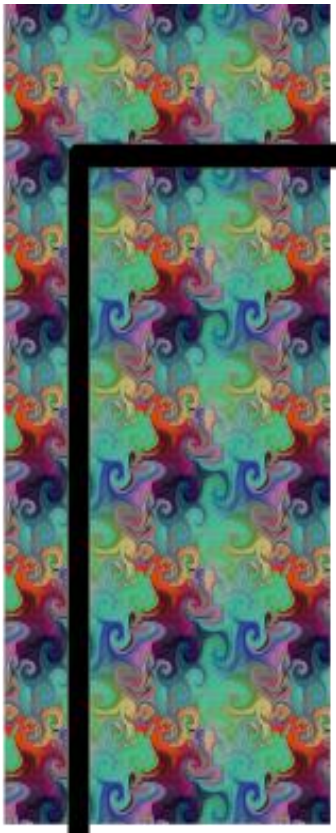


Actividad 2: Sistemas Expertos

CE Inteligencia Artificial y Big Data

Modelos de Inteligencia Artificial

2024/2025

Daniel Marín López

Índice

1. Problema 1

Se pretende construir un Sistema Experto que decida si se concede o no un crédito al consumo a cierto cliente. Este Sistema Experto debe trabajar como lo haría un bancario acostumbrado a dicha operación.
Un bancario con experiencia indica que únicamente se conceden este tipo de créditos a mayores de 18 años que dispongan de nómina y cuyo contrato sea bien de carácter indefinido o que la duración del mismo sea superior al tiempo necesario para la devolución de todas las cuotas del préstamo.

Para crear un Sistema Experto que determine si se concede o no un crédito al consumo a un cliente, basándose en los criterios establecidos por un bancario con experiencia podemos plantear las siguientes reglas:

- **SI** edad < 18 **ENTONCES** rechazar crédito (nos aseguramos que el cliente sea mayor de 18 años)
- **SI** no tiene nómina **ENTONCES** rechazar crédito (comprobamos si tiene nómina)
- Comprobamos que tipo de contrato tiene:
    - **SI** tipo de contrato == 'indefinido' **ENTONCES** aceptar condición laboral (se acepta condición)
    - **SI** tipo de contrato == 'temporal' **Y** duración del contrato >= duración del préstamo **ENTONCES** aceptar condición laboral (se acepta condición)
    - **SI** tipo de contrato == 'temporal' **Y** duración del contrato < duración del préstamo **ENTONCES** rechazar crédito (NO se acepta condición)
- **SI** edad >= 18 **Y** tiene nómina **Y** condición laboral aceptada **ENTONCES** conceder crédito (concedemos el crédito)

Un ejemplo que cumpliría estas condiciones sería:

- Edad: 30
- Nómina: Sí
- Tipo de contrato: temporal
- Duración del contrato: 24 meses
- Duración del préstamo: 18 meses

