www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 15(2), May-Aug 2024, 26-44. ISSN: 2007-1558. https://doi.org/10.61467/2007.1558.2024.v15i2.470

Diagnosis of the Car Brake System with Fuzzy-Bayesian Expert System

Misael Pérez Hernández [1], Martín Montes Rivera [1], Ricardo Perez Hernández [2], Roberto Macias Escobar[1]
[1]Researchand Postgraduate Studies Department in Universidad Politécnica de Aguascalientes
[2]Colegio Bosques de Aguascalientes

mc220003@alumnos.upa.edu.mx, martin.montes@upa.edu.mx
jrperez@bosques.edu.mx, mc220005@alumnos.upa.edu.mx

Abstract. Brakes are essential for vehicle safety, acting as the primary protection on the road. A malfunction can cause accidents, highlighting the importance of regular checks for issues like unusual noises, abnormal movements, slow response, and warning lights. Often, drivers may not link these symptoms to brake problems, delaying necessary checks. However, identifying these issues as brake-related allows for immediate action. This paper proposes a Fuzzy-Bayesian expert system to aid drivers in maintaining car brakes. This system combines fuzzy logic and Bayesian reasoning to manage uncertainty and make informed decisions. It utilizes UPAFuzzySystems for fuzzy rule description and Twilio for SMS integration, enabling drivers to access brake system information via mobile. Our Python-based tool aims to revolutionize brake system diagnostics and maintenance, ensuring enhanced safety through timely and effective decision-making.
Keywords: expert systems, brake maintenance, fuzzy systems, forward chaining, backward chaining

Article Info
Received May 1, 2024
Accepted June 1, 2024

1 Introduction

The primary purpose of the brake is to perform a unique and crucial function: to help reduce speed to completely stop the vehicle or prevent possible collisions with obstacles on the road. Brakes perform the fundamental function of counteracting the friction between the wheels and the pavement, thus preventing the wheels from slipping (Elian Francisco Chapi-Chamorro et al., 2022).

This is why it is one of the most important systems in vehicles. Being the first safety system in cars, it is imperative to keep it in good condition to prevent collisions and accidents (Borawski et al., 2023).The system itself sends signs of a malfunction before it fails completely or requires a preventive change. This type of signals is presented through sound, (such as squeaks), sensations (both in the pedal and in driving), as well as visual aspects (irregularities in the brake disc, light on the dashboard) (Sebastián Aldair Calderón Guerra et al., n.d.).

These failures sometimes share symptoms that can make the diagnosis for the repair of the failure delayed, in addition, several factors such as distance traveled and time elapsed since last maintenance will affect which component has the highest probability of failing. The procedure to begin an automotive diagnosis is based on the mechanic's memory and experience, which could be biased due to the limitation of the number of vehicles with that fault that the automotive technician has repaired. To solve this problem, an expert system was designed which emulates the decision making of the automotive technician to give a diagnosis with the most probable failure depending on the client's input variables (Bousdekis et al., 2021a).

The expert system will help customers obtain a diagnosis at any time they want and eliminating the bias of the technician's experience by using fuzzy logic that helps solve uncertainty problems where each fuzzy set is determining for a temperature range where it varies. the membership value of said set moving it further or further away from that set, for example the temperature, where there would be several fuzzy sets such as "cold", "warm" and "hot", in which the membership value decreases for a set and increasing for another, the range of values of belonged is a closed interval from 0 to 1 or mathematically written as [0,1] (Amirkhani & Molaie, 2023).

On the other hand, for probabilities the Bayes theorem will be used since it is a mathematical formula that describes how we can update our beliefs about the probability of a given event occurring as we obtain new relevant evidence. It is based on two key elements: a priori probability (our initial belief) and conditional probability (the probability of the evidence given the hypothesis), which will help improve precision given that several conditional events, such as several failures at the same time, squeal, braking feel, etc., to determine which part is most likely to fail (Knaiber & Alawieh, 2023).

For the user interface, tkinter was used as it is a standard Python library used to create graphical user interfaces (GUIs). Provides tools and widgets for designing windows, buttons, text boxes, menus, and other user interface elements in desktop applications. Tkinter is based on the Tcl/Tk toolset and is a popular way to create simple, functional user interfaces in Python. Tkinter is easy to learn and suitable for small and medium-sized projects that require a basic GUI (Naik et al., 2023).

In addition, an option to send the diagnosis via SMS through Twilio was incorporated since it is a cloud communications platform that allows developers to integrate communication functions, such as text messages, voice and video calls, into their applications. and websites. With Twilio, developers can send automated text messages, make programmatic phone calls, create chatbots, and enable two-factor authentication, among many other features. The platform is widely used in a variety of industries, including customer services, marketing, telemedicine, and more (Timko & Rahman, 2023).

Furthermore, this paper delves into the specific steps involved in developing an expert system for brake system fault diagnosis. These steps encompass defining the system's objectives, acquiring relevant knowledge from experts, representing the knowledge in a suitable format, designing a user-friendly interface, implementing an inference engine, and rigorously testing and validating the system's performance.

1.1 Objective

This paper focuses on the development of an expert system for fault diagnosis in brake systems, a critical component of automotive vehicles. .The objective of this research is to leverage the power of expert systems and Bayesian reasoning to accurately diagnose brake system faults and provide effective recommendations for repairs or maintenance.

Overall, the development and implementation of an expert system for brake system fault diagnosis using Bayesian reasoning, fuzzy systems, inference Mamdani system, in addition to the visual interface with tkinter and communication via SMS with TWILIO hold great potential in improving automotive maintenance practices.

2 State of the art

The development of expert systems for vehicle brake maintenance represents a significant advancement in the automotive industry. These systems combine the power of artificial intelligence (AI) and domain expertise to assist mechanics, technicians, and vehicle owners in diagnosing, maintaining, and repairing brake systems (Singh, 2023).
Trends and advancements have emerged in this field:

Integration of AI and Brake Maintenance: Expert systems leverage AI techniques such as machine learning, rule-based reasoning, and data analysis to interpret sensor data, historical maintenance records, and real-time vehicle performance metrics. This integration enhances the accuracy and speed of brake system diagnostics (Daniyan et al., 2022).

Data-Driven Decision Making: The availability of extensive data from modern vehicles has paved the way for data-driven decision-making in brake maintenance. Expert systems can analyze sensor data, wear patterns, temperature fluctuations, and more to provide informed recommendations for maintenance or repair(Bousdekis et al., 2021b).

Real-time Monitoring: Advanced sensor technologies have enabled real-time monitoring of brake components, ensuring continuous assessment of brake health while the vehicle is in operation. Expert systems can process this data in real-time to detect anomalies or potential issues (Alamelu Manghai et al., 2019).

