Problème de la surréservation

Problème de la surréservation

Étant donné une variable aléatoire X suivant une loi binomiale $\mathcal{B}(n\,;\,p)$, on cherche à déterminer le plus petit entier k tel que $p(X > k) \leqslant \alpha$ où α est un réel strictement positif.

L'activité suivante est fortement inspiré des travaux de l'IREM de la Réunion ainsi que ceux de L. Didier et R. Cabane.

Certaines compagnie aérienne vendent plus de billets qu'il n'y a de place dans leurs avion. L'idée de cette pratique est de compenser les éventuelles pertes quand certains voyageurs annulent leur voyage (maladie, retard, ...). Le taux de personnes qui ne se présentent pas semble se situer en moyenne autour de 5%. Ce taux était plus important il y a quelques années car on pouvait annuler sa réservation sans pénalités.

On appelle cela le surbooking ou surréservation.
Le risque est que certains voyageurs ne puissent pas embarquer en raison d'un manque de place. La compagnie est alors obligé des les indemniser.

Le but est le suivant : comment optimiser le nombre de places en surréservation sans dépasser un risque raisonnable ?

On note n le nombre de personnes ayant acheté un billet pour un vol donné. On note X la variable aléatoire égale au nombre de personnes qui se présentent à l'embarquement pour ce vol. Les comportements des clients sont indépendants les uns des autres. La probabilité qu'une personne ayant acheté un billet se présentera à l'embarquement avec une probabilité de 0,9. Il y a 470 places dans l'avion et chaque billet coûte 500€. Si un client ne peut pas embarquer, il est remboursé à hauteur de 600€.

Partie A On suppose que la compagnie a vendu $n = 470$ billets.

1. Quelle est la loi suivie par X ?
2. Calculer la probabilité que l'avion ne soit pas plein.
3. Calculer $E(X)$ et interpréter le résultat.
4. Calculer la recette moyenne du vol.
5. Le programme suivant permet de calculer $\displaystyle\sum_{i=0}^{k} p(X = i)$.

```python
def coeff(n,k):
  if k>n:
    return 0
  e=1
  if 2*k>n:
    k=n-k
  for i in range(1,k+1):
    e=(e*(n-k+i))//i
  return(e)
```

```python

def binomiale(n,p,k):
  prob=p**k*(1-p)**(n-k)
  return coeff(n,k)*prob

def SommeBinom(n,p,k):
  S=0
  for i in range(k+1):
    S=binomiale(n,p,i)+S
  return S
```

Que permet de faire le programme « coeff » ?

6. Que calcule le programme « binomiale » ?
7. A l'aide du programme complet, calculer $p(X \leqslant 440)$. Donner une interprétation de cette valeur et en déduire la nécessité pour la compagnie de pratique le surbooking.
8. Déterminer le plus petit entier k tel que $p(X \leqslant k) \geqslant 0,95$ et interpréter le résultat.

Partie B

Pour optimiser ses coûts, la compagnie décide de limiter à une certaine valeur p la probabilité de ne pas pouvoir embarquer tous les passagers qui se présenteront.
On cherche donc le nombre maximum de billets que l'on peut vendre, sachant que l'on doit avoir $p(X > 470) \leqslant p$, autrement dit, $p(0 \leqslant X \leqslant 470) \geqslant 1 - p$.

1. Dans le programme Python de la première partie, ajouter un sous programme qui permet de déterminer le nombre maximum de billets que peut vendre la compagnie quand $p = 0,01$, puis $p = 0,02$ puis $p = 0,05$.
2. Comparer les recettes moyennes de la compagnie lorsqu'elle n'effectue pas de surréservation et lorsqu'elle en pratique avec un seuil de risque de 5%.

> Correction des exercices

Partie A

1. X suit $\mathcal{B}(470\,;\,0,9)$.

2. La probabilité que l'avion soit plein est $p(X=470)=\binom{470}{470}\times 0,9^{470}\times 0,1^{0}\approx 3,12\times 10^{-22}$.

   La probabilité qu'il ne soit pas plein est donc de $1-3,12\times 10^{-22}\approx 1$.
   Il est donc quasiment certain que l'avion ne sera pas plein.

3. $E(X)=np=470\times 0,9=423$. En moyenne, il y aura 423 passagers dans l'avion.

4. $423\times 500=211\ 500$.
   La recette moyenne est alors de 211 500€.

5. Il permet de calculer le coefficient binomial $\binom{n}{k}$.

6. Il permet de calculer la probabilité $p(X=k)$.

7. \>> print(SommeBinom(470,0.9,440))
   \>> 0.9977937218759524

   La probabilité qu'il y ait au maximum 440 passagers dans l'avion est très proche de 1. Il y a donc 30 places de perdues ce qui représente une perte de $30\times$ 500€ = 15 000€.

8. Toujours en utilisant le programme Ptyhon :

   \>> print(SommeBinom(470,0.9,433))
   \>> 0.9507459769224057

   \>> print(SommeBinom(470,0.9,432))
   \>> 0.9315192041683489

   Ce plus petit entier k est donc 433.
   Cela signifie que la probabilité qu'il y ait au maximum 433 passagers dans l'avion est au moins de 95%.
   En dessus de ce nombre de passages, on passe sous la barre des 95% (93% en réalité).

Partie B

1. Voici le programme complété.

```python
def coeff(n,k):
  if k>n:
    return 0
  e=1
  if 2*k>n:
    k=n-k
```

```
    for i in range(1,k+1):
        e=(e*(n-k+i))//i
    return(e)

def binomiale(n,p,k):
    prob=p**k*(1-p)**(n-k)
    return coeff(n,k)*prob

def SommeBinom(n,p,k):
    S=0
    for i in range(k+1):
        S=binomiale(n,p,i)+S
    return S

def surreservation(alpha,n,p):
    proba=0
    i=n
    while proba<alpha:
        proba=1-SommeBinom(i,p,n)
        i=i+1
    return i-2
```

>> print(surreservation(0.01,470,0.9))
>> 506

>> print(surreservation(0.02,470,0.9))
>> 507

>> print(surreservation(0.05,470,0.9))
>> 510

2. Sans surbooking, on avait trouvé 211 500€.
   Avec $\alpha = 0,05$, la nouvelle espérance est $E(X) = np = 510 \times 0,9 = 459$.
   Puis $459 \times 500 = 229\ 500$.

   Cela représente un gain de 18 000€.