Question 1 - Describe a data project you worked on recently.

A data project I worked on recently was with the Enron data set in my Udacity nanodegree program. Its a dataset of all Enron employees along with a variety of information describing their compensation and emails. The task was to identify whether or not someone was a person of interest based on this particular data with precision and recall scores both above 0.3. Some individuals had already been identified as persons of interest like Ken Lay and Jeff Skilling however some individuals had fields missing. Instead of going through every record, I used a machine learning algorithm to identify whether or not an individual was a person of interest. Given the dataset was small there was a high likelihood that any model I fit would be overfit. To counter for this I first excluded outliers that fell outside of 3 standard deviations of a mean for a particular feature. After cleaning these outliers I made sure to stick some extra known persons of interest into the training data so that the classifier could have enough data to train on. I then ran the data through a pipeline composed of a feature selection tool to identify the most relevant features for the training and a Linear Support Vector Classifier with a balanced weighting so that the persons of interest were weighted as equally as non persons of interest. The resulting model yielded precision and recall scores of .4 and .6 respectively. The LinearSVC does a good job at classifying data in small feature spaces using linear algebra techniques to introduce higher dimensional space highlighting a margin of distance between two classes without having to perform all of the calculations to actually produce that higher dimensional feature set. While the model can be improved to bump up both precision and recall, it beat the threshold by which the project was being measured by a considerable amount.

Question 2 - You are given a ten piece box of chocolate truffles. You know based on the label that six of the pieces have an orange cream filling and four of the pieces have a coconut filling. If you were to eat four pieces in a row, what is the probability that the first two pieces you eat have an orange cream filling and the last two have a coconut filling?

Follow up question: If you were given an identical box of chocolates and again eat four pieces in a row, what is the probability that exactly two contain coconut filling?

Given a box of 10 chocolates (6: orange, 4: coconut), what is the probability that the first 2 pieces will be orange and last 2 coconut. This problem details sampling without replacement so let's draw out the probabilities at the start. 6/10 orange, 4/10 coconut. In order of sampling I will have a 6/10 probability of an orange, then a 5/9 probability of orange. Then I will have a 4/9 probability of coconut and finally a 3/9 probability of coconut. Then we multiply all these together to get the probability that we will select two orange and two coconut. [6/10 * 5/9 * 4/9 * 3/9] = .049. Now we want a particular order, namely that the first two selected will be orange and the last two coconut. We do this by identifying how many combinations of 2 orange and 2

coconut pieces there are. A quick way to do this is sketching it out. Given the order of the particular orange pieces and particular coconut pieces doesn't matter, we'll have half the number of total combinations we can come up with. The resulting combinations are:
oocc, occo, ccoo, ococ, cooc, coco. As only 1 out of 6 of these combinations satisfy the original requirement we then divide our probability by 6 resulting in .0082 or 41/5000.

For the follow up, given an identical box and selecting exactly two coconut filling we can take the probability from before, 0.049, and now change the denominator as order doesn't matter. Instead of 1 particular position out of 6 combinations, we now have 6 different possible ways we can select exactly two. The probability of selecting exactly two becomes our initial probability of 0.049.

Question 3 - Given the table `users`:

```
      Table "users"
+-------------+-----------+
| Column      | Type      |
+-------------+-----------+
| id          | integer   |
| username    | character |
| email       | character |
| city        | character |
| state       | character |
| zip         | integer   |
| active      | boolean   |
+-------------+-----------+
```

construct a query to find the top 5 states with the highest number of active users. Include the number for each state in the query result. Example result:

```
+------------+-----------------+
| state      | num_active_users |
```

```
+------------+-----------------+
| New Mexico | 502             |
| Alabama    | 495             |
| California | 300             |
| Maine      | 201             |
| Texas      | 189             |
+------------+-----------------+
```

Given the table users with the mentioned columns and types, construct a query to find the top 5 states with the highest number of active users including the number for each state in the query result. The result shows the query returning state and num_active_users. So to break this down… We will want to select state and active from users. We'll want to select active as a sum however. We also want to name it num_active_users per the results shown. Then we will order the results by active in descending order. Always followed by a semicolon when we finish. The query will be constructed like so:

```
SELECT state, SUM(active) AS num_active_users FROM users
ORDER BY active DESC;
```

Question 4 - Define a function `first_unique` that takes a string as input and returns the first non-repeated (unique) character in the input string. If there are no unique characters return `None`.Note: Your code should be in Python.

