Randomness and Computation: Written Problem Set 1

John Marangola

February 26, 2021

1.

(a)

$$b(n,p,k) = \binom{n}{k} p^k \left(1-p\right)^{n-k} \tag{1}$$

Under the assumption that Latouch has the ability that she claims, the sample space Ω is given by the number of times Latouch guesses correctly:

$$\Omega = \{0,1,2,3,4,5,6,7,8,9,10\}, \text{ such that } m(\omega) = b(10, 0.75, \omega) \; \forall \omega \in \Omega$$

Let E_1 be the event that Latouch wins the bet such that:

$$E_1 = \{7,8,9,10\}$$

$$\begin{aligned} P(E_1) &= \sum_{k=7}^{10} b(10, 0.75, k) = \sum_{k=7}^{10} \binom{10}{k} (0.75)^k \left(0.25\right)^{10-k} \\ &= \binom{10}{7} (0.75)^7 (0.25)^3 + \;\ldots\; + \binom{10}{10} (0.75)^{10} \\ P(E_1) &= 0.7758750916 \approx 0.776 \end{aligned}$$

(b) Now, assume that Latouch is bluffing. If Latouch has no special ability, the probability that she makes a single guess correctly is 0.5. Likewise, the mean of the binomial distribution corresponding to the probabilities of X is given by $np = (10)(0.5) = 5$. Letting E_2 be the event that Latouch loses the bet given that she is lying about her ability:

$$E_2 = \{0, 1, 2, 3, 4, 5, 6\}$$
$$P(E_2) = \sum_{i=0}^{6} b(10, 0.5, i) = 0.828125 \approx 0.828$$

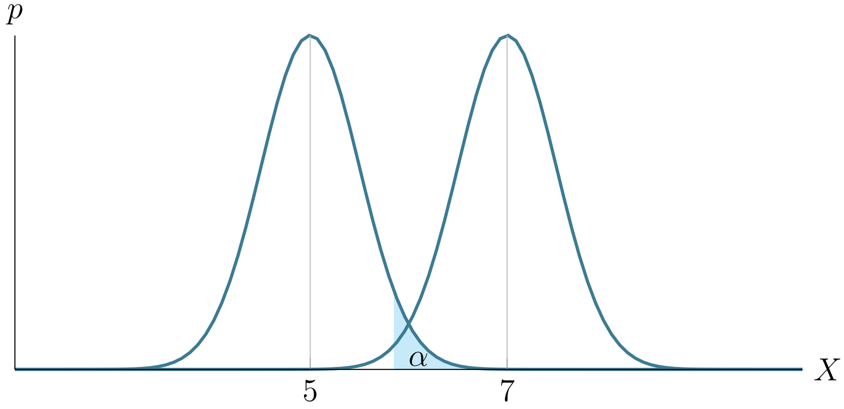
2.

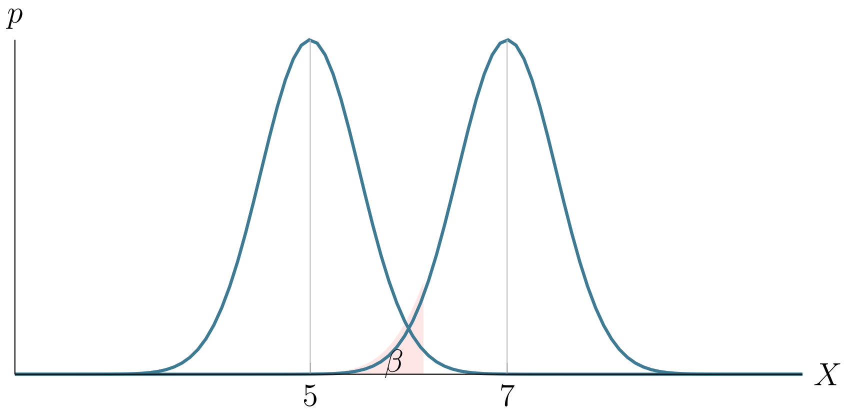
We define the null hypothesis H_0 to be that Dr. Venkman is guessing. It follows naturally that α, β are the probabilities of Type I and Type II errors, respectively. More precisely, α is the probability that H_0 is rejected when Dr. Venkman is simply guessing and β is the probability that we fail to reject the null hypothesis when Dr. Venkman does indeed possess ultrasensory powers.

From eqn (1) above,

$$\alpha(p) = \sum_{k=m}^{N} b(N, p, k),\ \beta(p) = 1 - \alpha(p)$$

Since $\overline{x} = np$, for N = 10 trials, the scenario above may be illustrated as:





To carry out a test such that the probability of making Type I, and II errors is less than 0.05, we must find a target number of guesses, m out of N that Dr. Venkman must guess correctly or incorrectly in order to ensure that α, $\beta < 0.05$. More precisely,

