Y broblem Monty Hall ac effaith y geifr

Vince Knight

1 Cyflwyniad: Y Broblem Monty Hall

Problem mewn tebygolrwydd yw'r broblem Monty Hall, yn seiliedig ar y sioe gêm [1]. Yn y sioe cyflwynir cystadleuydd gyda thri drws. Tu ôl un o'r drysau oedd car, a thu ôl y gweddill oedd geifr. Bydd y cystadleuydd yn dewis drws (y bydd yn cuddio naill ai gafr neu gar) a bydd y cyflwynydd yna'n agor drws i dangos leoliad yr afr. Ar y cam yma mae gan y cystadleuydd dewis strategol:

- Aros: os oedd y newid gwreiddiol yn gar yna maent wedi ennill!
- Newid: cymerwch beth bynnag sydd tu ôl i'r drws gweddill.

Fe wnaeth hwn drysu mathemategwyr am flynyddoedd oherwydd mae'n i'w weld bod gan y cystadleuydd dewis ffug a bod y tebygolrwydd o ennill yr un peth os ydynt yn newid neu beidio. Nid yw hyn yn wir: os yw'r cystadleuydd yn newid maent yn dyblu ei siawns o ennill!

Yn y gwaith hwn byddwn yn gobeithio egluro hwn trwy ystyried beth sy'n digwydd os ydyn ni'n newid y gêm: gadewch i ni gael n gafr tu ôl i n drws. Mae popeth arall yn aros yr un peth: mae'r cyflwynwr dal yn agor un o'r drysau sydd â gafr tu ôl iddo.

Yn Adran 2 defnyddir Python i efelychu'r ddwy strategaeth a chanfod y tebygolrwydd o ennill. Yna yn Adran 3 byddwn yn deillio a gwirio fformiwla fathemategol ar gyfer y tebygolrwydd o ennill.

2 Efelychu mwy o geifr

Dyma ddau ffwythiant python sy'n efelychu aros neu newid gyda nifer penodol o eifr:

```
import random

def aros(nifer_geifr=2):
    """Ffwythiant i efelychu chwarae'r gem os ydyn yn aros"""
    drysau = ['Car'] + nifer_geifr * ['Gafr']

    dewis_gwreiddiol = random.choice(drysau)  # gwneud dewis
    return dewis_gwreiddiol == 'Car'

def newid(nifer_geifr=2):
    """Ffwythiant i efelychu chwarae'r gem os ydyn yn newid"""
    drysau = ['Car'] + nifer_geifr * ['Gafr']

    dewis_gwreiddiol = random.choice(drysau)  # gwneud dewis

    drysau.remove(dewis_gwreiddiol)  # Newid: cael gwared a'r dewis gwreiddiol
    drysau.remove('Gafr')  # Mae'r cyflwynwr yn dangos gafr i ni

    dewis_newydd = random.choice(drysau)   # Rydym yn dewis yr un opsiwn sydd ar ol

    return dewis_newydd == 'Car'
```

Yn defnyddio hwn gallwn efelychu'r tebygolrwyddau ar gyfer yr achos o 2 gafr (y sioe teledu wreiddiol!):

```
>>> ailadroddiadau = 10000
>>> random.seed(0)
>>> tebyg_ennill_aros = sum([aros() for ail in range(ailadroddiadau)]) / ailadroddiadau
>>> tebyg_ennill_newid = sum([newid() for ail in range(ailadroddiadau)]) / ailadroddiadau
>>> tebyg_ennill_aros, tebyg_ennill_newid
(0.3346, 0.6636)
```

Mae hwn wir yn edrych fel bod y cystadleuydd dwywaith mwy tebygol o ennill os ydynt yn newid.

3 Fformiwlâu analytig

Gallwn gyfrifo'r tebygolrwydd p_n o ennill pan fyddwn yn newid pan mae n gafr. Er mwyn ennill wrth newid:

- Mae angen i'r dewis cyntaf ddim bod yn gar. Os oes n gafr, mae cyfanswm o $n+1$ drws. Felly'r tebygolrwydd o ganfod car yw $\frac{1}{n+1}$. Felly'r tebygolrwydd o **ddim** canfod car yw $1-\frac{1}{n+1}$.
- Mae newid yn awgrymu dewis car o'r drysau gweddill. Ond rydym wedi cael gwared ar ddau ddrws (y dewis gwreiddiol a'r drys y mae'r cyflwynwr yn agor). Felly'r tebygolrwydd o ddewis car yw $\frac{1}{n-1}$.