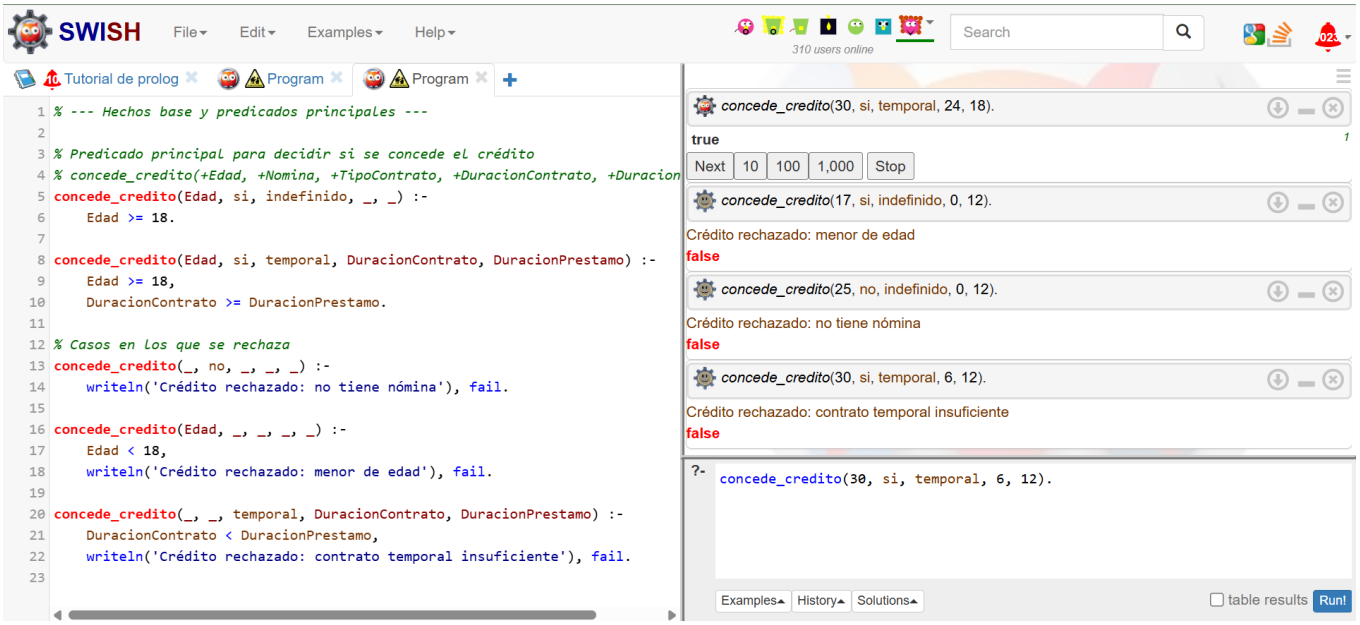
Para trasladar esto a Prolog con una plataforma como SWISH tenemos el siguiente conocimiento:

```
% --- Hechos base y predicados principales ---

% Predicado principal para decidir si se concede el crédito
% concede_credito(+Edad, +Nomina, +TipoContrato, +DuracionContrato, +DuracionPrestamo)
concede_credito(Edad, si, indefinido, _, _) :-
    Edad >= 18.

concede_credito(Edad, si, temporal, DuracionContrato, DuracionPrestamo) :-
    Edad >= 18,
    DuracionContrato >= DuracionPrestamo.

% Casos en los que se rechaza
concede_credito(_, no, _, _, _) :-
    writeln('Crédito rechazado: no tiene nómina'), fail.

concede_credito(Edad, _, _, _, _) :-
    Edad < 18,
    writeln('Crédito rechazado: menor de edad'), fail.
```

```
concede_credito(_, _, temporal, DuracionContrato, DuracionPrestamo) :-
    DuracionContrato < DuracionPrestamo,
    writeln('Crédito rechazado: contrato temporal insuficiente'), fail.
```

Y las siguientes consultas:

```
% Caso aceptado
?- concede_credito(30, si, temporal, 24, 18).

% Caso rechazado (menor de edad)
?- concede_credito(17, si, indefinido, 0, 12).

% Caso rechazado (sin nómina)
?- concede_credito(25, no, indefinido, 0, 12).

% Caso rechazado (contrato demasiado corto)
?- concede_credito(30, si, temporal, 6, 12).
```



Vemos que en varias consultas, la primera es el ejemplo anterior donde daba válido el crédito. Y las demás rechazan el crédito por distintos motivos.

2. Problema 2

Se desea desarrollar un Sistema Experto que evalúe solicitudes de admisión a una universidad. Este sistema considerará el desempeño académico del estudiante, resultados de exámenes estandarizados, y actividades extracurriculares, pero también admitirá excepciones para talentos especiales o circunstancias únicas.
El comité de admisiones indica que generalmente aceptan estudiantes con un promedio académico superior a 85/100, que hayan obtenido más de 1200 en el SAT (o su equivalente), participen activamente en actividades extracurriculares, o demuestren talentos excepcionales en áreas específicas como deportes, música, etc.

