



# Proyecto

[https://github.com/BoozSL/Sistemas\\_expertos](https://github.com/BoozSL/Sistemas_expertos)

Pedro Booz Salazar Ledezma

**Entregar un PDF en el cual den la explicación de sus sistema y demuestren el funcionamiento (hagan un vídeo subanlo a youtube y adjunten el enlace en el documento, así como el repositorio de GitHub al PDF del proyecto)**

**Codigo:**

```
import pandas as pd
import os
import tkinter as tk
from tkinter import messagebox
# =====
# 1. CARGA DE BASE DE DATOS
# =====
BASE_DIR = os.path.dirname(os.path.abspath(__file__))
CSV_PATH = os.path.join(BASE_DIR, "train.csv")
try:
    db = pd.read_csv(CSV_PATH)
except Exception as e:
    print("Error al cargar el CSV:", e)
    exit()
numeric_cols = [
    "Loan Amount", "Funded Amount", "Interest Rate",
    "Debit to Income", "Delinquency - two years",
    "Inquires - six months", "Revolving Balance",
    "Accounts Delinquent"
]
for col in numeric_cols:
    if col in db.columns:
        db[col] = pd.to_numeric(db[col], errors='coerce')
# =====
# TOOLTIP PARA LOS "?"
# =====

class ToolTip:
    def __init__(self, widget, text):
        self.widget = widget
        self.text = text
        self.tip_window = None
        widget.bind("<Enter>", self.show_tip)
        widget.bind("<Leave>", self.hide_tip)

    def show_tip(self, event=None):
        if self.tip_window:
            return
        x, y, _, _ = self.widget.bbox("insert")
        x += self.widget.winfo_rootx() + 25
        y += self.widget.winfo_rooty() + 20

        self.tip_window = tw = tk.Toplevel(self.widget)
        tw.wm_overrideredirect(True)
        tw.geometry(f"+{x}+{y}")

        label = tk.Label(tw, text=self.text, justify="left",
                        background="#ffffe0", relief="solid",
                        borderwidth=1, font=("Arial", 10))
        label.pack(ipadx=1)

    def hide_tip(self, event=None):
        tw = self.tip_window
        self.tip_window = None
        if tw:
            tw.destroy()
```

```

# =====
# 2. REGLAS DEL SISTEMA EXPERTO
# =====

def evaluate_rules(data):
    risk_score = 0

    if data["Debit to Income"] > db["Debit to Income"].mean():
        risk_score += 1

    if data["Inquires - six months"] > 3:
        risk_score += 1

    if data["Delinquency - two years"] > 0:
        risk_score += 2

    if data["Interest Rate"] > db["Interest Rate"].mean():
        risk_score += 1

    if data["Accounts Delinquent"] > 0:
        risk_score += 2

    if risk_score >= 4:
        return "ALTO RIESGO"
    elif risk_score >= 2:
        return "RIESGO MEDIO"
    else:
        return "BAJO RIESGO"

# =====
# 3. INTERFAZ CON TRADUCCIÓN
# =====

def create_ui():
    root = tk.Tk()
    root.geometry("520x650")

    # ----- Textos inglés y español -----
    texts_es = {
        "title": "Evaluación de Riesgo de Préstamo",
        "loan": "Monto del Préstamo",
        "interest": "Tasa de Interés (%)",
        "income": "Ingreso Mensual",
        "delinq2": "Retraso en 2 años",
        "inq": "Consultas en 6 meses",
        "revbal": "Balance Revolvente",
        "acctdel": "Cuentas Morosas",
        "eval": "Evaluar Riesgo",
        "lang_btn": "Cambiar a Inglés",
    }

    # Tooltips
    "t_loan": "Cantidad total solicitada del préstamo.",
    "t_interest": "Porcentaje anual que se paga por el préstamo.",
    "t_income": "Cuánto ganas al mes. Se usa para calcular tu DTI automáticamente.",
    "t_delinq2": "Número de veces que estuviste en mora en 2 años.",
    "t_inq": "Cuántas veces revisaron tu crédito en 6 meses.",
    "t_revbal": "Cantidad de Deuda revolvente (tarjetas, prestamos, etc).",
    "t_acctdel": "Número de cuentas actualmente en mora."
}

texts_en = {
    "title": "Loan Default Risk Assessment",
    "loan": "Loan Amount",
    "interest": "Interest Rate (%)",
    "income": "Monthly Income",
    "delinq2": "Delinquency - two years",
    "inq": "Inquires - six months",
}

