Alisa Antypova – technical test

Part 1. Logic

1. First of all, let's recall that every cube has 6 faces,12 edges, and 8 vertices.
   a) Answer: 8 small cubes.
   Solution: three coloured faces have cubes that are located at vertices. So, the answer is number of vertices.
   b) Answer: 36 small cubes.
   Solution: two coloured faces have small cubes located on edges, but not on vertices of the cube. At each edge, there are 5 small cubes, but two of them are also at vertices. So there are 3 small cubes at each of 12 edges that have two coloured faces.
   In total: 12*3 = 36.
   c) Answer: 54 small cubes.
   Solution: one coloured face have cubes that are located at a face but not at any edge of the cube. At each face, there are 5 * 5 = 25 small cubes, but at each dimension, we should subtract 2 to receive cubes that are not at edges (5 - 2) * (5 - 2) = 9.
   In total: 9 * 6 = 54.
   d) Answer: 27 small cubes.
   Solution: no coloured faces have cubes that are not at any face of the cube. They forme smaller cube with each dimension smaller by two.
   In total: 3 * 3 * 3 = 27.

Validation: there are no small cubes with 4 or more coloured faces. So if we sum all answers from previous questions we should receive a total number of small cubes.
8 + 36 + 54 + 27 =125
Looks good!

2. Answer: 0.7(7).
Solution: To find a probability of picking a small cube with 1 or more Blue faces we should divide the number of events when a given condition is satisfied by the number of all possible events. In our case, an event is picking a small cube. There are 125 different cubes, means 125 possible events. 8 + 36 + 54 is a number of cubes with 1 or more coloured faces and also a number of events, that satisfied given condition.
In total: 98 / 125 = 0.7(7)
Validation: the probability of picking a small cube with 0 Blue faces is 27/125. That means that the probability of picking any cube is 98 / 125 + 27 / 125 = 1. That is right.

3. Answer: 1.2
Solution: An average number of Blue faces can be found by summing all Blue faces and dividing by the total number of small cubes.
(0 * 27 + 1 * 54 + 2 * 36 + 3 * 8) / 125 = 6 / 5 = 1.2
Validation: let's count an average number of not coloured faces:
((6 - 0) * 27 + (6 - 1) * 54 + (6 - 2) * 36 + (6 - 3) * 8) / 125 = (162 + 270 + 144 + 24) / 125 = 600 / 125 = 4.8

Checking whether the sum of the average number of coloured and not coloured faces will be equal to the number of faces at a cube: 1.2 + 4.8 == 6. True.

4. Answer: max(0, 12 * (N - 2))
Solution: As far as small cubes with 2 coloured faces are located only at edges, but not on vertices we should multiply the number of edges, that is constant for every cube and equals 12 by a number of 2 faces coloured small cubes at one edge. The last number depends on the cube size. In fact, it equals a number of cubes at a face (which is exactly N) minus two since there are always two cubes at a face that located at vertices and because of that have 3 coloured faces.
The formula should look like: 12 * (N-2).
However, for N = 1 and N = 0, it gives negative numbers instead of zero. We can handle it by choosing a maximum between zero and the result from the above formula.
So, the final formula is: max(0, 12 * (N - 2)).

5. Answer: the formula is correct for rational (we can include zero) N = {0, 1, 2, ...}.
Speaking about whole numbers, the formula will be correct if we agree that the number of small cubes at the cube with a negative N is zero. If we prefer to define the number of small cubes at the cube with a negative N as a number of small cubes at the cube with -N cubes at each dimension we can modify the formula this way: max(0, 12 * (abs(N) - 2)).
For numbers that are not whole, I would say that a solution, as well as a task, is incorrect.

