12^{nd} Buyside Study Group - Quiz 2

Tian Zhang

June 9, 2019

1 Math

1. What do you know about χ^2 test?

Answer: Briefly speaking χ^2 test is a goodness-of-fit test, which is used to test if the distribution of sample data follows the required distribution. Suppose that there are k possible values for each observation, and we observe N_i with value i for $i = 1, ..., k$. Suppose that the null hypothesis says that the probability of the ith possible value is p_i. Then we compute

$$Q = \sum_{i=1}^{k} \frac{(N_i - np_i)^2}{np_i}$$

To test H_0 at level α, we would compare Q to the $1 - \alpha$ quantile of the χ^2 distribution with $k - 1$ degrees of freedom. Let $C^k_{1-\alpha}$ be the critical value at $1 - \alpha$ quantile of the χ^2 distribution with $k - 1$ degrees of freedom. We reject H_0 if $Q > C^k_{1-\alpha}$ and we do not reject H_0 if $Q < C^k_{1-\alpha}$.

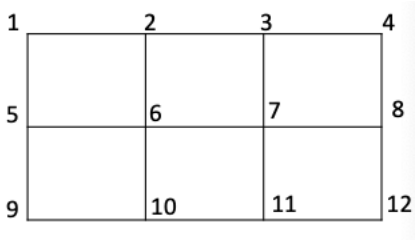
Alternatively, we can compute the p-value, which would be the smallest α at which we could reject H_0. In the case of the χ^2 goodness of fit test, the p-value equals $1 - \chi^2_{k-1}(Q)$, where $1 - \chi^2_{k-1}$ is the c.d.f of the χ^2 distribution with k-1 degrees of freedom. If p-value $> \alpha$, we do not reject H_0, otherwise, we reject H_0.

2. Given a 2 by 3 grid (which has 6 blocks and 17 edges), shortest route to visit all edges (assuming edge length is 1).

Answer: I spent a lot of time trying to find a solution based on an algorithm. For instance, Breathe-First-Search is used to find the shortest path between the source and the destination. Dijkstra's algorithm can be used for the weighted graph to cover all vertex. But none of them are fit for this question.

So my solution is only based on the human brute-force approach, but have some intuition as vertices $(2, 3, 5, 8, 10, 11)$ have the odd edges. We can treat 5 and 8 as the beginning and ending vertex respectively because they allow to have odd edges. Then we can only repeat the edges $(2 - 3)$ and $(10 - 11)$ to achieve the shortest path with length 19 as follows: $5- > 1- > 2- > 3- >$

$4- > 8- > 7- > 6- > 5- > 9- > 10- > 6- > 2- > 3- > 7- > 11- > 10- > 11- > 12- > 8$



3. X, Y are $i.i.d$ $N(0,1)$, calculate $P(X|X+Y>0)$, try not use density function of joint distribution.

Answer:

$$\begin{aligned} P(X=x|X+Y>0) &= \frac{P(X=x, X+Y>0)}{P(X+Y>0)} \\ &= 2P(X+Y>0|X=x)P(X=x) \\ &= 2P(Y>-X|X=x)P(X=x) \\ &= 2P(Y>-x)\phi(x) = 2\Phi(x)\phi(x) \end{aligned}$$

$$P(X \leq x|X+Y>0) = \int_{-\infty}^{x} 2\Phi(x)\phi(x)dx = \Phi(x)^2$$

$$P(X>x|X+Y>0) = 1-\Phi(x)^2$$

4. You have a six sided dice, you can keep rolling the dice and you get the dollars equal to the mount of the sum. However, if at some point, the sum is a square number, you must stop and will get zero dollars. (1) If at some point, your sum is 35, should you stop or keep rolling? (2) in (1), if you choose to continue and this is your strategy: you will keep rolling until you exceed 43, what is the most probable amount of dollar you win when you stop? (3) Is there a best strategy for this game, any number that you should stop ?

Answer:

1. Keep rolling, using the strategy in point 2 we can see the expected gain by keep rolling until exceed 43 is greater than 35.
2. Let $f(x)$ be the expectation of final sum at x. As we know f(44) = 44, f(45) = 45, ..., f(48) = 48 and f(49) = 0, so we can recursively solve this question.

$$f(43) = (f(44)+f(45)+f(46)+f(47)+f(48)+f(49))/6 = 38.33$$

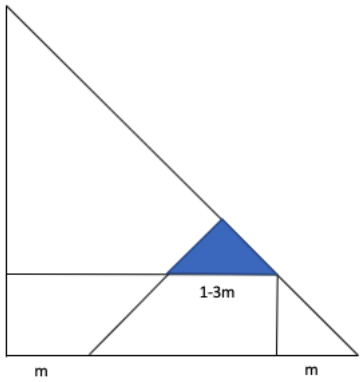
Similarly we can get $f(42) = 44.72, f(41) = 44.18, f(40) = 43.70, f(39) = 43.32, f(38) = 43.04, f(37) = 42.88, f(36) = 0, f(35) = 36.19$, thus the most probable amount is 36.19

3.