Predictive Maintenance: Predictive maintenance models have gained prominence, allowing expert systems to predict when specific brake components are likely to fail based on historical data and usage patterns. This proactive approach reduces downtime and prevents unexpected failures (Arena et al., 2021).

User-Friendly Interfaces: User interfaces for these expert systems have evolved to become more intuitive and user-friendly. Mechanics and technicians can interact with the system through graphical interfaces, voice commands, and mobile applications, enhancing accessibility and usability (Le & Le, 2021).
The integration of AI, real-time monitoring, predictive analytics, and user-friendly interfaces has revolutionized brake maintenance practices. As vehicles continue to evolve, these systems are poised to play a crucial role in ensuring safe and efficient brake system operation(Singh, 2023).

3 Theorical Framework

3.1 Expert Systems

Expert systems (SE) are computer programs that aim to solve a specific problem and use Artificial Intelligence (AI) to simulate the reasoning of a human being. They are called expert systems because these programs mimic the decision-making of a professional in the field (*¿Qué Es Un Sistema Experto? Usos y Aplicaciones En La IA*, n.d.).

Every expert system consists of two main parts: the knowledge base; and reasoning, or inference engine.
The knowledge base of expert systems contains actual and heuristic knowledge. Effective knowledge is task domain knowledge that is widely shared,
typically found in textbooks.
Heuristic knowledge is the least rigorous, most experimental, most critical knowledge of functioning. In contrast to factual knowledge, heuristic knowledge is rarely discussed and is largely individualistic. It is knowledge of good practice, good judgment, and admissible reasoning in the field. It is the knowledge that is the basis of the "art of good inference" (Tecnológica Nacional et al., n.d.).

The development of an expert system in the field of programming involves the following steps:

- Definition of the objective: Determines the specific objective of the expert system in the field of programming. For example, it might be helping programmers solve programming problems, providing software design recommendations, or diagnosing bugs in code.
- Knowledge acquisition: Identifies and collects relevant knowledge from programming experts. You can conduct interviews, review technical documentation, analyze existing code, among other methods. Make sure you capture the knowledge in a format suitable for further processing by the expert system.
- Knowledge Representation: Choose a way of representing knowledge that best suits your programming domain. Some common options include production rules, decision trees, semantic networks, or fuzzy logic. The choice will depend on the type of knowledge and the characteristics of the problem you are addressing.
- User interface design: Create an interface that allows users to interact with the expert system. It can be a command line interface, a GUI, or a web-based interface. Make sure that the interface is intuitive and facilitates communication between the user and the system.
- Implementation of the inference engine: Develops the inference engine that will be used by the expert system to process the knowledge and provide answers or solutions. You can implement rule-based reasoning algorithms, search algorithms, or even machine learning techniques, depending on the complexity of the problem and the resources available.
- Testing and validation: Perform extensive testing of the expert system to ensure that it is working correctly and producing accurate and reliable results. Validates the results of the expert system by comparing them with the knowledge of human experts or by using test data sets (Horvitz et al., 1988a, 1988b).

3.2 Faults and diagnosis for the brake system

Brake systems play a crucial role in ensuring the safety and performance of vehicles, and timely detection and resolution of faults are essential for maintaining optimal functionality (Caycedo Guzmán Juan José et al., 22 C.E.). The most common failures in the brake system are the following: in addition to adding the most common way to diagnose and correct said failure of each one, these failures must also take into account various factors such as the brand and material of the parts, time. and vehicle route. Proper maintenance and prompt repair of brake system problems are essential to ensure safe driving on the road and prevent severe accidents (Avliyokulov et al., 2023).

Noises when braking
The wear of components eventually generates noise in the braking system, which causes the metal section to come into contact with the brake disc and emit a high-pitched sound.
Causes: They may be due to wear of the brake pads or metallic contact between the pads and the discs.
Diagnosis: The problem occurs if hearing squeaks, squeals, or growls when braking.
Repair: Replace worn brake pads and, if necessary, grind or replace the discs (Avliyokulov et al., 2023).

Spongy Brake Pedal
This impression tends to arise when the brake pedals appear closer than usual. The brake pedal should have a firm or sturdy feel in typical situations. It should resist pressure, allowing it to be applied progressively rather than instantly.
Causes: It may be due to an air or brake fluid leak in the hydraulic system.
Diagnosis: If the pedal feels spongy or sinks to the floor, it is a sign of this problem.
Repair: Repair leaks in the brake lines, bleed the system to remove air, and replace the brake fluid if necessary (Avliyokulov et al., 2023).

Steering Wheel Vibration When Braking
The brake disc tends to experience deformations since it is a constantly rotating piece that comes into contact with the brake pads with each advance. Additionally, applying the brakes could produce uneven pressure, causing vibrations in the steering wheel, brake pedal, or even the entire vehicle.
Causes: Detected if warped or unevenly worn brake discs.
Diagnosis: Feeling vibrations in the steering wheel when braking is a clear sign of damaged brake discs.
Repair: Grind or replace the brake discs and ensure the calipers work correctly (Avliyokulov et al., 2023).

Brake Fluid Leak
Hydraulic lines are robust components in the brake system but are subject to corrosion, wear damage, or punctures. If these parts deteriorate, brake fluid leaks are likely. Additionally, if the piston seals, which have the function of containing the brake fluid, are damaged, this can cause leaks.
Causes: May be caused by deteriorated seals in the brake system or damaged lines.
Diagnosis: Seeing brake fluid on the ground or under the vehicle is an obvious sign of a leak.
Repair: Replace faulty seals or damaged lines and bleed the brake system (Avliyokulov et al., 2023).

Irregular Brake Pad Wear
The two brake pads within the same caliper may have different levels of wear, but this disparity should not be significant. If there is a marked discrepancy in wear levels between the two pads, it is essential to perform an inspection on the brake system to determine if any component, such as the cylinder piston or caliper bolt, is experiencing any type of restriction in their mobility.
Causes: May be due to stuck calipers, hydraulic system problems, or incorrect brake alignment.
Diagnosis: Visually inspect the brake pads for uneven wear.
Repair: Fix the underlying cause, such as repairing or replacing calipers, bleeding the system or correcting brake alignment, and replacing worn pads (Avliyokulov et al., 2023).

Dashboard Brake System Warning Lights
If the ABS warning light illuminates during everyday driving, it indicates a problem with the proper functioning of the ABS system. Although the brakes should continue to operate normally, there is a strong possibility that the ABS system will not activate in emergency braking situations.
Causes: This may be due to problems with the ABS (Anti-lock Braking System) system or low brake fluid level.
Diagnosis: The warning light on the dashboard will come on if there is a problem in the brake system.
Repair: Diagnose the underlying cause using a diagnostic scanner and make necessary repairs (Avliyokulov et al., 2023).

