# PEAS

## Self-driving cars

Self-driving cars acquire data about their surroundings using different sensors such as cameras, lidar, and radar. The AI algorithms then evaluate this data to recognize items such as pedestrians, automobiles, and traffic signs, as well as estimate their location and velocity. These algorithms employ machine learning approaches to enhance their performance over time, allowing self-driving cars to adapt to diverse driving conditions and circumstances.

### Performance measure:

* **Safety** - how well the AI system avoids accidents and ensures the safety of passengers and pedestrians.
* **Efficiency** - how efficiently the AI system navigates and reaches its destination, considering factors like fuel consumption and time taken.
* **Comfort** - how comfortable the ride is for passengers, taking into account smooth acceleration, breaking, and handling of the vehicle.
* **Compliance** - how well the AI system follows traffic rules and regulations.

### Environment:

* **Observable** - the environment is partially observable, as the AI system relies on sensors to perceive the surroundings (e.g., cameras, radar).
* **Multiagent** - the self-driving car AI interacts with other agents, such as other vehicles, pedestrians, and traffic signals.
* **Nondeterministic** - the environment is nondeterministic due to factors like unpredictable traffic conditions, weather, and road conditions.
* **Sequential** - the AI system operates in a sequential environment, making decisions and taking actions based on the current state and future predictions.
* **Dynamic** - the environment is dynamic, with continuous changes in traffic patterns, road conditions, and the presence of other agents.
* **Continuous** - the AI system operates in a continuous environment, with a continuous range of possible actions and sensor inputs.
* **Unknown** - the self-driving car AI does not know the exact rules that govern the behavior of other agents or the environment and must learn and adapt in real-time.

### Actuators:

* **Steering** **control** - the AI system controls the steering mechanism of the car to change its direction.
* **Acceleration** **and** **breaking** **control** - the AI system regulates the speed of the vehicle by controlling the acceleration and braking mechanisms.
* **Turn** **signals** - the AI system activates turn signals to indicate its intentions to other drivers and pedestrians.

### Sensors:

* **Cameras** - capture visual information about the surroundings, including road signs, traffic lights, and other vehicles.
* **Lidar** - uses laser beams to measure distances and create a detailed 3D map of the environment.
* **Radar** - detects objects and measures their distances and velocities using radio waves.
* **GPS** - provides the current location of the vehicle.
* **Inertial** **Measurement** **Unit** **(IMU)** - measures acceleration, orientation, and rotational forces.
* **Ultrasonic** **sensors** - detect nearby objects and assist in parking maneuvers.
* **Microphones** - capture audio information, such as sirens or honking, for situational awareness.

## Recommendation System

A recommendation system AI is a type of artificial intelligence that is designed to provide personalized recommendations to users based on their preferences, interests, and past behavior. These systems are commonly used in e-commerce, media streaming, and social media platforms to suggest products, movies, TV shows, music, and other content that the user is likely to enjoy.

### Performance measure

* **Accuracy** – how often the stem provides relevant recommendations to users
* **Personalization** – how well the system provides tailored recommendations to user’s specific preferences
* **Diversity** – how varied the recommendations are (wider range of options are preferred)
* **Novelty** – how often the system suggests new options
* **User satisfaction** – how happy is the user with recommendations
* **Revenue** – how much money the platform makes as result of user interaction with the system
* **Time spent on platform** – how long the user stays on the platform

### Environment

* **Observability –** for recommendation system, the environment is **partially observable.** What the system can observe is limited to the data collected from the user, which sometimes might not give us the whole picture about the user. Thus, there is degree of uncertainty that must be accounted for in recommendation process.
* **Single agent / Multiagent –** some recommendation systems are single agents, where the system only analyzes single users in isolation and characteristics of items to recommend. There are also multiagent recommendation systems, where the system could take others preferences to consideration when recommending items to user.
* **Determinism –** recommendation systems usually work in **nondeterministic environment.** User’s preferences could vary as a function of several circumstances.
* **Sequential –** the system is generally considered to operate in sequential environment than episodic environment. It continuously analyzes the user’s behavior and preferences in real-time which could affect future recommendations.
* **Dynamic –** the system operates in a dynamic environment that changes overtime. Contextual information change continuously that could affect user’s behavior.
* **Continuous –** user’s preference for certain item lies in a continuous spectrum thus the system is operating in continuous environment.
* **Unknown –** the recommendation system doesn’t know rules that will define user’s preferences, it has to learn it as it goes.

### Actuators

Recommendation system uses website, email or mobile apps as “actuators” to display recommended items. It can push notifications or send emails to users. They can display personalized ads.

### Sensors

In, recommendation systems we can consider inputs or data as sensors of the system.

* User rating and feedback
* Browsing and purchase history of user
* Demographic information
* Items characteristics
* Contextual information
* Social network data

# Project ideas

## AI powered Course Recommendation System

The system involves the development of an intelligent system that can recommend courses to students based on their academic interests and goals. The system would be integrated into a mobile app or website and would use machine learning algorithms to provide personalized recommendations for courses that are relevant to the student's academic background and career aspirations. The system would analyze student data such as performance on quizzes, exams, and assignments to identify areas where the student has excelled or struggled, and use this information to generate a personalized list of course recommendations. The system could use more advanced machine learning algorithms and social network analysis to improve the accuracy of its recommendations and provide students with a diverse range of course options. This project would be an opportunity for college students to learn about machine learning, data analysis, and academic advising while also helping their peers discover new courses that are relevant to their interests and goals.

## Article categorizer

An AI system that categorizes text topics and automatically categorize articles into different topics such as politics, sports, entertainment, and more. This can help the user organize and filtering text content for reader (user).

## AI that can play Rock paper-scissor

An AI agent that can play a game of Rock-Paper-Scissors against a human player or another AI agent. Rock-Paper-Scissors is a two-player game where each player chooses either rock, paper, or scissors. and the winner is decided by a set of rules: rock beats scissors, scissor beats paper, and paper beats rock. The AI agent will use a decision tree algorithm to make optimal decisions.

# Knapsack

For the knapsack problem, we have implemented hill climbing algorithm, simulated annealing and genetic algorithm. While benchmarking these algorithms, we have seen that **genetic** **algorithm** usually generates answers that are closest to actual optimal answers than the other two. **Hill climbing algorithm** is the lightest weight algorithm in terms of computational resource usage. The following table summarizes our observations.

|  |  |  |
| --- | --- | --- |
| 10 Items | Algorithm | Time (s) |
| Hill climbing | 0.001 |
| Simulated annealing | 0.16 |
| Genetic algorithm | 0.82 |
| 15 Items | Hill climbing | 0.01 |
| Simulated annealing | 0.34 |
| Genetic algorithm | 0.93 |
| 20 Items | Hill climbing | 0.01 |
| Simulated annealing | 0.42 |
| Genetic algorithm | 1.04 |



