**COS60011 - Technology Design Project**

**Semester 2 - 2024**

**Deliverable 2**

**Design Concept Report**

**Student members:**

Arun Ragavendhar Arunachalam Palaniyappan 104837257

Gurlivleen Singh Kainth 104796002

Amirajsinh Pradhyumansinh Sonagara 104801333

Henil Mukeshbhai Pistolwala 105065800

**Workshop:** Monday – 04:30 PM -06:30 PM - Room AGSE 108

**Facilitator:** Dr. Xinyi Cai

Contents

[Acknowledgement of Country 3](#_Toc177913021)

[1.Introduction 3](#_Toc177913022)

[1.1 Project Overview 3](#_Toc177913023)

[1.2 Background 3](#_Toc177913024)

[1.3 Project Objectives and Scope 4](#_Toc177913025)

[2. Design Concept 4](#_Toc177913026)

[2.1 Preliminary Design 4](#_Toc177913027)

[2.2 Methodology and Technical Specifications 5](#_Toc177913028)

[2.2.1 Front-End Web Application Using Streamlit 5](#_Toc177913029)

[2.2.2 Dataset Collection, Cleaning and Preparation 8](#_Toc177913030)

[2.2.3 Technical Design of the Machine Learning model 12](#_Toc177913031)

[2.3 Design Constraints, challenges and Mitigation Strategies 15](#_Toc177913032)

[2.4 Risk Mitigation Strategies for System Design improvement 17](#_Toc177913033)

[2.5 Future Plans for Scalability and Extension of the Application 18](#_Toc177913034)

[3.Project Management Plan 18](#_Toc177913035)

[3.1 Project Timeline 18](#_Toc177913036)

[3.2 Goals and Milestones 20](#_Toc177913037)

[3.3 Team breakdown and duties 20](#_Toc177913038)

[4. Conclusion 22](#_Toc177913039)

[5. Appendix 23](#_Toc177913040)

[6. References 23](#_Toc177913041)

# Acknowledgement of Country

I accept that the Wurundjeri People, the original occupants of this land, are the Kulin Nation, and that Swinburne University of Technology, situated in Melbourne, Australia, is situated on their traditional territory. I am happy to be a student at Swinburne University and would want to express my sincere appreciation to the Wurundjeri People.

My heartfelt gratitude also goes out to the Aboriginal and Torres Strait Islander students, staff, partners, and visitors of Swinburne University. It is a privilege for me to acknowledge and value the Wurundjeri People's deep ties to this land, its culture, history, and spirit.

# 1.Introduction

## 1.1 Project Overview

In today’s rapidly evolving world, the process of choosing and buying a car is often a challenging task due to the numerous factors that must be considered, such as the budget, brand, car type, engine specifications, safety features, and overall comfort. With countless models and options available, the decision-making process can become exhausting, making it challenging for potential buyers to identify the best vehicle that meets their requirements.

This project aims to design and develop a **Car Purchase Recommendation system** to simplify the car-buying process by leveraging artificial intelligence to provide personalized recommendations, saving time and reducing stress for potential buyers.”

## 1.2 Background

Car buyers often have to go through a lot of information and compare different models to find the best fit, which can be overwhelming, especially for those who are not familiar with technical details. The old way of researching cars manually is not only time-consuming but also difficult to navigate. However, with the recent advancements in machine learning and artificial intelligence, this process can be simplified by using this technology to build a custom recommendation system.

Current car recommendation platforms available in the market rely heavily on basic filtering methods, often failing to consider intricate user preferences and the dynamic nature of the buying process. Unlike traditional car recommendation platforms, this project is to use an advanced **Multi-Layer Perceptron (MLP) neural network** integrated with a dynamic Web Application. This approach enhances user engagement by providing immediate, personalised recommendations tailored to the user’s specific needs and preferences. The core idea is to analyse complex user inputs and requirements to recommend the best car options, making the car selection process faster, easier, and more personalised.

## 1.3 Project Objectives and Scope

The core idea is to create an interactive web application that uses an efficient and accurate **Multi-Layer Perceptron (MLP) neural network model** to predict and suggest the most suitable car for a user, based on the user’s preferences. The model is to be trained on a prepared dataset and integrated to the web application to interact with the user.

This report aims to cover the project’s design concept, detailed methodologies, and expected outcomes, providing a comprehensive guide to the proposed solution. It also provides the project management plan, project delivery timelines and task for respective team members.

# 2. Design Concept

## Preliminary Design

**1. System Overview**

The system aims to simplify the car buying process by using machine learning to provide personalised car recommendations. Users can easily input their preferences through a straightforward web interface, and the system quickly processes this information to rank the best car options.

**2. System Architecture**

The system consists of three main parts:

* **Web Application**:
  + It is the front end of the system and is to be built using **Streamlit** **(python library)**.
  + The web application enables users to input details such as budget, car type, and key features of their personal preference. Once all preferences are submitted, the application processes the data and feeds it into the model for analysis and recommendations.
* **Dataset:** The system is to use car dataset that has 20,000 unique car records with attributes like make, model, price, body type, engine type. The dataset is to be cleaned and validated before training the model.
* **MLP Neural Network**:

The Multi-Layer Perceptron (MLP) has been selected due to its proven efficiency in handling multi-dimensional data and its capacity to learn from complex patterns, making it ideal for this application. It is the core backend of the system that processes the data through different layers:

* + **Input Layer**: Takes the user’s inputs and passes them to the network.
  + **Hidden Layers**: Uses **ReLU (Rectified Linear Unit)** activation functions to learn patterns in the data. It helps the network handle complex data by passing positive values and ignoring negatives.
  + **Output Layer**: Uses the **Softmax** function to create a ranked list of car suggestions, showing which options are most suitable based on the user’s inputs.
* **Deployment Platform**: The completed project is planned to be hosted on Digital Ocean, providing a reliable and scalable environment that supports smooth performance and easy user access.