5. Given a stick, if randomly cut into 3 pieces, what's the average size of the smallest, of the middle-sized, and of the largest pieces?

Answer: Let X_1, X_2 be the independent random variables with $(0, 1)$ uniform distribution. $A = min(X_1, X_2)$, $B = max(X_1, X_2)$. Then we can define $C = min(A, B - A, 1 - B)$ is the smallest size of the 3 pieces.

$$\begin{aligned} P(C \leq m) &= 1 - P(C > m) = 1 - P(A > m, B - A > m, 1 - B > m) \\ &= 1 - \frac{1}{2}P(X_2 > m, X_1 - X_2 > m, 1 - X_1 > m | X_1 > X_2) \\ &\quad -\frac{1}{2}P(X_1 > m, X_2 - X_1 > m, 1 - X_2 > m | X_2 > X_1) \\ &= 1 - P(X_2 > m, X_1 - X_2 > m, 1 - X_1 > m | X_1 > X_2) = 1 - (1 - 3m)^2 \end{aligned}$$



The last equation comes from the blue area. Then we have pdf of C

$$f(m) = 6(1 - 3m)$$

Now we can calculate the expectation of C

$$E(C) = \int_0^{1/3} mf(m)dm = \frac{1}{9}$$

Similarly, we can calculate the expected length of the largest piece is $\frac{11}{18}$ and the expected length of the middle piece is $1 - \frac{1}{9} - \frac{11}{18} = \frac{5}{18}$

6. At a party, N people throw their hats (all hats are different) into the center of room. The hats are mixed up and each people randomly selects one. Let Y be the number of people who select their own hats. Now ask (1) what is the expectation of Y ? (2) what is the variance of Y ? Now, the picking hats game rule is extended. For each hats pick round, the people choosing their own hats quit the game, while others (those picked wrong hats) put their selected hats back in the center of room, mix them up, and then reselect. Also, suppose that this game continues until each individual has his own hat. Suppose N individuals initially join the game, let $R(N)$ be the number of rounds that are run and S(N) be the total number of selections made by the these N individuals, $(N > 1)$. (3) Find the expectation of $R(N)$. (4) Find the expectation of $S(N)$. (5) Find the expected number of false selections made by one of the N people.

Answer:

1. Let $Y = X_1 + X_2 + ... + X_N$ where X_i is the indicator of ith person whether he gets his own hat. As each person has $1/N$ probability to get his own hat, by the linearity of expectation, $E(Y) = E(\sum X_i) = \sum E(X_i) = N \times \frac{1}{N} = 1$

2. $Var(Y) = E(Y^2) - (E(Y))^2 = \sum_{i.j} E(X_i X_j) - 1 = NE(X_1^2) + N(N-1)E(X_1 X_2) - 1 = N \times \frac{1}{N} + N(N-1) \times \frac{1}{N(N-1)} - 1 = 2 - 1 = 1$

3. From (1) we could know that the expected number of people who select their own hats is 1 for $N = 1, 2, ..., N$. It indicates each round will reduce 1 man on average and thus we need N rounds.

4. For every round i, the number of selections is i so in total we have $S(N) = 1 + 2 + ... + N = \frac{N(N+1)}{2}$

5. From (3) we know the expected number of rounds is N. Let $Z = Z_1 + Z_2 + ...Z_N$ be the number of false selections for the first man and Z_i is the indicator whether he has the false selection in round i. We can easily get $E(Z_1) = \frac{N-1}{N}$ and $E(Z_2) = E(Z_2|Z_1 = 1)P(Z_1 = 1) + E(Z_2|Z_1 = 0)P(Z_1 = 0) = \frac{N-2}{N-1}\frac{N-1}{N} + 0 = \frac{N-2}{N}$. Finally we can get $E(Z) = \frac{N-1}{N} + \frac{N-1}{N}\frac{N-2}{N-1} + ... + \frac{2}{N}\frac{1}{2} = \frac{N-1}{N} + \frac{N-2}{N} + \frac{1}{N} = \frac{N-1}{2}$

7. Consider linear regression of Y on features $X1, X2$: Model1-$(Y, X1), R2 = 0.1$; Model2- $(Y, X2), R2 = 0.2$; Model3-$(Y, X1, X2)$, calculate the range of R^2 of Model3.

Answer: The lower bound will be greater than 0.2 as we can use (Y, X_1) as the base and X_2 will bring more information. For upper bound, it is bit complicated. First, to simplify this question, we need to standardize variable Y, X_1, X_2 and we have the property for standardized variable z that $\bar{z} = 1$ and $S_{zz} = \sum z^2 = \hat{\sigma_z}^2(n-1) = n - 1$.

