M3 Challenge 2026:

The Rise of Online Gambling: *What's at Stake?*

Team #18612

Executive Summary

The global gambling market surpassed $643 billion in 2025, with online gambling accounting for $121 billion of it. Due to the ease of accessing online platforms and the huge range of things to bet on, the market is only growing over time. Sports betting alone takes up a 29% share in the European betting market, and individual events like the 2026 Super Bowl had over $1.7 billion bet on them. With this growth in gambling comes increased financial risk, particularly for younger individuals and lower-income groups, who are statistically more vulnerable to long-term losses.

Since betting expenditure is funded from disposable income, we modelled disposable income using linear regression based on data from England and Wales, taking two inputs — age and income — to predict how much money would be left over after tax and essential spending. The model shows a strong positive correlation between income and disposable income: for every extra £1 earned, approximately 52p becomes disposable income, reflected in a coefficient of 0.5206. The model also shows a slight negative correlation from the interaction term (age × income) of −0.000251, suggesting that older households benefit marginally less from each additional £1 of income than younger ones — consistent with higher fixed costs such as insurance and healthcare. The model achieves an R^2 of 0.9808 on unseen test data, confirming strong predictive accuracy across the full range of ages and income levels in the dataset.

When considering how much an individual will gain or lose through sports betting, we consider the strategy of the house. Operators offer a free betting bonus averaging roughly £60 on new sign-ups, which temporarily offsets the expected loss in year one for low-frequency bettors, creating a positive expected first-year experience. This functions as a behavioural acquisition mechanism rather than a genuine profitability opportunity — it disappears entirely in year two, after which all demographic profiles show net losses. Regardless of short-term outcomes, the structural house edge of approximately 10% means expected returns remain negative over time. This base edge varies with education and research habits: the effective house edge ranges from 7% for BA-educated bettors who research regularly to 12% for those with no college degree who never research. Beyond the house edge, behavioural patterns such as loss chasing introduce exponential exposure growth through geometric stake doubling — a small decrease in win probability from 0.15 to 0.05 increases expected chasing exposure by more than 10×. For a moderate bettor betting 1–2 times per week on a £35,000 salary, the model predicts annual losses of £522.

Finally, we investigated the pension pot gap caused by gambling, designing a model that integrates the disposable income model from Part 1 and the non-linear loss model from Part 2 to quantify the long-run effect of sustained gambling on an individual's pension pot by age 67 — the age at which the UK state pension becomes available. We introduced a compounding factor to capture how money lost to gambling forgoes future investment growth: the younger a gambler is when they start, the greater the pension damage, since each pound lost is denied decades of compound returns. For a moderate bettor starting at 25, an annual loss of £522 produces a pension gap of £120,811 by age 67 — a 5.5× compounding multiplier on the £21,924 nominally lost to gambling over a lifetime. For a problem gambler betting daily with severe escalation, the gap reaches £401,679, effectively wiping out 58.8% of what their pension pot would otherwise have been. These findings suggest that the true cost of gambling is not the money lost today, but the retirement security forfeited tomorrow.

Table of Contents

Q1: Playing with House Money

1.1 Defining the Problem

We are tasked with making a model that predicts the disposable income of a household. We chose to look at weekly disposable income, which we defined as the money left after taxes and essential spending. The output of this model feeds directly into Q2, where disposable income is used as the budget constraint on annual gambling expenditure.

1.2 Assumptions

1-1. Households have no additional income from benefits or allowances.

- **Justification:** Benefits and allowances would either decrease the cost of necessities or increase effective income. We exclude them due to a lack of sufficiently granular data — the available dataset does not break down benefit receipts at the income and age group level required by the model.

1-2. The households' only debt repayments are their mortgages, included as part of housing costs.

- **Justification:** Mortgage repayments are the dominant form of debt for most UK households and are captured within the housing expenditure category. Other forms of debt such as car finance or personal loans vary widely across individuals and are not available in the dataset at the required level of disaggregation.

1-3. The dataset is representative of the wider population.

- **Justification:** If certain age or income groups are under-represented, predictions for those groups will be less reliable. The data we found did not report sample sizes broken down to every age-income combination, so we assume the ONS survey methodology produces a broadly representative sample.

1-4. Only income is spent on essential goods, not savings.

- **Justification:** We only had data for income, not for savings behaviour. The model therefore treats all non-essential expenditure as available disposable income, which is the standard definition used in ONS household finance statistics.

1.3 Variables

Variable	Definition	Units
I	Income	£/wk
D	Disposable income	£/wk
T	Gross tax (National Insurance contribution included)	£/wk
C_e	Cost of essentials	£/wk
A	Age	years

Table 1: Variable definitions for Question 1

1.4 The Model

1.4.1 Developing the Model

We began by considering the demographics we would use as inputs to the model. After income, we decided on age, since age is a significant factor in lifestyle and therefore a considerable factor in disposable income. Older generations are more likely to own their home outright and thus spend less on housing costs. Similarly, young adults might be more likely to live with housemates and thus also spend less on housing costs per person. We also considered using region as an input, as that would affect cost of living, however we decided against it due to the lack of data available surrounding expenditure by region.

We calculated disposable income with the equation:

$$D = I - (T + C\_e)$$

We planned to fit a linear regression model to the data, so we plotted income on the x-axis against disposable income on the y-axis. After plotting the data it was clear that a linear function was appropriate.

1.4.2 Executing the Model

After identifying income and age as the two key predictors, we fitted a multiple linear regression model with an interaction term to capture the age-income relationship:

$$D = \beta_0 + \beta_1 \cdot A + \beta_2 \cdot I + \beta_3 \cdot (A \times I)$$

$$D = -47.84 + 0.5514 \cdot A + 0.5216 \cdot I - 0.000251 \cdot (A \times I)$$

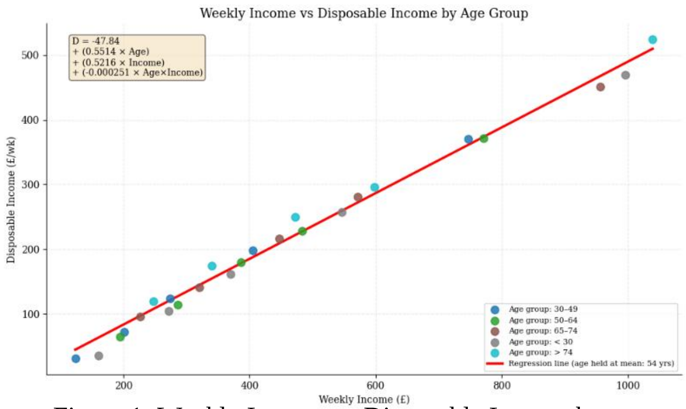


Figure 1: Weekly Income vs Disposable Income by age group

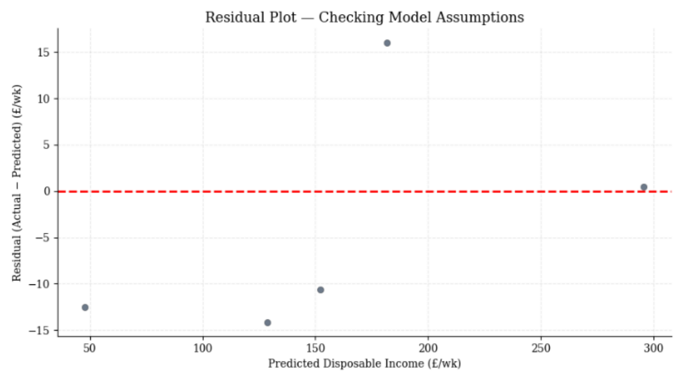


Figure 2: Residual plot to check assumptions

Where D is the predicted weekly disposable income (£/wk), β_0 is the intercept, β_1 is the coefficient on age (midpoint), β_2 is the coefficient on weekly income (£), and β_3 is the coefficient on the interaction term between age and income.

Fitting this linear model gives an R^2 of 0.9952 on the training set and 0.9808 on the test set. The small gap between train and test performance confirms the model is not overfitting, a test R^2 of 0.9808 means the model explains 98.1% of the variance in disposable income on unseen data.

1.5 Results

According to the model, a 27-year-old with an income of £500/wk would have a disposable income of £224.46/wk. This is reasonable as it is approximately 50% of income after tax. The difference between income and disposable income is nearly all taken up by rent and mortgages, with a lower proportion for food, giving a realistic figure for the disposable income of a young adult.

The model also gives reasonable predictions for older age groups. A 71-year-old on £1,200/wk would have a disposable income of £595/wk, which is around 70% of after-tax income. This is reasonable as most elderly people own their home outright so housing costs are significantly lower than for younger people. Food costs do not increase as quickly with income as housing costs do, since people tend to spend proportionally more of additional income on better housing rather than food (and because non-essential food products such as alcohol are discretionary rather than essential).

1.6 Model Validation

To validate the model, we examine the R^2 values, sensitivity analysis, and residual plots in combination, as no single metric alone is sufficient to confirm a model's reliability.

The model achieves an R^2 of 0.9952 on the training set and 0.9808 on the test set, giving a gap of just 0.0144. A test R^2 of 0.9808 means the model explains 98.1% of the variance in disposable income on data it has not seen, which is a strong result for household financial data. The MAE of £10.76/wk further supports this, predictions deviate from actual values by roughly £10.76 on average, representing only 2.4% of the mean weekly income of £457, indicating the model is accurate in practical terms.

To test whether the model responds logically to changes in its inputs, we conducted a sensitivity analysis holding each variable at the mean while varying the other. At the mean values of age 54 and weekly income £457, the model predicts a disposable income of £214.28/wk. When income is increased while age is held constant, disposable income rises smoothly and linearly, which is the expected behaviour. The income coefficient of 0.52 implies that for every additional £1 earned, approximately 52p becomes disposable income, with the remaining 48p attributed to essential costs on average.

When age is varied while income is held constant, the model shows a slight negative relationship, meaning older households retain marginally less disposable income at the same income level. This is consistent with real-world expectations, where older households tend to carry higher fixed costs such as insurance and healthcare premiums. The Age × Income interaction term of -0.000251 captures this: there is a marginal negative effect when income increases at higher ages.

Examining the residual plot, the residuals are randomly scattered around zero across the full range with no visible trend. There are no systematic curves in the residuals, which validates the assumption that a linear model is appropriate for this relationship. No large outliers are visible that would suggest the model is being pulled by extreme data points. Taken together, the residual plot supports the conclusion that the model assumptions are satisfied and predictions are reliable across the full range of incomes in the dataset.

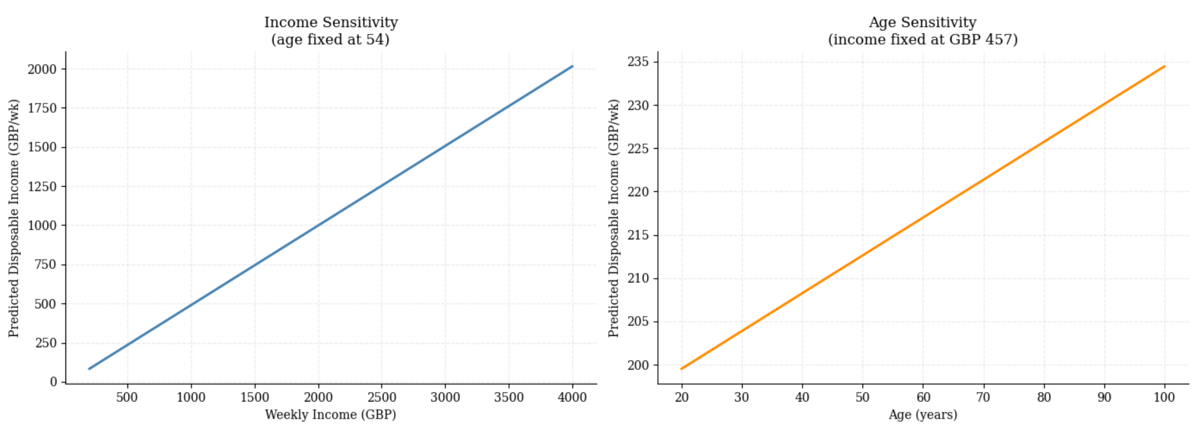


Figure 3: Income and Age Sensitivity

1.7 Strengths and Weaknesses

The strength of our model comes from its ability to predict disposable income across a wide range of demographics. It is also easy to adapt for other countries as it only requires two inputs — income and age — which minimises the amount of data needed to make an accurate prediction. The model uses 25 data points to fit, which is sufficient to produce a robust regression with the degrees of freedom available.

However, the model is limited by the same simplicity that makes it flexible. Considering only two inputs (and not others such as location, debt, or homeownership) means it is generalised for England and Wales as a whole and does not differentiate between obvious differences in cost of living between heavily urbanised areas such as London and rural areas such as Cumbria. This limitation extends to other variables like non-mortgage debt: if someone has significant repayments for a car or personal loan, that represents a necessary cost our model does not capture. If we had more time, we would wish to add another input such as location, due to its significance for cost of living and therefore disposable income.

Q2: Know The Spread

2.1 Defining the Problem

For Question 2, we were tasked with developing a model to predict an individual's profits or losses from online sports gambling over the course of one year, based on their demographic profile and self-reported behavioural tendencies. Our model extends a standard expected value framework by incorporating two phenomena that make real gambling behaviour fundamentally non-linear: addiction-driven stake escalation and loss chasing. The central finding is that the house always wins in expectation, but the non-linear components explain why real losses can be an order of magnitude larger than the baseline 10% house edge alone predicts.

2.2 Assumptions

2-1. The base house edge is 10%, derived from the problem statement data.

- **Justification:** \$150 billion was wagered on online sports betting in 2024, generating \$15 billion in revenue for operators (problem statement). This gives h_base = \$15bn / \$150bn = 10%. This structural advantage is built into every bet through odds set below fair value — the overround — and applies regardless of the bettor's strategy.

2-2. Better-informed bettors face a lower effective house edge.

- **Justification:** Survey rows 12–16 show frequent researchers bet more accurately, and row 29 shows BA+ bettors are 13 percentage points more likely to report winning. These adjustments shift h_eff between 7% and 12% depending on education and research habits, rather than treating all bettors identically at 10%.

2-3. Stake size scales linearly with disposable income and aggression score.

- **Justification:** A bettor with higher disposable income can afford larger stakes, and higher aggression reflects a deliberate choice to wager a greater fraction of income per bet.

2-4. Addiction escalation follows an exponential growth function over the year.

- **Justification:** Tolerance build-up is a clinical feature of gambling disorder, requiring progressively larger stakes for the same psychological effect.

2-5. Individual bets are modelled as independent events.

- **Justification:** Independence is the standard assumption in expected value analysis and is mathematically necessary for the geometric distribution of losing streak lengths. Parlays and correlated accumulators would increase variance but are not captured in this model.

2.3 Variables

Symbol	Definition	Units / Source
P&L	Annual profit or loss from gambling	£/year — primary output
B	Free bet bonus for new customers	£
h_base	Base house edge = $15bn / $150bn	Dimensionless
h_eff	Effective house edge = h_base + edu_adj + res_adj	Dimensionless
a	Aggression score (1=cautious, 10=high risk)	Integer 1–10
p	Win probability = 0.45 − (a−1) × (0.40/9)	Dimensionless
s0	Base stake per bet	£/bet
lambda	Addiction escalation rate parameter	Dimensionless
s_avg	Average escalated stake = s0 × (exp(lambda)−1) / lambda	£/bet
n	Bets per year — scaled by demographic multipliers	Integer
r_chase	Demographic loss-chasing rate	Proportion
k	Expected losing streak length = (1−p)/p	Dimensionless
N_ep	Chasing episodes per year = n × r_chase × 0.15	Integer
DI	Annual disposable income — from Q1 model	£/year

Table 2: Variable definitions for Question 2

2.4 The Model

2.4.1 Developing the Model

We define annual profit or loss from gambling per person as:

```
P&L = B - L_base - L_chasing - L_escalation - L_debt
```

We chose these five terms because together they capture the complete financial story of gambling for any demographic profile. B is included because the new customer bonus makes year 1 appear profitable for low-frequency bettors. This stops the model missing the deliberate acquisition trap set by operators. L_base is the unavoidable mathematical floor that proves the house always wins by construction. L_chasing is included because survey row 40 shows 41–58% of bettors chase losses, so ignoring it would massively understate real losses for the majority of gamblers. L_escalation is included because without it the model predicts identical losses for a recreational bettor and an addicted one on the same salary, which is clearly wrong. L_debt accounts for the 2–3% problem gamblers (Lancet ref 9) who bet beyond their income entirely — the most financially catastrophic group.