Para crear un sistema experto que evalúe las solicitudes de admisión y recomendar una decisión (Aceptar, Rechazar, Evaluación adicional) con base en criterios académicos, pruebas estandarizadas, actividades extracurriculares, y talentos especiales se pueden definir las siguientes reglas:

- **SI** Promedio ≥ 85 **Y** SAT ≥ 1200 **Y** Actividades extracurriculares = Sí **ENTONCES** → Aceptar
- **SI** Promedio ≥ 85 **Y** SAT ≥ 1200 **ENTONCES** → Evaluar participación extracurricular
- **SI** Promedio < 85 **O** SAT < 1200 **PERO** Talento especial = Sí **ENTONCES** → Enviar a evaluación especial
- **SI** Promedio < 85 **Y** SAT < 1200 **Y** Sin actividades extracurriculares **Y** Sin talento especial **ENTONCES** → Rechazar
- **SI** Circunstancia única documentada = Sí **ENTONCES** → Enviar a comité para revisión humana

Para hacer esto en SWISH podríamos hacer los siguiente:

1. Primero definimos un solicitante:

```
% Datos del solicitante: promedio, puntaje_SAT, extracurriculares, talento_especial,
circunstancia_unica
% Todos los valores son simbolizados para facilidad de evaluación

solicitante(
    nombre("Juan Perez"),
    promedio(88),
    sat(1300),
    extracurriculares(si),
    talento(no),
    circunstancia(no)
).
```

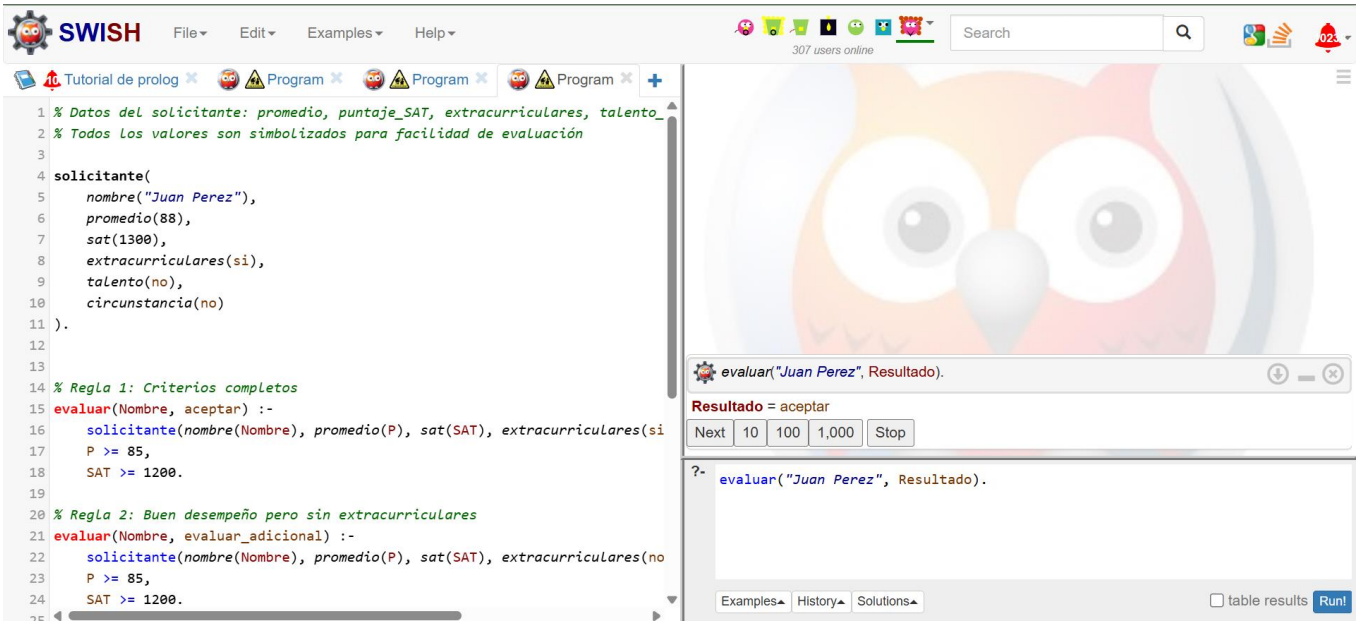
2. Definimos las reglas del sistema:

```
% Regla 1: Criterios completos
evaluar(Nombre, aceptar) :-
    solicitante(nombre(Nombre), promedio(P), sat(SAT), extracurriculares(si), talento(_),
circunstancia(_)),
    P >= 85,
    SAT >= 1200.

% Regla 2: Buen desempeño pero sin extracurriculares
evaluar(Nombre, evaluar_adicional) :-
    solicitante(nombre(Nombre), promedio(P), sat(SAT), extracurriculares(no), talento(_),
circunstancia(_)),
    P >= 85,
    SAT >= 1200.
```

```
% Regla 3: Talento especial compensa deficiencias
evaluar(Nombre, evaluar_especial) :-
    solicitante(nombre(Nombre), promedio(P), sat(SAT), extracurriculares(_), talento(si),
circunstancia(_)),
    (P < 85 ; SAT < 1200).

% Regla 4: Circunstancia única válida
evaluar(Nombre, comite) :-
    solicitante(nombre(Nombre), promedio(_), sat(_), extracurriculares(_), talento(_),
circunstancia(si)).

% Regla 5: No cumple con requisitos mínimos
evaluar(Nombre, rechazar) :-
    solicitante(nombre(Nombre), promedio(P), sat(SAT), extracurriculares(no), talento(no),
circunstancia(no)),
    P < 85,
    SAT < 1200.
```

