Data Masked - Customized Take-home Challenges

How to build a data science take-home challenge

1) Define in advance the skills you are looking for

2) Evaluate skills objectively

3) Define what's needed to pass the challenge

4) General Guidelines

1) Define in advance the skills you are looking for

Let's assume you define that the required skills for your data scientists to make it to the onsite are (in no particular order):

- Ability to code in Python or R
- Attention to detail
- Knowledge of machine learning
- Product sense

2) Evaluate skills objectively

At least, you should be able to rank each skill on a 0-2 scale (average or below, good, great).

Ability to code in Python or R

Build an exercise that involves creating a metric. An example could be:

- Give them two tables. One with two columns: user_id and sign-up timestamp (basically when they created the account). And the other table could be the conversion table. That is, two columns: user_id and conversion timestamp (when they converted, according to whatever definition of conversion your team has)
- Ask them to find for each user the time difference between when they signed-up and the second time they converted. If a given user has never converted or just once, it should return NA for that user
- Possible outcomes:
    - Wrong code -> assign 0
    - Correct code, but clearly inefficient. Inefficient can mean, for instance, performing tasks that can be removed without affecting the output -> assign 1
    - Correct and efficient code -> assign 2

Attention to detail

In the previous example, user_id in the sign-up table should be unique and conversion timestamps should always happen after the sign-up, obviously. You could:

- Add a few wrong duplicate user_ids in the sign-up table. Add a few conversion events that happen before the sign-up date for given users. Have all these wrong entries happen on a specific date.
- Possible outcomes:
    - Doesn't find all the wrong entries -> assign 0

- Finds all the wrong entries and cleans the data before evaluating the metric from above -> assign 1
- Finds all the wrong entries, cleans the data, and notices the pattern (i.e. all wrong entries happen on the same date) -> assign 2

Knowledge of machine learning

Here you want to weed out those who build models without understanding what they are doing.

- Give them a data set with a binary label, for instance has_converted -> 1/0, and a bunch of variables related to the conversion event (user country, age, sex, time of the day, source, clickstream behavior). Add also a variable that chronologically happens after the conversion event, for instance feedback left by the user (1-5 stars and NA if the user never converted). Set conversion rate in the data set at around 1%.
- Ask them to:
  - Build a model to predict conversion rate. This is meant to test whether they can properly design a ML project. Emphasize that they don't have to spend too much time on optimizing the model parameters (that hardly matters in real life).
  - Report model False Positive Rate and True Positive Rate for two different prediction cut-off points: i.e. 0.5 (meaning if model output > 0.5, classify as 1, else 0) and 0.1. Meant to test understanding of how models work internally.
  - Explain in 1 sentence why they chose that model. Meant to test theoretical knowledge, i.e. which models do well in high-dimension, with outliers, etc.
- Possible outcomes:
  - Fails to complete either point 1 or 2 from above. Point 1 failure would include not removing the feedback variable from the model, not testing on independent data or cross-validating, building a model that predicts the majority class no matter what, or choosing a totally wrong model. Point 2 failure would be not knowing how to change the model cut-off point or how to evaluate Positive Rates -> assign 0
  - Correctly completes point 1 and 2, but answer to point 3 doesn't give a concrete example of why the chosen model is appropriate for that task -> assign 1
  - Everything correct -> assign 2

Product Sense

- Based on the model they just built, ask them to come up with product suggestions to increase conversion rate.
- Possible outcomes:
  - Doesn't come up with any suggestions or can't extract insights from a model -> assign 0
  - Identifies segments of users that are interesting to improve conversion rate, but doesn't explain the next steps (i.e. correctly states that young users from given countries have much lower conversion rate. But doesn't explain possible ways to fix that) -> assign 1
  - Identifies interesting user segments and comes up with ideas and test methodology to make that actionable -> assign 2

3) Define what's needed to pass the challenge

Fix in advance the score required to pass the challenge. A common approach is something like:

- No zeros and at least a couple of twos in the 4 sections -> pass
- Else -> fail

This means that in order to make it to the onsite someone will need to excel in certain skills and not being bad in any skill. As the team size increases and you want to make sure you have diversity of skills, you can modify the cut-off focusing on the specific skills you are looking for. For instance, if you notice the team needs a great coder, you can start requiring a 2 in that section.

Whatever cut-off for passing the challenge you have chosen, do not make exceptions after seeing a given candidate solution. Or this will dramatically increase false positives (by probability only, similarly to when you keep changing a metric after you run an A/B test to find a scenario where the test won).

4) General Guidelines

- Treat the data challenge just like any other data-driven test you are running in your company: develop hypotheses, find the right metric to measure them, test them, evaluate results, iterate. As a general rule, a false positive is way more expensive than a false negative
- Everything in the challenge should be there to assess a pre-defined skill. If not, you are wasting both your and your candidate time. Start from the skills you want to evaluate and then work backwards to build the challenge. Don't build a random data science challenge and then try to find signal of skills within the solution (would you run an A/B test without having defined the target metric in advance?)
- Keep collecting data, such as percentage of candidates who complete the challenge and onsite-to-offer conversion rate. Then, optimize accordingly. A healthy onsite-to-offer conversion rate is ~50%. Adjust the challenge, and the score required to pass it, based on the metrics you choose
- Do send candidates real and useful feedbacks about their solutions. Don't have to tell them the actual mistakes, but the area where they need improvement. When the voice spreads that people learn thanks to the challenge and the feedback, you will save tons of money in terms of marketing and recruiting efforts. It is the most cost-effective way to increase your candidate pipeline numbers. Do not send fake feedbacks, be honest with them
- **Do not send actual work pretending is a challenge and do not use real data**. Even if it is not actual work, do not send challenges that can be mistaken for actual work. Candidates should be able to complete the challenge in less than 4 hours.

Data Masked - Customized Take-home Challenges