Next we can derive β_1 and β_2 by minimizing the SSE:

$$\frac{\partial \sum(y-\beta_1 x_1-\beta_2 x_2)^2}{\partial \beta_1} = \sum x_1(y-\beta_1 x_1-\beta_2 x_2) = r_{yx1}-\beta_1 r_{x_1x_1}-\beta_2 r_{x_1x_2} = 0$$

$$\frac{\partial \sum(y-\beta_1 x_1-\beta_2 x_2)^2}{\partial \beta_2} = \sum x_2(y-\beta_1 x_1-\beta_2 x_2) = r_{yx2}-\beta_1 r_{x_1x_2}-\beta_2 r_{x_2x_2} = 0$$

We get

$$\beta_1 = \frac{r_{yx_1}-r_{yx_2}r_{x_1x_2}}{1-r_{x_1x_2}^2}$$

and

$$\beta_2 = \frac{r_{yx_2}-r_{yx_1}r_{x_1x_2}}{1-r_{x_1x_2}^2}$$

Knowing that

$$R^2 = \frac{SSR}{SST} = \frac{\sum(\hat{y}-\bar{y})^2}{\sum(y-\bar{y})^2} = \frac{\sum \hat{y}^2}{\sum y^2} = \frac{\sum \hat{y}^2}{n-1}$$

We also know that

$$R = r_{y\hat{y}} = \frac{\sum y\hat{y}}{\sqrt{\sum y^2 \sum \hat{y}^2}} = \frac{\beta_1 \sum x_1 y + \beta_2 \sum x_2 y}{\sqrt{(n-1)\sum \hat{y}^2}} = \frac{\beta_1 \sum x_1 y/(n-1) + \beta_2 \sum x_2 y/(n-1)}{\sqrt{\sum \hat{y}^2/(n-1)}} = \frac{\beta_1 r_{yx_1} + \beta_2 r_{yx_2}}{R}$$

Hence we have

$$R^2 = \beta_1 r_{yx_1} + \beta_2 r_{yx_2} = \frac{r_{yx_1}^2 + r_{yx_2}^2 - 2r_{yx_1}r_{yx_2}r_{x_1x_2}}{1-r_{x_1x_2}^2} = \frac{0.3 \pm \frac{\sqrt{2}}{5} r_{x_1x_2}}{1-r_{x_1x_2}^2}$$

Theoretically, we can reach the upper bound 1 when $r_{x_1x_2} = \frac{7\sqrt{2}}{10}$ or $-\frac{\sqrt{2}}{2}$ and lower bound is 0.2 as we mention in the beginning. We can also tell from the equation that if x_1 and x_2 are uncorrelated, i.e. $r_{x_1x_2} = 0$, we have $R^2 = 0.3$.

8. Given a function for a fair coin, write a function for a biased coin that returns heads with probability $\frac{1}{n}$ (n is a param).

Answer: First we can express the probability of win p as a binary number $0.p_1p_2p_3...p_n$. Then we toss the fair coin and denote q_i with 1 if head and 0 if tail. Next we compare q_i with p_i in the following way:

$$i^{th}\text{coin} = \begin{cases} \text{we win} & q_i < p_i \\ \text{we loss} & q_i < p_i \\ \text{continue to toss} & q_i = p_i \end{cases} \tag{1}$$

If we have an infinite number such as $\frac{1}{3}$, n will be infinite and eventually we will either win or loss. If we have a finite number, for instance, $\frac{7}{8}$ and n is finite as well. Then we will loss if we toss n^th coin and $q_n = p_n$.

9. 10 islands with 9 bridges. The bridges are either strong or weak (half half). Weak bridge falls if stepped on and the man is drifted to the 1st island, then all the bridges are miraculously fixed. To arrive the 10^{th} island, how many bridges on average the man has to cross?

Answer: Based on the Markov chain, we can list the following equations, where S_i represent the expected number of bridges need to cross in island i:

$$S_1 = \frac{1}{2}S_2 + \frac{1}{2}S_1 + 1 \tag{2}$$

$$S_2 = \frac{1}{2}S_3 + \frac{1}{2}S_1 + 1 \tag{3}$$

$$\ldots$$

$$S_9 = \frac{1}{2}S_{10} + \frac{1}{2}S_1 + 1 \tag{4}$$

After arrangement via eq (3) - eq (2).., eq (n) - eq (n-1) and then sum them up, we can get

$$S_9 - S_{10} = 2^8(S_1 - S_2)$$

We know that S_{10} is 0 as 10^{th} island is the destination and get $S_2 = S_1 - 2$ from eq (2), thus $S_9 = 2^9$. Substitute back into eq (4) finally we can get

$$S_1 = 2^{10} - 2 = 1022$$

2 Programming

10. Explain the following code: const int* const fun(const int* const& p) const;

Answer: p is a reference to a const pointer to a const int. And the return value of the function is a constant pointer pointing to the constant int.