Define a function first_unique that takes a string as input and returns the first non-repeated (unique) character in the input string. If there are no unique characters return None.

Let's first build a simple, maybe non efficient algorithm to get a test case working. The first check we want to see is whether or not the string passed to the function contains a character. Let's use a manual regex type expression and construct a list of characters [a,b,c,...,x,y,z]. We can then check to see if the string contains any character in our alphabet list. If not, return None. If so, continue. We will iterate over each element of the string provided and check to see if that element is in the alphabet. If so we add it to a new string. If that new string is blank at the end of our loop, return None. Otherwise we have a new cleaned string of only characters. Let's move on to process unique characters. Taking our new string, we want to find the first non-repeated character. What I don't know at this point is if the string's characters are sorted or

if there may be two identical characters separated by several spaces. This isn't contained in the examples but I imagine this must be a test case. Let's proceed if this were the case. In our brute force example let's check to see from the start of the list if a particular character is repeated somewhere else in the list. We will enumerate our new string so that we can take a character and then check to see if it is contained somewhere else in the string. If it is, we will continue iterating through the string. If it is not, this will be the first example of a unique character, satisfying our requirement. We will set this character as our unique character and return it. Another gotcha that just came to me is that if the string is blank from the get go, we need to catch that up top and return None. Additionally we need to set a catch in the new_string loop such that if there are no unique characters, we don't run into an index error. We will here set i to only run until it is less than one minus the length of the string. If this situation is generated, we will return None as there will be no non-repeated characters in the string.

Here is the code:

```
def first_unique(string):
	alphabet = ['abcdefghijklmnopqrstuvwxyz']
	new_string = ""

	if string == "":
		return None

	for elem in string:
		if elem in alphabet:
			new_string+=elem

	if new_string == "":
		return None


	for i, elem in enumerate(new_string):
		if i == len(new_string)-1:
			return None
		if elem in new_string[i+1:]:
			continue
		else:
			unique_char = elem


	return unique_char
```

```
>first_unique('aabbcdd123')
>c

>first_unique('a')
>a

>first_unique('112233')
>None
```

Underfitting and overfitting in machine learning are quite prevalent. They both refer to the fit of the machine learning model relative to unseen data. An underfit model does not capture enough variance in the training data to generalize well to unseen data while an overfit model captures so much of the variance in the training data that it does not generalize well to unseen data. There are a number of different techniques to use to balance these but in the end it comes down to understanding the data. If you have a tiny dataset, any model you throw at it will overfit. There just isn't enough data to show a larger trend. In statistics this is a power problem. One has to have a certain amount of power to investigate statistical significance appropriately. With machine learning we replace statistical significance with predictive power. Model complexity interferes with this as well. Think of two data points on a chart. We can fit thousands of different lines to fit those two points. But a straight line fits just as well. There just isn't enough data to know what sort of pattern those two data points are representing (if any!). Underfitting always means get more data. And if there isn't more data, like with stock or classified data (time series is a culprit… we can't generate more time!), we have to assume simple models lest we run into a problem where super complex models infer wild scenarios. The opposite problem of underfitting is using too simple a model to accurately capture the variance in the data. Here we may have run a linear regression on a problem best served for natural language processing methods. We'll end up with junk predictions with little relevance to the data. Here we have to be sneaky. We can either engineer the data to make the data conform to what can be captured by a simple linear regression model. We can do this by scaling the data, selecting the most relevant features, using principal component analysis or transforming the data through scalers like a log transform or square. These types of engineering solutions allow us to take a more simple model and apply it in a way that makes sense. Alternatively we can use raw forms of the data and use more sophisticated models like gradient boosting or support vector machines. This way we can utilize the more complex models in an effort to more accurately capture trends. In all situations it takes a bit of finesse to make sure the data is being appropriately handled.

Question 6 - If you were to start your data analyst position today, what would be your goals a year from now?

