Job: https://g.co/kgs/PMYNzz

1. We A/B tested two styles for a sign-up button on our company's product page. ***100*** *visitors viewed page* ***A****, out of which* ***20*** *clicked on the button; whereas,* ***70*** *visitors viewed page* ***B****, and only* ***15*** *of them clicked on the button. Can you confidently say that page* ***A*** *is a better choice, or page* ***B****? Why?*

No. The standard error for samples is defined as the root of p times (1-p) over n, where p is the probability of some event and n is the number of samples. For page A, p is 20%, and the standard error is 4.0%. For page B, the probability is 21.4% and the standard error is 4.9%. We can see that both probabilities for pages A and B are well within 1 standard error of each other. Therefore, I can't say with confidence that page A or B is a better choice. However, if we were to increase our samples, and thereby decrease both the standard error and our confidence intervals, then I might be able to conclude with confidence the result of this A/B test.

2. Can you devise a scheme to group Twitter users by looking only at their tweets? No demographic, geographic or other identifying information is available to you, just the messages they've posted, in plain text, and a timestamp for each message.

In JSON format, they look like this:

```json
{
 "user_id": 3,
 "timestamp": "2016-03-22_11-31-20",
 "tweet": "It's #dinner-time!"
}
```

Assuming you have a stream of these tweets coming in, describe the process of collecting and analyzing them, what transformations/algorithms you would apply, how you would train and test your model, and present the results.

There are some interesting pieces of information we can get from each tweet that we can use to group users. First, we have all their tweets, which can tell us how long a user has been active. The timestamps can show us their specific tweeting activities over time – when they tweet, how frequently they tweet. The hashtag provides meaning to the tweet as it's an author-created index for other Twitter users. Even combining the timestamp with the hashtag can provide insights into the inner thoughts and personal daily activities of Twitter users.

I would use this data and organize it to reflect information at the user level. Through data preprocessing, I would extrapolate metrics on frequency, user engagement, user longevity, group interests by hashtag, and group common hashtags by date or time of day to understand how the user acts during the day. I would use principal components analysis to reduce the number of dimensions and data into composite dimensions that represent archetypes of Twitter users. From this PCA-reduced data, I would either run a maximum expectation clustering algorithm for clustering or K-means if we know in advance the number of desired clusters.

To present the results, I would display the clusters visually in multidimensional space if possible with clear boundaries for the clusters, and I would display the clusters as a table that showed how each cluster corresponds to the principal components, and what types of Twitter users this implies.

3. In a classification setting, given a dataset of labeled examples and a machine learning model you're trying to fit, describe a strategy to detect and prevent overfitting.

One excellent strategy to detect and prevent overfitting is the use of dropout layers combined with cross-validated grid search using dropout rates as hyperparameters. Dropout layers are vectors of 0 or 1 that are multiplied to tensors in neural networks. For each training epoch, dropout weights are assigned 0 at the dropout rate. When multiplied by a 0-dropout weight, output from preceding nodes essentially becomes ignored by the network. Thus, we can 'shut off' nodes randomly during training.

Because we are 'shutting off' nodes randomly, and thereby allowing only random pieces of information for training, the model will be less prone to train on one specific piece or category of information and instead be forced to account for all other pieces of information as well. That is, the model is less likely to be overfitted to any one piece from a dataset

Setting dropout rates as hyperparameters would allow us to monitor how differences between testing and validation log loss change with respect to the dropout rate. On the one hand, a low dropout rate can result in overfitting; on the other hand, a high dropout rate may omit too much information and underfit the model. It's up to the engineer to assess.

*4. Your team is designing the next generation user experience for your flagship 3D modeling tool. Specifically, you have been tasked with implementing a smart context menu that learns from a modeler's usage of menu options and shows the ones that would be most beneficial. E.g. I often use **Edit** > **Surface** > **Smooth Surface**, and wish I could just right click and there would be a **Smooth Surface** option just like **Cut**, **Copy** and **Paste**. Note that not all commands make sense in all contexts, for instance I need to have a surface selected to smooth it. How would you go about designing a learning system/agent to enable this behavior?*

I would set up this problem like so: for a given selected type of object, what commands maximize the usefulness to the user? The type of object selected by the user provides a context or environment in which the agent acts. With this context defined, the agent can record information a couple different ways. The first way simply keeps a running count of commands for each given context and return the top commands. The downside is that memory would be used for this algorithm.