We considered a purely linear model where annual loss = n × s × h_eff — but rejected it because it produces the same loss rate regardless of behavioural inputs. Doubling bets simply doubles the loss. Real gambling behaviour is fundamentally non-linear: chasing produces a geometric explosion in stakes and escalation compounds the stake baseline exponentially. The non-linear model captures all three.

2.4.2 Executing the Model

Stage 1: Effective house edge. The base house edge h_base = 10% is adjusted per individual based on education and research habits:

```
h_eff  =  h_base  +  edu_adj  +  res_adj
```

Factor	Adjustment	Rationale
BA or higher education	**−2%**	Survey row 29: BA+ bettors 13pp more likely to report winning
No college degree	**+1%**	Less-informed decisions; lower ability to identify value bets
Researches 2x+ per day	**−1%**	Survey rows 12–16: frequent researchers show better accuracy
Never researches	**+1%**	Purely recreational betting with no analytical basis

Table 3: Effective house edge adjustments by education and research frequency

Therefore h_eff ranges from 7% (BA+, researches regularly) to 12% (no college, never researches). We can also prove that E(V) = −h_eff regardless of aggression score. With win probability p = 0.45 − (a−1) × (0.40/9) and multiplier = (1−h_eff)/p:

```
E(V) per £1  =  p × (1 − h_eff)/p  −  (1 − p)  =  −h_eff
```

This holds for all values of p. There is no betting strategy that produces a positive E(V) — risk profile determines volatility of outcomes but not the loss rate per pound wagered.

Stage 2: Base stake and bets per year. The base stake scales with aggression score and monthly disposable income:

```
s0  =  [ 0.005  +  (a−1) × (0.045/9) ]  ×  (DI / 12)
```

This gives stake percentages from 0.5% of monthly DI at aggression 1 to 5% at aggression 10. Bets per year are derived from a frequency input scaled by demographic account-ownership multipliers, using age 35–49 as the 1.00× baseline reference multiplier. 22% is the overall participation rate across both genders, so by dividing by 22% we could compare how active this gender is compared to the average person.

```
n  =  n_freq  ×  (age account rate / 34%)  ×  (gender account rate / 22%)
```

Frequency Input	Base n/yr	Account Rate	Demographic Multipliers
< Once/month	6	—	Age 18-34: 35% → 1.03×
Once/month	12	—	Age 35-49: 34% → 1.00× (reference)
Few times/month	30	22%	Age 50-64: 10% → 0.29×
1–2x per week	78	—	Age 65+: 4% → 0.12×
3–4x per week	182	—	Male: 30% → 1.36×
Daily	365	—	Female: 15% → 0.68×

Table 4: Frequency inputs and demographic multipliers for bets per year

Stage 3: Free bet bonus (B). New customers receive a one-off promotional bonus upon registration:

$$B = £60 \quad \text{if new customer,} \quad \text{else} \quad B = 0$$

£60 is the midpoint of the UK average new customer bonus range of £50–£75.

Stage 4: Addiction escalation (L_escalation). Problem gambling is characterised by tolerance build-up, requiring progressively larger stakes for the same psychological effect. We model this as exponential stake growth over the year:

$$s(t) = s0 \times \exp(\lambda \times t) \qquad t \in [0, 1]$$

The average stake over the full year is the integral of s(t) over [0,1]:

$$s\_avg = s0 \times (\exp(\lambda) - 1) / \lambda$$

This is always greater than s0 for any lambda > 0. The additional loss from escalation above the flat-stake baseline is:

$$L\_escalation = n \times (s\_avg - s0) \times h\_eff$$

Level	lambda	Year-end stake	Interpretation
None	0.00	1.00 × s0	Linear model — stake unchanged all year
Mild	0.03	1.03 × s0	Slight creep — recreational gambler
Moderate	0.07	1.07 × s0	Noticeable escalation — emerging dependency
Severe	0.15	1.16 × s0	Significant addiction — rapidly growing exposure

Table 5: Addiction escalation parameter values and interpretation

Stage 5: Loss chasing (L_chasing). Loss chasing is where a larger bet is placed after a loss to recover previous losses.

$$s\_k = s \times 2^k \qquad \text{(stake after k consecutive losses)}$$

The expected length of a losing streak before a win follows a geometric distribution with mean E[k] = (1−p)/p. For a high-risk gambler (a=10, p=0.05), E[k] = 19 (meaning 19 consecutive losses before a win on average). The total extra stake in one chasing episode is the geometric series sum:

$$\text{Extra stake (one episode)} = s \times (2 + 4 + \ldots + 2^k) = s \times (2^{(k+1)} - 2)$$

$$L\_chasing = N\_ep \times s\_avg \times (2^{(k+1)} - 2) \times h\_eff$$

Where N_ep = n × r_chase × 0.15 is the number of chasing episodes per year, and r_chase is the demographic chase rate from the survey:

Demographic	Chase Rate	Source	Notes
Age 18–34	**58%**	Survey row 40	Highest rate — impulsive, high FOMO
Age 35–49	**54%**	Survey row 40	Still majority chase losses

Demographic	Chase Rate	Source	Notes
Age 50–64	38%	Survey row 40	Reduced chasing with age
Age 65+	15%	Survey row 40	Lowest rate of all groups
Male	**57%**	Survey row 40	Substantially higher than female
Female	41%	Survey row 40	Lower but still substantial

Table 6: Loss chasing rates by demographic group

Stage 6: Debt and addiction extra (L_debt). Problem gamblers bet beyond their disposable income using credit or savings. If flagged, an additional loss term is applied:

$$L\_debt = 0.50 \times DI \times h\_eff$$

This models an additional 50% of DI wagered on credit. Problem gambling prevalence is approximately 2–3% of the adult population. The Kellogg Northwestern study (ref 10) documents a measurable reduction in household savings for this group.

2.5 Results

Substituting all terms into the core equation:

$$P\&L = B - n \cdot s\_avg \cdot h\_eff - N\_ep \cdot s \cdot (2^{(k+1)}-2) \cdot h\_eff - 0.50 \cdot DI \cdot h\_eff \cdot [debt]$$

Factoring out h_eff from all loss terms, the full factored model equation therefore is:

$$P\&L = B - h\_eff \times [n \cdot s\_avg + N\_ep \cdot s \cdot (2^{(k+1)}-2) + 0.50 \cdot DI \cdot [debt]]$$

This factored form shows that h_eff multiplies the entire bracket (a governing parameter of the model). Inside the bracket, the loss chasing term (2^(k+1) − 2) is a geometric series that grows exponentially with k = (1−p)/p. For high aggression scores where p is small, this term dominates (a single change from aggression 5 to aggression 9 increases the chasing bracket from 6× to 126×). This superlinear growth is the core non-linearity of the model, thus highlighting the extreme losses experienced by bettors.

Table 6 below presents predicted annual losses for three representative profiles (male, 18-34, no college degree, occasional research, no chasing, new customer in year 1):

Profile	Salary	Frequency	Aggression	Escalation	Annual Loss	h_eff
Casual bettor	£28,000	Once/month	3	None	£65/yr	11%
Moderate bettor	£35,000	1–2x/week	5	None	£522/yr	11%
Frequent bettor	£35,000	3–4x/week	7	Mild	£1,735/yr	11%

Table 7: Predicted annual gambling loss for three representative profiles

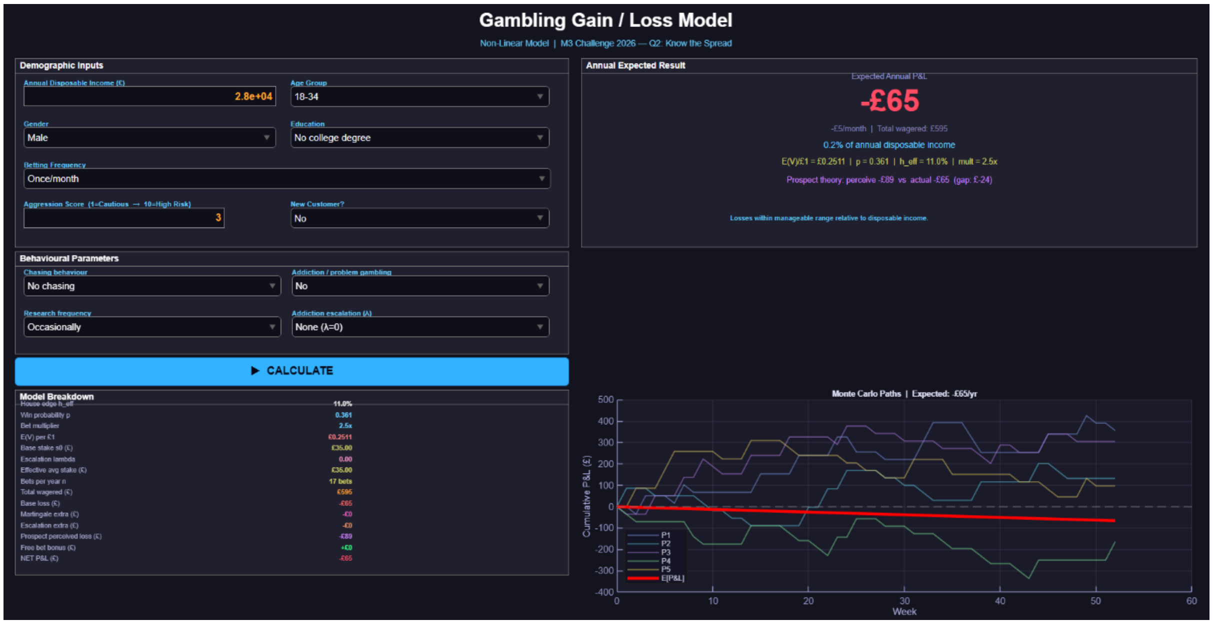


Figure 4: Casual Bettor

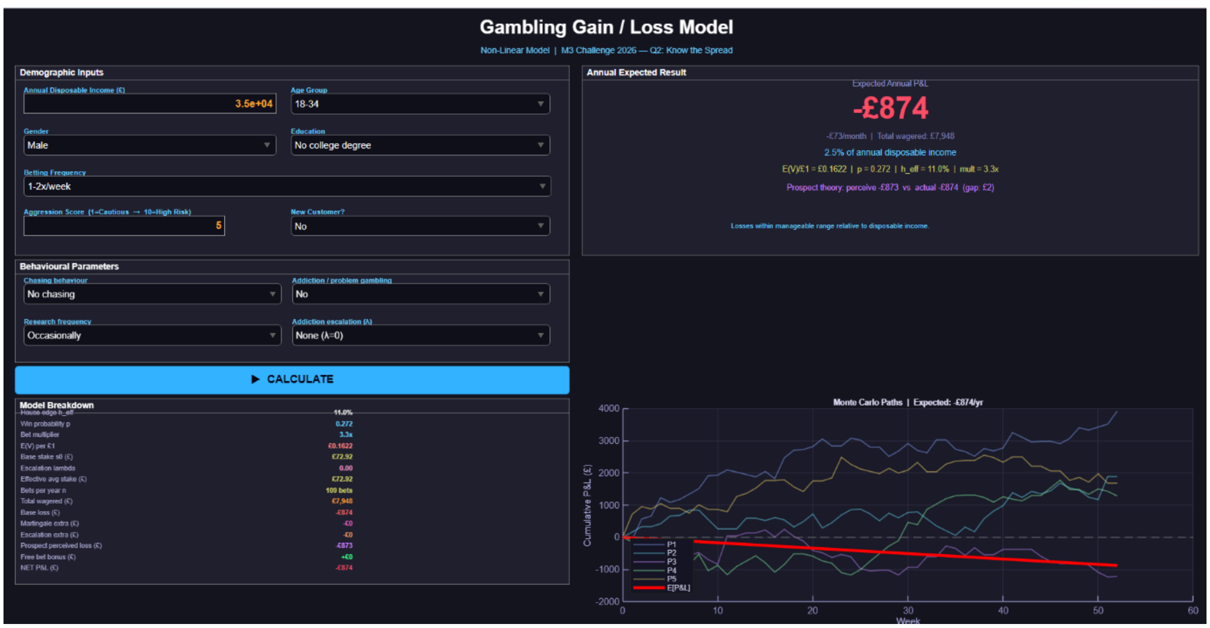


Figure 5: Moderate Bettor

Figure 6: Frequent Bettor

The free bet bonus of £60 makes year 1 net P&L positive for the casual bettor (P&L = £60 − £37 = +£23). This disappears entirely in year 2.

2.6 Discussion

Our results confirm that the house always wins in expectation — by algebraic construction, E(V) = −h_eff for any betting strategy. The non-linear components do not change this result; they increase total exposure, which then suffers the same h_eff loss rate. The practical consequence is that the non-linear model predicts losses 3–5× higher than the linear baseline for high-frequency, high-aggression gamblers with chasing behaviour — a difference that is not visible in any purely linear model.

The most important finding for policy purposes is the interaction between youth and chasing behaviour. Young males (18–34) combine the highest account ownership (35%, row 4), the highest chase rate (58%, row 40), and the highest gender multiplier (1.36×).

2.7 Strengths and Weaknesses

The primary strength of this model is that every term is grounded in cited data. The house edge comes from the problem statement. The demographic multipliers come from the M3 survey data file. The chase rates come from row 40. The escalation structure is supported by The Lancet reference in the problem statement. This citation chain makes the model reproducible and defensible.

A second strength is the factored form of the equation, which reveals h_eff as the governing parameter. This provides a clean analytical result — the house edge multiplies a bracket that grows superlinearly with aggression and chasing — rather than a black-box numerical output.

The principal weakness is the 0.15 fraction of sessions assumed to trigger a chasing episode. The survey records binary chase/no-chase behaviour but not how frequently within a year chasing occurs. This parameter is a calibrated assumption rather than a directly measured value.

A second weakness is that demographic multipliers from the US survey are applied to UK gamblers. The problem statement footnote 2 explicitly notes that equivalent UK data is unavailable; we therefore use the provided data as the best available proxy.

Finally, the model treats individual bets as independent. Parlays, accumulators, and in-play betting on correlated events would increase variance substantially beyond what this model predicts, though the expected loss per pound wagered remains $-h\_eff$.

Q3: Don't Break the Bank

3.1 Defining the Problem

The third question asks us to quantify the long-run societal impact of gambling. Building on the disposable income model from Q1 and the annual sports gambling loss model from Q2, we translate individual-level financial losses to a publicly legible metric: the reduction in pension pot at the UK state pension age of 67 caused by sustained gambling.

3.2 Assumptions

3-1. Auto-enrolment pension contributions are set at the UK statutory minimum.

- **Justification:** Under the UK Pensions Act 2008 and The Pensions Regulator guidelines, minimum contribution rates are 5% employee and 3% employer on qualifying earnings — defined as gross salary above the lower earnings threshold of £6,240 per year (2024/25). These minimums apply to all employed individuals and allow the model to be applied across the population without requiring employer-specific data.