## Methodology and Technical Specifications

Having established the design concept, the next step involves detailing the methodologies employed to achieve a fully functional car purchase recommendation system.

### 2.2.1 Front-End Web Application Using Streamlit

**Introduction to Streamlit**

Streamlit has been chosen as the development framework for its powerful, open-source capabilities that simplify building interactive web applications, particularly for data-driven and machine learning projects. It offers a user-friendly interface, rapid prototyping, and built-in tools like easy form handling and real-time data updates, which reduce development time when compared to other python frameworks like Flask or Django. This makes Streamlit ideal for quickly creating visually appealing and responsive interfaces, perfectly aligning with the system’s goal of delivering fast and personalized car recommendations without needing extensive web development skills.

**Design and Implementation of the Car Purchase Recommendation System**

The Car Purchase Recommendation System's front-end is built using Streamlit to interactively gather user preferences that the model uses for car recommendations. The web app is designed to guide users through a series of carefully structured questions, each targeting a specific car attribute that will help the model make an accurate suggestion.

**Step-by-Step Question Flow**

To keep the interaction straightforward, the application displays one question at a time. This approach ensures that users can focus on providing one response at a time, enhancing clarity and minimizing errors. Each question corresponds to a key attribute considered by the recommendation model.

**1.Sequential Question Display:**

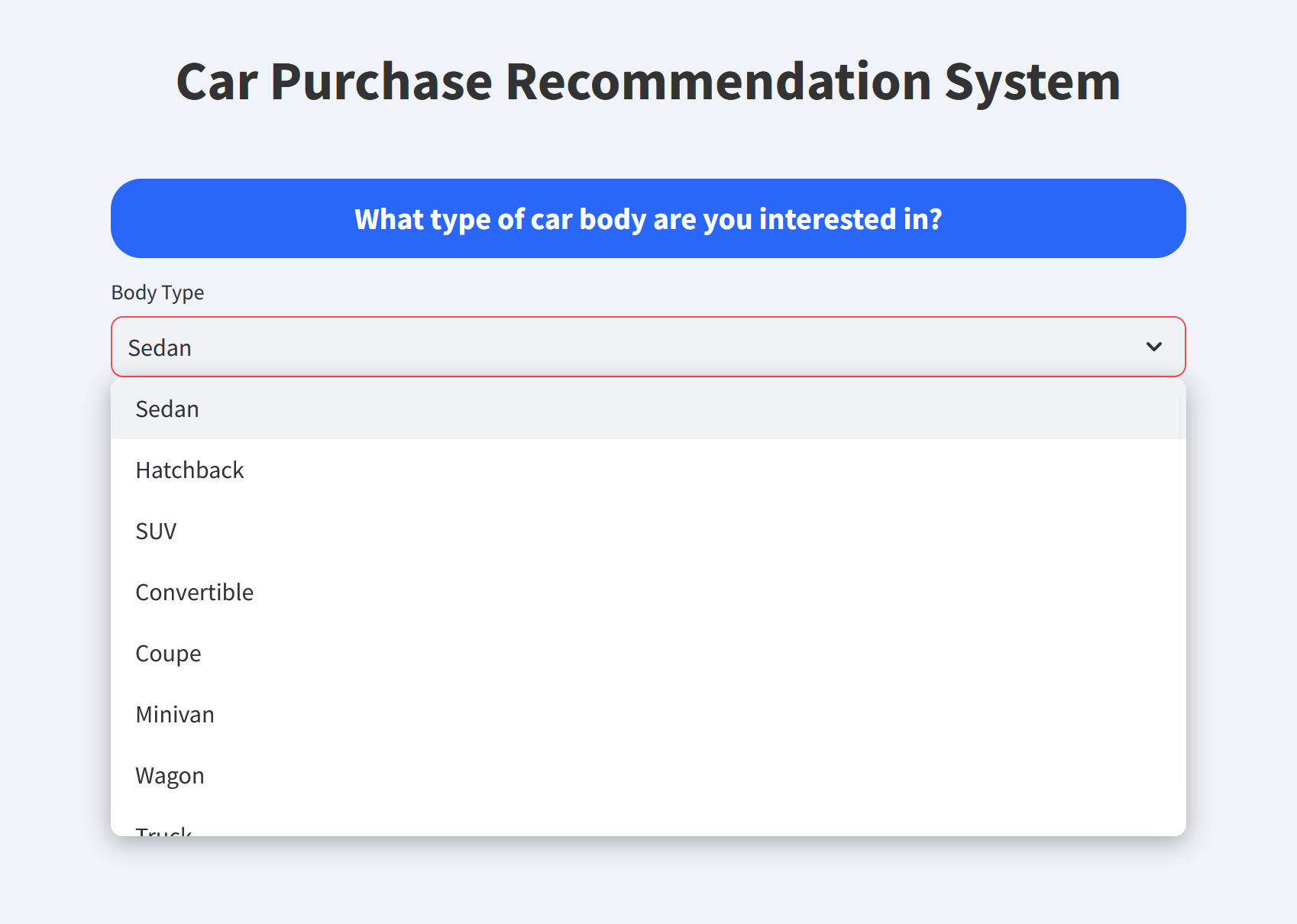
* + Only a single question appears on the screen at any given time. This sequential approach simplifies the user journey by breaking down the input process into manageable steps.
  + After selecting their response, users press a "Next" button, which automatically moves them to the following question.