3.3 Experta

To develop the expert system, Python and the library called expert are used to develop it. This library helps to generate an expert system (*The Basics — Experta Unknown Documentation*, n.d.).
Fundamentals: An expert system is a program capable of combining a set of facts with a set of rules and executing actions based on the rules that match the facts.

Facts: Facts are the basic unit of information in Expert. They are used to reason about the problem and are represented by Python classes.

Rules: Rules in Expert are defined as functions decorated with @Rule. The rules have two components: the LHS (left side) and the RHS (right side). The LHS describes the conditions that must be met for the rule to run, and the RHS contains the actions that will be performed when the rule is triggered.

DefFacts: It is a decorator used to declare a set of initial facts that are needed for the system to work correctly. Methods decorated with @DefFacts must be generators that generate fact instances.

KnowledgeEngine: It is the main class where the execution of the expert system occurs. You must create subclasses of this class and use the @Rule decoration in the methods to define the rules. The knowledge engine execution cycle consists of selecting an active rule and executing the actions defined in its RHS.

Fact manipulation: Methods such as declare, retract, modify, and duplicate are provided to manipulate facts in the knowledge engine. These methods allow you to add, delete, and modify facts in working memory.

Engine Execution Cycle: The Knowledge Engine execution cycle involves selecting an active rule from the rule set, executing the corresponding actions, and updating the active rule set based on changes in the facts (*The Basics — Experta Unknown Documentation*, n.d.).

3.4 Bayesian Rasoning

Bayesian reasoning is a method of logical inference that is based on Bayes' theorem, a mathematical formula that relates the probability of a hypothesis to the available evidence. Bayesian reasoning allows you to update beliefs about a hypothesis as new information is obtained, so that you can estimate the posterior probability of the hypothesis given the evidence(Gigerenzer & Hoffrage, 1995).
Bayesian reasoning is a logical, probabilistic approach to inference and decision making, based on Bayes' theorem. This approach uses probabilities to quantify uncertainty and update them as new information is obtained.
Bayesian reasoning is based on the following elements:

1. Priori: It is the initial probability or belief about an event or a hypothesis before obtaining new information. It is denoted as P(H), where H represents the hypothesis.
2. Likelihood: It is the probability of observing the data or evidence given a hypothesis. It is denoted as P(D|H), where D represents the data.

3. Posteriori: It is the revised or updated probability of a hypothesis after considering the data. It is calculated using Bayes' theorem and is denoted as P(H|D).

Bayes' theorem states that the posterior probability of a hypothesis is proportional to the product of the prior probability and the likelihood:

$$\mathrm{P}(H|D) = \frac{P(H)\times P(D|H)}{P(D)} \quad (1)$$

Where P(D) is the marginal probability of the data, which can be calculated by adding the probabilities of all possible hypotheses multiplied by their respective likelihoods.
Bayesian reasoning involves continually updating the posterior probability as new information is obtained. This is accomplished by incorporating new data into the posterior probability calculation, leading to a revision and refinement of initial beliefs (Ayal & Beyth-Marom, 1930).

3.5 UPAFuzzySystems

The UPAFuzzySystems library used in this research serves as an instrument for generating inference systems using fuzzy logic. Additionally, it facilitates the execution of simulations and control assignments involving fuzzy controllers, transfer functions, and state-space models, encompassing both discrete and continuous domains. This library effectively fills a notable void in the

realm of open-source resources by seamlessly incorporating these capabilities within a Python-based framework.(Montes Rivera et al., 2023a, 2023b).

The capabilities of this library are manifold. It permits the formulation of fuzzy universes tailored to diverse scenarios. Furthermore, it streamlines the process of defining rules within a Fuzzy Inference System (FIS), creating connections between premises, connectives, and consequences. Moreover, the library stands out in its capacity to replicate complex problem scenarios by computing outcomes based on input arrays, establishing a mapping between input premises and corresponding consequences within a continuous realm(Montes Rivera et al., 2023b).

Furthermore, the UPAFuzzySystems library demonstrates adeptness in governing and emulating the conduct of a DC motor plant through the utilization of an array of fuzzy controllers. These encompass single-input and dual-input Mamdani and Fuzzy Logic System (FLS) controllers, along with Takagi-Sugeno controllers available in both single-input and dual-input configurations. This comprehensive suite of controllers efficiently rectifies errors, accomplishing reduction of errors by less than 1% under standard circumstances, and by less than 4% even when subjected to random disturbances of 10% uniform intensity (Montes Rivera et al., 2023a).

The library empowers users to delve into pivotal attributes of control systems, encompassing aspects like overshoot, steady time, time rising, and time peak. The manifestation of these attributes is contingent upon the controller chosen, due to discernible modifications within control structures, which encompass premises, consequences, connectives, implication, fuzzification, and defuzzification techniques. Furthermore, the library is equipped to incorporate derivatives that enable projection of changes in error behavior, as well as integrals that facilitate incremental error reduction (Montes Rivera et al., 2023a).

4 Methodology

For the development of the following expert system, the methodology described below was followed:
Identification and Definition of Relevant Facts: The first step involves identifying and defining the facts that are essential for the problem domain. These facts represent the information or data that the expert system will reason with. For example, in the context of brake maintenance, relevant facts may include the condition of brake pads, the level of brake fluid, etc. Each fact is defined with its possible values and attributes. Code1 represents an example of the identified facts.

```
class Spedal(Fact):
    "Sensacion del pedal"
    pass
class Cfrenar(Fact):
    "Chillido al frenar"
    pass
class Rfrenar(Fact):
    "Vacio al frenar"
    pass
class Nivel(Fact):
    "Nivel de liquido de freno"
    pass
class Recorrido(Fact):
    "Distancia sin mantenimiento"
    pass
class Rnivel(Fact):
    "Rellenaste el nivel"
    pass
class NivelBajo(Fact):
    "Nivel Bajo"
    pass
class ok(Fact):
    "Todo bien"
    pass
class MalaB(Fact):
```

```
        "Balatas malas"
        pass
    class MU(Fact):
        "Mantenimiento Urgente"
     pass
```