JOB DESCRIPTION:

Do you have a passion for creating data-driven solutions to the world's most difficult problems? The CIA needs technically-savvy specialists to organize and interpret Big Data to inform US decision makers, drive successful operations, and shape CIA technology and resource investments. The CIA is looking for individuals from diverse educational backgrounds to fill the role of data scientist. If you have experience in data analytics, computer science, mathematics, statistics, economics, operations research, computational social science, quantitative finance, engineering or other data analysis fields, consider a career as a Data Scientist at CIA.

As a Data Scientist at CIA you will get to work with advanced hardware, software, and techniques to develop computational algorithms and statistical methods that find patterns and relationships in large volumes of data. Through CIA's global mission, the agency has access to unique data sets that can be analyzed in one computational environment. Successful applicants will have keen technical insight, creativity, initiative, and a curious mind.

Data Scientists will be expected to communicate their conclusions clearly to a lay audience and become experts through continued education, attending academic and technical conferences, and collaboration with the Intelligence Community.

Positions available range from entry level to full performance.

Qualifications:

Entry Level: Bachelor's degree and experience with applied quantitative research working with real world data, either through thesis research, internships, or work experience. Applicants should have demonstrated creativity, initiative, and leadership abilities.

Developmental: A Bachelor's or Master's degree and 2-5 years of work experience in a Data Science equivalent field or sub-field, working with data rich problems either through research or programs and experience with computer programming. Applicants should have demonstrated ability to successfully complete projects with large or incomplete data sets and be able to provide solutions.

Full Performance: A Master's degree, or equivalent work experience, and 5+ years experience in a field where you have applied technical methods to a substantive problem. Applicants should be an expert in their field and have demonstrated ability leading Interdisciplinary teams throughout the full course of a project's life-cycle.

Minimum Requirements: A Bachelor's degree with a GPA of 3.0 or better on a 4.0 scale as well as initiative, creativity, integrity, technical excellence and strong interpersonal and communication skills. A self-starter attitude, the ability to work independently and in a group, demonstrated initiative, and writing/briefing skills are also required.

The following items must be attached to your online application:

- **Your resume.**
- **A cover letter in which you specify your qualifications for one or more positions.**
- **Unofficial transcripts for all degrees.**
- **A writing sample, five (5) pages MAXIMUM, single spaced, technical or analytic paper that focuses on your current area of expertise or interests and is related to your interest in positions at CIA. You can excerpt longer papers.**

Preferred: Some advanced statistical or computational programming and familiarity with modern databases and distributed computing systems is strongly preferred. Programming expertise with the leading Big Data information management platforms and statistical software packages is a plus. Some domestic and overseas mobility is also preferred.

ALL POSITIONS REQUIRE RELOCATION TO THE WASHINGTON DC METROPOLITAN AREA.

All applicants must successfully complete a thorough medical and psychological exam, a polygraph interview and an extensive background investigation. US citizenship is required.

To be considered suitable for Agency employment, applicants must generally not have used illegal drugs within the last twelve months. The issue of illegal drug use prior to twelve months ago is carefully evaluated during the medical and security processing.

Important Notice: Friends, family, individuals, or organizations may be interested to learn that you are an applicant for or an employee of the CIA. Their interest, however, may not be benign or in your best interest. You cannot control whom they would tell. We therefore ask you to exercise discretion and good judgment in disclosing your interest in a position with the Agency. You will receive further guidance on this topic as you proceed through your CIA employment processing.

To Apply:

Save the position(s) that interest you in the job cart. You can add up to four (4) positions. Job cart selections will only be retained during this site visit, so be sure to click “Apply Now” before closing the browser window. After clicking "Apply Now" you will be taken to the application account creation page. The positions will appear in the cart once you have created an account.**DO NOT** submit multiple applications; this will only slow the review of your application and delay processing. Please read the Application Instructions carefully before you begin the online application process.

If I were to start the data science position today, my goals a year from now would be as follows:

1) Have accomplished a major project with a team
2) Written an automatic pipeline to initially scrub, train, and test common datasets being generated
3) Developed a new series of projects based on a particular area of interest within the data being provided in the agency