.92       700
 weighted avg       0.93      0.93      0.93       700


Test set classification_report: /n/n               precision    recall  f1-score   support

Not_Purchased       0.88      0.93      0.91       171
    Purchased       0.90      0.84      0.87       129

     accuracy                           0.89       300
    macro avg       0.89      0.88      0.89       300
 weighted avg       0.89      0.89      0.89       300
```

9. RESULTS

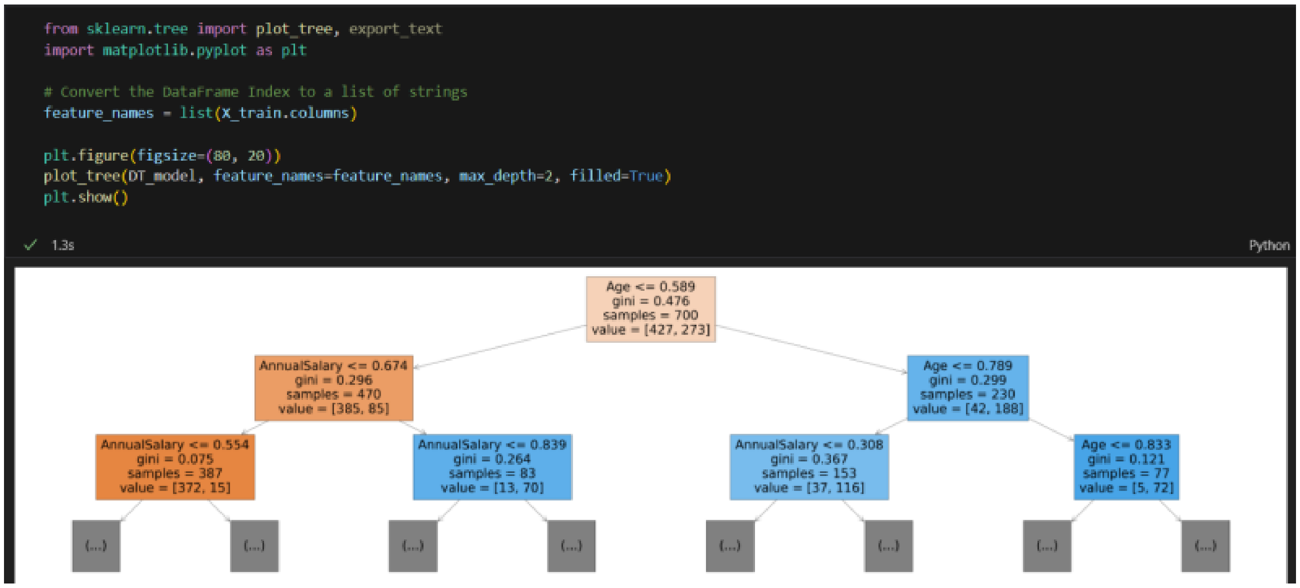
9.1 Output Screenshots



```
from sklearn.tree import DecisionTreeClassifier
DT_model = DecisionTreeClassifier(random_state=192)
DT_model.fit(X_train, y_train)
```

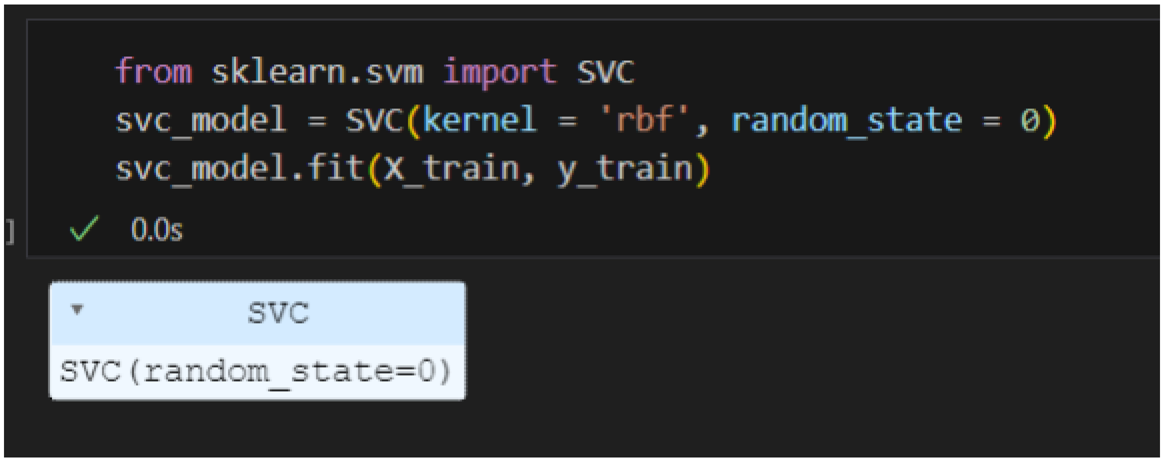
```
✓ 0.0s
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(random_state=192)
```