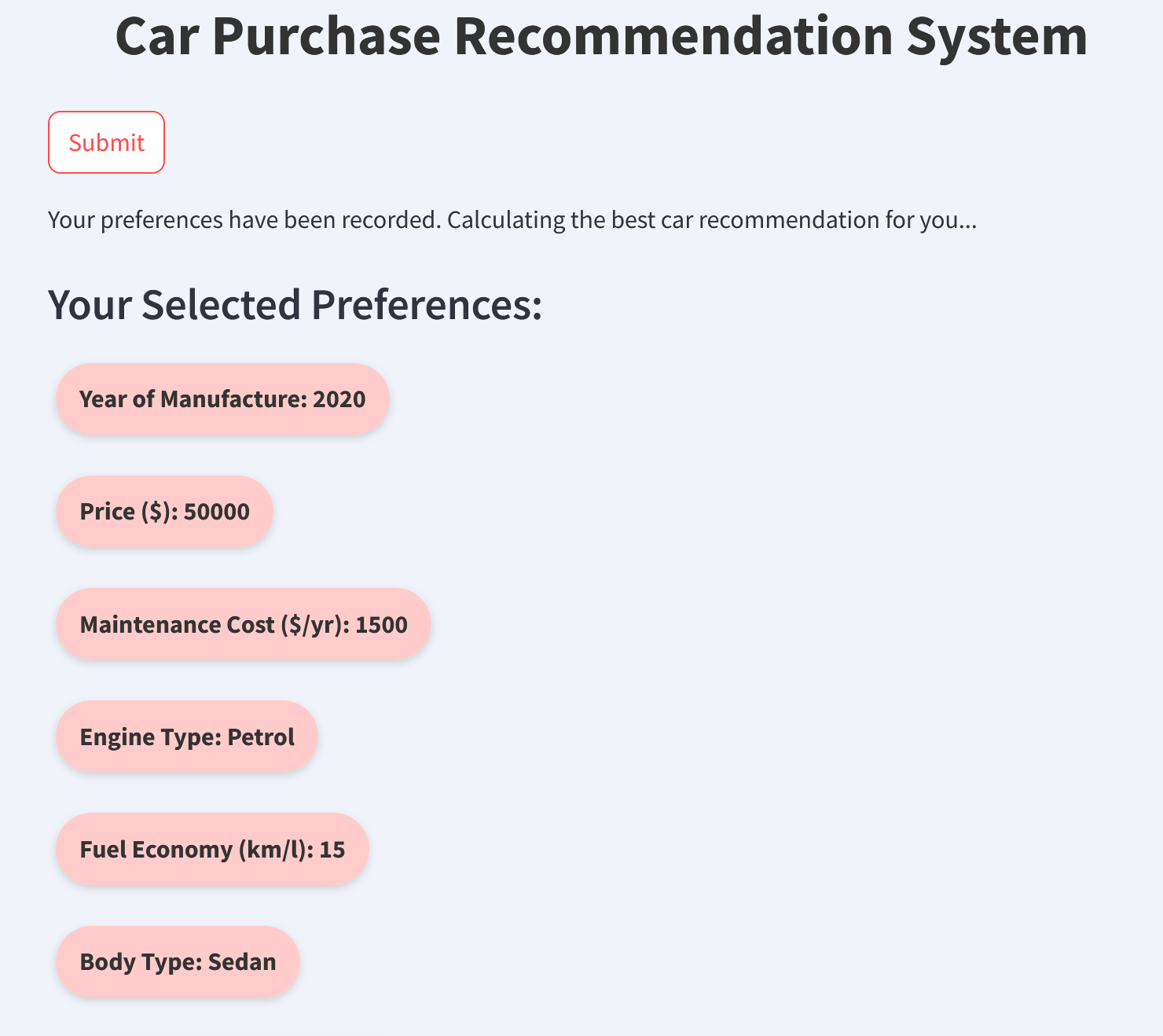
A screen shot of a computer

Description automatically generated

**2.Next and Submit Button Mechanics:**

* + Each answer is saved using Streamlit’s session state feature, allowing the application to track the user's progress and maintain the flow of questions.
  + After the final question, a "Submit" button appears. Once clicked, the application gathers all the responses, prepares them for input into the neural network model, and provides a confirmation message displaying the selected options.





**Design and User Interface Enhancements**

To make the application user-friendly and visually appealing, custom styling and layout techniques are used. Streamlit’s Markdown and CSS capabilities allow for a polished look that enhances the overall experience.

* **Header and Introduction**: The app begins with a prominent, centred title ("Car Purchase Recommendation System") and can include a related image to set the tone.
* **Question Presentation**: Each question is displayed in a distinctively styled pill-shaped box with soft colours and rounded edges, making the interaction visually clear and attractive.
* **Interactive Input Options**: Each question’s answers are presented as easily clickable options, sliders, or dropdowns, ensuring smooth navigation through the application.
* **Confirmation of Responses**: Upon submission, the user’s selected preferences are displayed in a clean, organized format with pill-shaped badges, enhancing readability and providing a neat summary of their choices.