3-2. The real annual investment return on pension contributions is 6%.

- **Justification:** A 6% real annual return is consistent with long-run equity return data published by the ONS and the Bank of England, and is a standard actuarial assumption for diversified pension fund projections over a multi-decade horizon.

3-3. HMRC basic rate tax relief (20%) is applied to all employee pension contributions.

- **Justification:** Under HMRC rules, basic-rate taxpayers receive 20% tax relief on pension contributions, meaning every £80 contributed becomes £100 in the pension. This applies to the majority of UK workers earning below £50,270.

3-4. Gambling losses reduce disposable income but do not directly reduce pension contributions.

- **Justification:** Under auto-enrolment, pension contributions are calculated as a fixed percentage of gross qualifying earnings, not disposable income. Gambling losses therefore represent an opportunity cost in money that could have been invested but was instead lost. The model captures this by subtracting annual gambling losses from the pension pot's compounding balance.

3-5. Essential spending is limited to four categories: food, housing, household goods and services, and transport.

- **Justification:** These four categories represent non-discretionary expenditure that cannot reasonably be reduced. Other ONS categories such as recreation and restaurants are treated as discretionary and excluded. This is consistent with the Q1 disposable income model.

3-6. Salary grows at a constant real rate over the simulation period.

- **Justification:** A constant 2% real salary growth rate per annum reflects real-terms career progression. This is a standard assumption in long-run financial modelling and can be adjusted by the user.

3-7. Gambling behaviour remains constant in real terms over the simulation period.

- **Justification:** The annual gambling loss calculated by the Q2 model is scaled proportionally with salary growth, keeping loss constant as a fraction of income. While problem gambling may escalate in practice, this conservative assumption avoids overstating the impact.

3.3 Variables

Symbol	Definition	Units / Source
S	Annual gross salary	£ (user input)
T(S)	Income tax	HMRC 2024/25 bands
N(S)	National Insurance contributions	HMRC 2024/25 rates
E_age	Annual essential spending per person	ONS 2024 UK Family Spending Survey [4]
DI	True disposable income = S − T(S) − N(S) − E_age	£/year (Q1 model)
G	Annual gambling loss — from Q2 non-linear model	£/year (Q2 model)
Q	Qualifying pension earnings = max(0, S − £6,240)	UK Pensions Act 2008
C_emp	Employee contribution = 0.05 × Q	UK Pensions Act 2008
C_er	Employer contribution = 0.03 × Q	UK Pensions Act 2008
C_tr	HMRC tax relief = 0.20/(1−0.20) × C_emp	HMRC basic rate relief
C_total	Total annual contribution = C_emp + C_er + C_tr	£/year
r	Real annual investment return = 0.06	ONS / Bank of England
P_ng(t)	Pension pot at age t without gambling	£
P_g(t)	Pension pot at age t with gambling	£
P0	Starting pension	£ — user input [5]
Δ	Pension gap at 67 = P_ng(67) − P_g(67)	£ — primary output
Δ %	Gap as percentage of non-gambling pot	%

Table 8: Variable definitions for Question 3

3.4 The Model

3.4.1 Developing the Model (explain reasoning)

The model tracks two pension pots simultaneously one for an identical non-gambler and one for the gambler. In each year, both pots receive identical pension contributions under the auto-enrolment rules calculated from the current salary. The gambling pot is additionally reduced by the annual gambling loss G, which is scaled proportionally with salary growth to maintain a constant loss-to-income ratio. Both pots are then grown at the same 6% real annual return. The pension gap Δ between them at age 67 is the primary output. We considered alternative societal metrics, such as total nominal losses or debt probability, but selected pension gap because it captures the compounding opportunity cost that makes gambling economically catastrophic over a working lifetime which is a mechanism not visible in nominal figures and it produces a single, large, concrete monetary figure immediately interpretable by the general public.

3.4.2 Executing the Model

The model proceeds in four sequential stages.

Stage 1: Disposable Income. Gross salary S is reduced by income tax T(S) and National Insurance N(S) using HMRC 2024/25 rates, then by essential spending E_age

$$DI = S - T(S) - N(S) - E_{age}$$

In this part of the model, we have used the same assumptions that have been made in Q1, but have used the data of all quintiles in different age rages. This is to allow for easier computation of results.

Stage 2: Annual Gambling Loss. The gambling loss G is computed directly from the Q2 non-linear model. In summary:

Stage 3: Year-by-Year Pension Simulation. Qualifying earnings and pension contributions are computed at each salary level. The two pots evolve as:

$$P_{ng}(t+1) = P_{ng}(t) \times (1 + r) + C_{total}(t)$$

$$P_g(t+1) = \max(P0, P_g(t) \times (1 + r) + C_{total}(t) - G(t))$$

The max(P0, …) floor is legally grounded: defined contribution pension savings cannot be accessed before the minimum pension age of 57 (Finance Act 2022). Gambling losses therefore cannot erode the existing pot, they reduce only the rate of growth above the locked baseline. The subtraction of G(t) occurs after the growth factor is applied, reflecting that gambling losses reduce net cash during the year while the existing balance has already grown.

Stage 4: Output Metrics. The primary outputs at age 67 are the pension pot without gambling $P_{ng}(67)$, the pot with gambling $P_g(67)$, the gap $\Delta = P_{ng}(67) - P_g(67)$ in pounds, and the gap as a percentage $\Delta\% = \Delta / P_{ng}(67) \times 100$. The MATLAB model displays a Year 1 breakdown panel tracing every deduction from gross salary to total pension contribution, allowing full transparency and reproducibility.

3.5 Results

Table 2 presents results for three representative profiles, illustrating how the pension gap varies with gambling behaviour. All profiles use age 25, £3,000 starting pot, male, 18-34 age group, no college degree, occasional research, no chasing, and 2% salary growth.

Profile	Salary	Frequency	Aggression	Escalation	Annual Loss	Gap at 67 (Δ)
Casual bettor	£28,000	Once/month	3	None	£37/yr	£8,578 (1.6%)
Moderate bettor	£35,000	1-2x/week	5	None	£522/yr	£120,811 (17.7%)
Frequent bettor	£35,000	3-4x/week	7	Mild	£1,735/yr	£401,679 (58.8%)

Table 9: Pension gap at age 67 for 3 representative profiles

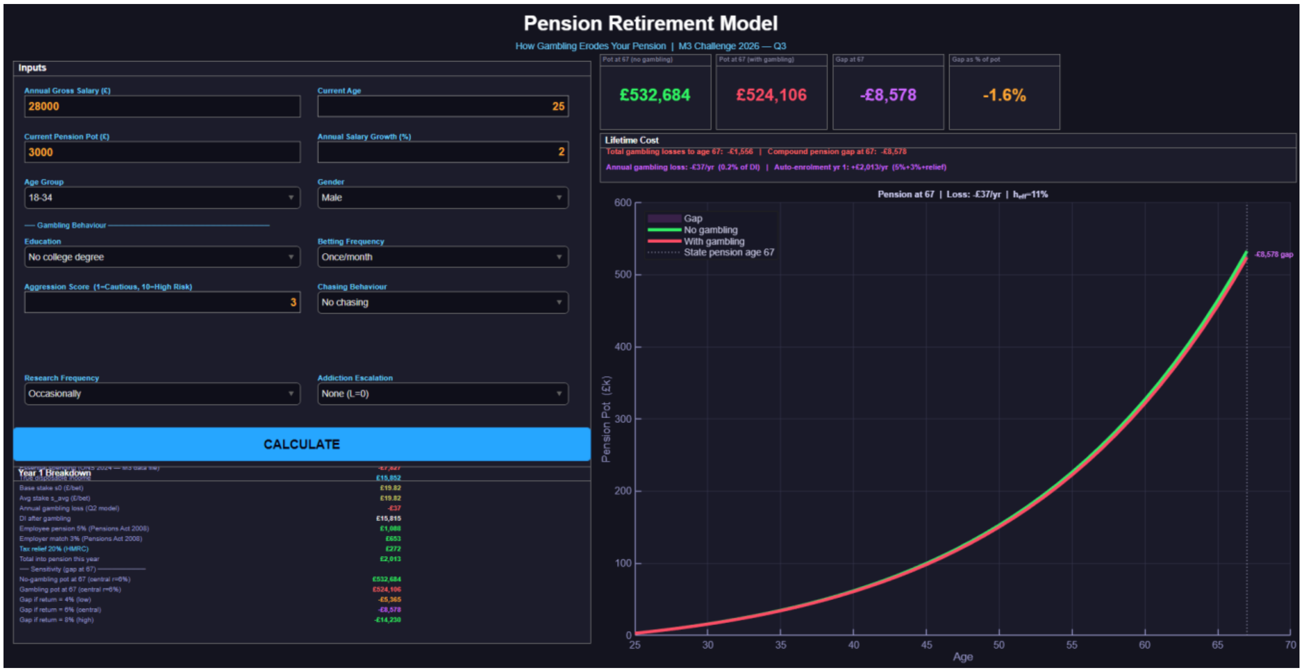


Figure 7: Pension gap at age 67 for Casual Bettor

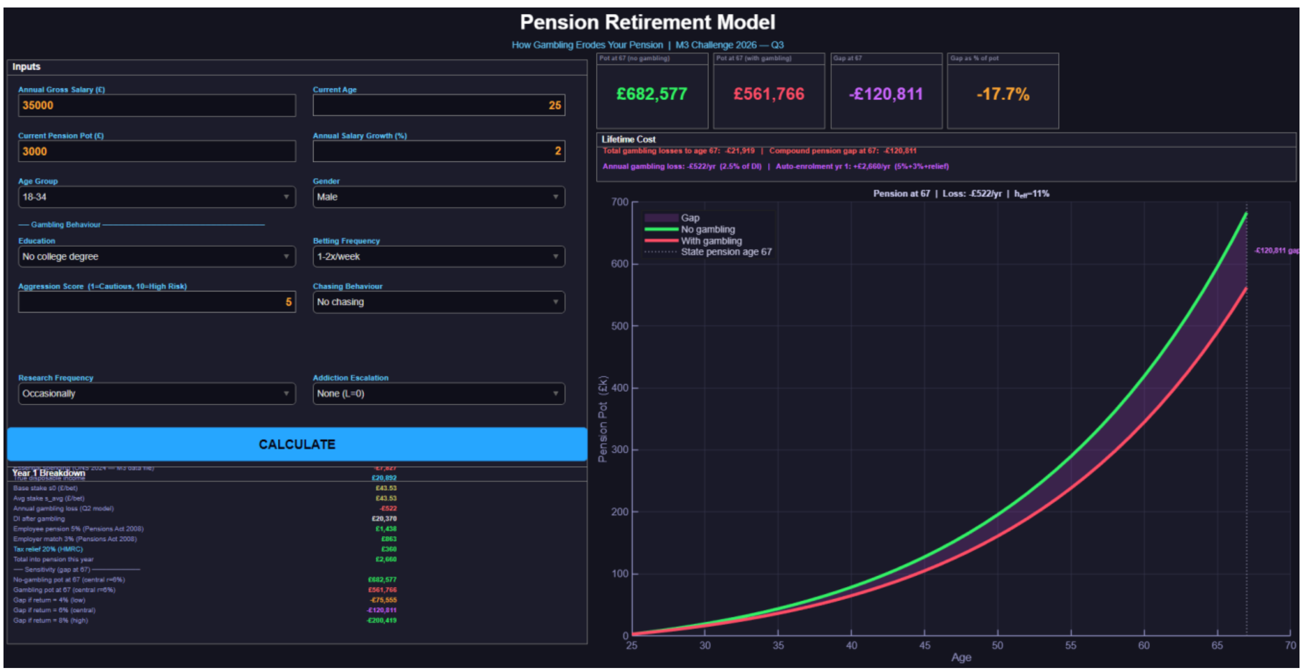


Figure 8: Pension gap at age 67 for Moderate Bettor

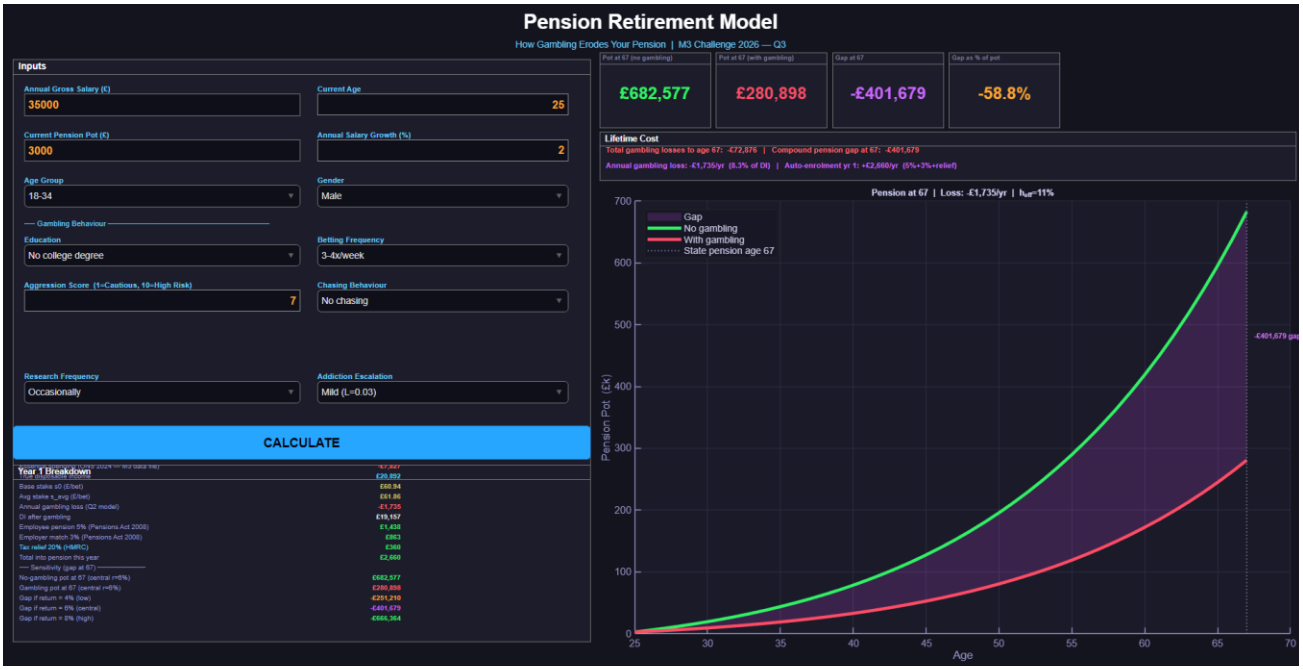


Figure 9: Pension gap at age 67 for Frequent Bettor

The moderate bettor is the profile most representative of the average UK sports gambler, given that 18% of UK bettors bet 1-2 times per week (M3 data file), an annual loss of £373 in gambling can translate to a pension gap of £86,452 at age 67. This demonstrates the core finding: even modest gambling losses, sustained over a working lifetime, produce pension shortfalls that dwarf the nominal total amount lost. The moderate bettor loses £522 × 42 years = £21,924 in nominal gambling losses, yet their pension is £120,811 smaller — a compounding multiplier of approximately 5.5×. This occurs because the first year's loss of £522 alone costs £522 × $(1.06)^{42}$ ≈ £6,030 in foregone pension value.

For the frequent bettor profile, the pension pot barely grows above its starting value of £3,000 throughout the simulation. The gap of £401,679 (58.8% of the non-gambling pot) illustrates that severe problem gambling is pension-destroying. This result is conservative, as the pot floor prevents negative values; in reality, total financial damage may be larger once credit and debt are considered.