The second way requires much less memory – the agent simply recommends the past X unique commands used. When the user issues a new command, the command enters a first-in, last-out queue of the past X commands. If an already-used command is in the queue, then that command goes to the top and the other elements shift. The downside is that commands will seldom stay in the same place on the menu. Depending on the type of user (power vs. casual), one design may be preferable over another.

5. Give an example of a situation where regularization is necessary for learning a good model. How about one where regularization doesn't make sense?

Variance can be a good indicator of when to use regularization. When a feature on a dataset has a high variance, training will result in overfitting since more extreme values will be used in the training. Therefore, a regularization factor, lambda, is used to penalize large errors. Furthermore, high variance can be symptomatic of bias in the data (i.e. some unmeasured phenomenon) in which case it would be best to regularize, and therefore minimize, the effect of this bias. Situations that could lead to a feature with high variance include outliers and large datasets with limited features.

When a feature on a dataset has values within a contained set or when variance of the data is low suggesting a well-designed dataset with quality information, regularization is not necessary. We regularize data to eliminate error, so in cases where our feature options are limited, or all required information is included to make a classification, and therefore little to no error, we don't regularize. Example situations include simple multiple-choice datasets where only one of a few answers can be selected, and situations where the model is designed knowing the process by which data is generated

6. Your neighborhood grocery store would like to give targeted coupons to its customers, ones that are likely to be useful to them. Given that you can access the purchase history of each customer and catalog of store items, how would you design a system that suggests which coupons they should be given? Can you measure how well the system is performing?

I would build a two-tier system with different objectives: 1. retain customer business and 2. grow customer value.

Grocery store customers are cyclical and somewhat predictable on how they use the items that are most important to them. This importance, I assume, is a large part of the value proposition of the grocery store for the customer. Whenever a customer is approaching the time when he or she should purchase a staple item again, I would want to send a competitive coupon to swing their business our way once again. In short, you want to make their next trip to the grocery store more important to nudge them inside the store. Patterns among purchase history to watch would be length of history, frequency of purchase, and if possible, the price sensitivity of the customer.

The second objective, growing customer value, can be delivered with a different set of coupons. However, instead of looking at the customer, we look at other customers within his or her segment and see what products they buy or often buy. In this coupon, we conduct experiments by suggesting a product to a customer, then monitoring whether the customer uses the coupon. If the customer uses the coupon, then perhaps a second coupon is sent to see whether the customer is likely to consume the purchase again. How the customer responds can be used to update future recommendations. Data used here would focus on the consumer at large versus his or her individual spending habits. Demographic, psychographic, and aggregate data would be key.

A technical metric to use in evaluating success would be the percentage of coupons sent that were used. This would indicate that our assumptions about customer behavior would be correct. However, beyond that, the key metric behind this system is really the average cart value per grocery trip. Higher cart values mean more items are being purchased by each customer, which means the grocery is effectively monetizing the value propositions of the coupons.

7. If you were hired for your machine learning position starting today, how do you see your role evolving over the next year? What are your long-term career goals, and how does this position help you achieve them?

If I started today, I would anticipate the first month of my work mainly helping my teammates with their projects. This way, I would get to know my teammates, the IT infrastructure, the purpose and scopes of our current projects, the company culture, and department protocol. Once I've onboarded successfully, I would be handed small projects to own. My manager and teammates would provide feedback which I would incorporate into additional small projects. Within three or four months, I hope to obtain my AWS certification and to lead a regular workflow. Over the course of the next year, I want to prove myself through the quality of my work and value to my team. My professional one-year goal is to earn a mid-manager position where I can interact with cross-functional stakeholders within the company.

For me, this job provides great technical and leadership opportunities. My long-term career goal is to be involved in a leadership role at a company that specializes in providing AI-empowered tools in traditionally non-technology fields. AI, by its ability to receive complex inputs and by its ability to learn complex tasks in an efficient manner, opens new opportunities in every industry and promises the disruptive power of the Internet – everything from decision making to tedious workload balancing. Clayton Christensen defines what a product is by the job that it does. There are plenty of jobs AI can do, which means there are plenty of non-tech market opportunities for AI.