```

```

"revbal": "Revolving Balance",
"acctdel": "Accounts Delinquent",
"eval": "Evaluate Risk",
"lang_btn": "Switch to Spanish",

# Tooltips
"t_loan": "Total loan amount requested.",
"t_interest": "Annual percentage rate charged on the loan.",
"t_income": "How much you earn monthly. Used to compute DTI automatically.",
"t_delinq2": "Delinquencies in the last 2 years.",
"t_inq": "Times your credit was checked in 6 months.",
"t_revbal": "Revolving debt such as credit cards.",
"t_acctdel": "Number of currently delinquent accounts."
}

current_lang = {"lang": "es"}

# ----- Función para traducir -----
def translate():
    lang = current_lang["lang"]
    new = "en" if lang == "es" else "es"
    current_lang["lang"] = new

    T = texts_en if new == "en" else texts_es

    root.title(T["title"])
    lbl_title.config(text=T["title"])
    btn_eval.config(text=T["eval"])
    btn_lang.config(text=T["lang_btn"])

    fields = [
        (lbl_loan, "loan"),
        (lbl_interest, "interest"),
        (lbl_income, "income"),
        (lbl_delinq2, "delinq2"),
        (lbl_inq, "inq"),
        (lbl_revbal, "revbal"),
        (lbl_acctdel, "acctdel")
    ]
    for lbl, key in fields:
        lbl.config(text=T[key])

    tooltips = [
        (tt_loan, "t_loan"),
        (tt_interest, "t_interest"),
        (tt_income, "t_income"),
        (tt_delinq2, "t_delinq2"),
        (tt_inq, "t_inq"),
        (tt_revbal, "t_revbal"),
        (tt_acctdel, "t_acctdel")
    ]
    for tip, key in tooltips:
        tip.text = T[key]

# =====
# Estructura visual
# =====

T = texts_es
root.title(T["title"])

lbl_title = tk.Label(root, text=T["title"], font=("Arial", 16))
lbl_title.pack(pady=15)

# Campo con etiqueta + entrada + botón tooltip
def make_field(name_key, tooltip_text):
    frame = tk.Frame(root)

```

```

frame.pack(pady=7)

lbl = tk.Label(frame, text=T[name_key] + ": ", width=25, anchor="w")
lbl.pack(side=tk.LEFT)

entry = tk.Entry(frame, width=15)
entry.pack(side=tk.LEFT)

q_btn = tk.Label(frame, text=" (?) ", fg="blue", cursor="question_arrow")
q_btn.pack(side=tk.LEFT)
tip = ToolTip(q_btn, tooltip_text)

return entry, lbl, tip

entry_loan, lbl_loan, tt_loan = make_field("loan", T["t_loan"])
entry_interest, lbl_interest, tt_interest = make_field("interest", T["t_interest"])
entry_income, lbl_income, tt_income = make_field("income", T["t_income"])
entry_delinq2, lbl_delinq2, tt_delinq2 = make_field("delinq2", T["t_delinq2"])
entry_inq, lbl_inq, tt_inq = make_field("inq", T["t_inq"])
entry_revbal, lbl_revbal, tt_revbal = make_field("revbal", T["t_revbal"])
entry_acctdel, lbl_acctdel, tt_acctdel = make_field("acctdel", T["t_acctdel"])

# Evaluar
def evaluar():
    try:
        income = float(entry_income.get())
        loan = float(entry_loan.get())
        rate = float(entry_interest.get())

        debt = loan * (rate / 100)
        dti = debt / income

        user_data = {
            "Loan Amount": loan,
            "Interest Rate": rate,
            "Debit to Income": dti,
            "Delinquency - two years": int(entry_delinq2.get()),
            "Inquires - six months": int(entry_inq.get()),
            "Revolving Balance": float(entry_revbal.get()),
            "Accounts Delinquent": int(entry_acctdel.get())
        }
    except:
        messagebox.showerror("Error", "Por favor ingresa valores válidos.")
        return

    result = evaluate_rules(user_data)
    messagebox.showinfo("Resultado",
                       f"DTI Calculado: {dti:.3f}\n\nResultado del sistema experto:\n\n{result}")

btn_eval = tk.Button(root, text=T["eval"], font=("Arial", 14),
                     bg="lightblue", command=evaluar)
btn_eval.pack(pady=25)

# Botón para cambiar idioma
btn_lang = tk.Button(root, text=T["lang_btn"], command=translate)
btn_lang.pack(pady=10)

root.mainloop()

# =====
# 4. EJECUCIÓN
# =====

if __name__ == "__main__":
    create_ui()