**Technical Implementation**

1. **Session State Management**: Streamlit’s session state is used to manage the user’s journey, saving responses and tracking which question the user is currently answering. This ensures consistency and a seamless flow throughout the process.
2. **Dynamic Question Handling**: The application dynamically adjusts based on prior responses. For instance, it will only ask about battery capacity if the user selects an electric or hybrid engine type.
3. **Input Validation**: The system checks those inputs match expected data types and formats, ensuring that only valid data is collected and sent to the model.
4. **Preparing Data for the Model**: After collecting all the inputs, the data is formatted correctly and encoded to match the requirements of the neural network model, which then processes the information to provide a personalized car recommendation.

This design approach not only ensures a smooth user experience but also guarantees that the collected data is reliable and ready for the model’s predictive process. Streamlit’s capabilities are effectively utilized to build an engaging and functional web application that aligns with the system’s overall design goals.

### 2.2.2 Dataset Collection, Cleaning and Preparation

**Dataset Collection, Cleaning, and Preparation**

This section explains the steps taken to collect, clean, and prepare the dataset, ensuring that the Car Purchase Recommendation System operates effectively. Proper data handling is crucial for developing a reliable model that can accurately suggest cars based on user preferences.

**1. Collecting the Data**

**Goal**: The primary goal of data collection is to gather comprehensive car information, covering essential features like price, engine type, fuel efficiency, safety ratings, and more, to cater to various buyer needs.

**Sources**:

* **Web Scraping**: Data was collected using automated tools from several reliable sources:
  + **Car Dealership Websites**: Provided detailed information on car prices, models, and specific features.
  + **Automotive Review Sites**: Offered insights into user reviews, expert ratings, and performance evaluations, providing a broad view of each car's market standing.
  + **Manufacturer Databases**: Contained official specifications, such as engine details, warranty periods, and other technical attributes, ensuring data accuracy.
* **Data Aggregation**: The collected data from different sources was merged into one large dataset, capturing a wide variety of car brands and models to ensure the data was comprehensive and diverse.

**Sample Data Points**:

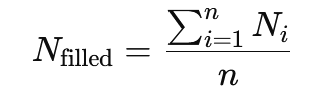
* **Attributes Collected**: The dataset includes information on car make and model, price, engine type, body style, fuel economy, performance and safety ratings, user reviews, transmission type, and comfort levels, among other features.
* **Dataset Size**: The initial dataset consists of approximately 20,000 car entries, providing a substantial base for training the recommendation model.

**2. Cleaning the Data**

Data cleaning is a critical step to eliminate errors and inconsistencies that could negatively impact the model’s performance. Here’s the approach:

**Handling Missing Data**:

* **Importance**: Missing data can lead to skewed results and inaccurate predictions, making it essential to address these gaps.
  + **Numerical Data**: Missing numerical values (e.g., price or performance ratings) are filled using the average value of the available data. This method keeps the dataset balanced without biasing towards any entry.
  + **Categorical Data**: Missing categorical information (e.g., car type or transmission) is filled using the most frequent value in the dataset, preserving the natural distribution of categories.
  + **Mathematical Approach**:
    - For missing numerical values:



* + - For missing categorical values:

A black and white text

Description automatically generated

* **Removing Duplicates and Outliers**:
* **Duplicates**: Identical records are identified and removed to prevent redundancy, which could distort the learning process of the model.
* **Outliers**: Extreme values, such as abnormally high prices or unrealistic performance ratings, are detected and adjusted or removed to maintain data consistency.
  + **Detection Methods**:
    - **IQR (Interquartile Range) Method**: Flags values outside a reasonable range.

A math equations with black text

Description automatically generated with medium confidence

* + - **Z-Score Method**: Identifies values that significantly deviate from the average.

A black text on a white background

Description automatically generated

**Standardizing Data Formats**:

* Units were standardized throughout the dataset, ensuring that all measurements (e.g., distances in kilometres and prices in AUD) are consistent, making the data comparable and uniform.

A screenshot of a computer

Description automatically generated

**3. Preparing the Data**

After cleaning, the next step is preparing the data to ensure it is in the right format and quality for use in the neural network.

* **Normalization**: User inputs are normalized using Min-Max Scaling, which scales each attribute’s values between 0 and 1. This ensures that no single input disproportionately influences the neural network due to a larger numerical range.

A black text on a white background

Description automatically generated

* + **Example**: If the user inputs a budget of $30,000, with a minimum value of $10,000 and a maximum of $50,000 in the dataset, the normalized value is calculated as:

A number with numbers on it

Description automatically generated with medium confidence

* **Reasoning**: Normalization prevents features with larger scales, such as price, from overshadowing others like user ratings or safety scores, allowing the network to treat all attributes on an equal footing during learning.
* **One-Hot Encoding**: Categorical attributes, such as Engine Type, Body Type, and Transmission Type, are transformed using One-Hot Encoding. This technique converts categorical variables into binary vectors that can be processed by the neural network.
  + **Example**: For Engine Type with categories "Petrol", "Diesel", and "Electric":
    - Original Data: Petrol, Diesel, Electric
    - One-Hot Encoded:
      * Petrol: [1, 0, 0]
      * Diesel: [0, 1, 0]
      * Electric: [0, 0, 1]
* **Reasoning**: One-Hot Encoding ensures that the neural network treats each category as a distinct attribute without implying any ordinal relationship, which would mislead the mod.

