from ml\_model import predict\_ml

import csv

from datetime import datetime

# Parametri di input (esempio)

frequenza\_storica = 0.39

ritardo\_attuale = 16

ritardo\_medio = 11

ritardo\_massimo = 18

quota = 3.20

unit\_base = 5

storico\_esiti = []

# Funzioni metodo classico

def indice\_anomalia\_ritardo(ritardo\_attuale, ritardo\_medio, ritardo\_massimo):

delta = (ritardo\_attuale - ritardo\_medio) / (ritardo\_massimo - ritardo\_medio + 0.01)

return max(0.0, min(delta, 1.0))

def calcola\_probabilita\_migliorata(frequenza\_storica, ritardo\_attuale, ritardo\_medio, ritardo\_massimo):

anomalia = indice\_anomalia\_ritardo(ritardo\_attuale, ritardo\_medio, ritardo\_massimo)

boost = 1 + (anomalia \* 0.10)

p = frequenza\_storica \* boost

return min(p, 1.0)

def calcola\_ev(p, quota):

return p \* (quota - 1) - (1 - p)

def calcola\_puntata\_dalembert(esiti, unit\_base):

step = sum(1 if not esito else -1 for esito in esiti)

step = max(step, 0)

return unit\_base \* (1 + step)

# Combina probabilità

def combina\_probabilita(p\_classica, p\_ml, peso\_classica=0.6, peso\_ml=0.4):

return (peso\_classica \* p\_classica) + (peso\_ml \* p\_ml)

# Funzione di log automatico nel file CSV

def log\_scommessa(data, p\_classica, p\_ml, p\_comb, ev\_comb, quota, puntata, risultato=""):

nome\_file = "log\_scommesse.csv"

intestazione = ["DataOra", "Prob\_Classica", "Prob\_ML", "Prob\_Combinata", "EV\_Combinato", "Quota", "Puntata", "Risultato"]

try:

# Verifica se il file esiste

with open(nome\_file, "r", newline="") as file:

pass

except FileNotFoundError:

# Se non esiste, crea e scrive l'intestazione

with open(nome\_file, "w", newline="") as file:

writer = csv.writer(file)

writer.writerow(intestazione)

# Scrive i dati della scommessa

with open(nome\_file, "a", newline="") as file:

writer = csv.writer(file)

writer.writerow([data, f"{p\_classica:.4f}", f"{p\_ml:.4f}", f"{p\_comb:.4f}", f"{ev\_comb:.4f}", f"{quota:.2f}", f"{puntata:.2f}", risultato])

# --- ESECUZIONE ---

p\_classica = calcola\_probabilita\_migliorata(frequenza\_storica, ritardo\_attuale, ritardo\_medio, ritardo\_massimo)

p\_ml = predict\_ml(ritardo\_attuale, ritardo\_medio, ritardo\_massimo, frequenza\_storica, quota)

p\_comb = combina\_probabilita(p\_classica, p\_ml)

ev\_comb = calcola\_ev(p\_comb, quota)

soglia\_ev = 0.05

giocare = ev\_comb >= soglia\_ev

print(f"Probabilità classica: {p\_classica:.2%}")

print(f"Probabilità ML: {p\_ml:.2%}")

print(f"Probabilità combinata: {p\_comb:.2%}")

print(f"Valore Atteso combinato (EV): {ev\_comb:.3f}")

print(f"Giocare? {'SÌ' if giocare else 'NO'}")

if giocare:

puntata = calcola\_puntata\_dalembert(storico\_esiti, unit\_base)

print(f"Puntata consigliata: €{puntata:.2f}")

else:

puntata = 0

print("Non giocare: EV insufficiente.")

# Registra il log (risultato da aggiornare a posteriori)

log\_scommessa(datetime.now().strftime("%Y-%m-%d %H:%M:%S"), p\_classica, p\_ml, p\_comb, ev\_comb, quota, puntata, risultato="")

# Nota:

# Quando sai l'esito della scommessa aggiorna la riga nel file CSV o crea una funzione separata per aggiornare

# oppure aggiungi una nuova riga con l'esito corretto dopo la partita.

Cosa fa il log?

Se il file log\_scommesse.csv non esiste, lo crea con intestazione

Appende ogni volta una riga con i valori di probabilità, EV, quota, puntata e risultato (inizialmente vuoto)

Puoi aggiornare poi il risultato (vinto/perso) in un secondo momento, oppure aggiungere nuove righe dopo ogni partita.