```
from sklearn.tree import plot_tree, export_text
import matplotlib.pyplot as plt

# Convert the DataFrame Index to a list of strings
feature_names = list(X_train.columns)

plt.figure(figsize=(80, 20))
plot_tree(DT_model, feature_names=feature_names, max_depth=2, filled=True)
plt.show()
```

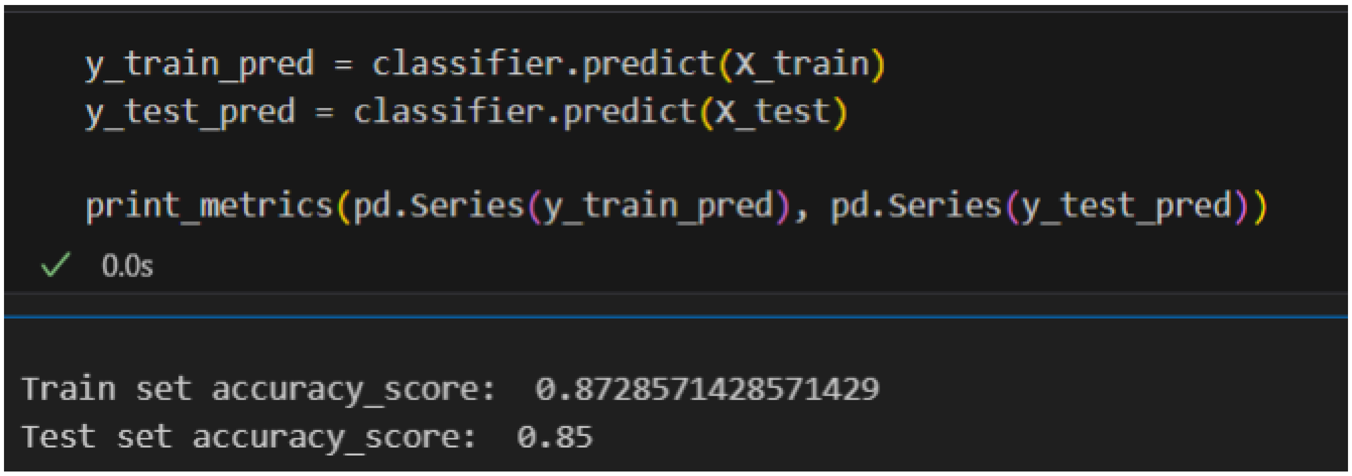
```
✓ 1.3s                                                              Python
```



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

```
✓ 0.0s
```

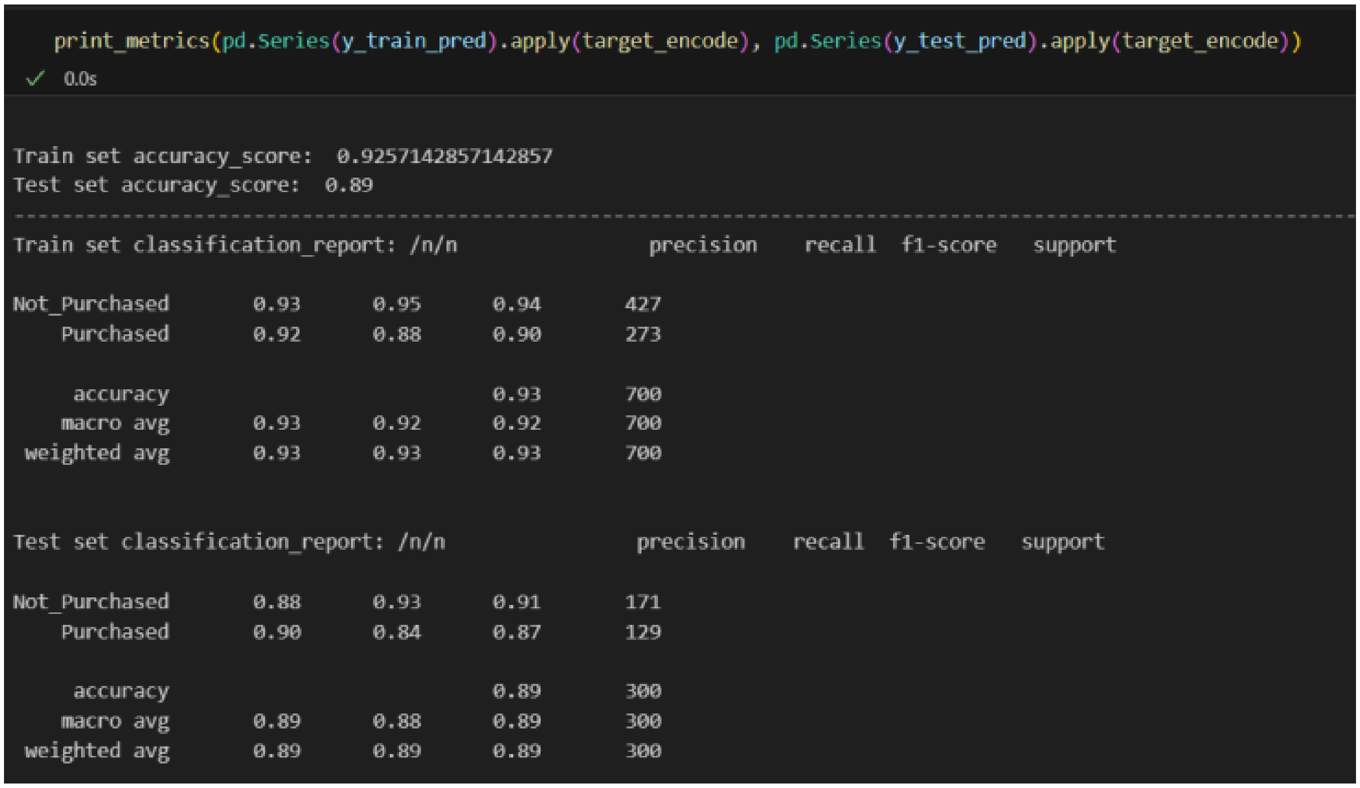
```
▾ SVC
SVC(random_state=0)
```