Design of Inference Rules: Inference rules capture the knowledge and relationships between the facts in the expert system. These rules define the logic and conditions under which certain actions or recommendations are made. The rules are designed based on the expertise and experience of human experts in the field. The activation and execution conditions for each rule are established to ensure that they are triggered when the necessary conditions are met. Code2 illustrates an example of such inference rules.

```
@Rule(NOT(Rpedal(r_ped=W())))
    def ask_rpedal(self):
        self.declare(Rpedal(r_ped=ask_info("¿El pedal recorre más de lo normal? (si/no)
")))

    @Rule(NOT(Spedal(s_ped=W())))
    def ask_spedal(self):
        self.declare(Spedal(s_ped=ask_info("¿Qué sensación tiene el pedal? (mas
duro/mas blando/normal) ")))

    @Rule(NOT(Dfrenar(d_frenar=W())))
    def ask_Dfrenar(self):
        self.declare(Dfrenar(d_frenar=ask_info("¿Tarda más en frenar? (si/no) ")))

    @Rule(NOT(Cfrenar(c_frenar=W())))
    def ask_cfrenar(self):
        self.declare(Cfrenar(c_frenar=ask_info("Limpia las balatas y el disco, ¿Se
escucha un chillido cuando frenas? (si/no) ")))

    @Rule(NOT(Vibracion(vibe=W())))
    def ask_vibrar(self):
        self.declare(Vibracion(vibe=ask_info("¿El vehiculo vibra al frenar? (si/no)
")))

    @Rule(NOT(Rfrenar(r_frenar=W())))
    def ask_Rfrenar(self):
        self.declare(Rfrenar(r_frenar=ask_info("¿Se escucha un ruido de vacio o como
una fuga de aire cuando frenas? (si/no) ")))

    @Rule(NOT(Nivel(nivel_bajo=W())))
    def ask_Nivel_de_liquido(self):
        self.declare(Nivel(nivel_bajo=ask_info("¿El nivel del liquido de frenos esta
bajo? (si/no) ")))

    @Rule(NOT(Tpastillas(tiempoc=W())))
    def ask_tiemposincambio(self):
        self.declare(Tpastillas(tiempoc=ask_info("¿Ha pasado mucho tiempo sin hacerle
cambio de balatas? (si/no) ")))
```

Implementation of the Inference Engine: The inference engine is responsible for controlling the reasoning process of the expert system. It processes the facts and applies the inference rules to make informed decisions or provide recommendations. The engine evaluates the conditions of the rules, matches them with the available facts, and triggers the appropriate rules. The inference engine is implemented using a programming language, such as Python, and suitable libraries or frameworks, such as the “Experta” library mentioned earlier. The engine interacts with the knowledge base, applies the rules, and produces the desired outputs.

Fig. 1. Example of request and engine created.

4.1 Bayesian Rules

The objective of the Bayesian system is initially selected, with the hypothesis being that brake pads need to be replaced, and the evidence being the vehicle's screeching sound when braking. For the probabilistic calculations, a maintenance record was obtained, which includes documented instances of failure symptoms and the corresponding actions taken for repairs. The maintenance record was provided by "Julian's Mechanical Workshop," a well-established automotive maintenance facility with over 20 years of experience in providing general vehicle maintenance services.

Table 1. Brake system maintenance record.

Vehiculo No.	El pedal del freno se siente esponjoso/suave.	Se escucha un chirrido al frenar.	Vehiculo tarda en detenerse.	El pedal del freno se hunde hasta el fondo.		Accion
1	no	si	si	no		Cambio de balatas
2	si	si	si	no		Cambio de balatas
3	no	no	si	no		Cambio de cilindro
4	no	no	si	no		Cambio de cilindro
5	no	si	no	no		Exceso de suciedad
6	no	no	no	no		Sin accion
7	si	si	si	no		Cambio de balatas
8	no	no	si	no		Reparacion de Booster
9	si	si	si	no		Cambio de balatas
10	no	no	no	no		Sin accion
11	no	no	no	si		Nivel bajo de liquido
12	si	no	no	si		Reparacion de fuga
13	si	no	no	si		Reparacion de fuga
14	no	si	no	no		Exceso de suciedad
15	si	no	no	no		Cambio de balatas
16	no	si	no	no		Exceso de suciedad
17	no	no	si	no		Exceso de suciedad
18	si	no	si	no		Cambio de balatas
19	si	no	no	si		Reparacion de fuga
20	si	no	si	no		Cambio de cilindro

In the present study, Bayesian reasoning was employed to calculate the probability of the event by dividing it into two components: LS (Likelihood of Sufficiency) and LN (Likelihood of Necessity). LS (Likelihood of Sufficiency): In the context of risk analysis or safety assessment, "Likelihood of Sufficiency" refers to the probability that a quantity or level of resources, safety measures, or any other factor is sufficient to meet certain criteria or standards.
LN (Likelihood of Necessity): could be interpreted as the probability that necessity occurs in certain actions or processes. Necessity generally involves failing to fulfill a duty or responsibility. These components are determined using the following equations(Mandel, 2014)(*Vista de SEDFE: Un Sistema Experto Para El Diagnóstico Fitosanitario Del Espárrago Usando Redes Bayesianas*, n.d.):

$$LS = \frac{P(E|H)}{P(E|\neg H)} \tag{2}$$

$$LN = \frac{P(\neg E|H)}{P(\neg E|\neg H)} \tag{3}$$

Where:

P(E|H) = Probability that the evidence given the hypothesis.
P(E|¬H) = Probability that the evidence does not give the hypothesis.
P(¬E|H) = Probability that the evidence is not present given the hypothesis
P(¬E|¬H) = Probability that the evidence is not present given the non-hypothesis.

By substituting the values into the formula, the expression can be represented as follows:

$$LS = \frac{P(E|H)}{P(E|\neg H)} = \frac{0.66}{0.214} = 30.8 \tag{4}$$

$$LN = \frac{P(\neg E|H)}{P(\neg E|\neg H)} = \frac{0.33}{0.785} = 0.42 \tag{5}$$

To conclude the probabilistic calculation, the following equations need to be applied, which involve probability assignment and normalization in order to obtain a specific probability.

$$O(H) = \frac{P(H)}{1-P(H)} \tag{6}$$

$$\mathrm{O}(H|E) = \mathrm{LS} \cdot O(H) \text{ and } O(H|\neg E) = LN \cdot O(H) \tag{7}$$

Applying Bayesian reasoning in the Python-based expert system results in the implementation illustrated in the following figure.

```
class BY(KnowledgeEngine):

    @DefFacts()
    def inicializar_hechos(self):
        yield cambiopastillas()
        yield cambiodiscos()
        yield estado()

    @Rule(NOT(estado(state=W())))
    def AsKCHILLA(self):
        self.chilla_ahora=estado(state=ask_info("Chilla       al       frenar       el
vehiculo?(si/no)"))
        self.declare(self.chilla_ahora)



    @Rule(estado(state="si")| estado(state="no"))
    def probabilidaddecambiopastillas(self):
        LS=3.08
        LN=0.42
        OS=0.5
        OS=OS/(1-OS)
        if self.chilla_ahora['state']=='si':
            OS_T=OS*LS
            OS=OS_T/(1+OS_T)

        if self.chilla_ahora['state']=='no':
            OS_N=OS*LN
            OS=OS_N/(1+OS_N)
        self.cambiopastillas=cambio(OSr=OS)
        self.declare(self.cambiopastillas)
        respuesta=("Probabilidad   de   cambio   de   pastillas   de   freno      %f"
%self.cambiopastillas['OSr'])
```

```
listas_resp.append(respuesta)
show_info(respuesta)
if self.chilla_ahora['state']=='si':
    respuesta=("Limpia los discos y las balatas")
    listas_resp.append(respuesta)
    show_info(respuesta)
```

4.2 Fuzzy Sets

It is important to note that, in the context of brake maintenance, the kilometers traveled play a crucial role in determining the need for a brake change. On the other hand, the elapsed time is considered as a maintenance standard, even if the vehicle has not traveled a considerable distance. The importance of maintaining, at least once a year, an adequate maintenance of the brake system is highlighted since this constitutes one of the fundamental components to guarantee the safety in the operation of the vehicle.
In this context, the development of an expert system for the maintenance of brakes in vehicles, proceeds in the first instance to the creation of fuzzy sets of inputs. These fuzzy sets represent two universes: one related to the number of kilometers traveled since the last brake maintenance and another concerning the amount of time elapsed in months since said last maintenance, as can be visually appreciated in Figures 1 and 2, respectively.