Felly:

$$p_n = \left(1 - \frac{1}{n+1}\right)\left(\frac{1}{n-1}\right) \tag{1}$$
$$p_n = \left(\frac{n+1-1}{n+1}\right)\left(\frac{1}{n-1}\right) \tag{2}$$
$$p_n = \frac{n}{(n+1)(n-1)} \tag{3}$$
$$p_n = \frac{n}{(n^2-1)} \tag{4}$$

Yn defnyddio Python i wirio'r algebra:

```
>>> import sympy as sym
>>> n = sym.symbols('n')
>>> p_n = (1 - 1 / (n + 1)) * (1 / (n - 1))
>>> p_n.simplify()
n/(n**2 - 1)
```

Felly rhoddir y gymhareb i faint yn well yr yw i newid gan:

$$\alpha_n = \frac{p_n}{1/(n+1)} = \frac{n}{n-1}$$

Yn olaf, gadewch i ni gymharu'r fformiwla hon i'r gwerthoedd o'r efelychiad.

```
def cymhareb(ailadroddiadau=50000, nifer_geifr=2):
    """Cael cymhareb y tebygolrwyddau o ennill"""
    tebyg_ennill_aros = sum([aros(nifer_geifr=nifer_geifr)
                             for ail in range(ailadroddiadau)]) / ailadroddiadau
    tebyg_ennill_newid = sum([newid(nifer_geifr=nifer_geifr)
                              for ail in range(ailadroddiadau)]) / ailadroddiadau
    return tebyg_ennill_newid / tebyg_ennill_aros
```

Dyma god Python sy'n plotio'r gymhareb hon (y gwerthoedd o'r efelychiad a'r fformiwlâu) a ddangosir yn Darlun 1.

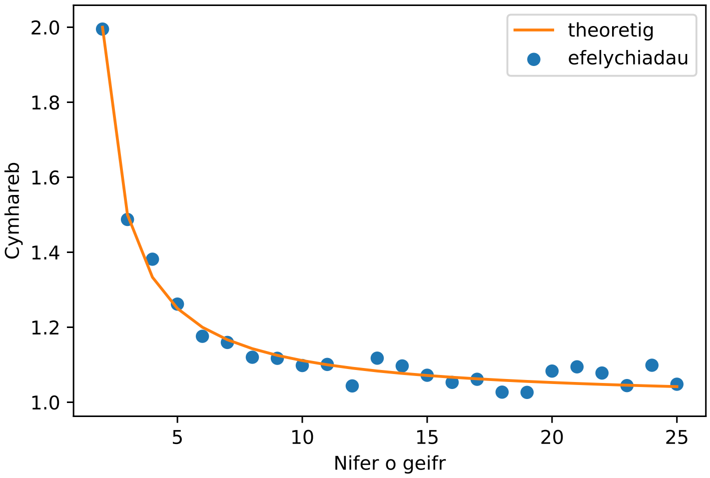
```
>>> random.seed(0)
>>> geifr = range(2, 25 + 1)
>>> cymarebau =>>>  [cymhareb(nifer_geifr=n) for n in geifr]
>>> cymhareb_theoretig = [(n / (n - 1)) for n in geifr]
>>> plt.figure()
>>> plt.scatter(geifr, cymarebau, label="efelychiadau")
>>> plt.plot(geifr, cymhareb_theoretig, color="C1", label="theoretig")
>>> plt.xlabel("Nifer o geifr")
>>> plt.ylabel("Cymhareb")
>>> plt.legend()
>>> plt.savefig("simulated_v_expected_ratio_of_win_probability.pdf")
```

Yn olaf gadewch i ni wirio wrth i nifer o eifr cynyddu mae'r effaith o newid yn mynd yn llai ac yn llai:

$$\lim_{n\to\infty} \alpha_n = \lim_{n\to\infty} \frac{n}{n-1} \tag{5}$$
$$\lim_{n\to\infty} \alpha_n = \lim_{n\to\infty} \frac{1}{1-1/n} \tag{6}$$
$$\lim_{n\to\infty} \alpha_n = 1 \tag{7}$$

Yn defnyddio Python i wirio'r cyfrifiad:

```
>>> import sympy as sym
>>> alpha_n = p_n / (1 / (n + 1))
>>> sym.limit(alpha_n, n, sym.oo)
```



Darlun 1: Yr effaith disgwyliedig ac o'r efelychiad o eifr ar y tebygolrwydd o ennill

1

4 Casgliad

Mae'r adroddiad yma wedi defnyddio mathemateg a Python i uwcholeuo'r problem Monty Hall. Mae fersiwn o'r gêm wedi addasu i ystyried r effaith o gael unrhyw nifer o eifr yn y gêm.

Cawn fformiwla analytig ar gyfer y tebygolrwydd o ennill os yw'r cystadleuydd yn newid.

Wrth i nifer y geifr cynyddu mae'r budd-effaith o newid yn lleihau: mae hwn yn gwneud synnwyr oherwydd po fwyaf nifer y geifr, po leiaf yw effaith y cyflwynwr yn dangos drws gyda gafr tu ôl iddo.

Gall cyfeiriadau ymchwil eraill cynnwys edrych ar effaith ychwanegu mwy o geir, a beth sy'n digwydd os yw'r cyflwynwr yn agor mwy o ddrysau gyda geifr tu ôl iddynt.

Cyfeiriadau

[1] Jason Rosenhouse. *The Monty Hall problem: the remarkable story of Math's most contentious brain teaser*. Oxford University Press, 2009.