**Analysing Correlations and Feature Selection**:

* The data is analysed to identify which attributes most strongly influence car recommendations. Attributes with minimal impact may be removed to streamline the model and enhance performance.

**Splitting the Data**:

* The cleaned dataset is split into three subsets:
  + **Training Set (70%)**: Used to train the neural network and help it learn patterns in the data.
  + **Validation Set (15%)**: Used to fine-tune the model and adjust its parameters.
  + **Test Set (15%)**: Used to evaluate the model’s accuracy and performance after training.

**4. Ensuring Data Quality and Relevance**

**Consistency Checks**:

* Final checks are conducted to verify that all data points are correct and consistent, including checking ranges and ensuring all categorical variables are properly encoded.

**Suitability for Use**:

* The dataset is designed to mirror real-world car buying scenarios, providing the neural network with relevant and high-quality inputs, thus enhancing the accuracy of the model’s recommendations.

**Validation Techniques**:

* Techniques like cross-validation are used to confirm that the model performs consistently across different subsets of the data, helping prevent overfitting and ensuring that the model can generalize well to new data.

### 2.2.3 Technical Design of the Machine Learning model

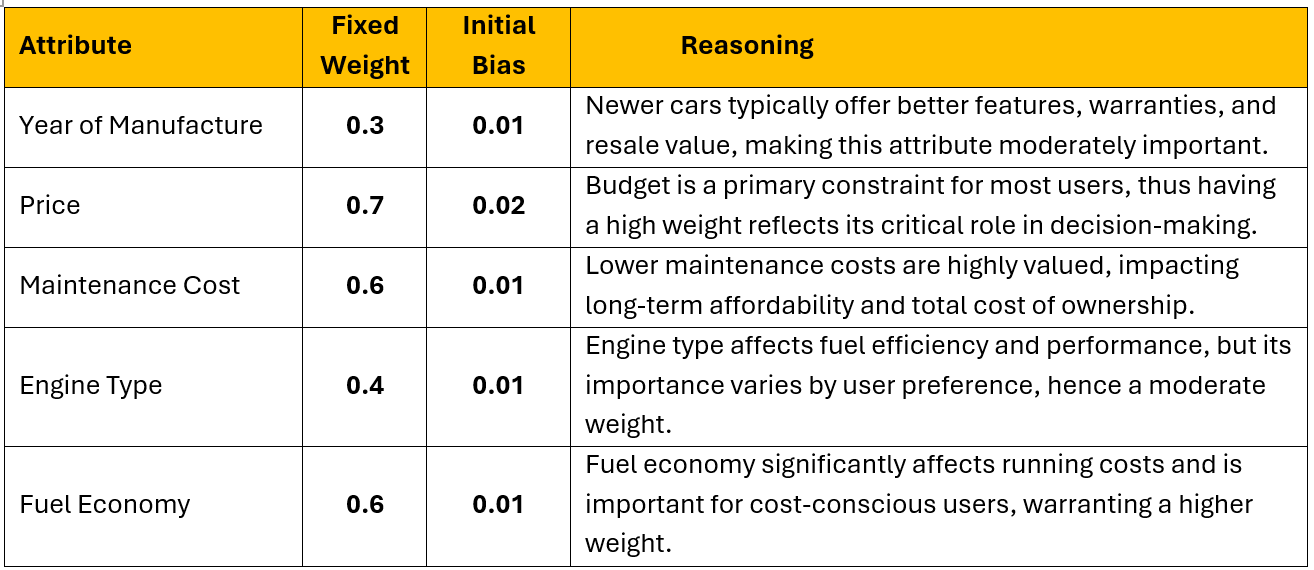
The User answers 16 questions related to car attributes, which are mapped directly to the neural network’s input layer. The network then processes these inputs through hidden layers, adjusting its behaviour based on the importance of each attribute as specified by the user.

**Input Layer Design**

* **Number of Nodes**: The input layer consists of 16 nodes, each representing a specific attribute that the user provides input on, such as budget, engine type, safety rating, performance, and more.

**Initial Weights and Biases**

Weights and biases are initialized based on the criticality of each attribute in the car buying decision process. Weights determine the importance of each input, and biases allow the network to adjust its decision threshold. The suggested weights and biases are chosen to reflect the relative impact of each attribute.



A screenshot of a report

Description automatically generated

**4. Hidden Layers Design**

**ReLU (Rectified Linear Unit)** has been chosen for its computational efficiency and ability to address the vanishing gradient problem, common with Sigmoid and Tanh functions. It accelerates training by maintaining strong gradients for positive inputs, enabling the model to capture complex relationships and enhance prediction accuracy in our car recommendation system.

* **Structure**:
  + **First Hidden Layer**: Contains 32 nodes, each using ReLU (Rectified Linear Unit) activation. This layer captures complex interactions among user inputs, allowing the model to detect intricate patterns that relate to car preferences.
  + **Second Hidden Layer**: Contains 16 nodes with ReLU activation, refining the patterns identified by the first layer, focusing on amplifying features that align with the user's top priorities.
* **ReLU Activation Function**:
  + **Purpose**: ReLU helps the network learn effectively by introducing non-linearity, which allows it to model complex relationships between inputs and outputs.