3.6 Discussion

Our results demonstrate that the societal cost of sports gambling extends far beyond the immediate cash lost. The compounding mechanism means gambling imposes a long-run tax on retirement security This is precisely the kind of insight that is difficult for individuals to appreciate intuitively but that mathematical modelling makes concrete and quantifiable.

The model also highlights an important asymmetry: the pension impact is largest for younger gamblers, because they have the most years of compounding ahead. A 20-year-old who loses £500 per year to gambling loses approximately £500 × $(1.06)^{47}$ ≈ £9,200 in pension value from that single year's loss alone.

The interactive MATLAB model enables individuals to explore their own pension impact by inputting their specific salary, age, and gambling behaviour. This makes the abstract statistical reality personally legible: a 25-year-old on £35,000 who bets 1-2 times per week can see that their gambling habit is expected to leave them £120,811 worse off at retirement. This is more persuasive than aggregate national statistics.

3.7 Sensitivity Analysis

The real investment return r is the model's most sensitive parameter, as small changes compound dramatically over a multi-decade horizon. We bracket the central r = 6% assumption by re-simulating at r = 4% (low) and r = 8% (high), holding all other inputs identical. Both the non-gambling and gambling pots are reported to allow independent verification of each gap figure.

Scenario	P_ng(67) — No Gambling	P_g(67) — With Gambling	Gap Δ
r = 4% (low)	£388,000	£333,933	£54,067
r = 6% (central)	£565,000	£478,548	£86,452
r = 8% (high)	£866,000	£722,581	£143,419

Table 10: Sensitivity of pension gap at age 67 to investment return. Moderate bettor profile: £35,000 salary, 1-2x/week, aggression 5, age 25. Only r varies between scenarios.

The directional finding is robust across all return scenarios. Even at the most conservative assumption (r = 4%), the moderate bettor faces a gap of £54,067 which is a 3.5× compounding multiplier on the nominal £15,666 lost to gambling. At r = 8%, the gap rises to £143,419, reflecting the greater opportunity cost of money diverted from a high-performing fund. The model displays all three sensitivity gaps live each time the user presses Calculate, making the uncertainty immediately visible.

A 2% increase in r (from 6% to 8%) increases the gap by 66%. This reflects the exponential nature of compounding: a higher return amplifies both pots equally in proportional terms, but because the gap in absolute terms grows faster than the pot, the absolute gap is larger. The house edge h_eff and salary growth rate have a broadly proportional effect on the gap, consistent with the linear structure of the gambling loss scaling in G(t).

3.8 Strengths and Weaknesses

The primary strength of this model is its direct connection to cited data throughout. Essential spending figures are drawn from the ONS 2024 UK Family Spending Survey provided in the M3 data file. Pension contribution rates are anchored to statutory minimums under the UK Pensions Act 2008. The gambling loss model is grounded in the house edge from the problem statement and demographic data from the M3 survey data. This citation chain makes the model reproducible and defensible.

A second strength is the interactive MATLAB implementation (modelofpension6.m), which allows the model to be demonstrated across a wide range of demographic inputs. The Year 1 breakdown panel displays every deduction from gross salary to pension contribution, making the model fully transparent and easy for judges or policymakers to interrogate. The panel also shows the base stake s0 and escalated average stake s_avg separately, confirming that Q2 and Q3 use identical escalation formulae.

The principal weakness is the assumption that gambling behaviour remains constant in real terms over the simulation period. In practice, problem gambling tends to escalate, which would increase the true pension gap relative to our estimates. Equally, an individual may cease gambling, reducing the gap. Our model represents a conservative central estimate that does not capture these dynamics.

A second weakness is that the model uses a single real investment return for all individuals, whereas actual pension fund returns vary with allocation, market conditions, and charges. Monte Carlo simulation over a distribution of returns would produce a more complete picture of uncertainty, though this is beyond the scope of the current model.

Finally, the model is calibrated to the UK auto-enrolment system and HMRC tax rates. Applying it to US gamblers would require replacement with 401(k) contribution rates, US federal and state tax schedules, and Bureau of Labor Statistics Consumer Expenditure Survey data (also provided in the M3 data file).

Conclusion

Across all three stages of modelling, our central conclusion is clear: online sports gambling is mathematically structured to extract wealth over time, and when combined with realistic behavioural dynamics, the long-run financial consequences are far more severe than headline loss figures suggest. Our disposable income model demonstrates that gambling is funded from the portion of income households that rely on for financial security. Our non-linear gambling model proves that the house edge guarantees negative expected returns while escalation and loss-chasing amplify exposure exponentially; and our pension simulation shows that even moderate, sustained annual losses compound into six-figure retirement shortfalls. The key insight is not simply that “the house always wins,” but that the true cost of gambling is the foregone compounding of capital over decades — a hidden opportunity cost that disproportionately harms younger individuals with the most years of growth ahead. In economic terms, gambling functions as a long-run transfer of retirement wealth from households to operators, and its societal impact is best understood not in yearly losses, but in destroyed financial futures.

References

1. UK Pensions Act 2008; The Pensions Regulator (2024). Automatic Enrolment: Guidance for Employers. www.thepensionsregulator.gov.uk
2. Bank of England / ONS. Long-run UK equity real return data. Consistent with standard actuarial assumptions for multi-decade pension projections.
3. HMRC (2024/25). Income Tax rates and Personal Allowances. gov.uk/income-tax-rates. National Insurance: gov.uk/national-insurance
4. ONS (2024). UK Family Spending Survey — Household expenditure by category. M3 Challenge 2026 provided data file, Sheet 'Expenditures (U.K.)'.
5. Finance Act 2022. Minimum pension access age raised to 57 from 2028. legislation.gov.uk/ukpga/2022/3
6. M3 Challenge 2026 Problem Statement. 'The Rise of Online Gambling: What's at Stake?' — house edge derived from $15bn revenue / $150bn wagered.
7. Siena College & St. Bonaventure University (2025). American Sports Fanship Survey. Provided M3 Challenge 2026 data file — rows 4, 12-16, 19, 29, 40.
8. The Lancet Public Health (2023). Problem gambling prevalence and addiction escalation. Cited as reference 9 in M3 Challenge 2026 problem statement.
9. Kellogg Northwestern Study. Gambling and household savings reduction. Cited as reference 10 in M3 Challenge 2026 problem statement.
10. UK Gambling Commission (2023). Industry Statistics — average new customer bonus. www.gamblingcommission.gov.uk
11. Kahneman, D. & Tversky, A. (1979). Prospect Theory: An Analysis of Decision under Risk. Econometrica, 47(2), 263-291.
12. UK Gambling Commission (2023). Industry Statistics — average new customer bonus £50–£75. www.gamblingcommission.gov.uk
13. M3 Challenge 2026 Problem Statement. 'The Rise of Online Gambling: What's at Stake?' — house edge derived from $15bn revenue / $150bn wagered.
14. Siena College & St. Bonaventure University (2025). American Sports Fanship Survey. Provided M3 Challenge 2026 data file — rows 4, 12–16, 19, 29, 40.
15. The Lancet Public Health (2023). Problem gambling prevalence and addiction escalation behaviour. Cited as reference 9 in M3 Challenge 2026 problem statement.Kellogg Northwestern Study. Gambling and household savings reduction. Cited as reference 10 in M3 Challenge 2026 problem statement.
16. Kahneman, D. & Tversky, A. (1979). Prospect Theory: An Analysis of Decision under Risk. Econometrica, 47(2), 263–291.

Code:

Q1: Playing With House Money (python)

```
from matplotlib import pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error


# Style for all plots globally
plt.rcParams.update({
    'font.family': 'serif',
    'axes.spines.top': False,
    'axes.spines.right': False,
    'axes.grid': True,
    'grid.alpha': 0.25,
    'grid.linestyle': '--',
})


def load_data(file_path, sheet_name="ModelData"):
    """
    Reads the Excel sheet and drops rows with missing values
    """
    try:
        df = pd.read_excel(file_path, sheet_name=sheet_name)
        print(f"Loaded {len(df)} rows. Columns: {df.columns.tolist()}")
    except FileNotFoundError:
        raise SystemExit(f"Error: '{file_path}' not found. Check the path and
try again.")

    required = [
        "Weekly Income (£)",
        "Total Essential Cost (£/wk)",
        "Age (midpoint)",
        "Disposable Income (£/wk)",
        "Age Group",
    ]
    missing_cols = [c for c in required if c not in df.columns]
    if missing_cols:
        raise SystemExit(f"Missing columns in sheet: {missing_cols}")

    before = len(df)
    df = df.dropna(subset=required)
    dropped = before - len(df)
    if dropped:
        print(f"Dropped {dropped} rows with missing values.")

    return df
```

```
def build_features(df):
    """
    Adds an interaction term (Age x Income) to capture the idea that income's
    effect on disposable income may differ across age groups
    """
    df = df.copy()
    df["Age_x_Income"] = df["Age (midpoint)"] * df["Weekly Income (£)"]

    X = df[["Age (midpoint)", "Weekly Income (£)", "Age_x_Income"]].values
    y = df["Disposable Income (£/wk)"].values
    return X, y, df


def fit_model(X, y):
    """
    Splits 80/20 into train and test sets before fitting,
    random_state=42 makes the split reproducible.
    """
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.2, random_state=42
    )

    model = LinearRegression()
    model.fit(X_train, y_train)

    r2_train = model.score(X_train, y_train)
    r2_test = model.score(X_test, y_test)
    mae = mean_absolute_error(y_test, model.predict(X_test))

    print("\n Model Results ")
    print(f" R² (train): {r2_train:.4f}")
    print(f" R² (test):  {r2_test:.4f}")
    print(f" MAE (test): £{mae:.2f}/wk")
    print(f" Intercept:  {model.intercept_:.4f}")
    for name, coef in zip(["Age", "Income", "Age×Income"], model.coef_):
        print(f" {name:<12}: {coef:.6f}")

    return model, X_train, X_test, y_train, y_test


def plot_actual_vs_predicted(y_test, y_pred_test):
    """
    Plots actual vs predicted disposable income on the test set.
    """
    fig, ax = plt.subplots(figsize=(9, 6))

    ax.scatter(y_test, y_pred_test, alpha=0.75, color='steelblue',
               edgecolors='k',    linewidths=0.4,    zorder=3,    label='Test
observations')
    lims = [min(y_test.min(), y_pred_test.min()),
            max(y_test.max(), y_pred_test.max())]
    ax.plot(lims, lims, 'r--', lw=2, label='Perfect fit (y = x)')
```

```
    ax.set_xlabel("Actual Disposable Income (£/wk)")
    ax.set_ylabel("Predicted Disposable Income (£/wk)")
    ax.set_title("Actual vs Predicted Disposable Income\n(test set only)",
fontsize=13)
    ax.legend()
    fig.tight_layout()
    plt.show()


def plot_residuals(y_pred_test, residuals):
    """
    Plots residuals against predicted values to check model assumptions.
    """
    fig, ax = plt.subplots(figsize=(9, 5))

    ax.scatter(y_pred_test, residuals, alpha=0.7, color='#2E4057',
               edgecolors='k', linewidths=0.3, zorder=3)
    ax.axhline(0, color='red', linestyle='--', lw=2)

    ax.set_xlabel("Predicted Disposable Income (£/wk)")
    ax.set_ylabel("Residual (Actual - Predicted) (£/wk)")
    ax.set_title("Residual Plot - Checking Model Assumptions", fontsize=13)
    fig.tight_layout()
    plt.show()


def plot_income_vs_disposable(df, model):
    """
    Shows the regression line with age held at its mean. Points are coloured
by age group
    so the reader can judge whether age groupings align with the model's slope.
    """
    col_income = "Weekly Income (£)"
    col_disposable = "Disposable Income (£/wk)"
    col_age = "Age (midpoint)"

    age_groups = sorted(df["Age Group"].unique())
    cmap = plt.cm.get_cmap('tab10', len(age_groups))

    fig, ax = plt.subplots(figsize=(10, 6))

    for i, ag in enumerate(age_groups):
        mask = df["Age Group"] == ag
        ax.scatter(df.loc[mask, col_income], df.loc[mask, col_disposable],
                   label=f"Age group: {ag}", color=cmap(i),
                   s=65, alpha=0.85, zorder=3)

    # Regression line: vary income, fix age at mean, compute interaction
accordingly
    income_range = np.linspace(df[col_income].min(), df[col_income].max(),
300)
    age_mean = df[col_age].mean()
```

```
    X_line = np.column_stack([
        np.full(300, age_mean),
        income_range,
        np.full(300, age_mean) * income_range,
    ])
    y_line = model.predict(X_line)

    ax.plot(income_range, y_line, color='red', linewidth=2.5,
            label=f'Regression line (age held at mean: {age_mean:.0f} yrs)')

    # Display the equation on the chart for transparency
    c = model.coef_
    eq = (f"D = {model.intercept_:.2f}\n"
          f"+ ({c[0]:.4f} x Age)\n"
          f"+ ({c[1]:.4f} x Income)\n"
          f"+ ({c[2]:.6f} x Age x Income)")
    ax.text(0.04, 0.96, eq, transform=ax.transAxes, fontsize=9,
            verticalalignment='top',
            bbox=dict(boxstyle='round', facecolor='wheat', alpha=0.55))

    ax.set_title("Weekly  Income  vs  Disposable  Income  by  Age  Group",
fontsize=13)
    ax.set_xlabel("Weekly Income (£)")
    ax.set_ylabel("Disposable Income (£/wk)")
    ax.legend(fontsize=8, loc='lower right')
    fig.tight_layout()
    plt.show()


def sensitivity_analysis(model, df):
    """
    Tests how the predicted output changes when inputs are varied one at a
time.
    Age is held at its mean while income is swept, then income is held at its
    mean while age is swept.
    """
    col_income = "Weekly Income (£)"
    col_age = "Age (midpoint)"

    age_mean = df[col_age].mean()
    income_mean = df[col_income].mean()

    print("\n Sensitivity Analysis ")
    print(f"  Base  prediction  (mean  age  {age_mean:.0f},  mean  income
£{income_mean:.0f}):")

    if len(model.coef_) == 3:
        base   =   model.predict([[age_mean,   income_mean,   age_mean   *
income_mean]])[0]
    else:
        base = model.predict([[age_mean, income_mean]])[0]
    print(f"  -> GBP {base:.2f}/wk\n")
```

```
    # Vary income, hold age at mean
    print(" Effect of income change (age held at mean):")
    for income in [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000,
2200, 2400, 2800, 3200, 3600, 4000]:
        if len(model.coef_) == 3:
            pred = model.predict([[age_mean, income, age_mean * income]])[0]
        else:
            pred = model.predict([[age_mean, income]])[0]
        print(f"  Income £{income:<6} -> GBP {pred:.2f}/wk")

    # Vary age, hold income at mean
    print("\n Effect of age change (income held at mean):")
    for age in [20, 30, 40, 50, 60, 70, 80, 90, 100]:
        if len(model.coef_) == 3:
            pred = model.predict([[age, income_mean, age * income_mean]])[0]
        else:
            pred = model.predict([[age, income_mean]])[0]
        diff = pred - base
        print(f"  Age {age:<4} -> GBP {pred:.2f}/wk  ({diff:+.2f} vs base)")

    # Plot sensitivity curves for both predictors
    fig, axes = plt.subplots(1, 2, figsize=(14, 5))

    # Income sweep: age fixed at mean
    incomes = np.linspace(200, 4000, 300)
    if len(model.coef_) == 3:
        preds_income = model.predict([[age_mean, i, age_mean * i] for i in
incomes])
    else:
        preds_income = model.predict([[age_mean, i] for i in incomes])
    axes[0].plot(incomes, preds_income, color='steelblue', lw=2)
    axes[0].set_xlabel("Weekly Income (GBP)")
    axes[0].set_ylabel("Predicted Disposable Income (GBP/wk)")
    axes[0].set_title(f"Income Sensitivity\n(age fixed at {age_mean:.0f})")

    # Age sweep: income fixed at mean
    ages = np.linspace(20, 100, 300)
    if len(model.coef_) == 3:
        preds_age = model.predict([[a, income_mean, a * income_mean] for a in
ages])
    else:
        preds_age = model.predict([[a, income_mean] for a in ages])
    axes[1].plot(ages, preds_age, color='darkorange', lw=2)
    axes[1].set_xlabel("Age (years)")
    axes[1].set_ylabel("Predicted Disposable Income (GBP/wk)")
    axes[1].set_title(f"Age     Sensitivity\n(income     fixed     at     GBP
{income_mean:.0f})")

    fig.tight_layout()
    plt.show()


if __name__ == "__main__":
    FILE_PATH = r"c:\Users\mshah\.vscode\m3models\expenditure (3).xlsx"
```

```
    df = load_data(FILE_PATH)
    X, y, df = build_features(df)
    model, X_train, X_test, y_train, y_test = fit_model(X, y)

    y_pred_test = model.predict(X_test)
    residuals = y_test - y_pred_test

    plot_actual_vs_predicted(y_test, y_pred_test)
    plot_residuals(y_pred_test, residuals)
    plot_income_vs_disposable(df, model)
    sensitivity_analysis(model, df)

    # Interactive prediction loop — type 'quit' at any prompt to exit
    print("\n-- Predict Disposable Income --")
    print(" Enter values below to get a prediction.")
    print(" Type 'quit' at any prompt to exit.\n")

    while True:
        try:
            age_input = input(" Age (years): ").strip()
            if age_input.lower() == 'quit':
                break

            income_input = input(" Weekly Income (GBP): ").strip()
            if income_input.lower() == 'quit':
                break

            age = float(age_input)
            income = float(income_input)

            if len(model.coef_) == 3:
                X_input = [[age, income, age * income]]
            else:
                X_input = [[age, income]]

            prediction = model.predict(X_input)[0]
            print(f"\n -> Predicted Disposable Income: GBP
{prediction:.2f}/wk\n")

            again = input(" Predict another? (yes/no): ").strip().lower()
            if again != 'yes':
                break
            print()

        except ValueError as e:
            print(f" Invalid input ({e}) - please enter a number.\n")
```