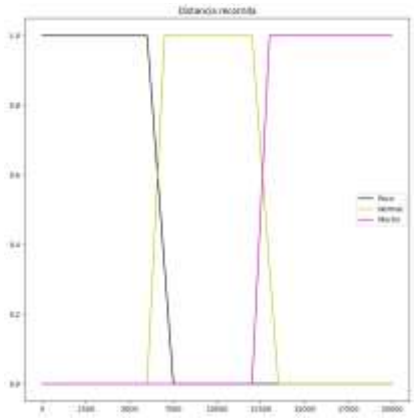

Fig. 2. Fuzzy set kilometers driven input no.1

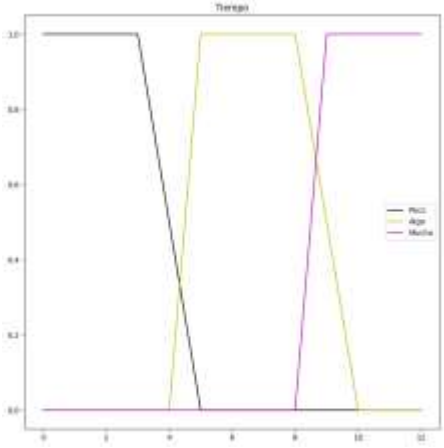

Fig. 3. Fuzzy set time elapsed input no.2.

For the generation of the output universe, a fuzzy set related to the remaining months for the next maintenance is established, as illustrated in Figure 3.

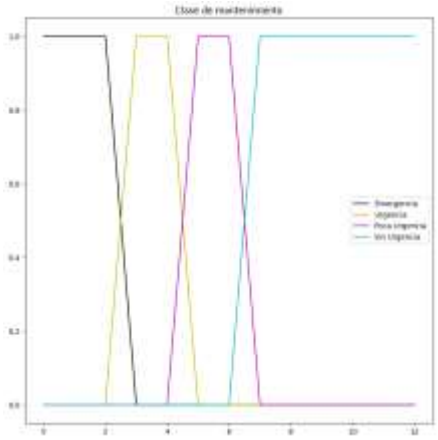

Fig. 4. Fuzzy set remaining months output

4.3 FIS

In the inference process, it was possible to observe that as more time elapses since the last maintenance and as the vehicle travels a greater number of kilometers, the number of months remaining for the next maintenance decreases. That is, these two factors are inversely related to the time remaining to carry out brake maintenance again.

This inverse relationship identified in the rules of inference highlights the importance of periodic and proper maintenance of the brake system. As the time and distance traveled increase, it is essential to be aware of the need for maintenance, since the safety and optimal performance of the vehicle are closely linked to proper care of the braking system. As shown in figure 4.

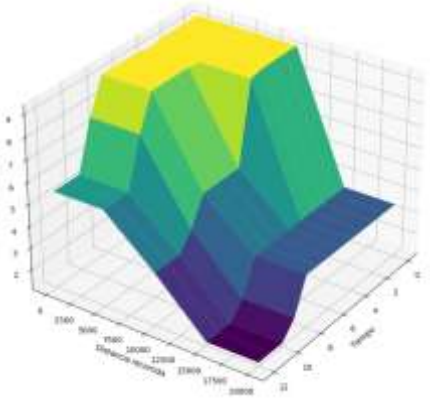

Fig. 5. FIS For the relationship between the inputs with the output

Everything described above is shown in code4

```
#Fuzzy Logic
tiempo_universe = np.arange(0,12.1,0.1)
TiempoSinMantenimiento=UPAfs.fuzzy_universe('Tiempo',tiempo_universe,'continuous')
TiempoSinMantenimiento.add_fuzzyset('Poco','trapmf',[0,0,3,5])
TiempoSinMantenimiento.add_fuzzyset('Algo','trapmf',[4,5,8,10])
```

```
TiempoSinMantenimiento.add_fuzzyset('Mucho','trapmf',[8,9,12,12])
TiempoSinMantenimiento.view_fuzzy()

#ask_info2 Recorrido
distancia_universe = np.arange(0,20000.1,0.1)
DistanciaRecorrida=UPAfs.fuzzy_universe('Distancia
recorrida',distancia_universe,'continuous')
DistanciaRecorrida.add_fuzzyset('Poco','trapmf',[0,0,6000,7500])
DistanciaRecorrida.add_fuzzyset('Normal','trapmf',[6000,7000,12000,13500])
DistanciaRecorrida.add_fuzzyset('Mucho','trapmf',[12000,13000,20000,20000])
DistanciaRecorrida.view_fuzzy()

#Output
mantenimiento_universe = np.arange(0,12.1,0.1)
NiveldeUrgencia=UPAfs.fuzzy_universe('Clase                                                    de
mantenimiento',mantenimiento_universe,'continuous')
NiveldeUrgencia.add_fuzzyset('Emergencia','trapmf',[0,0,2,3])
NiveldeUrgencia.add_fuzzyset('Urgencia','trapmf',[2,3,4,5])
NiveldeUrgencia.add_fuzzyset('Poca Urgencia','trapmf',[4,5,6,7])
NiveldeUrgencia.add_fuzzyset('Sin Urgencia','trapmf',[6,7,12,12])
NiveldeUrgencia.view_fuzzy()

#FIS
MantenimientoFrenos_Inference = UPAfs.inference_system('Mantenimineto de Frenos')
MantenimientoFrenos_Inference.add_premise(TiempoSinMantenimiento)
MantenimientoFrenos_Inference.add_premise(DistanciaRecorrida)
MantenimientoFrenos_Inference.add_consequence(NiveldeUrgencia)
MantenimientoFrenos_Inference.add_rule([['Tiempo','Poco'],['Distancia
recorrida','Poco']],['and'],[['Clase de mantenimiento','Sin Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Poco'],['Distancia
recorrida','Normal']],['and'],[['Clase de mantenimiento','Sin Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Poco'],['Distancia
recorrida','Mucho']],['and'],[['Clase de mantenimiento','Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Algo'],['Distancia
recorrida','Poco']],['and'],[['Clase de mantenimiento','Sin Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Algo'],['Distancia
recorrida','Normal']],['and'],[['Clase de mantenimiento','Poca Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Algo'],['Distancia
recorrida','Mucho']],['and'],[['Clase de mantenimiento','Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Mucho'],['Distancia
recorrida','Poco']],['and'],[['Clase de mantenimiento','Poca Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Mucho'],['Distancia
recorrida','Normal']],['and'],[['Clase de mantenimiento','Urgencia']])
MantenimientoFrenos_Inference.add_rule([['Tiempo','Mucho'],['Distancia
recorrida','Mucho']],['and'],[['Clase de mantenimiento','Emergencia']])
MantenimientoFrenos_Inference.configure('Mamdani')
MantenimientoFrenos_Inference.build()
```