3. Realizamos la consulta:

```
?- evaluar("Juan Perez", Resultado).
```



Aquí la consulta muestra que la solicitud ha sido aceptada.

3. Problema 3

Se busca crear un Sistema Experto que ayude a diagnosticar si un paciente tiene diabetes tipo 2. Este sistema evaluará factores como los niveles de glucosa en ayunas, la edad, el índice de masa corporal (IMC), el historial familiar y considerará un factor adicional: pacientes que ya presentan síntomas relacionados con la diabetes.
Los médicos consideran que un paciente es susceptible de tener diabetes tipo 2 si sus niveles de glucosa en ayunas son superiores a 126 mg/dL, tienen más de 45 años de edad, un IMC mayor de 25, antecedentes familiares de diabetes, o ya presentan síntomas de diabetes como sed excesiva, visión borrosa, o fatiga frecuente.

Para hacer un sistema experto sencillo basado en reglas para diagnosticar la posibilidad de que un paciente tenga diabetes tipo 2 se pueden definir las siguientes reglas:

- **SI** glucosa > 126 mg/dL → Riesgo alto de diabetes tipo 2
- **SI** edad > 45 Y IMC > 25 → Riesgo moderado de diabetes tipo 2
- **SI** antecedentes familiares → Riesgo moderado de diabetes tipo 2
- **SI** presenta síntomas relacionados → Riesgo moderado de diabetes tipo 2
- **SI** se cumplen al menos 3 factores → Riesgo alto de diabetes tipo 2
- **SI** no se cumple ningún factor → Riesgo bajo de diabetes tipo 2

Para hacer esto en SWISH haríamos lo siguiente:

1. Definir un paciente:

```
% Hechos: información del paciente (ejemplo)
paciente(glucosa_alta).
paciente(mayor_de_45).
paciente(imc_alto).
paciente(antecedentes_familiares).
paciente(sintomas_diabetes).
```