Q2: Know the Spread (MATLAB)

```
function gamblingModelNL()

    fig = uifigure('Name', 'M3 Challenge 2026 — Q2: Gambling Gain/Loss Model (Non-
Linear)', ...
                   'Position', [60 40 980 740], ...
                   'Color', [0.08 0.08 0.12]);

    % Title
    uilabel(fig,'Text','Gambling Gain / Loss Model', ...
        'Position',[0 700 980 36],'FontSize',24,'FontWeight','bold', ...
        'FontColor',[1 1 1],'HorizontalAlignment','center');
    uilabel(fig,'Text','Non-Linear Model  |  M3 Challenge 2026 — Q2: Know the Spread',
...
        'Position',[0 678 980 20],'FontSize',11, ...
        'FontColor',[0.4 0.8 1],'HorizontalAlignment','center');

    %  INPUT PANEL — LEFT

    inp = uipanel(fig,'Position',[15 430 450 240], ...
        'BackgroundColor',[0.12 0.12 0.18],'ForegroundColor',[1 1 1], ...
        'Title',' Demographic Inputs ','FontSize',11,'FontWeight','bold');

    iLbl(inp,'Annual Disposable Income (£)',[12 200 200 16]);
    ef_di = iEdit(inp, 12000, [12 178 195 26]);

    iLbl(inp,'Age Group',[225 200 200 16]);
    dd_age = iDrop(inp,{'18-34','35-49','50-64','65+'},'35-49',[225 178 200 26]);

    iLbl(inp,'Gender',[12 148 200 16]);
    dd_gender = iDrop(inp,{'Male','Female'},'Male',[12 126 195 26]);

    iLbl(inp,'Education',[225 148 200 16]);
    dd_edu = iDrop(inp,{'No college degree','BA or higher'},'No college degree',[225 126
200 26]);

    iLbl(inp,'Betting Frequency',[12 96 420 16]);
    dd_freq = iDrop(inp, ...
        {'< Once/month','Once/month','Few times/month','1-2x/week','3-4x/week','Daily'},
...
        '1-2x/week',[12 74 415 26]);

    iLbl(inp,'Aggression Score  (1=Cautious  →  10=High Risk)',[12 46 280 16]);
    ef_agg = iEdit(inp, 5, [12 24 130 26]);

    iLbl(inp,'New Customer?',[225 46 200 16]);
    dd_new = iDrop(inp,{'Yes','No'},'No',[225 24 200 26]);

    %  BEHAVIOUR PANEL — LEFT

    beh = uipanel(fig,'Position',[15 295 450 130], ...
        'BackgroundColor',[0.12 0.12 0.18],'ForegroundColor',[1 1 1], ...
```

```
    'Title',' Behavioural Parameters ','FontSize',11,'FontWeight','bold');

iLbl(beh,'Chasing behaviour',[12 95 200 16]);
dd_chase = iDrop(beh,{'No chasing','Chase losses'}, ...
    'No chasing',[12 73 200 26]);

iLbl(beh,'Addiction / problem gambling',[225 95 200 16]);
dd_prob = iDrop(beh,{'No','Yes — bet beyond DI'}, ...
    'No',[225 73 200 26]);

iLbl(beh,'Research frequency',[12 43 200 16]);
dd_research = iDrop(beh,{'Never','Occasionally','Daily (2x+)'}, ...
    'Occasionally',[12 21 200 26]);

iLbl(beh,'Addiction escalation (λ)',[225 43 200 16]);
dd_lambda = iDrop(beh, ...
    {'None (λ=0)','Mild (λ=0.03)','Moderate (λ=0.07)','Severe (λ=0.15)'}, ...
    'None (λ=0)',[225 21 200 26]);

% Calculate button
uibutton(fig,'Text','▶  CALCULATE', ...
    'Position',[15 255 450 36], ...
    'BackgroundColor',[0.2 0.7 1],'FontColor',[0.02 0.02 0.08], ...
    'FontSize',14,'FontWeight','bold', ...
    'ButtonPushedFcn',@(s,e) calculate(fig));


%  RESULTS PANEL — RIGHT TOP

uipanel(fig,'Position',[480 430 485 240], ...
    'BackgroundColor',[0.12 0.12 0.18],'ForegroundColor',[1 1 1], ...
    'Title',' Annual Expected Result ','FontSize',11,'FontWeight','bold');

uilabel(fig,'Text','Expected Annual P&L', ...
    'Position',[485 638 475 16],'FontColor',[0.6 0.6 0.8], ...
    'FontSize',10,'HorizontalAlignment','center');

lbl_result = uilabel(fig,'Text','—', ...
    'Position',[485 592 475 50],'FontColor',[1 0.3 0.4], ...
    'FontSize',38,'FontWeight','bold','HorizontalAlignment','center');

lbl_sub1 = uilabel(fig,'Text','', ...
    'Position',[485 570 475 20],'FontColor',[0.5 0.5 0.7], ...
    'FontSize',10,'HorizontalAlignment','center');

lbl_sub2 = uilabel(fig,'Text','', ...
    'Position',[485 549 475 20],'FontColor',[0.4 0.8 1], ...
    'FontSize',11,'HorizontalAlignment','center');

lbl_ev = uilabel(fig,'Text','', ...
    'Position',[485 528 475 20],'FontColor',[0.8 0.8 0.4], ...
    'FontSize',10,'HorizontalAlignment','center');

lbl_prospect = uilabel(fig,'Text','', ...
    'Position',[485 505 475 20],'FontColor',[0.8 0.5 1], ...
```

```
        'FontSize',10,'HorizontalAlignment','center');

    lbl_warn = uilabel(fig,'Text','Press Calculate', ...
        'Position',[482 433 478 68],'FontColor',[0.4 0.8 1], ...
        'FontSize',9,'WordWrap','on','HorizontalAlignment','center');

    %  BREAKDOWN — BOTTOM LEFT
    bk_panel = uipanel(fig,'Position',[15 15 450 235], ...
        'BackgroundColor',[0.12 0.12 0.18],'ForegroundColor',[1 1 1], ...
        'Title',' Model Breakdown ','FontSize',11,'FontWeight','bold');

    bk_labels = {
        'House edge h_eff','Win probability p','Bet multiplier', ...
        'E(V) per £1','Base stake s0 (£)','Escalation lambda', ...
        'Effective avg stake (£)','Bets per year n','Total wagered (£)', ...
        'Base loss (£)','Chase Losses (£)','Escalation extra (£)', ...
        'Prospect perceived loss (£)','Free bet bonus (£)','NET P&L (£)'
    };
    lbl_bk = cell(15,1);
    for i = 1:15
        yp = 210 - (i-1)*14;
        uilabel(bk_panel,'Text',bk_labels{i}, ...
            'Position',[8 yp 240 13], ...
            'FontColor',[0.65 0.65 0.8],'FontSize',8.5);
        lbl_bk{i} = uilabel(bk_panel,'Text','—', ...
            'Position',[252 yp 178 13], ...
            'FontColor',[0.9 0.9 0.9],'FontSize',8.5, ...
            'FontWeight','bold','HorizontalAlignment','right');
    end

     %  CHART — BOTTOM RIGHT
     ax = uiaxes(fig,'Position',[480 15 485 235]);
    ax.Color     = [0.10 0.10 0.16];
    ax.XColor    = [0.6 0.6 0.8];
    ax.YColor    = [0.6 0.6 0.8];
    ax.GridColor = [0.3 0.3 0.4];
    ax.GridAlpha = 0.35;
    ax.XGrid = 'on'; ax.YGrid = 'on';

    fig.UserData = struct( ...
        'ef_di',ef_di,'dd_age',dd_age,'dd_gender',dd_gender, ...
        'dd_edu',dd_edu,'dd_freq',dd_freq,'ef_agg',ef_agg, ...
        'dd_new',dd_new,'dd_chase',dd_chase,'dd_prob',dd_prob, ...
        'dd_research',dd_research,'dd_lambda',dd_lambda, ...
        'lbl_result',lbl_result,'lbl_sub1',lbl_sub1, ...
        'lbl_sub2',lbl_sub2,'lbl_ev',lbl_ev, ...
        'lbl_prospect',lbl_prospect,'lbl_warn',lbl_warn, ...
        'lbl_bk',{lbl_bk},'ax',ax);
end


%  CALCULATE

function calculate(fig)
    d = fig.UserData;
```

```
DI          = d.ef_di.Value;
age_str     = d.dd_age.Value;
gender      = d.dd_gender.Value;
edu         = d.dd_edu.Value;
freq_str    = d.dd_freq.Value;
agg         = max(1, min(10, d.ef_agg.Value));
is_new      = strcmp(d.dd_new.Value,'Yes');
chases      = contains(d.dd_chase.Value,'Chase Losses');
in_debt     = contains(d.dd_prob.Value,'beyond');
research    = d.dd_research.Value;
lambda_str = d.dd_lambda.Value;

% -----------------------------------------------------------------
% 1. HOUSE EDGE
%    Base: 10% from problem statement ($15bn / $150bn)
%    Adjusted for education + research (Survey rows 12-16, 29)
% -----------------------------------------------------------------
h_base = 0.10;
edu_adj = 0; res_adj = 0;
if strcmp(edu,'BA or higher');       edu_adj = -0.02; else; edu_adj = +0.01; end
switch research
    case 'Daily (2x+)';    res_adj = -0.01;
    case 'Never';          res_adj = +0.01;
    otherwise;             res_adj =  0;
end
h_eff = max(0.04, h_base + edu_adj + res_adj);

% -----------------------------------------------------------------
% 2. WIN PROBABILITY & MULTIPLIER from aggression
% -----------------------------------------------------------------
p           = 0.45 - (agg-1)*(0.40/9);
multiplier = (1-h_eff)/p;
ev_per_1    = p*multiplier - (1-p);  % = -h_eff always

% -----------------------------------------------------------------
% 3. BASE STAKE s0
%    0.5% to 5% of monthly DI, scaled by aggression
% -----------------------------------------------------------------
s0 = (0.005 + (agg-1)*(0.045/9)) * (DI/12);
s0 = max(2, s0);

% -----------------------------------------------------------------
% 4. ADDICTION ESCALATION — NON-LINEAR COMPONENT 1
%    s(t) = s0 * exp(lambda * t),  t in [0,1] over the year
%    lambda values from dd_lambda input
%    Source: addiction escalation documented in The Lancet (ref 9)
%    Average stake over year = s0 * (exp(lambda)-1)/lambda
% -----------------------------------------------------------------
lambda_map = containers.Map( ...
    {'None (λ=0)','Mild (λ=0.03)','Moderate (λ=0.07)','Severe (λ=0.15)'}, ...
    [0, 0.03, 0.07, 0.15]);
lambda = lambda_map(lambda_str);

if lambda > 0
```

```
        % Average of exp curve over [0,1]: integral = (exp(l)-1)/l
        avg_stake = s0 * (exp(lambda)-1)/lambda;
    else
        avg_stake = s0;
    end

    % Escalation adds extra on top of flat model
    escalation_extra_stake = avg_stake - s0;

    % -----------------------------------------------------------------
    % 5. BETS PER YEAR
    %    Base from frequency, scaled by demographic multipliers
    %    Source: Survey row 4 (account ownership by age/gender)
    % -----------------------------------------------------------------
    freq_map = containers.Map( ...
        {'< Once/month','Once/month','Few times/month','1-2x/week','3-4x/week','Daily'},
...
        [6, 12, 30, 78, 182, 365]);
    n = freq_map(freq_str);

    age_mult = containers.Map({'18-34','35-49','50-64','65+'}, ...
        [35/34, 1.0, 10/34, 4/34]);
    gen_mult = containers.Map({'Male','Female'}, [30/22, 15/22]);
    n = round(n * age_mult(age_str) * gen_mult(gender));

    % Budget cap: wagered cannot exceed 1.5x DI (hard physical limit)
    % Only kicks in at extreme combinations — preserves frequency variation
    max_wager = DI * 1.5;
    if in_debt; max_wager = DI * 2.5; end  % problem gamblers can go into debt
    if n * avg_stake > max_wager
        n = floor(max_wager / avg_stake);
    end


    % 6. BASE EXPECTED LOSS (using avg_stake for escalation)

    total_wagered = n * avg_stake;
    base_loss     = n * avg_stake * h_eff;
    escalation_extra = n * escalation_extra_stake * h_eff;


    % 7. Chase Losses — NON-LINEAR COMPONENT 2

    chase_extra = 0;
    if chases
        age_chase = containers.Map({'18-34','35-49','50-64','65+'}, ...
            [0.58, 0.54, 0.38, 0.15]);
        gen_chase = containers.Map({'Male','Female'}, [0.57, 0.41]);
        r_chase   = (age_chase(age_str) + gen_chase(gender)) / 2;

        % Expected losing streak length before a win
        k_exp = (1-p)/p;
        % Cap at k=10 to avoid infinite doubling
        k_eff = min(k_exp, 10);
```