The final and crucial stage in this development of the expert system consists in the creation of the graphical interface by using the renowned Tkinter library. As experts in the field of programming, we understand the importance of an attractive, intuitive and functional graphical interface to improve the user experience and ensure application efficiency.

By using the power and versatility of Tkinter, we can design and develop a graphical interface that perfectly suits the needs and objectives of the project. Widely recognized in the developer community, this library offers a variety of widgets and tools that make it easy to create windows, buttons, labels, text boxes, and many other essential visual elements for a robust user interface.

In the process of developing the graphical interface, we focus on ensuring consistency with the visual identity and design standards of the project. Our experience in interface design allows us to create attractive color schemes, efficient element distribution and a harmonious layout that promotes a high-quality and pleasant user experience.

Additionally, by working with Tkinter, we have the ability to implement interactive and dynamic functionality such as user event response, input validation, and application logic management. This real-time responsiveness adds significant value to the application, allowing for fluid and friendly interaction with the user.

```
from tkinter import *
from tkinter import ttk
def center_window(window):
    window.update_idletasks()  # Actualizar la ventana antes de centrarla
    screen_width = window.winfo_screenwidth()
    screen_height = window.winfo_screenheight()
    x = (screen_width // 2) - (window.winfo_width() // 2)
    y = (screen_height // 2) - (window.winfo_height() // 2)
    window.geometry(f"+{x}+{y}")
def sendInfo():
    global information, root, info
    info = information.get()
    print(info)
    root.destroy()
def showInfo():
    global root
    root.destroy()
def closeAllWindows():
    global root
    root.destroy()
def ask_info(question):
    global information, info, root
    root = Tk()
    root.title("Ask information")
    Ask_information = StringVar()
    Ask_information.set(question)
    Ask_information_label = ttk.Label(root, textvariable=Ask_information)
    Ask_information_label.pack(pady=5)
    information = StringVar()
    information_entry = ttk.Entry(root, width=30, textvariable=information)
    information_entry.pack(pady=5)
    Send_information_button = ttk.Button(root, text="Siguiente", command=sendInfo)
    Send_information_button.pack()
    # Agregar el botón "Cerrar"
    Close_button = ttk.Button(root, text="Cerrar", command=closeAllWindows)
    Close_button.pack()
    root.update_idletasks()  # Actualizar la ventana antes de obtener las dimensiones
    root.geometry(f"{root.winfo_width()}x{root.winfo_height()}")
    center_window(root)
    root.mainloop()
    return info
def show_info(info):
    global root
    root = Tk()
    root.title("Show information")

    show_information = StringVar()
    show_information.set(info)
    show_information_label = ttk.Label(root, textvariable=show_information)
    show_information_label.pack(pady=5)
    Send_information_button = ttk.Button(root, text="Siguiente", command=showInfo)
    Send_information_button.pack()
    # Agregar el botón "Cerrar"
```

```
    Close_button = ttk.Button(root, text="Cerrar", command=closeAllWindows)
    Close_button.pack()
    root.update_idletasks()  # Actualizar la ventana antes de obtener las dimensiones
    root.geometry(f"{root.winfo_width()}x{root.winfo_height()}")
    center_window(root)
    root.mainloop()
```

Finally, we proceed to implement the SMS interaction with TWILIO to confirm the user's phone number and receive responses from the expert system, allowing efficient access and management of information related to the status of the vehicle's braking system.
The following figure shows the implementation in python code of the interaction code as well as the functions to receive and send SMS.

```
from twilio.rest import Client
from twilio.twiml.messaging_response import MessagingResponse
import time
account_sid = 'AC200585a739c26d58bda368d19e478b45'
auth_token = '9634eed0e78e8317595cedbffeb15525'
client = Client(account_sid, auth_token)
def send_sms(ask, numero_a_enviar):
    message = client.messages.create(
        from_='+14706137168',
        body=ask,
        to= numero_a_enviar
    )
    print(message.sid)
def recive_sms(numero_a_recibir):
    time.sleep(20)  # Espera 20 segundos antes de leer los mensajes
    messages = client.messages.list(from_=numero_a_recibir, limit=5)   # Ajusta el
límite según tus necesidades
    messages_list = list(messages)  # Almacenar los mensajes en una lista
    if messages_list:
        ultimo_mensaje = messages_list[0]  # El último mensaje será el primer elemento
de la lista
        respose = ultimo_mensaje.body
        return str(respose)
    else:
        respose = "No se encontraron mensajes."
        return str(respose)

#Mandar Resultados por SMS
Pregunta_Numero = ask_info("Desea recibir las respuestas por SMS")
if Pregunta_Numero.lower() == 'si':
    numero = ask_info("Ingrese su número telefónico")
    numero = "+52" + numero
    send_sms('Confirme su número telefónico respondiendo con un: Si', numero)
    numero_confirmado = recive_sms(numero)
    if numero_confirmado.lower() == 'si':
        for respuesta in listas_resp:
            send_sms(respuesta, numero)
```

5 Results

The results shown below are the results of one input tests, the first uses where the user gives as input that the pedal travels more than normal and that its travel since the last change has been more than normal, as it does not present Another symptom of failure,

the probabilistic conclusion reaches natural wear and only brake fluid is added to compensate for the wear of the pad with the fluid inside the piston.

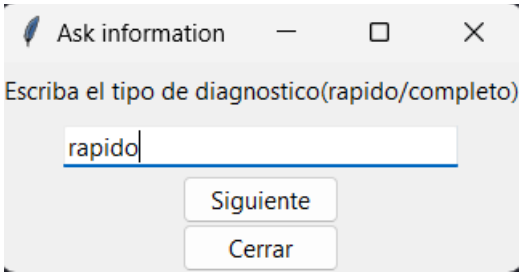

Fig. 6. Choice of diagnosis type.