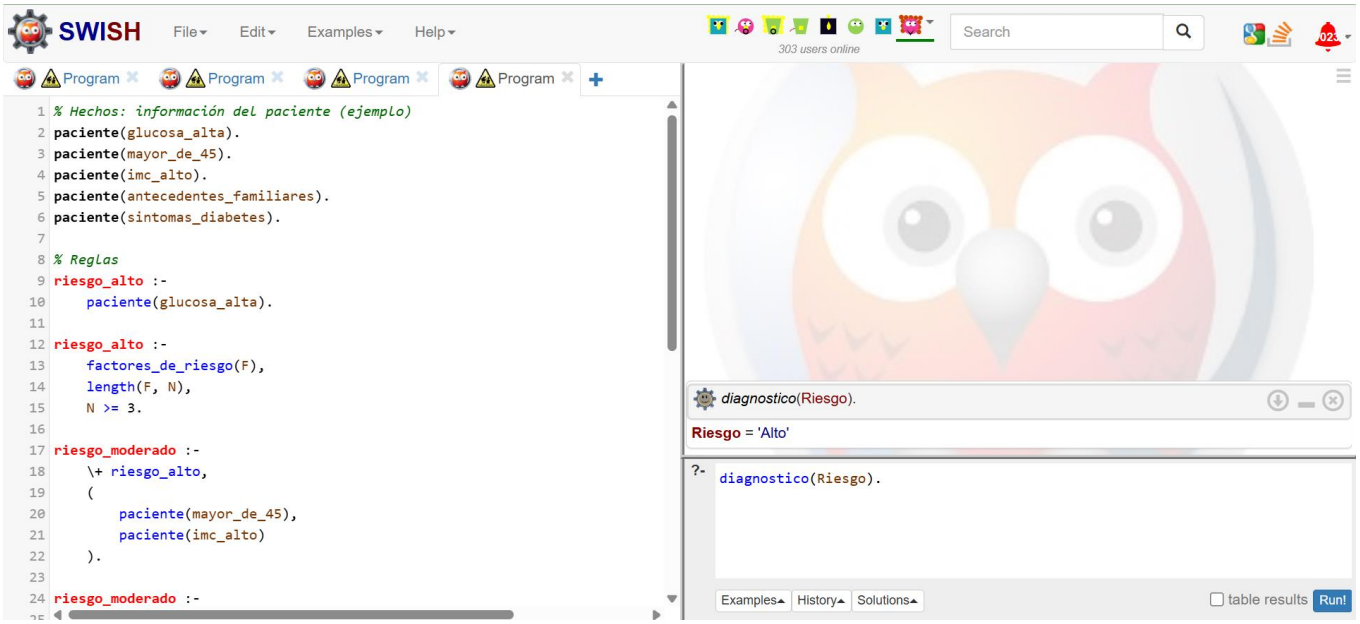
2. Definir las reglas del sistema:

```
% Reglas
riesgo_alto :-
    paciente(glucosa_alta).

riesgo_alto :-
    factores_de_riesgo(F),
    length(F, N),
    N >= 3.

riesgo_moderado :-
    \+ riesgo_alto,
    (
        paciente(mayor_de_45),
        paciente(imc_alto)
    ).

riesgo_moderado :-
    \+ riesgo_alto,
```

```
    paciente(antecedentes_familiares).

riesgo_moderado :-
    \+ riesgo_alto,
    paciente(sintomas_diabetes).

riesgo_bajo :-
    \+ riesgo_alto,
    \+ riesgo_moderado.

% Utilidad: lista de factores presentes
factores_de_riesgo(Factores) :-
    findall(Factor,
            (member(Factor, [mayor_de_45, imc_alto, antecedentes_familiares,
sintomas_diabetes]),
             paciente(Factor)),
            Factores).

% Diagnóstico general
diagnostico(Riesgo) :-
    (riesgo_alto -> Riesgo = 'Alto';
     riesgo_moderado -> Riesgo = 'Moderado';
     riesgo_bajo -> Riesgo = 'Bajo').
```

3. Realizamos la consulta

```
?- diagnostico(Riesgo).
```



Vemos que sale que el riesgo es alto porque su glucosa es demasiado alta.