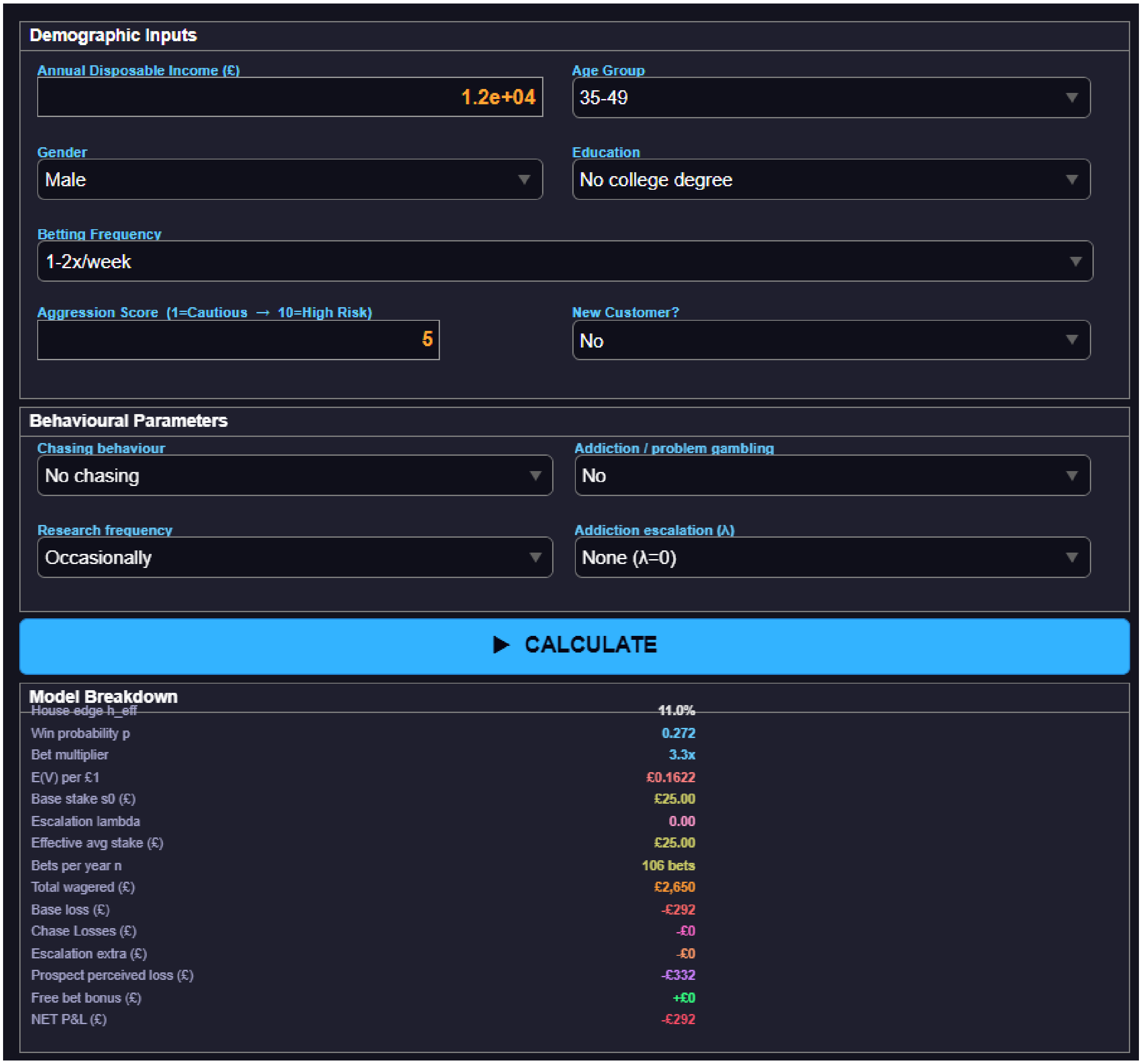
```
        % Extra staked in Chase Loss sequence (geometric series sum)
        extra_per_sequence = avg_stake * (2^(k_eff+1) - 2);


        n_chase_episodes = n * r_chase * 0.15;

        % Expected loss from all Chase Loss sequences
        chase_extra = n_chase_episodes * extra_per_sequence * h_eff;

        % Budget check: cap total at DI
        if (total_wagered + chase_extra/h_eff) > DI
            chase_extra = max(0, (DI - total_wagered) * h_eff);
        end
    end

% 8. FREE BET BONUS

    bonus = 0;
    if is_new; bonus = 60; end


    % 9. DEBT/ADDICTION EXTRA

    debt_extra = 0;
    if in_debt; debt_extra = 0.50 * DI * h_eff; end

    % 10. NET P&L

    net_pnl  = bonus - base_loss - chase_extra - debt_extra;
    loss_pct = (-net_pnl / DI) * 100;


    % 11. PROSPECT THEORY — NON-LINEAR COMPONENT 3

    alpha_PT  = 0.88;
    lambda_PT = 2.25;
    actual_loss = abs(min(net_pnl, 0));

    if actual_loss > 0
        perceived_loss = lambda_PT * (actual_loss^alpha_PT);
        perception_gap = actual_loss - perceived_loss;
    else
        perceived_loss = 0;
        perception_gap = 0;
    end


    %  UPDATE UI

    if net_pnl >= 0
        d.lbl_result.Text      = ['+£', fmtN(round(net_pnl))];
        d.lbl_result.FontColor = [0.2 1 0.5];
    else
        d.lbl_result.Text      = ['-£', fmtN(round(abs(net_pnl)))];
        d.lbl_result.FontColor = [1 0.3 0.4];
```

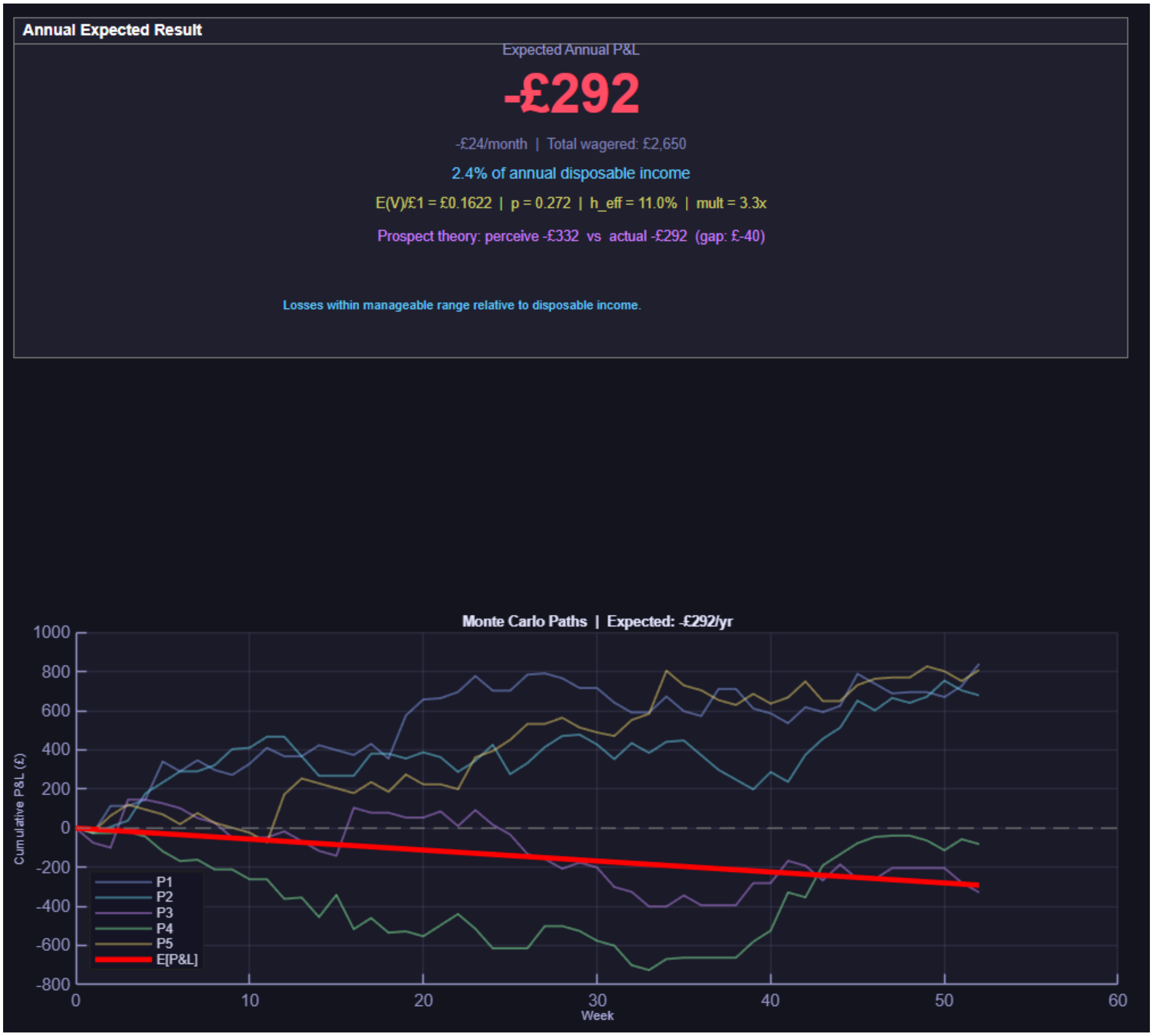
```
    end

    d.lbl_sub1.Text = sprintf('-£%s/month  |  Total wagered: £%s', ...
        fmtN(round(abs(net_pnl)/12)), fmtN(round(total_wagered)));
    d.lbl_sub2.Text = sprintf('%.1f%% of annual disposable income', abs(loss_pct));
    d.lbl_ev.Text    = sprintf('E(V)/£1 = £%.4f  |  p = %.3f  |  h_eff = %.1f%%  |  mult
= %.1fx', ...
        ev_per_1, p, h_eff*100, multiplier);
    d.lbl_prospect.Text = sprintf('Prospect theory: perceive -£%s  vs  actual -£%s
(gap: £%s)', ...
        fmtN(round(perceived_loss)), fmtN(round(actual_loss)),
fmtN(round(perception_gap)));

    % Warning
    if in_debt
        wt = '⚠ PROBLEM GAMBLING: Betting beyond DI. Risk of significant debt. Source:
The Lancet, ref 9.';
        wc = [1 0.3 0.4];
    elseif abs(loss_pct) > 20
        wt = '⚠ HIGH RISK: Losses exceed 20% of DI. Kellogg NW study links this to
measurable savings reduction.';
        wc = [1 0.3 0.4];
    elseif abs(loss_pct) > 10
        wt = '⚠ Losses exceed 10% of DI — significant financial strain likely.';
        wc = [1 0.65 0.2];
    elseif net_pnl > 0
        wt = sprintf('✓ Year 1 positive due to £%.0f bonus. Reverses in year 2 once
promotion expires.', bonus);
        wc = [0.2 1 0.5];
    else
        wt = 'Losses within manageable range relative to disposable income.';
        wc = [0.4 0.8 1];
    end
    d.lbl_warn.Text      = wt;
    d.lbl_warn.FontColor = wc;

    % Breakdown values
    bk_v = {sprintf('%.1f%%',h_eff*100), sprintf('%.3f',p), ...
            sprintf('%.1fx',multiplier), sprintf('£%.4f',ev_per_1), ...
            sprintf('£%.2f',s0), sprintf('%.2f',lambda), ...
            sprintf('£%.2f',avg_stake), sprintf('%d bets',n), ...
            ['£',fmtN(round(total_wagered))], ...
            ['-£',fmtN(round(base_loss))], ...
            ['-£',fmtN(round(chase_extra))], ...
            ['-£',fmtN(round(escalation_extra))], ...
            ['-£',fmtN(round(perceived_loss))], ...
            ['+£',fmtN(round(bonus))], ...
            ''};
    if net_pnl >= 0
        bk_v{15} = ['+£', fmtN(round(net_pnl))];
    else
        bk_v{15} = ['-£', fmtN(round(abs(net_pnl)))];
    end
```

```
bk_c = {[0.9 0.9 0.9],[0.4 0.8 1],[0.4 0.8 1],[1 0.5 0.5], ...
        [0.8 0.8 0.4],[1 0.6 0.8],[0.8 0.8 0.4],[0.8 0.8 0.4], ...
        [1 0.6 0.2],[1 0.4 0.4],[1 0.4 0.8],[1 0.6 0.4], ...
        [0.8 0.5 1],[0.2 1 0.5], ...
        [0 1 0.5]};
if net_pnl < 0; bk_c{15} = [1 0.3 0.4]; end

for i = 1:15
    d.lbl_bk{i}.Text      = bk_v{i};
    d.lbl_bk{i}.FontColor = bk_c{i};
end


% 5 paths with all non-linear effects

rng(42);
n_paths  = 5;
weeks    = 0:52;
bets_pw  = n / 52;
paths    = zeros(n_paths, 53);

% Expected P&L curve (analytical, week by week)
exp_curve = zeros(1,53);
exp_curve(1) = bonus;
for w = 1:52
    t = w/52;
    s_t = s0 * exp(lambda*t);          % escalated stake at week w
    wk_loss = bets_pw * s_t * h_eff;
    exp_curve(w+1) = exp_curve(w) - wk_loss;
end
if in_debt; exp_curve = exp_curve - debt_extra*(0:52)/52; end

for path = 1:n_paths
    cum    = 0;
    ruined = false;
    bonus_done = false;
    consec_losses = 0;

    for w = 1:52
        if ruined
            if in_debt
                % Problem gambler continues on credit
                cum = cum - bets_pw * avg_stake * h_eff;
            end
            paths(path, w+1) = cum;
            continue;
        end

        if is_new && ~bonus_done
            cum = cum + bonus;
            bonus_done = true;
        end

        % Escalated stake this week
        t = w/52;
```

```
            s_t = s0 * exp(lambda * t);
            s_t = max(2, s_t);

            n_w = max(0, round(bets_pw + randn*sqrt(max(bets_pw,0.1))));

            for b = 1:n_w
                if cum <= -DI && ~in_debt
                    ruined = true; break;
                end

                % Chases: stake doubles after each consecutive loss
                if chases & consec_losses > 0
                    curr_stake = s_t * (2^min(consec_losses, 8));
                else
                    curr_stake = s_t;
                end

                if rand < p
                    cum = cum + curr_stake * multiplier;
                    consec_losses = 0;  % reset streak on win
                else
                    cum = cum - curr_stake;
                    consec_losses = consec_losses + 1;
                end
            end
            paths(path, w+1) = cum;
        end
    end

    % Plot
    path_cols = {[0.5 0.6 0.9],[0.4 0.8 0.9],[0.7 0.5 0.9], ...
                 [0.5 0.9 0.6],[0.9 0.8 0.4]};

    cla(d.ax); hold(d.ax,'on');
    for path = 1:n_paths
        plot(d.ax, weeks, paths(path,:), ...
            'Color',[path_cols{path}, 0.5],'LineWidth',1.2);
    end
    plot(d.ax, weeks, exp_curve, 'r-','LineWidth',2.8);
    yline(d.ax, 0,'--','Color',[0.5 0.5 0.5],'LineWidth',1);
    hold(d.ax,'off');

    if net_pnl >= 0; sgn = '+'; else; sgn = '-'; end
    xlabel(d.ax,'Week','Color',[0.7 0.7 0.9],'FontSize',9);
    ylabel(d.ax,'Cumulative P&L (£)','Color',[0.7 0.7 0.9],'FontSize',9);
    title(d.ax, sprintf('Monte Carlo Paths  |  Expected: %s£%s/yr', ...
        sgn, fmtN(round(abs(net_pnl)))), ...
        'Color',[0.9 0.9 1],'FontSize',10);
    legend(d.ax,{'P1','P2','P3','P4','P5','E[P&L]'}, ...
        'TextColor',[0.8 0.8 0.9],'Color',[0.08 0.08 0.14], ...
        'Location','southwest','FontSize',7);
end


%  UI HELPERS
```

```
function c = iLbl(p,t,pos)
    c = uilabel(p,'Text',t,'Position',pos,'FontColor',[0.4 0.8 1], ...
        'FontWeight','bold','FontSize',9);
end
function c = iEdit(p,v,pos)
    c = uieditfield(p,'numeric','Value',v,'Limits',[0 Inf],'Position',pos, ...
        'BackgroundColor',[0.06 0.06 0.10],'FontColor',[1 0.65 0.2], ...
        'FontSize',13,'FontWeight','bold');
end
function c = iDrop(p,items,val,pos)
    c = uidropdown(p,'Items',items,'Value',val,'Position',pos, ...
        'BackgroundColor',[0.06 0.06 0.10],'FontColor',[1 1 1]);
end
function s = fmtN(n)
    str = num2str(abs(n)); len = length(str);
    s = '';
    for i = 1:len
        s = [s, str(i)];
        if mod(len-i,3)==0 && i<len; s=[s,',']; end
    end
    if n<0; s=['-',s]; end
end
```