In this entry the client commented that he has traveled approximately 10,000 km

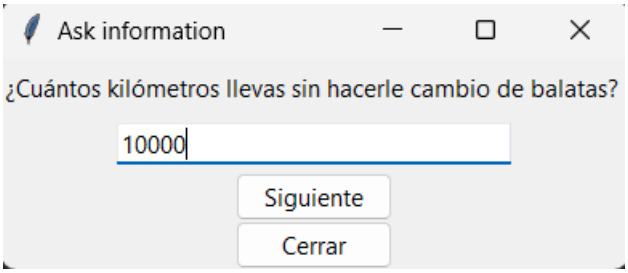

Fig. 7. Distance Traveled Entry.

Then the client specifies that he has no problems with noise.

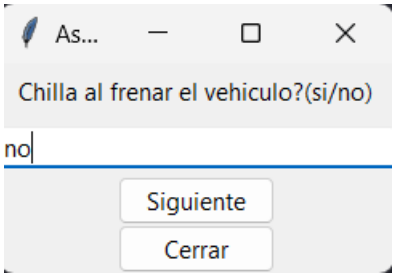

Fig. 8. Input, customer without sound problems

The client mentions that the brake fluid level is below the maximum level, which indicates that the brake pistons are traveling more than normal.

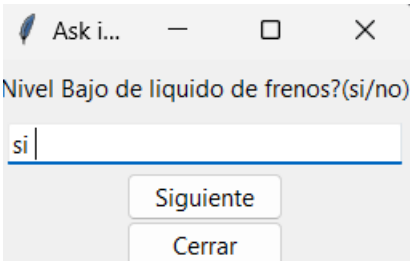

Fig. 9. positive response of low brake fluid level.

The expert system returns responses based on the database and probabilistic calculations, which shows that it is normal wear since it does not present any other failure symptom other than the low fluid level.

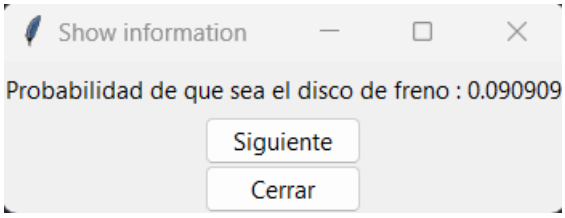

Fig. 10. Shows the probability of changing the brake disc.

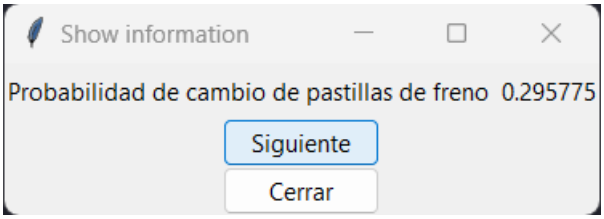

Fig. 11. Shows the probability of changing the brake pads.

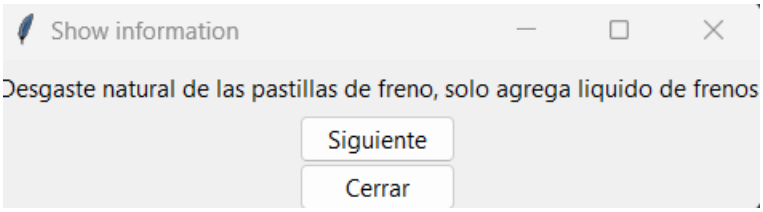

Fig. 12. It shows the verdict which is: only natural wear, just add brake fluid.

Finally, it asks the client if they want the answers by SMS, to which if the client does require it, it will ask them to enter their phone number, then the client has to confirm the number by responding via SMS to obtain the diagnostic answers

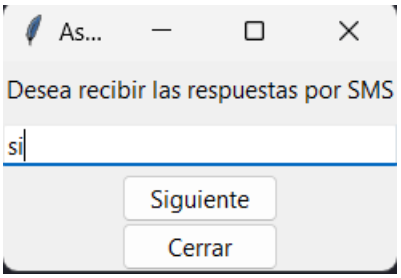

Fig. 13. Affirmative response to the request to send responses via SMS.

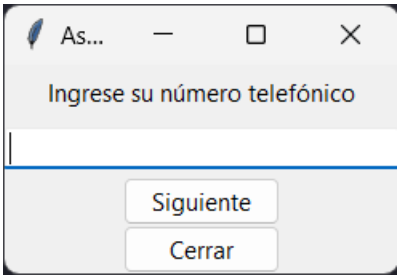

Fig. 14. Box to enter the phone number to receive the answers.

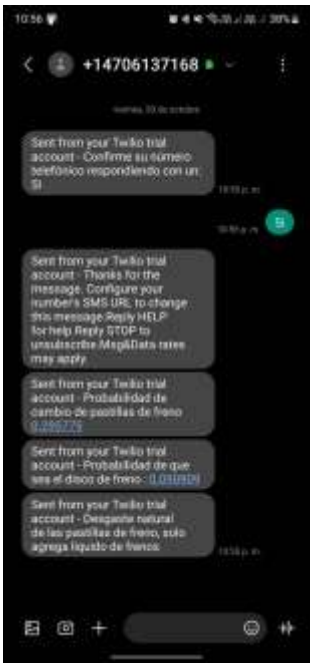

Fig. 152. Confirmation of cell phone number and receipt of answers and diagnosis.

6 Conclusions

In summary, expert systems have proven to be valuable tools in the automotive industry. By combining the specialized knowledge of auto mechanic experts with the power of computational processing, these systems offer accurate diagnosis, component condition assessment, and repair guidelines. They simulate human decision-making, incorporating both actual and heuristic knowledge into their knowledge bases, and utilize inference engines to provide informed recommendations.

Furthermore, the integration of Bayesian reasoning in brake system fault diagnosis shows promising results. The use of probabilistic calculations, based on previous maintenance records, improves accuracy in assessing probabilities of recurring failures and identifying potential solutions, such as brake pad replacement.

Another innovative approach involves combining fuzzy logic and expert systems for vehicle brake maintenance. This method aims to enhance vehicle safety and performance by utilizing intelligent maintenance systems that can adapt to various situations and handle uncertainty in data. It lays the groundwork for developing advanced maintenance systems capable of adjusting to changing conditions and dealing with imprecision in data.

Moreover, the utilization of TWILIO for SMS communication enhances the system's capabilities by providing users with real-time results and maintenance recommendations directly on their mobile devices. This combination of technology ensures a seamless and efficient user experience, allowing users to make informed decisions regarding the maintenance and safety of their brake systems.