$$\alpha = \sum_{k=m}^{N} b(N, p_0, k) < 0.05, \beta = 1 - \alpha(p_1) < 0.05$$

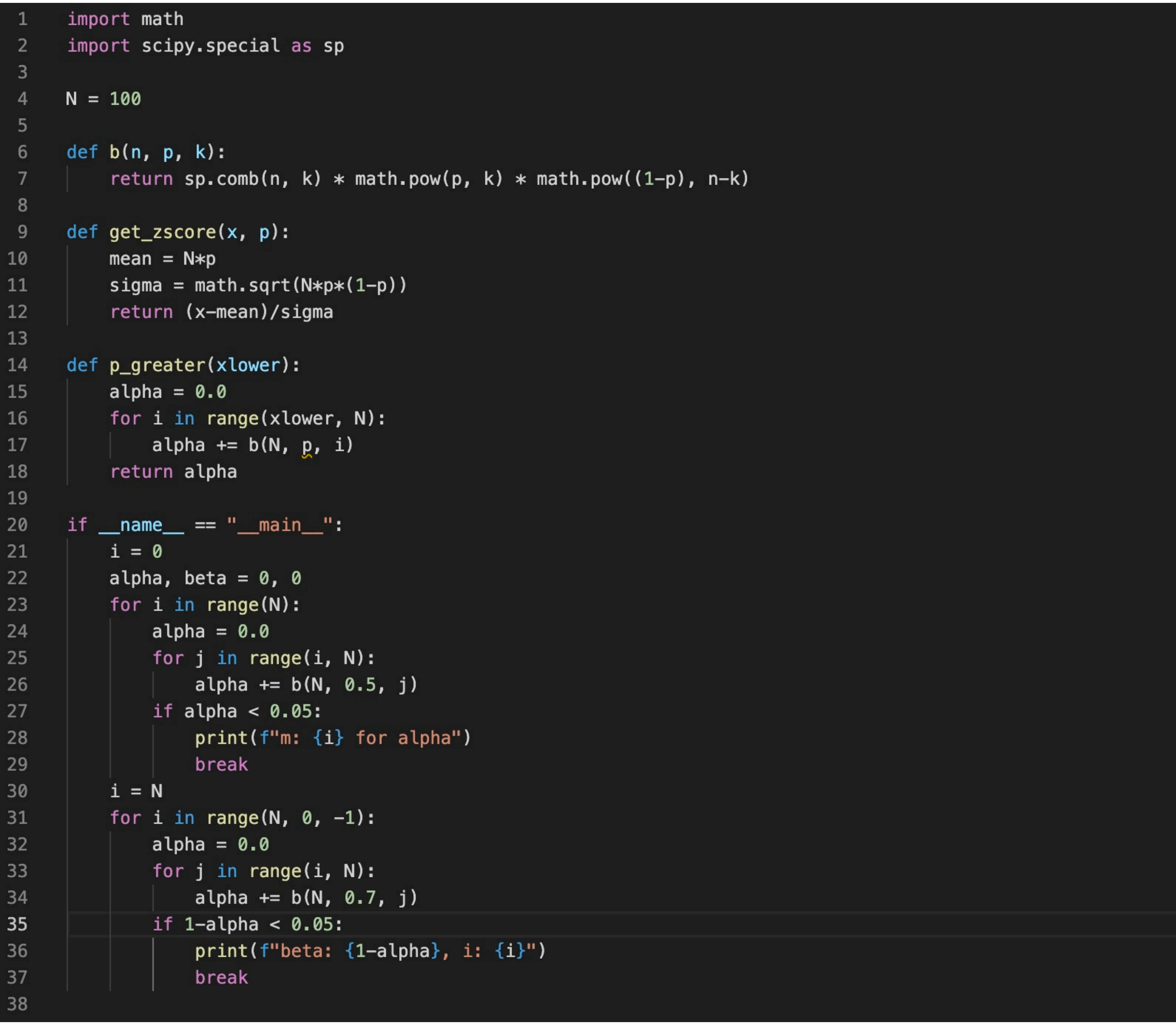
Assume that Dr. Venkman is indeed guessing as we suspect. If his true probability of guessing correctly $p_0 = 0.5$, we can find a target number of correct guesses m for some N trials such that:

$$\sum_{k=m}^{N} b(N, 0.5, k) < 0.05$$

At this number of correct guesses, the probability that we are making a type I error in assesing Dr. Venkman's abilities would be less than 0.05. Likewise, if we assume he does have this ability, we can determine values for m, N such that:

$$\beta(0.7) = 1 - \alpha(0.7) < 0.05 \rightarrow 1 - \sum_{k=m}^{N} b(N, 0.7, k) < 0.05$$

In other words, if he guessed m times correctly in N trials, the probability of making a type II error and stating that he doesn't have this ability would be less than 0.05. Using N=100 trials, we compute this value numerically. Since increasing m will make α smaller and while decreasing m will make β smaller, there is a range of m for which $\alpha, \beta < 0.05$ Numerically, it was found that for subjecting Dr. Venkman to N=100 trials m must be less than 62 for $\beta < 0.05$ and m must be greater than 59 for $\alpha < 0.05$. Thus, we may devise a test that if $59 \leq X \leq 62 \rightarrow \alpha, \beta < 0.05$ and we reject the null hypothesis that he is making fradulent claims. Otherwise, we fail to reject H_0 as we do not possess adequate certainty about the result ie. $\max(\alpha, \beta) \geq 0.05$.



```python
import math
import scipy.special as sp

N = 100

def b(n, p, k):
    return sp.comb(n, k) * math.pow(p, k) * math.pow((1-p), n-k)

def get_zscore(x, p):
    mean = N*p
    sigma = math.sqrt(N*p*(1-p))
    return (x-mean)/sigma

def p_greater(xlower):
    alpha = 0.0
    for i in range(xlower, N):
        alpha += b(N, p, i)
    return alpha

if __name__ == "__main__":
    i = 0
    alpha, beta = 0, 0
    for i in range(N):
        alpha = 0.0
        for j in range(i, N):
            alpha += b(N, 0.5, j)
        if alpha < 0.05:
            print(f"m: {i} for alpha")
            break
    i = N
    for i in range(N, 0, -1):
        alpha = 0.0
        for j in range(i, N):
            alpha += b(N, 0.7, j)
        if 1-alpha < 0.05:
            print(f"beta: {1-alpha}, i: {i}")
            break
```