A black and white math symbol

Description automatically generated

* + **Example**: For an input x=-3, ReLU outputs 0, effectively ignoring negative influences. For an input x=5, ReLU outputs 5, allowing positive contributions to pass through.
* **Reasoning**: ReLU accelerates training by mitigating issues such as vanishing gradients, which are common with older activation functions like Sigmoid or Tanh. This ensures that the network remains efficient and responsive to new data.
* **Forward Propagation and Importance Adjustment**:
  + As inputs are fed through the hidden layers, the network adjusts its internal weights based on the importance ratings given by the user. If a user rates safety as highly important, the hidden layers will amplify connections to safety-related attributes, ensuring these are prioritized in the final recommendation.

**5. Output Layer Design**

* **Number of Nodes**: The output layer consists of a single node, providing a final score that determines the best car recommendation based on the processed inputs from the hidden layers.
* **Softmax Activation Function (Optional)**:
  + If multiple outputs are considered, Softmax can be used to convert the final scores into probabilities, ranking car options according to how well they match user preferences.

A black and white math equation

Description automatically generated with medium confidence

* **Reasoning**: A single output allows for a clear, definitive recommendation that is directly influenced by the user’s stated priorities and the model’s learned patterns.

**6. Training and Backpropagation**

* **Training Objective**: The model is trained to minimize the discrepancy between predicted and actual outcomes, refining its ability to recommend the most suitable car.
* **Loss Function**:
  + **Cross-Entropy Loss**: Measures the model’s prediction accuracy by comparing the output probabilities against the true preference distribution.

A black and white math symbol

Description automatically generated

* **Backpropagation**:
  + During backpropagation, the model calculates the gradient of the loss with respect to each weight, updating them iteratively to reduce the error and improve accuracy.
  + **Weight Update Formula**:

A black and white math equation

Description automatically generated

* **Reasoning**: Backpropagation enables the network to fine-tune its connections based on feedback from the training data, ensuring that it continuously learns and adapts to prioritize user preferences accurately.

A black text on a white background

Description automatically generated

## 2.3 Design Constraints, challenges and Mitigation Strategies

The Car Purchase Recommendation System's design and implementation involve several technical constraints and challenges that must be addressed to ensure optimal functionality and performance. Below are the primary challenges:

1. **Data Quality and Completeness**
   * **Challenge:** The accuracy of the model’s predictions heavily depends on the quality and completeness of the dataset. Missing values, inconsistencies, or anomalies in the data can skew results.
   * **Constraint:** Comprehensive data cleaning and validation processes are crucial. Missing numerical values (e.g., price) can be handled using imputation techniques such as mean substitution. It ensure that no data gaps lead to skewed predictions.
2. **Scalability of the System**
   * **Challenge:** As the user base grows, the system must efficiently handle increased data volume and concurrent user requests without performance degradation.
   * **Constraint:** The design incorporates cloud-based deployment (DigitalOcean), with auto-scaling capabilities to adjust resources dynamically. Efficient data management strategies, such as indexed databases and optimized queries, will be implemented to support scalability.
3. **Model Performance and Efficiency**
   * **Challenge:** Balancing model complexity and computational efficiency is vital. A more complex model can yield higher accuracy but may also require excessive computational resources, slowing down the response time.
   * **Constraint:** The neural network’s architecture will be fine-tuned using techniques such as dropout regularization (randomly dropping neurons during training to prevent overfitting) and early stopping to halt training when performance stagnates.
4. **User Experience and Interface Design**
   * **Challenge:** A user-friendly interface is essential to ensure that users can easily navigate and interact with the system without confusion.
   * **Constraint:** The UI/UX will be designed with Streamlit, focusing on simplicity and clear guidance for inputting preferences. User feedback will guide iterative improvements.
5. **Security and Privacy**
   * **Challenge:** Ensuring secure handling of user data is paramount, particularly when dealing with personal preferences and potentially sensitive information.
   * **Constraint:** The system will employ encryption algorithms, such as AES (Advanced Encryption Standard) for data storage and SSL/TLS protocols for data transmission, ensuring secure and private user interactions.
6. **Bias in Recommendations**
   * **Challenge:** Training data biases can lead to skewed recommendations that favour certain car types or brands, potentially impacting the model’s fairness.
   * **Constraint:** Continuous monitoring of the model outputs using fairness metrics, such as disparate impact analysis, will help detect and mitigate biases. Techniques like data augmentation will balance underrepresented categories.

## 2.4 Risk Mitigation Strategies for System Design improvement

To address these challenges, specific strategies and techniques will be implemented to minimize risks and optimize system performance:

1. **Improving Data Quality and Validation**
   * **Strategy:** Comprehensive data cleaning, normalization, and imputation will be performed to enhance data quality.
   * This approach flags inputs that could impact predictions, while tuning backpropagation settings ensures fast, real-time processing.A black text on a white background

     Description automatically generated
   * **Mitigation:** High data quality ensures more reliable and accurate model predictions, reducing errors in recommendations.
2. **Optimizing Model Performance**
   * **Strategy:** Techniques like Stochastic Gradient Descent (SGD) with momentum will be used to optimize the training process. Tuning the learning rate and using momentum can help speed up convergence and improve accuracy.
   * **Mitigation:** These methods ensure efficient learning, preventing the model from getting stuck in local minima and enhancing overall performance.
3. **Enhancing User Experience**
   * **Strategy:** Iterative testing and feedback loops will be established, allowing users to provide input on the interface, which will then be used to make adjustments that enhance usability.
   * **Mitigation:** A well-designed interface minimizes user errors and enhances satisfaction, encouraging continuous engagement with the system.
4. **Addressing Bias in Recommendations**
   * **Strategy:** Using fairness-aware machine learning techniques and continuously updating the training dataset with unbiased, balanced samples will help reduce biases in model outputs.
   * **Mitigation:** These strategies will ensure that recommendations are fair, equitable, and reflective of a diverse set of user preferences.
5. **Managing Model Overfitting and Underfitting**
   * **Strategy:** Cross-validation and techniques such as L2 regularization (penalizing large weights

A black text on a white background

Description automatically generated

* + **Mitigation:** These strategies help the model generalize well, avoiding overfitting while maintaining high accuracy.