Part 2. SQL

1. SELECT COUNT(id), country_code
FROM user
GROUP BY country_code

2.SELECT ((SELECT COUNT(*) FROM user WHERE id IN
 (SELECT user_id FROM payment
 WHERE user.id = payment.user_id
   AND payment.created_at > user.date_joined
   AND payment.created_at < user.date_joined + INTERVAL 3 DAY))
 * 100 / (SELECT COUNT(*) FROM user));

3.SELECT ((SELECT COUNT(*) FROM user WHERE id IN
 (SELECT user_id FROM payment
   WHERE user.id = payment.user_id
   AND payment.created_at > user.date_joined
   AND payment.created_at < user.date_joined + INTERVAL 3 DAY
   AND (SELECT COUNT(*) AS les FROM lesson WHERE user.id = lesson.user_id
  AND lesson.created_at > user.date_joined
      AND lesson.created_at < user.date_joined + INTERVAL 7 DAY
      HAVING les > 1))) * 100 / (SELECT COUNT(*) FROM user));

Part 3. Statistics

1. Since “a” seems to show the highest point of distribution, it most likely to denote the mode - value that occurs most frequently. It seems like “b” divides the distribution into two with same area, which means it is probably denoting a median. Then the obvious option for “c” is an average.

However, not to be confused by eyes, it should be noticed that one-mode distribution is shifted compared to normal: it has more values at the left and long tail at the right. That leads to median and average be shifted to the right. Moreover, since the median is robust(stable to outliers, in our case long tail of the distribution) it will be closer to the mode. So previous guesses seem reasonable.

2. Let’s drew n = 1 sample from the distribution (we can skip a step with mean measuring) and repeat it many times. The distribution of our samples will be really close to the initial distribution.

Now let’s drew $n \to \infty$ samples from the distribution and find their mean, repeat it many times. If n is really large, then every time we will average values, that have distribution really close to the initial one we will get almost the same mean - a mean of the initial distribution! So the result will be a distribution with one really high peak with a density close to 1.

For some middle value of n, there are still high chances to receive a mean of samples close to the mean of the original distribution, however, cases when it is not as close also occur. Moreover, the farther from the mean of the original distribution, the fewer chances, and it works the same at both dimensions. If plot this distribution we will see something like normal distribution!

To sum up, the resulting distribution will be similar to the initial at small n. When n start to grow the resulting distribution will be closer to normal, with a peak becoming higher.

3. Standard deviation with increasing of n will be smaller (comparing to the initial distribution as well) because more values will concentrate around the mean of initial values. And this means that deviation will be smaller.

Validation: check empirical evidence in the file “3_Statistics.ipynb”.

Part 4. Storytelling

Hello! I want to explain “Voice assistant” to you. You must have noticed how your parents ask their phones questions and it answers them and sometimes does more. For example, turn the music on!
Well, let’s understand how a voice assistant can hear you (it doesn't have any ears, right?). When you speak your mouth creates little winds. And your mouse is so good in creating different little winds so it can produce different letters. Every letter you know is a specific wind you make with your mouse. And your ear and brain are very good at understanding those little

winds of every letter. Your brain can remember all those little winds, put them together into words, and put those words into sentences.
Now let's turn back to the voice assistant. It sure has no ears, but it has a thing called a microphone - it can detect every little wind and convert it into numbers. And numbers is the language our assistant can understand.
And as you and I were trained to understand the wind as words, the voice assistant is trained to understand numbers from the wind.
But how does it answer those words? Well, in the same way you were trained by your parents to be polite and say: Hello, when someone greets you, and say “You are welcome” when someone says ”Thank you” - Voice assistant also was trained to answer some of the questions - it remembers numbers, that often goes after another numbers. You can view him as a human that can answer a simple question, read you a book or play music if you politely ask him.

Part 5. Recommendation Engine

At the “5_Recommendation_engine.ipynb” file you can find a straightforward implementation of “users that viewed this tutor also viewed...” section.

1. Recommendations for “ff0d3fb21c00bc33f71187a2beec389e9eff5332”:

“6b0cd6a8094daf42e766ea257a2af3571831bb32”,
“7ee223009403f7450993fe5d79516f1fc841e75e”,
“340f1eaf7ad0c07f1491338ab68cbcab30c315ec”,
“bdf147e99ee57500eb2dabcbf3cfa24e1daef357”,
“0d3dc58ead1aa17dcc7d6481215d0e940f1cedad”,
“f75cd7a7339a029b9f1aef886f3ea9ddeb0a4525”,
“e9ee460fac3c729a7de68f933621a117878dad2d”,
“85ef93bda0f7fb6327bd1b5ad44da26246b4360d”,
“61bc35a6401829bd28a8da47a2f235944ba8d2df”,
“c093b1743115b3f9d368b2f7bdf54f367afccc7c”.