```

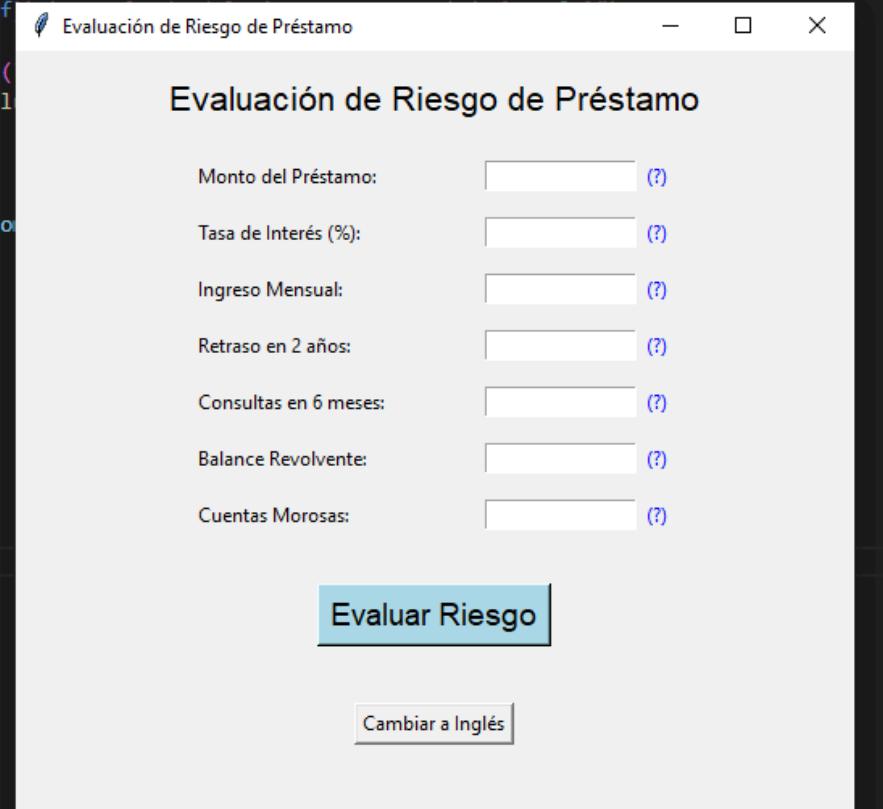
Interfaz (Tanto inglés como español)

Evaluación de Riesgo de Préstamo

Monto del Préstamo:	<input type="text"/> (?)
Tasa de Interés (%):	<input type="text"/> (?)
Ingreso Mensual:	<input type="text"/> (?)
Retraso en 2 años:	<input type="text"/> (?)
Consultas en 6 meses:	<input type="text"/> (?)
Balance Revolvente:	<input type="text"/> (?)
Cuentas Morosas:	<input type="text"/> (?)