11. How do you implement "delete" operation in a single-linked list?

```python
# Q11 (leetcode 19)
class ListNode:
    def __init__(self,x):
        self.val = x
        self.next = None

class Solution:
    def defNthFromEnd(self,head,n):
        x = ListNode(None)
        fast = slow = x
        x.next = head
        while fast.next:
            if n:
                n -= 1
            else:
                slow = slow.next
            fast = fast.next
        slow.next = slow.next.next
        return x.next
```

12. Implement the interface for matrix class in C++.

Answer: I am not able to do this question in C++, will revisit it later once I get more familiar with it.

13. Can the constructor of a class be virtual? How to realize a similar function as a virtual constructor?

Answer: Here is what I got online: "A virtual call is a mechanism to get work done given partial information. In particular, "virtual" allows us to call a function knowing only any interfaces and not the exact type of the object. To create an object you need complete information. In particular, you need to know the exact type of what you want to create. Consequently, a "call to a constructor" cannot be virtual." But I will revisit this question once I get more familiar with it.

14. Is it okay for a non-virtual function of the base class to call a virtual function?

Answer: I will revisit this question once I get more familiar with it.

15. Given a string, return the longest palindrome subsequence.

```python
# Q15(leetcode 5: substring)
# Manacher algorithm
def longestPalindrome(S):
    T = "#".join(S)
    T = "^#" + T + "#$"
    print(T)
    n = len(T)
    P = [0]*n
    C = R = 0
    for i in range(1, n-1):
        P[i] = max(0, min(R-i,P[2*C-i]))
        print(P)

        while T[i+P[i]+1]==T[i-P[i]-1]:
            P[i] += 1

        if i + P[i] > R:
            C = i
            R = i + P[i]
    maxLen, centerIndex = max((n,i) for i, n in enumerate(P))
    print(P)
    print(maxLen,centerIndex)

    return S[(centerIndex-maxLen)//2:(centerIndex+maxLen)//2]
print(longestPalindrome("cbcbccde"))
```

16. How to inverse a string of sentence (without reverse the word) ?

Answer: If sentence consists of words separated by space, then we can use one-line code in python:

$print(""".join(str.split(sentence)[:: -1]))$

17. Say you have an array for which the $i-th$ element is the price of a given stock on day i. Design an algorithm to find the maximum profit. You may complete at most two transactions. Note: You may not engage in multiple transactions at the same time (i.e., you must sell the stock before you buy again).

```python
# Q17 (leetcode 123)
def maxProfit(prices):
    import numpy as np
    buy1=buy2=-np.inf
    sell1=sell2=0
    for price in prices:
        sell2 = max(sell2,buy2+price)
        buy2 = max(buy2,sell1-price)
        sell1 = max(sell1,buy1+price)
        buy1 = max(buy1,-price)
    return sell2
print(maxProfit([3,3,5,0,0,3,1,4]))
```

18. The book problem: There is a group of $N(2 <= N <= 1000)$ people which are numbered 1 through N, and everyone of them has not less than $[\frac{N+1}{2}]$ friends. A man with number 1 has the book, which others want to read. Write the program which finds a way of transferring the book so that it will visit every man only once, passing from the friend to the friend, and, at last, has come back to the owner. Note: if A is a friend of B then B is a friend of A. INPUT: First line of input contains number N. Next N lines contain information about friendships. $(i+1)-th$ line of input contains a list of friends of $i-th$ man. OUTPUT: If there is no solution then your program must output 'No solution'. Else your program must output exactly $N+1$ number: this sequence should begin and should come to end by number 1, any two neighbors in it should be friends, and any two elements in it, except for the first and last, should not repeat.

Answer: This is equivalent to find a Hamilton cycle problem and we can use backtracking algorithm. Below code still has bug as I am run out of time but it can at least show the idea to solve this question.

```python
# Q18
def findHamCycleInPos(N,inputs,path,pos):
    if pos==N:
        print(path)
        return True
    for vertex in range(1,N):
        # if current man is not in the connected and current
        # man and previous man are friends
        if vertex not in path and path[vertex-1] in inputs[vertex+1]:
            path[pos]=vertex
            print(path)
            if findHamCycleInPos(N,inputs,path,pos+1) == True:
                print(path)
                return True
            path[pos] = -1
    return False

def findHamCycle(inputs):
    N = inputs[0]
    path = [-1]*N
    path[0]=1
    if findHamCycleInPos(N,inputs,path,1)==True:
        return True
    else:
        return False

inputs = [5,[2,4,5],[1,3,4,5],[2,4,5],[1,2,3],[1,2,3]]
print(findHamCycle(inputs))
```