Demographic Inputs
Annual Disposable Income (£)
1.2e+04
Age Group
35-49
Gender
Male
Education
No college degree
Betting Frequency
1-2x/week
Aggression Score (1=Cautious → 10=High Risk)
5
New Customer?
No
Behavioural Parameters
Chasing behaviour
No chasing
Addiction / problem gambling
No
Research frequency
Occasionally
Addiction escalation (λ)
None (λ=0)
CALCULATE
Model Breakdown
House edge h_eff 11.0%
Win probability p 0.272
Bet multiplier 3.3x
E(V) per £1 £0.1622
Base stake s0 (£) £25.00
Escalation lambda 0.00
Effective avg stake (£) £25.00
Bets per year n 106 bets
Total wagered (£) £2,650
Base loss (£) -£292
Chase Losses (£) -£0
Escalation extra (£) -£0
Prospect perceived loss (£) -£332
Free bet bonus (£) +£0
NET P&L (£) -£292


Annual Expected Result
Expected Annual P&L
-£292
-£24/month | Total wagered: £2,650
2.4% of annual disposable income
E(V)/£1 = £0.1622 | p = 0.272 | h_eff = 11.0% | mult = 3.3x
Prospect theory: perceive -£332 vs actual -£292 (gap: £-40)
Losses within manageable range relative to disposable income.
Monte Carlo Paths | Expected: -£292/yr
Cumulative P&L (£)
Week
P1
P2
P3
P4
P5
E[P&L]

Q3: Don't Break the Bank

```
function modelofpension6()

    fig = uifigure('Name','M3 Challenge 2026 — Q3: Pension & Retirement Model', ...
        'Position',[50 30 1060 770],'Color',[0.08 0.08 0.12]);

    uilabel(fig,'Text','Pension Retirement Model', ...
        'Position',[0 730 1060 32],'FontSize',24,'FontWeight','bold', ...
        'FontColor',[1 1 1],'HorizontalAlignment','center');
    uilabel(fig,'Text','How Gambling Erodes Your Pension  |  M3 Challenge 2026 — Q3',
...
        'Position',[0 708 1060 20],'FontSize',11, ...
        'FontColor',[0.4 0.8 1],'HorizontalAlignment','center');

    lp = uipanel(fig,'Position',[12 12 480 690], ...
        'BackgroundColor',[0.11 0.11 0.17],'ForegroundColor',[1 1 1], ...
        'Title',' Inputs ','FontSize',11,'FontWeight','bold');

    lbl(lp,'Annual Gross Salary (£)',[14 645 220 16]);
    ef_sal = etxt(lp,'35000',[14 622 220 26]);
    lbl(lp,'Current Age',[254 645 200 16]);
    ef_age = enum(lp,30,[254 622 200 26]);

    lbl(lp,'Current Pension Pot (£)',[14 591 220 16]);
    ef_pot = etxt(lp,'5000',[14 568 220 26]);
    lbl(lp,'Annual Salary Growth (%)',[254 591 200 16]);
    ef_grow = enum(lp,2,[254 568 200 26]);

    lbl(lp,'Age Group',[14 537 220 16]);
    dd_age = edrp(lp,{'18-34','35-49','50-64','65+'},'18-34',[14 514 220 26]);
    lbl(lp,'Gender',[254 537 200 16]);
    dd_sex = edrp(lp,{'Male','Female'},'Male',[254 514 200 26]);

    uilabel(lp,'Text','── Gambling Behaviour ───────────────────────────────', ...
        'Position',[14 486 440 16],'FontColor',[0.4 0.8 1],'FontSize',9);

    lbl(lp,'Education',[14 467 220 16]);
    dd_edu = edrp(lp,{'No college degree','BA or higher'},'No college degree',[14 444
220 26]);
    lbl(lp,'Betting Frequency',[254 467 200 16]);
    dd_frq = edrp(lp,{'< Once/month','Once/month','Few times/month','1-2x/week','3-
4x/week','Daily'}, ...
        '1-2x/week',[254 444 200 26]);

    lbl(lp,'Aggression Score  (1=Cautious, 10=High Risk)',[14 413 440 16]);
    ef_agg = enum(lp,5,[14 390 220 26]);
    lbl(lp,'Chasing Behaviour',[254 413 200 16]);
    dd_chs = edrp(lp,{'No chasing','Chase losses (Martingale)'},'No chasing',[254 390
200 26]);

    lbl(lp,'Research Frequency',[14 305 220 16]);
    dd_res = edrp(lp,{'Never','Occasionally','Daily (2x+)'},'Occasionally',[14 282 220
26]);
    lbl(lp,'Addiction Escalation',[254 305 200 16]);
```

```
    dd_lam = edrp(lp,{'None (L=0)','Mild (L=0.03)','Moderate (L=0.07)','Severe
(L=0.15)'}, ...
        'None (L=0)',[254 282 200 26]);

    uibutton(fig,'Text','CALCULATE', ...
        'Position',[12 228 480 40], ...
        'BackgroundColor',[0.15 0.65 1],'FontColor',[0.02 0.02 0.08], ...
        'FontSize',15,'FontWeight','bold', ...
        'ButtonPushedFcn',@(~,~) run_model(fig));

    % Breakdown panel
    bk = uipanel(fig,'Position',[12 12 480 210], ...
        'BackgroundColor',[0.11 0.11 0.17],'ForegroundColor',[1 1 1], ...
        'Title',' Year 1 Breakdown ','FontSize',10,'FontWeight','bold');
    rows = {'Gross salary','Income tax (HMRC 2024/25)','National Insurance (HMRC
2024/25)', ...
        'Net income','Essential spending (ONS 2024 — M3 data file)', ...
        'True disposable income','Base stake s0 (£/bet)','Avg stake s_avg (£/bet)', ...
        'Annual gambling loss (Q2 model)','DI after gambling', ...
        'Employee pension 5% (Pensions Act 2008)','Employer match 3% (Pensions Act
2008)', ...
        'Tax relief 20% (HMRC)','Total into pension this year', ...
        '—— Sensitivity (gap at 67) ——————————', ...
        'No-gambling pot at 67 (central r=6%)', ...
        'Gambling pot at 67 (central r=6%)', ...
        'Gap if return = 4% (low)', ...
        'Gap if return = 6% (central)','Gap if return = 8% (high)'};
    val_lbls = cell(20,1);
    for i = 1:20
        y = 250-(i-1)*12;
        uilabel(bk,'Text',rows{i},'Position',[8 y 260 12], ...
            'FontColor', [0.4 0.8 1]*(i==13) + [0.6 0.6 0.8]*(i~=13),'FontSize',8);
        val_lbls{i} = uilabel(bk,'Text','—','Position',[270 y 190 12], ...
            'FontColor',[0.9 0.9 0.9],'FontSize',8,'FontWeight','bold', ...
            'HorizontalAlignment','right');
    end

    % 4 result boxes
    boxes    = {'Pot at 67 (no gambling)','Pot at 67 (with gambling)','Gap at 67','Gap
as % of pot'};
    box_clrs = {[0.2 0.95 0.4],[1 0.3 0.4],[0.8 0.4 1],[1 0.65 0.2]};
    big_lbls = cell(4,1);
    for i = 1:4
        xb = 502+(i-1)*138;
        uipanel(fig,'Position',[xb 620 132 90], ...
            'BackgroundColor',[0.11 0.11 0.17],'ForegroundColor',[0.5 0.5 0.6], ...
            'Title',boxes{i},'FontSize',7.5,'FontWeight','bold');
        big_lbls{i} = uilabel(fig,'Text','—', ...
            'Position',[xb 632 132 60], ...
            'FontColor',box_clrs{i},'FontSize',20,'FontWeight','bold', ...
            'HorizontalAlignment','center');
    end

    sp = uipanel(fig,'Position',[502 558 555 58], ...
        'BackgroundColor',[0.11 0.11 0.17],'ForegroundColor',[1 1 1], ...
```

```
        'Title',' Lifetime Cost ','FontSize',10,'FontWeight','bold');
    lbl_lt1 = uilabel(sp,'Text','','Position',[8 28 535 16], ...
        'FontColor',[1 0.4 0.4],'FontSize',9,'FontWeight','bold');
    lbl_lt2 = uilabel(sp,'Text','','Position',[8 10 535 16], ...
        'FontColor',[0.8 0.4 1],'FontSize',9,'FontWeight','bold');

    ax = uiaxes(fig,'Position',[502 12 555 538]);
    ax.Color=[0.10 0.10 0.16]; ax.XColor=[0.6 0.6 0.8]; ax.YColor=[0.6 0.6 0.8];
    ax.GridColor=[0.25 0.25 0.35]; ax.GridAlpha=0.4;
    ax.XGrid='on'; ax.YGrid='on';
    xlabel(ax,'Age','Color',[0.6 0.6 0.8],'FontSize',10);
    ylabel(ax,'Pension Pot  (£k)','Color',[0.6 0.6 0.8],'FontSize',10);
    title(ax,'Press Calculate','Color',[0.6 0.6 0.8],'FontSize',11);

    fig.UserData = struct('ef_sal',ef_sal,'ef_age',ef_age,'ef_pot',ef_pot, ...
        'ef_grow',ef_grow,'dd_age',dd_age,'dd_sex',dd_sex,'dd_edu',dd_edu, ...
        'dd_frq',dd_frq,'ef_agg',ef_agg,'dd_chs',dd_chs,'dd_res',dd_res, ...
        'dd_lam',dd_lam,'big_lbls',{big_lbls},'val_lbls',{val_lbls}, ...
        'lbl_lt1',lbl_lt1,'lbl_lt2',lbl_lt2,'ax',ax);
end

function run_model(fig)
    d = fig.UserData;

    % READ ALL INPUTS FRESH
    salary   = parse_num(d.ef_sal.Value, 35000);
    cur_age  = round(d.ef_age.Value);
    cur_pot  = parse_num(d.ef_pot.Value, 5000);
    sal_grow = d.ef_grow.Value / 100;
    age_str  = d.dd_age.Value;
    gender   = d.dd_sex.Value;
    edu      = d.dd_edu.Value;
    freq_str = d.dd_frq.Value;
    agg      = max(1, min(10, round(d.ef_agg.Value)));
    chasing  = strcmp(d.dd_chs.Value,'Chase losses (Martingale)');
    research = d.dd_res.Value;
    lam_str  = d.dd_lam.Value;

    % TAX & NI — HMRC 2024/25
    tax = calc_tax(salary);
    NI  = calc_ni(salary);
    net = salary - tax - NI;

    % ESSENTIAL SPENDING — ONS 2024 UK Family Spending Survey

    age_mid = get_age_mid(age_str);
    if age_mid <= 30
        essential = (316.1 / 2.1) * 52;   % £7,827
    elseif age_mid <= 49
        essential = (317.3 / 3.0) * 52;   % £5,500
    else
        essential = (305.9 / 2.3) * 52;   % £6,916
    end
    DI = max(0, net - essential);
```

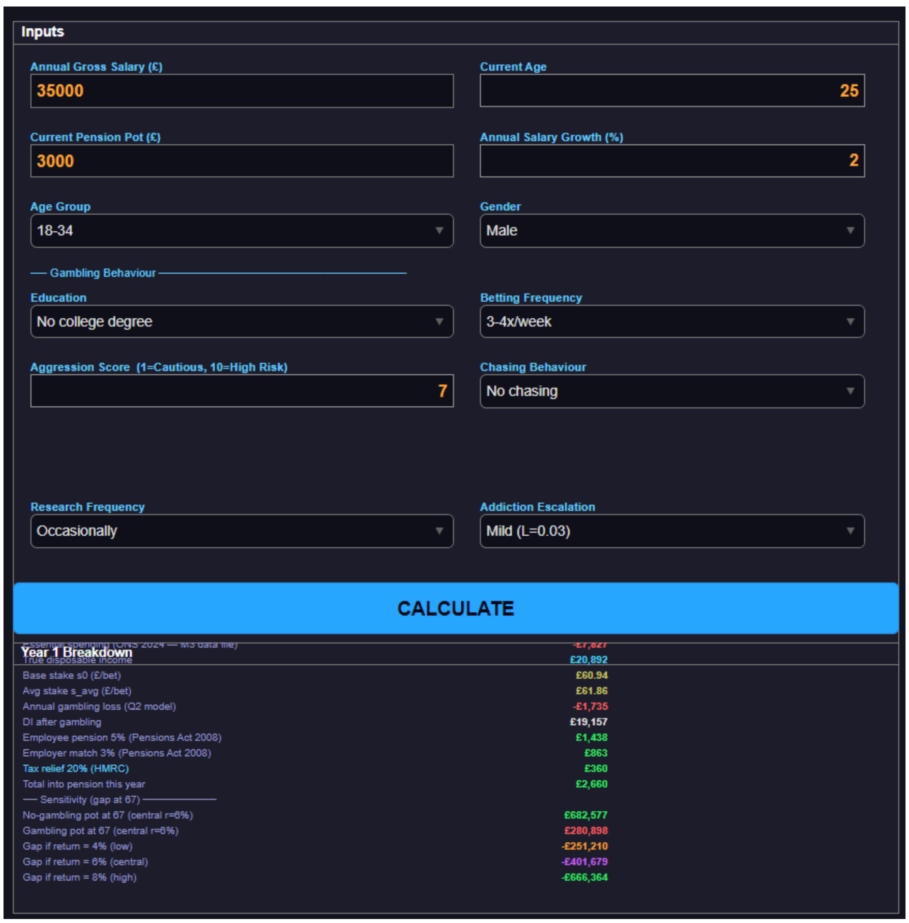
```matlab
% GAMBLING LOSS — Q2 model
[gloss, h_eff] = calc_gambling(DI, age_str, gender, edu, ...
                               freq_str, agg, chasing, research, lam_str);

% PENSION

qual_yr1  = max(0, salary - 6240);
emp_pct   = 0.05;
er_pct    = 0.03;
tr_rate   = 0.20;
growth    = 0.06;

% SIMULATE to age 67
end_age = 67;
n_yrs   = max(0, end_age - cur_age);

ages      = cur_age : (cur_age + max(n_yrs,1));
pot_g     = zeros(1, max(n_yrs,1)+1);
pot_ng    = zeros(1, max(n_yrs,1)+1);
pot_g(1)  = cur_pot;
pot_ng(1) = cur_pot;
sal = salary;

% FIX 2: hard cap — gambling loss cannot exceed 95% of DI
% FIX 3: pension pot floor = cur_pot
di_cap = DI * 0.95;

for yr = 1:max(n_yrs,1)
    if n_yrs == 0; break; end
    if yr > 1; sal = sal*(1+sal_grow); end
    qual     = max(0, sal - 6240);
    emp_c    = emp_pct * qual;
    er_c     = er_pct  * qual;
    tr_c     = emp_c * tr_rate / (1 - tr_rate);
    contrib = emp_c + er_c + tr_c;

    gloss_yr = min(gloss * (sal / salary), di_cap * (sal / salary));

    pot_ng(yr+1) = pot_ng(yr)*(1+growth) + contrib;


    pot_g(yr+1) = max(cur_pot, pot_g(yr)*(1+growth) + contrib - gloss_yr);
end

pot67_ng = pot_ng(end);
pot67_g  = pot_g(end);
gap67    = pot67_ng - pot67_g;
gap_pct  = 100 * gap67 / max(pot67_ng, 1);
total_lost = min(gloss, di_cap) * n_yrs;

% SENSITIVITY ANALYSIS
if n_yrs > 0
    gap_lo = run_sensitivity(cur_pot, n_yrs, gloss, di_cap, salary, sal_grow, 0.04);
    gap_hi = run_sensitivity(cur_pot, n_yrs, gloss, di_cap, salary, sal_grow, 0.08);
else
```