No, there are different results for different tutor_id. Moreover, I divided a dataset to train and test parts and ask a model to recommend other tutors for users, who watch given in test set tutors.
I found that 88% percent of tutors in the training data was recommended at least one time. However, a lot of them appeared several times, but from the other side several tutors appeared notably more often (up to 1.6% of all searches).

2. I would say that to avoid a cold start of tutors we should also create some measure of tutors similarity between each other. It can be based on languages, their country, subjects they teach, prefered level of students and student age, education and work experience and so on.
When we implement this measure we will be able to rather recommend “similar tutors” amount which new tutors can be (we still will not show tutors with empty page, but I feel like it should not be the problem).

Other idea is to measure similarity between users to define groups of people with diverse tastes. This can help us faster understand to whom tutor is interesting.

And finally if we implement models, using feature generation (neural networks, boostings) we can add relative measures in terms of user. Like, whether tutor is getting more popular and how fast.

3. First of all I should mention, that there are cases when approach “users that viewed this tutor also viewed...” will not give any results. The easiest way to handle it is to show the most popular tutors or randomly select them based on number of views, but it does not seems to be a personalization at all.
Fortunately, collaborative filtering algorithms are really diverse and we can change our section to “you may also like” or “similar tutors”. I would rather build the recommendation system based not only on current tutor that person is watching, but at previous history of users searches and similarity of tutors as well.

4. If the dataset was including the possibility of a user to view a tutor_id more than once, tutors who have more students would have much more views. Also, a person who is looking for tutor can briefly watch a lot of profiles, but then choose between small number of tutors visiting their page and checking information. In both situation it seems reasonable to recommend a tutor who has same number of unique (by users) views with other but more views in total.
But important point is that the first view and one hundred and first view should not add the same value to the tutors suitability because they have different nature, so a different contribution as well.
Moreover, we should think whether one view by person who viewed only one page equals to one view of person who checked a hundred of them.
I think that dataset that represent both number of unique views and average number of use will work better.

5. Removing rows where tutors are visiting their own tutors page:

+ We will have a consistent dataset of users views that are all have similar meaning in terms of model.
+ It makes a recommendation model built on this dataset not vulnerable to tutors who notices this pattern and begin to promote their page this way.

Leaving rows where tutors are visiting their own tutors page:

+ I suppose, that tutors which visited their page frequently care about their possibility to find students more (but maybe, it’s rather about duration and filling a profile).
+ It gives additional information about tutors and may be used for other models.

6. For sure! As I mentioned earlier, there is a wide range of recommendations algorithms that are worth to try: algorithms based on users similarity (find users similar to given and check which tutors have they viewed), singular value decomposition (based on matrix factorization), k-nearest neighbors, adapting neural networks and boosting algorithms, “Learning to Rank” algorithms.
As I described earlier to address cold start problem, similarity between different tutors can be evaluated. We can take this idea a step further and construct tutor embedding (we can use

something like https://github.com/facebookresearch/StarSpace for this purpose) this alone will allow us to address cold start, but on this step, when each tutor is represented with embedding, we can switch to the users and perform sequence modeling in a way, similar to NLP tasks as we now operate not with one-hot-encoded tutors sequence, but with meaningful embeddings much as FastText in the NLP domain (I mentioned FastText and not Word2Vec as it out-of-vocabulary words which is intiutively close to our cold start tutors).
Moreover is extremely important to define a metrics that we want to optimize.
Data that can be used for algorithm: for users/tutor - information from their profile, if available - information from their social media, devices information.
Btw, we should not only care of whether a user will open certain tutor profile, but whether user will have classes with this tutor and whether user will recommend this tutor to other.
In addition to a recommendation system we should try generating ideas of promotion a website and social medias, like challenges with bonuses and analyze their results.