### 2.5 Future Plans for Scalability and Extension of the Application

The car recommendation system is designed with scalability and future expansion in mind. As demand increases, the application can be scaled using cloud-based platforms like AWS, Google Cloud, or Microsoft Azure, enhancing its ability to handle more users with features like load balancing and auto-scaling to maintain optimal performance.

A key future enhancement is converting the neural network model into a standalone API, allowing it to operate independently of the current web app. This API can then be integrated by third-party companies, such as car dealerships or automotive websites, to offer personalized car recommendations directly to their users. This integration broadens the system’s reach and utility beyond its initial scope.

Additionally, the model’s adaptability allows it to be customized for other recommendation tasks, such as suggesting related car accessories or insurance plans, opening doors to new business opportunities. Continuous updates and feedback-driven improvements will ensure the system evolves to meet changing user needs, making it a valuable tool for both individual users and businesses in the automotive sector.

# 3.Project Management Plan

## 3.1 Project Timeline

A screenshot of a task description

Description automatically generated

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Team Member** | **Week 8** | **Week 9** | **Week 10** | **Week 11** | **Week 12** | **Week 13** |
| **Arun Ragavendhar Arunachalam Palaniyappan** | Lead the creation of the web application using Streamlit. | Oversee collecting, cleaning, and preparing the dataset. | Lead the system design and initial setup of the ML model. | - Train and fine-tune the ML model for accuracy. | - Integration of all components (web app, dataset, ML model). | - Final report preparation, review, and video creation. |
|  | - Coordinate initial project tasks and team efforts. | - Ensure high-quality data preparation and formatting. | - Design and develop the neural network. | - Monitor and refine model performance during training. | - Oversee integration and ensure seamless operation. | - Finalize documentation and presentation. |
|  |  |  |  |  |  |  |
| **Gurlivleen Singh Kainth** | - Support Streamlit web app development. | - Assist in dataset collection, cleaning, and preparation. | - Contribute to ML model design and development. | - Help with model training and tuning processes. | - Support integration of components and data handling. | - Assist in report writing and presentation development. |
|  | - Collaborate on front-end elements. | - Focus on handling missing data, standardization, and encoding. | - Work on model setup and adjustments. | - Implement data-driven refinements to the model. | - Participate in system integration and testing. | - Review and refine the final report and video. |
|  |  |  |  |  |  |  |
| **Amirajsinh Pradhyumansinh Sonagara** | - Begin developing the user interface on Streamlit. | - Continue with UI adjustments and interactive features. | - Test and debug the initial ML model and system setup. | - Conduct thorough testing of the ML model's predictions. | - Final testing, debugging, and user experience improvements. | - Ensure system stability for final review and demo. |
|  | - Focus on question flow and interface layout. | - Enhance sequential question flow for data input. | - Identify and resolve integration issues. | - Work on performance testing and bug fixes. | - Final adjustments before final testing. | - Assist with video creation and final system demo. |
|  |  |  |  |  |  |  |
| **Henil Mukeshbhai Pistolwala** | - Assist with initial web app design using Streamlit. | - Work on user interaction design and question sequencing. | - Support system design and debugging tasks. | - Collaborate on testing the ML model and overall system. | - Work on testing, debugging, and finalizing features. | - Participate in report preparation and final video edits. |
|  | - Focus on UI elements and usability. | - Improve usability and user flow consistency. | - Debugging and interface fixes. | - Identify any discrepancies in user inputs and predictions. | - Ensure the app meets all design and performance criteria. | - Prepare the system for final project form submission. |

## 3.2 Goals and Milestones

1. **Web Application Development (Weeks 8-9):** The initial version of the web application is planned to be developed using Streamlit, focusing on user interface design, sequential question flow, and basic functionality for collecting user input and displaying preliminary recommendations.
2. **Dataset Collection and Preparation (Week 9):** During this phase, the team will collect, clean, and organize the car dataset, handling missing values, standardizing data formats, and encoding categorical features to ensure the dataset is prepared for training the neural network.
3. **System Design and Initial Setup of the ML Model (Week 10):** The initial design and setup of the machine learning model will be undertaken, including defining model architecture, selecting relevant features, and preparing the training pipeline.
4. **Model Training and Fine-Tuning (Weeks 10-11):** The neural network model is to be trained and fine-tuned during this period, focusing on achieving the desired accuracy and refining predictions based on user inputs.
5. **System Integration (Week 12):** All key components, including the web application, dataset, and trained ML model, will be integrated to form a complete, functional car recommendation system ready for user testing.
6. **Testing and Debugging (Week 12):** Rigorous testing will be conducted to identify and fix any issues, ensuring all system elements work together seamlessly and that the user experience is smooth and reliable.
7. **Final Deliverables (Week 13):** The project will conclude with the preparation of the final report, including all documentation, along with a video presentation that highlights the project's outcomes, demonstrating how the system aids users in making personalized car purchase decisions.