References

Alamelu Manghai, T. M., Jegadeeshwaran, R., & Sakthivel, G. (2019). Real time condition monitoring of hydraulic brake system using naive bayes and bayes net algorithms. *IOP Conference Series: Materials Science and Engineering*, *624*(1), 012028. https://doi.org/10.1088/1757-899X/624/1/012028

Amirkhani, A., & Molaie, M. (2023). Fuzzy Controllers of Antilock Braking System: A Review. *International Journal of Fuzzy Systems*, *25*(1), 222–244. https://doi.org/10.1007/S40815-022-01376-Y/METRICS

Arena, F., Collotta, M., Luca, L., Ruggieri, M., & Termine, F. G. (2021). Predictive Maintenance in the Automotive Sector: A Literature Review. *Mathematical and Computational Applications 2022, Vol. 27, Page 2*, *27*(1), 2. https://doi.org/10.3390/MCA27010002

Avliyokulov, J. S., Pulatovich, M. S., & Rakhmatov, M. I. (2023). MAIN FAILURES OF THE VEHICLE BRAKE SYSTEM, MAINTENANCE AND REPAIR. *CENTRAL ASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES*, *4*(3), 63–69. https://doi.org/10.17605/OSF.IO/SMAUF

Ayal, S., & Beyth-Marom, R. (1930). The effects of mental steps and compatibility on Bayesian reasoning. *Judgment and Decision Making*, *9*(3), 226–242. https://doi.org/10.1017/S1930297500005775

Borawski, A., Mieczkowski, G., & Szpica, D. (2023). Composites in Vehicles Brake Systems-Selected Issues and Areas of Development. *Materials 2023, Vol. 16, Page 2264*, *16*(6), 2264. https://doi.org/10.3390/MA16062264

Bousdekis, A., Lepenioti, K., Apostolou, D., & Mentzas, G. (2021a). A Review of Data-Driven Decision-Making Methods for Industry 4.0 Maintenance Applications. *Electronics*, *10*(7), 828. https://doi.org/10.3390/electronics10070828

Bousdekis, A., Lepenioti, K., Apostolou, D., & Mentzas, G. (2021b). A Review of Data-Driven Decision-Making Methods for Industry 4.0 Maintenance Applications. *Electronics 2021, Vol. 10, Page 828*, *10*(7), 828. https://doi.org/10.3390/ELECTRONICS10070828

Caycedo Guzmán Juan José, Masmela Téllez Efraín, & Gómez Macias Mariana. (22 C.E.). Un software analítico de vehículos y un sonido de alerta la salvación de muchas vidas humanas. *JOURNAL OF SCIENCE AND RESEARCH*, *7*, 612–633.

Daniyan, I., Mpofu, K., Muvunzi, R., & Uchegbu, I. D. (2022). Implementation of Artificial intelligence for maintenance operation in the rail industry. *Procedia CIRP*, *109*, 449–453. https://doi.org/10.1016/J.PROCIR.2022.05.277

Elian Francisco Chapi-Chamorro, Jorge Andrés Fraga-Portilla, & Luis Caiza-Quispe. (2022). Existing influence on the viscosity of fluids in the anti-lock braking system (ABS). *Polo Del Conocimiento*, *7*(11), 619–629.

Gigerenzer, G., & Hoffrage, U. (1995). How to improve Bayesian reasoning without instruction: Frequency formats. *Psychological Review*, *102*(4), 684–704. https://doi.org/10.1037/0033-295X.102.4.684

Horvitz, E. J., Breese, J. S., & Henrion, M. (1988a). *Decision Theory in Expert Systems and Artificial Intelligence**.

Horvitz, E. J., Breese, J. S., & Henrion, M. (1988b). Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, *2*(3), 247–302. https://doi.org/10.1016/0888-613X(88)90120-X

Knaiber, M., & Alawieh, L. (2023). Bayesian inference using an adaptive neuro-fuzzy inference system. *Fuzzy Sets and Systems*, *459*, 43–66. https://doi.org/10.1016/J.FSS.2022.07.001

Le, T. T., & Le, M. V. (2021). Development of user-friendly kernel-based Gaussian process regression model for prediction of load-bearing capacity of square concrete-filled steel tubular members. *Materials and Structures/Materiaux et Constructions*, *54*(2), 1–24. https://doi.org/10.1617/S11527-021-01646-5/METRICS

Mandel, D. R. (2014). *The psychology of Bayesian reasoning*. https://doi.org/10.3389/fpsyg.2014.01144

Montes Rivera, M., Olvera-Gonzalez, E., & Escalante-Garcia, N. (2023a). UPAFuzzySystems: A Python Library for Control and Simulation with Fuzzy Inference Systems. *Machines*, *11*(5). https://doi.org/10.3390/machines11050572

Montes Rivera, M., Olvera-Gonzalez, E., & Escalante-Garcia, N. (2023b). UPAFuzzySystems: A Python Library for Control and Simulation with Fuzzy Inference Systems. *Machines*, *11*(5). https://doi.org/10.3390/machines11050572

Naik, K. N., Patil, A. R., Patil, K. N., Sankhe, V. R., More, S. S., & Brian Lobo, V. (2023). A Python-based Grade Converter Application. *Proceedings of the 2023 2nd International Conference on Electronics and Renewable Systems, ICEARS 2023*, 180–184. https://doi.org/10.1109/ICEARS56392.2023.10084961

¿Qué es un sistema experto? Usos y aplicaciones en la IA. (n.d.). Retrieved May 24, 2023, from https://www.unir.net/ingenieria/revista/sistema-experto/

Sebastián Aldair Calderón Guerra, Luis Alberto Santos Correa, & Diego Patricio Pineda Maigua. (n.d.). Eficiencia del sistema de frenos en vehículos eléctricos. *Open Journal Systems*.

Singh, A. (2023). *EVALUATING USER-FRIENDLY DASHBOARDS FOR DRIVERLESS VEHICLES: EVALUATION OF IN-CAR INFOTAINMENT IN TRANSITION*. https://doi.org/10.25394/PGS.23750994.V1

Tecnológica Nacional, U., Regional Rosario Autor, F., & Juan Manuel, P. (n.d.). *Sistemas Expertos Sistemas Expertos Sistemas Expertos Sistemas Expertos (Expert System) (Expert System) (Expert System) (Expert System) Orientación I: Informática aplicada a la Ingeniería de Procesos 1 Ingeniería Química*.

The Basics — experta unknown documentation. (n.d.). Retrieved May 24, 2023, from https://experta.readthedocs.io/en/latest/thebasics.html

Timko, D., & Rahman, M. L. (2023). Commercial Anti-Smishing Tools and Their Comparative Effectiveness Against Modern Threats. *WiSec 2023 - Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 1–12. https://doi.org/10.1145/3558482.3590173

Vista de SEDFE: Un Sistema Experto para el Diagnóstico Fitosanitario del Espárrago usando Redes Bayesianas. (n.d.). Retrieved June 29, 2023, from https://dspace.palermo.edu/ojs/index.php/cyt/article/view/785/687