**Evaluar Riesgo**

**Cambiar a Inglés**

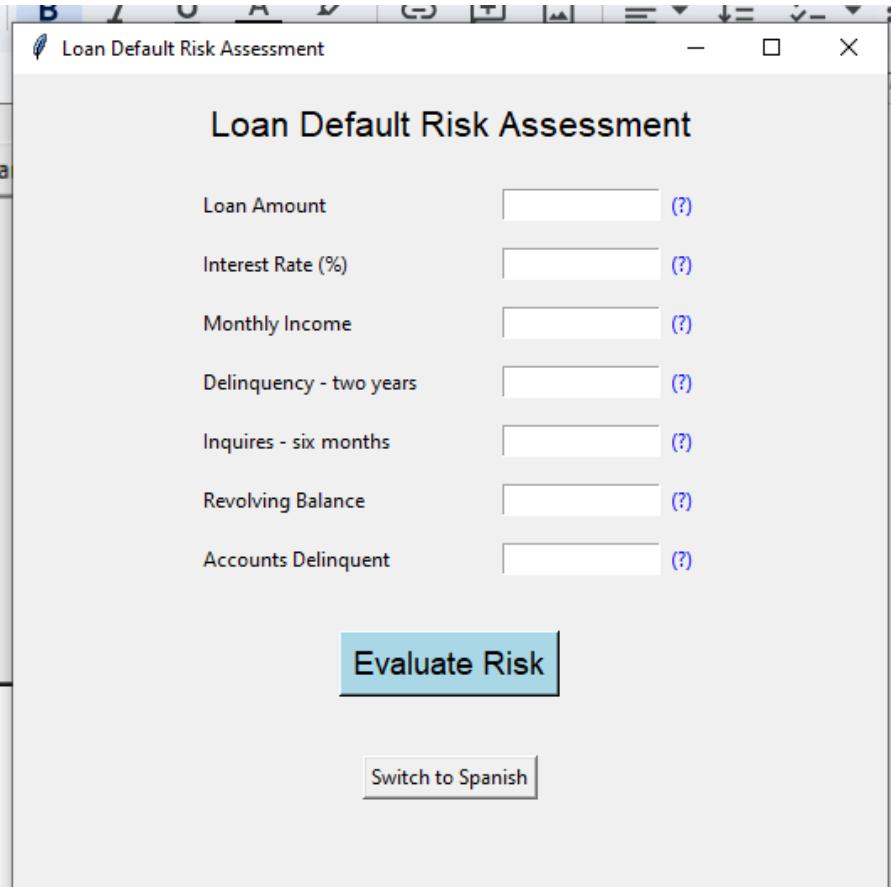


Loan Default Risk Assessment

Loan Amount	<input type="text"/> (?)
Interest Rate (%)	<input type="text"/> (?)
Monthly Income	<input type="text"/> (?)
Delinquency - two years	<input type="text"/> (?)
Inquires - six months	<input type="text"/> (?)
Revolving Balance	<input type="text"/> (?)
Accounts Delinquent	<input type="text"/> (?)

**Evaluate Risk**

**Switch to Spanish**



## **1.Carga del archivo CSV**

Primero, el programa busca el archivo train.csv en la misma carpeta donde está el script. Luego intenta cargarlo con pandas. Si falla, imprime el error y detiene la ejecución.

Después convierte a números algunas columnas del CSV, porque a veces vienen como texto y eso impediría hacer cálculos. Si alguna columna no existe, simplemente la ignora.

## **2. Sistema de ToolTips**

Aquí se define una pequeña clase llamada ToolTip.

Esta clase permite que cuando pasas el mouse sobre un símbolo de “(?)”, aparezca una ventanita explicando qué significa ese campo.

Tkinter no tiene tooltips por defecto, así que esta clase los crea manualmente usando una ventana emergente muy pequeña.

## **3. Reglas del sistema experto**

La función evaluate\_rules() recibe un diccionario con la información ingresada por el usuario.

Evaluá varios puntos:

- Si el DTI (debt-to-income) es mayor que el promedio de la base de datos → suma 1 punto.
- Si tiene más de 3 consultas de crédito → suma 1.
- Si ha tenido moras en 2 años → suma 2.
- Si su tasa de interés es más alta que el promedio → suma 1.
- Si tiene cuentas morosas → suma 2.
- Al final, según el total de puntos:
  - 4 o más = ALTO RIESGO
  - 2 a 3 = RIESGO MEDIO
  - Menos de 2 = BAJO RIESGO

Este es el “motor de inferencia”

## **4. Interfaz gráfica con traducción**

La función create\_ui() construye la ventana:

Se crean dos diccionarios (texts\_es y texts\_en) con todos los textos necesarios: etiquetas, títulos, botones y tooltips y por consiguiente hay un botón para cambiar idioma: Cuando presionas el botón, la función translate() cambia todos los textos visibles según el idioma seleccionado.

## **5. Creación de cada campo de entrada**

La función make\_field() sirve para no repetir código.

Cada fila del formulario contiene:

- una etiqueta,
- una caja de texto,
- un botón “(?)” con tooltip.
- Se usa para crear campos como:
- monto del préstamo
- interés
- ingreso mensual
- consultas
- moras
- etc.

## **6. Cálculo del DTI automático**

Cuando el usuario presiona “Evaluar riesgo”, la función evaluar():

Lee los valores ingresados.

Calcula el DTI automáticamente:

```
debt = loan * (rate / 100)
```

```
dti = debt / income
```

Construye un diccionario con los valores que el sistema experto necesita.

Llama a evaluate\_rules().

Muestra un cuadro con:

El DTI calculado, el nivel de riesgo final, Si algún valor no es válido, aparece un mensaje de error.

## **7. Ejecución del programa**