## 3.3 Team breakdown and duties

The project is planned to be executed by a team of four members, each assigned specific roles and responsibilities to ensure efficient completion of all tasks. The roles are divided based on individual expertise and interest areas, with a focus on collaborative efforts to meet the project deadlines.

**Team Members and Roles:**

1. **Arun Ragavendhar Arunachalam Palaniyappan (104837257)**

**Role: Project Leader, Researcher, Machine learning Engineer, Data Engineer**

**Responsibilities:**

* + Leading the project, ensuring all tasks are aligned with the timeline, and managing team coordination.
  + Collecting, cleaning, and organising the car dataset from various sources.
  + Preparing the data by handling missing values, standardising formats, and encoding categorical features for model training.
  + Incorporating high-quality data into the training process.
  + Designing and developing the neural network model that will predict the best car based on user inputs.
  + Training and fine-tuning the model using the prepared dataset, focusing on improving prediction accuracy and minimizing errors.
  + Integration of the Streamlit web application with the machine learning model.
  + Creating the design concept report, final project documentation, and video presentation.

1. **Gurlivleen Singh Kainth (104796002)**

**Role: Machine Learning Engineer, Data Engineer**

**Responsibilities:**

* + Collecting, cleaning, and organising the car dataset from various sources.
  + Preparing the data by handling missing values, standardising formats, and encoding categorical features for model training.
  + Ensuring the dataset is suitable for training the neural network.
  + Incorporating high-quality data into the training process.
  + Developing the neural network model that will predict the best car based on user inputs.
  + Ensuring that all deliverables are completed on time and meet the required quality standards.
  + Integration of the Streamlit web application with the machine learning model.

1. **Amirajsinh Pradhyumansinh Sonagara (104801333)**

**Role: Front End Developer and tester**

**Responsibilities:**

* Developing the Streamlit web application interface, focusing on user experience and interactive design.
* Implementing the sequential question flow to collect user preferences efficiently.
* Testing the integrated system to ensure all components work seamlessly and the user experience is smooth.
* Identifying and fixing bugs, focusing on system stability and performance during the testing phase.

1. **Henil Mukeshbhai Pistolwala (105065800)**

**Role: Front End developer and tester**

**Responsibilities:**

* Testing the integrated system to ensure all components work seamlessly and the user experience is smooth.
* Developing the Streamlit web application interface, focusing on user experience and interactive design.
* Implementing the sequential question flow to collect user preferences efficiently.
* Identifying and fixing bugs, focusing on system stability and performance during the testing phase.

**Collaboration Plan:**

* Weekly meetings will be scheduled to review progress, discuss challenges, and make necessary adjustments.
* The team will use collaborative tools such as Trello for task tracking and Microsoft Teams for communication, ensuring transparency and continuous updates.

# 4. Conclusion

The Car Purchase Recommendation System is carefully planned to use advanced machine learning, specifically a Multi-Layer Perceptron (MLP) neural network, to provide tailored car suggestions based on what users want. This report outlines a step-by-step approach to gathering, cleaning, and preparing a large dataset of 20,000 car records, designed to reflect real-world car market data accurately. The system includes an easy-to-use web interface built with Streamlit, where users can input their preferences, and a neural network model that is fine-tuned for accuracy and reliability through smart design choices, like ReLU activation functions, backpropagation, and careful risk management strategies.

The report also addresses key challenges, such as ensuring data quality, making the model scalable, and securing user information, with clear methods to tackle these issues. The system is set to be hosted on DigitalOcean cloud, which offers a flexible and reliable platform to grow with the number of users.

This project showcases the potential of AI in simplifying car buying by providing tailored recommendations that match user preferences. The current system design is effective in personalizing suggestions, but there is room for improvement by incorporating additional user feedback and refining the prediction model to boost accuracy. The plan establishes a solid groundwork for transitioning to a real-world application, making car selection more intuitive and user-friendly. By guiding users toward the best car options based on their specific needs and preferences, this project aims to enhance the overall car buying experience.

# 5. Appendix

# 6. References

[1] E. Alpaydin, *Introduction to Machine Learning*, 4th ed. Cambridge, MA: MIT Press, 2020.

[2] Y. Bengio and Y. LeCun, "Generalization in deep learning," *Journal of Machine Learning Research*, 2003.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

[4] F. Chollet, *Deep Learning with Python*. Shelter Island, NY: Manning Publications, 2017.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[6] J. Heaton, *Introduction to Neural Networks with Java*. St. Louis, MO: Heaton Research, Inc., 2018.

[7] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press, 2012.

[8] M. A. Nielsen, *Neural Networks and Deep Learning: A Textbook*. n.p.: Determination Press, 2015.

[9] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ: Pearson, 2021.

[10] I. Sutskever, J. Martens, and G. Hinton, "Learning with recurrent neural networks," in *Proceedings of the 28th International Conference on Machine Learning*, 2011.

[11] C. Zhang and Y. Ma, *Ensemble Machine Learning: Methods and Applications*. New York, NY: Springer, 2021.

[12] J. A. Smith and R. B. Doe, "Developing interactive web applications using Streamlit: A case study," *Journal of Web Technology*, vol. 15, no. 3, pp. 234-250, 2021.