```
y_train_pred = classifier.predict(X_train)
y_test_pred = classifier.predict(X_test)

print_metrics(pd.Series(y_train_pred), pd.Series(y_test_pred))
```

✓ 0.0s

```
Train set accuracy_score:  0.8728571428571429
Test set accuracy_score:  0.85
```



```
print_metrics(pd.Series(y_train_pred).apply(target_encode), pd.Series(y_test_pred).apply(target_encode))
```

✓ 0.0s

```
Train set accuracy_score:  0.9257142857142857
Test set accuracy_score:  0.89
--------------------------------------------------------------------------------------------------------
Train set classification_report: /n/n               precision    recall  f1-score   support

Not_Purchased       0.93      0.95      0.94       427
    Purchased       0.92      0.88      0.90       273

     accuracy                           0.93       700
    macro avg       0.93      0.92      0.92       700
 weighted avg       0.93      0.93      0.93       700


Test set classification_report: /n/n               precision    recall  f1-score   support

Not_Purchased       0.88      0.93      0.91       171
    Purchased       0.90      0.84      0.87       129

     accuracy                           0.89       300
    macro avg       0.89      0.88      0.89       300
 weighted avg       0.89      0.89      0.89       300
```

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

Data-Driven Decision Making:

Advantage: ML algorithms can analyze large datasets to identify patterns and trends, enabling more informed decision-making in the car purchasing process.

Personalized Recommendations:

Advantage: ML models can provide personalized recommendations based on user preferences, previous behavior, and market trends, enhancing the overall user experience.

Efficient Marketing:

Advantage: Car manufacturers and dealerships can use ML to target their marketing efforts more efficiently by identifying potential customers and tailoring advertisements to specific demographics.

Risk Assessment:

Advantage: ML models can assess the creditworthiness of potential buyers, helping financial institutions and dealerships make more accurate lending decisions.

Inventory Management:

Advantage: ML algorithms can predict demand for specific car models, helping dealerships optimize their inventory and reduce overstock or shortages.

Fraud Detection:

Advantage: ML can be employed to detect fraudulent activities related to car purchases, such as identity theft or financing fraud.

Streamlined Customer Experience:

Advantage: ML-powered chatbots and virtual assistants can enhance the customer experience by providing instant responses to queries, guiding users through the purchasing process, and offering post-purchase support.

DISADVANTAGE:

Data Privacy Concerns:

Disadvantage: The use of personal data for predictions raises concerns about privacy. Customers may be uncomfortable with the level of information that companies have about their preferences and behaviors.

Bias and Fairness Issues:

Disadvantage: ML models may inherit biases present in the training data, leading to unfair or discriminatory outcomes. For example, certain demographic groups may receive different recommendations or face different approval rates.

Complexity and Lack of Transparency:

Disadvantage: Many ML algorithms, especially complex ones like deep neural networks, lack transparency. It can be challenging to understand how the model reaches a particular decision, which may be a concern for regulatory compliance and user trust.

Over Reliance on Historical Data:

Disadvantage: ML models rely on historical data, and if market conditions change rapidly, the models may become less accurate. Unforeseen events or shifts in consumer behavior may not be well-captured by historical data.

High Implementation Costs:

Disadvantage: Developing and implementing ML models can be expensive. Small businesses and dealerships with limited resources may find it challenging to invest in the necessary technology and expertise.

User Resistance:

Disadvantage: Some users may be resistant to the idea of machines making decisions about their car purchases. Building trust in the accuracy and fairness of ML models is crucial.

Maintenance Challenges:

Disadvantage: ML models require ongoing maintenance to remain accurate and relevant. Keeping the models up-to-date with changing market dynamics and user preferences can be resource-intensive.

11. CONCLUSION