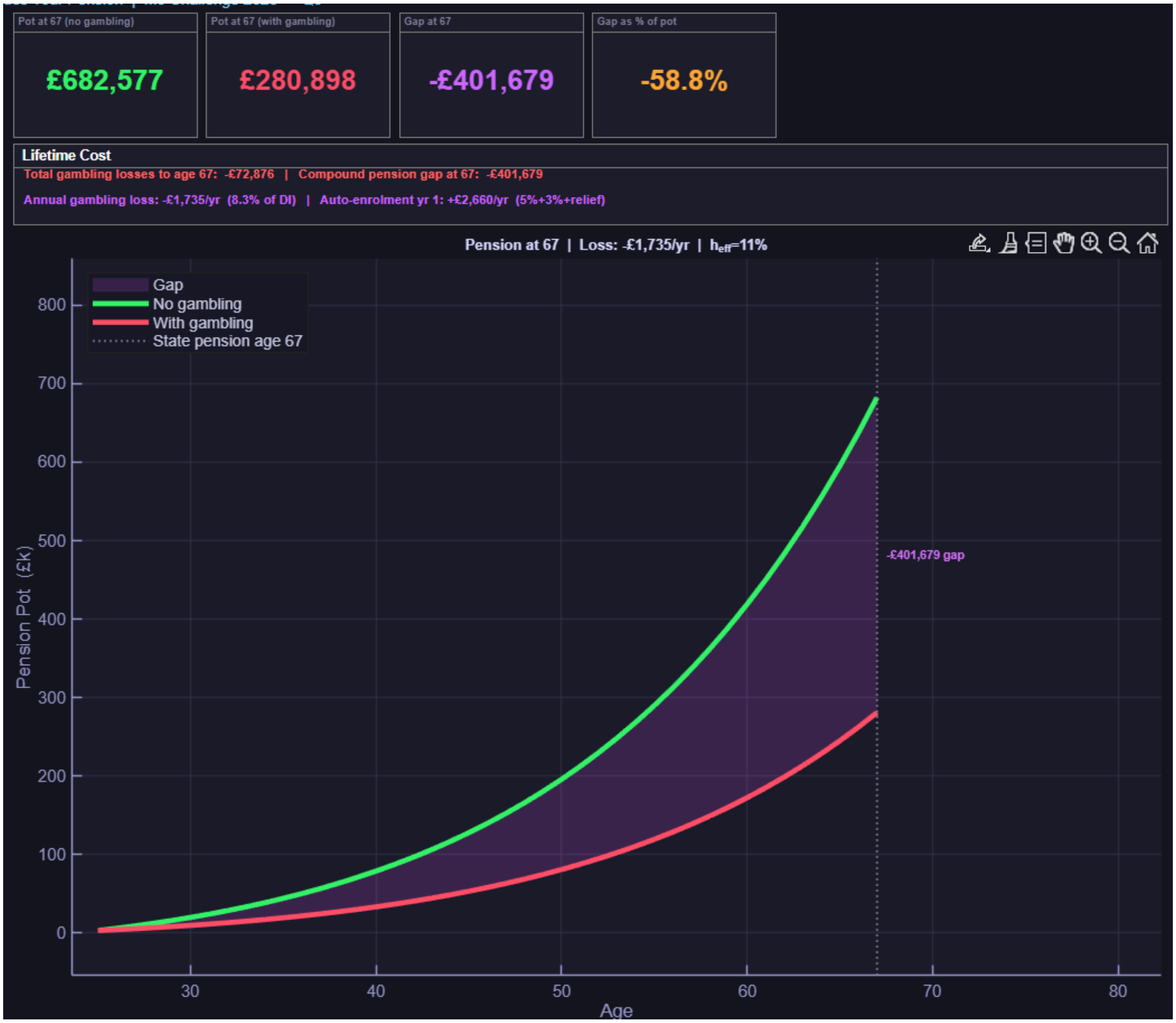
```
        gap_lo = 0; gap_hi = 0;
    end

    % UPDATE BOXES
    bl = d.big_lbls;
    bl{1}.Text = sprintf('£%s', fmt(round(pot67_ng)));
    bl{2}.Text = sprintf('£%s', fmt(round(pot67_g)));
    bl{3}.Text = sprintf('-£%s', fmt(round(gap67)));
    bl{4}.Text = sprintf('-%.1f%%', gap_pct);

    % LIFETIME LABELS
    d.lbl_lt1.Text = sprintf( ...
        'Total gambling losses to age 67:  -£%s   |   Compound pension gap at 67:  -
£%s', ...
        fmt(round(total_lost)), fmt(round(gap67)));
    auto1 = emp_pct*qual_yr1 + er_pct*qual_yr1 + emp_pct*qual_yr1*tr_rate/(1-tr_rate);
    d.lbl_lt2.Text = sprintf( ...
        'Annual gambling loss: -£%s/yr  (%.1f%% of DI)   |   Auto-enrolment yr 1:
+£%s/yr  (5%%+3%%+relief)', ...
        fmt(round(gloss)), 100*gloss/max(DI,1), fmt(round(auto1)));

    % YEAR 1 BREAKDOWN
    emp_c1 = emp_pct*qual_yr1; er_c1 = er_pct*qual_yr1;
    tr_c1  = emp_c1*tr_rate/(1-tr_rate);
    % Retrieve s0 and s_avg from gambling calc for display
    s0_disp   = max(2, (0.005+(agg-1)*(0.045/9))*(DI/12));
    lam_map2  = containers.Map( ...
        {'None (L=0)','Mild (L=0.03)','Moderate (L=0.07)','Severe (L=0.15)'}, ...
        [0, 0.03, 0.07, 0.15]);
    lam2 = lam_map2(d.dd_lam.Value);
    if lam2 > 0; s_avg_disp = s0_disp*(exp(lam2)-1)/lam2; else; s_avg_disp = s0_disp;
end

    vals = {sprintf('£%s',fmt(round(salary))), sprintf('-£%s',fmt(round(tax))), ...
        sprintf('-£%s',fmt(round(NI))), sprintf('£%s',fmt(round(net))), ...
        sprintf('-£%s',fmt(round(essential))), sprintf('£%s',fmt(round(DI))), ...
        sprintf('£%.2f',s0_disp), sprintf('£%.2f',s_avg_disp), ...
        sprintf('-£%s',fmt(round(gloss))), sprintf('£%s',fmt(round(max(0,DI-gloss)))),
...
        sprintf('£%s',fmt(round(emp_c1))), sprintf('£%s',fmt(round(er_c1))), ...
        sprintf('£%s',fmt(round(tr_c1))), sprintf('£%s',fmt(round(emp_c1+er_c1+tr_c1))),
...
        '', sprintf('£%s',fmt(round(pot67_ng))), sprintf('£%s',fmt(round(pot67_g))), ...
        sprintf('-£%s',fmt(round(gap_lo))), ...
        sprintf('-£%s',fmt(round(gap67))), sprintf('-£%s',fmt(round(gap_hi)))};
    clrs = {[0.9 0.9 0.9],[1 0.4 0.4],[1 0.4 0.4],[0.9 0.9 0.9], ...
            [1 0.4 0.4],[0.3 0.85 1],[0.8 0.8 0.4],[0.8 0.8 0.4], ...
            [1 0.4 0.4],[0.9 0.9 0.9], ...
            [0.2 0.95 0.4],[0.2 0.95 0.4],[0.2 0.95 0.4],[0.2 0.95 0.4], ...
            [0.4 0.8 1],[0.2 0.95 0.4],[1 0.4 0.4], ...
            [1 0.65 0.2],[0.8 0.4 1],[0.2 0.95 0.4]};
    for i = 1:20
        d.val_lbls{i}.Text = vals{i};
        d.val_lbls{i}.FontColor = clrs{i};
    end
```

```
    % CHART
    ax = d.ax; cla(ax); hold(ax,'on');
    fill(ax,[ages,fliplr(ages)],[pot_ng/1e3,fliplr(pot_g/1e3)], ...
        [0.8 0.3 0.9],'FaceAlpha',0.18,'EdgeColor','none');
    plot(ax,ages,pot_ng/1e3,'Color',[0.2 0.95 0.4],'LineWidth',2.8);
    plot(ax,ages,pot_g/1e3, 'Color',[1 0.3 0.4],   'LineWidth',2.8);
    xline(ax,67,':','Color',[0.55 0.55 0.65],'LineWidth',1.2);
    ymid = (pot67_ng+pot67_g)/2/1e3;
    if gap67 > 200
        text(ax,67.5,ymid,sprintf('-£%s gap',fmt(round(gap67))), ...
            'Color',[0.85 0.45 1],'FontSize',8.5,'FontWeight','bold');
    end
    hold(ax,'off');
    legend(ax,{'Gap','No gambling','With gambling','State pension age 67'}, ...
        'TextColor',[0.8 0.8 0.9],'Color',[0.09 0.09 0.15], ...
        'Location','northwest','FontSize',8.5);
    title(ax,sprintf('Pension at 67  |  Loss: -£%s/yr  |  h_{eff}=%.0f%%', ...
        fmt(round(gloss)),h_eff*100),'Color',[0.85 0.85 1],'FontSize',10);
    xlabel(ax,'Age','Color',[0.6 0.6 0.8],'FontSize',10);
    ylabel(ax,'Pension Pot  (£k)','Color',[0.6 0.6 0.8],'FontSize',10);
end

function [net_loss, h_eff] = calc_gambling(DI, age_str, gender, edu, ...
        freq_str, agg, chasing, research, lam_str)
    h = 0.10;
    if strcmp(edu,'BA or higher'); h=h-0.02; else; h=h+0.01; end
    if strcmp(research,'Daily (2x+)'); h=h-0.01; end
    if strcmp(research,'Never');       h=h+0.01; end
    h_eff = max(0.04,h);

    stake_pct = 0.005 + (agg-1)*(0.045/9);
    s0 = max(2, stake_pct*(DI/12));

    % Addiction escalation — matches Q2 non-linear model exactly

    lam_map = containers.Map( ...
        {'None (L=0)','Mild (L=0.03)','Moderate (L=0.07)','Severe (L=0.15)'}, ...
        [0, 0.03, 0.07, 0.15]);
    lambda = lam_map(lam_str);
    if lambda > 0
        s_avg = s0 * (exp(lambda)-1)/lambda;  % integral of exp curve over [0,1]
    else
        s_avg = s0;                            % no escalation — flat stake
    end

    freq_map = containers.Map( ...
        {'< Once/month','Once/month','Few times/month','1-2x/week','3-4x/week','Daily'},
...
        [6,12,30,78,182,365]);
    age_map = containers.Map({'18-34','35-49','50-64','65+'}, [35/34,1.0,10/34,4/34]);
    sex_map = containers.Map({'Male','Female'}, [30/22,15/22]);

    n = round(freq_map(freq_str)*age_map(age_str)*sex_map(gender));
    n = max(1, min(n, floor(DI*1.5/max(s_avg,1))));
```

```
    base_loss = n*s_avg*h_eff;  % escalation already embedded in s_avg

    mart = 0;
    if chasing
        ac_map = containers.Map({'18-34','35-49','50-64','65+'}, [0.58,0.54,0.38,0.15]);
        sc_map = containers.Map({'Male','Female'}, [0.57,0.41]);
        r_c = (ac_map(age_str)+sc_map(gender))/2;
        p   = 0.45-(agg-1)*(0.40/9);
        k   = min((1-p)/p,10);
        ep  = s_avg*(2^(k+1)-2);  % use escalated avg stake, matching Q2 model
        mart = n*r_c*0.15*ep*h_eff;
        mart = min(mart, max(0, DI - base_loss) * h_eff);
    end
    net_loss = min(base_loss+mart, DI*0.95);
end

function t = calc_tax(s)
    t=0;
    if s>125140; t=t+(s-125140)*0.45; s=125140; end
    if s>50270;  t=t+(s-50270)*0.40;  s=50270;  end
    if s>12570;  t=t+(s-12570)*0.20;             end
end
function n = calc_ni(s)
    n=0;
    if s>50270; n=n+(s-50270)*0.02; s=50270; end
    if s>12570; n=n+(s-12570)*0.08;           end
end
function m = get_age_mid(s)
    mp=containers.Map({'18-34','35-49','50-64','65+'}, [26,42,57,70]);
    m=mp(s);
end
function v = parse_num(s,fallback)
    v=str2double(s); if isnan(v)||v<=0; v=fallback; end
end
function s = fmt(n)
    if n>=1e6; s=sprintf('%.2fm',n/1e6);
    elseif n>=1e3; s=sprintf('%d,%03d',floor(n/1e3),mod(round(n),1000));
    else; s=num2str(round(n)); end
end
function c = lbl(p,t,pos)
    c=uilabel(p,'Text',t,'Position',pos,'FontColor',[0.4 0.8
1],'FontWeight','bold','FontSize',9);
end
function c = etxt(p,v,pos)
    c=uieditfield(p,'text','Value',v,'Position',pos,'BackgroundColor',[0.06 0.06 0.10],
...
        'FontColor',[1 0.65 0.2],'FontSize',13,'FontWeight','bold');
end
function c = enum(p,v,pos)
    c=uieditfield(p,'numeric','Value',v,'Limits',[0 Inf],'Position',pos, ...
        'BackgroundColor',[0.06 0.06 0.10],'FontColor',[1 0.65
0.2],'FontSize',13,'FontWeight','bold');
end
function c = edrp(p,items,def,pos)
```

```
    c=uidropdown(p,'Items',items,'Value',def,'Position',pos,'BackgroundColor',[0.06 0.06 0.10], ...
        'FontColor',[0.9 0.9 0.9],'FontSize',11);
end

function gap = run_sensitivity(cur_pot, n_yrs, gloss, di_cap, salary, sal_grow, r)
    % Re-simulate with alternate return rate for sensitivity analysis
    % gloss and di_cap already computed with correct s_avg formula — reuse directly
    pg = cur_pot; png = cur_pot; sal = salary;
    for yr = 1:n_yrs
        if yr > 1; sal = sal*(1+sal_grow); end
        q   = max(0, sal-6240);
        c   = 0.05*q + 0.03*q + 0.05*q*0.20/0.80;   % emp + er + tax relief
        g   = min(gloss*(sal/salary), di_cap*(sal/salary));
        png = png*(1+r) + c;
        pg  = max(cur_pot, pg*(1+r) + c - g);
    end
    gap = png - pg;
end
```




Pot at 67 (no gambling)
£682,577
Pot at 67 (with gambling)
£280,898
Gap at 67
-£401,679
Gap as % of pot
-58.8%
Lifetime Cost
Total gambling losses to age 67: -£72,876 | Compound pension gap at 67: -£401,679
Annual gambling loss: -£1,735/yr (8.3% of DI) | Auto-enrolment yr 1: +£2,660/yr (5%+3%+relief)
Pension at 67 | Loss: -£1,735/yr | heff=11%
Gap
No gambling
With gambling
State pension age 67
-£401,679 gap
Pension Pot (£k)
Age