The "Predictive Car Purchase Prediction Using Machine Learning" project aims to simplify the car purchasing process for users by leveraging machine learning algorithms. By analyzing user preferences and historical data, the system provides personalized recommendations, making the decision-making process more efficient and enjoyable for car buyers. This project not only showcases the power of machine learning in recommendation systems but also contributes to the ongoing innovation in the automotive industry.

12. FUTURE SCOPE

The future scope for car prediction using machine learning (ML) is promising, and there are several potential advancements and trends that may shape the field. Here are some key areas of future development:

Enhanced Personalization:

Future ML models can become even more adept at understanding individual preferences, considering factors such as lifestyle, social trends, and real-time behavior. This can lead to highly personalized recommendations for car features, models, and financing options.

Explainable AI (XAI):

There is a growing emphasis on developing ML models that are more transparent and explainable. Future systems may focus on providing clearer insights into how decisions are made, addressing concerns related to trust, bias, and fairness.

Continuous Learning Models:

ML models that can adapt and learn from ongoing changes in the automotive market, consumer behavior, and external factors will be crucial. Continuous learning systems can better adapt to evolving trends and provide more accurate predictions over time.

Collaboration with Autonomous Vehicles:

With the rise of autonomous vehicles, ML models can play a crucial role in predicting the demand for self-driving features and technologies. Car predictions may evolve to encompass not only traditional vehicle features but also advanced driver-assistance systems and autonomous capabilities.

Eco-Friendly and Sustainability Considerations:

ML models can take into account environmental concerns and evolving preferences for eco-friendly and sustainable vehicles. Predictions may include recommendations for electric or hybrid models based on user preferences and market trends.

Integration with Voice and Natural Language Processing:

Improved natural language processing capabilities can enable users to interact with ML-powered car prediction systems using voice commands. This can enhance the user experience and make the car-buying process more intuitive.

As technology continues to advance, the future scope for car prediction using ML will likely involve a combination of these trends, leading to more sophisticated, accurate, and user-friendly systems. Additionally, addressing ethical considerations and ensuring responsible AI practices will be essential in shaping the future of ML applications in the automotive industry

13. APPENDIX

Source Code

"It's too long to paste the whole code so here's the overview of the Source Code and the complete code is uploaded in the GitHub repository"

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder

# Load the dataset (replace 'your_dataset.csv' with your actual dataset)
data = pd.read_csv('your_dataset.csv')

# Display the first few rows of the dataset to understand its structure
print(data.head())

# Data preprocessing
# Assuming the dataset has features like 'Car_Name', 'Year', 'Selling_Price',Kms_Driven,
Fuel_Type, etc., and a target variable 'Car_Purchase'
# Convert categorical variables to numerical using Label Encoding
label_encoder = LabelEncoder()
data['Gender'] = label_encoder.fit_transform(data['Gender'])
data['Car_Model'] = label_encoder.fit_transform(data['Car_Model'])

# Split the dataset into features (X) and target variable (y)
X = data.drop('Car_Purchase', axis=1)
y = data['Car_Purchase']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Model training
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Model evaluation
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy}')
print('Classification Report:\n', classification_report(y_test, y_pred))
#Prediction on the Test Data
y_pred1 = lr.predict(X_test)
y_pred2 = rf.predict(X_test)
y_pred3 = xgb.predict(X_test)
y_pred4 = xg.predict(X_test)

#Evaluating the Algorithm
final_data = pd.DataFrame({'Models':['LR','RF','GBR','XG'],
         "R2_SCORE":[score1,score2,score3,score4]})

sns.barplot(final_data['Models'],final_data['R2_SCORE'])

#Prediction on New Data
model.predict(data_new)

#GUI
```

GitHub & Project Demo Link

GitHub Link : https://github.com/smartinternz02/SI-GuidedProject-581340-1694616636
https://github.com/smartinternz02/SI-GuidedProject-592688-1700578997

Project Demo Link : https://clipchamp.com/watch/EJqOlFv9nyT