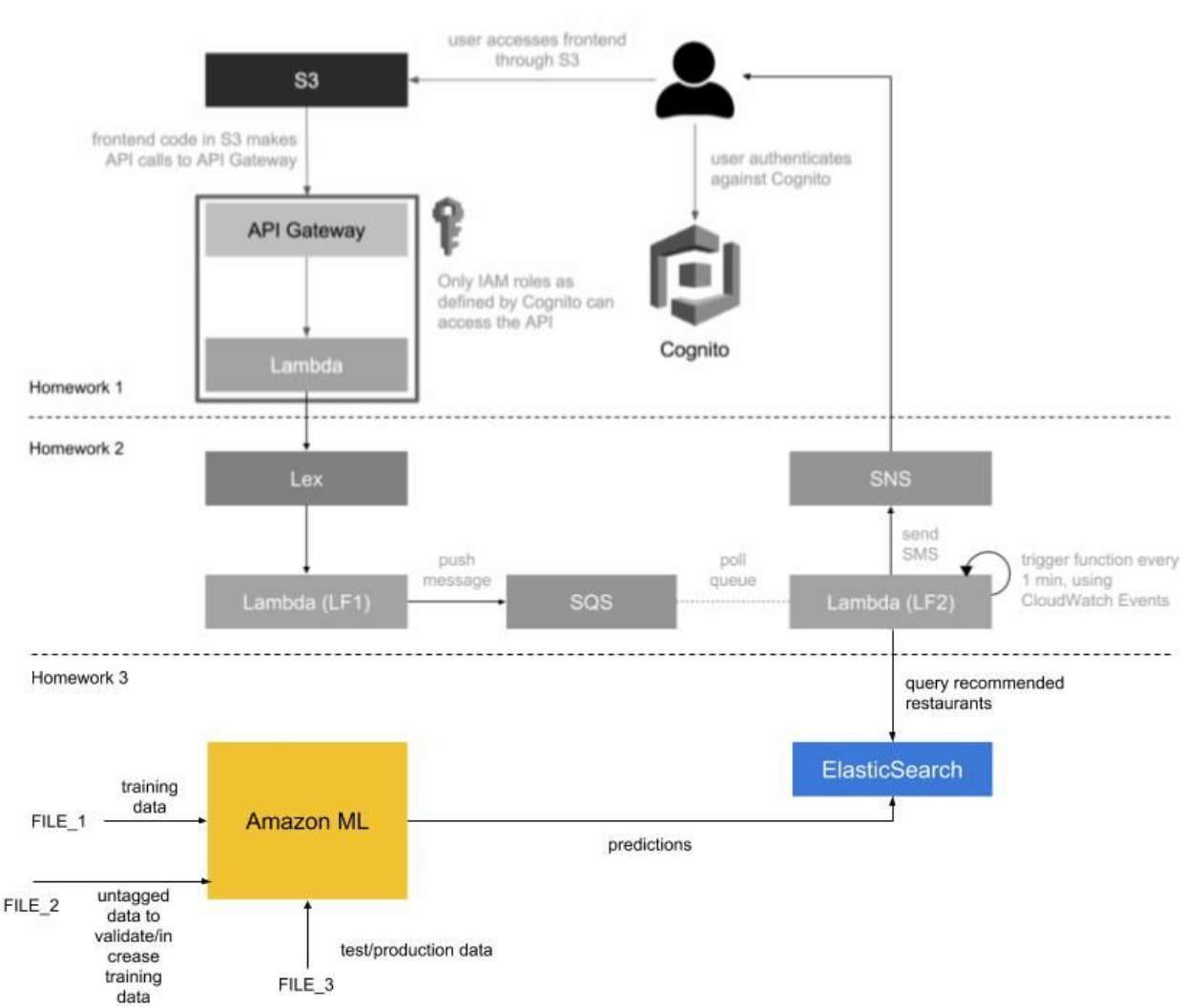
Cloud Computing and Big Data - Spring 18

Homework Assignment 3

In this assignment you will continue to iterate over your application from the first and second assignments. You have to build a restaurant recommendation engine, which you will then connect to your Lex chatbot from Assignment 2. Your Lex chatbot will filter from the restaurants which your recommendation engine will recommend.

The key learnings from this assignment will be on maintaining and searching through unstructured data (using DynamoDB), building a machine learning powered recommendation engine and connecting it to your AWS technology pipeline from hw1 and hw2.

Architecture Diagram



Outline of the steps:
For this assignment, you have to complete and/or implement the following tasks:

1. Use the Yelp API to collect 10,000 random restaurants from Manhattan.
    - Use the following tools:
        - Yelp API
            - Get restaurants
        - DynamoDB (a noSQL database)
            - Name the table “yelp-restaurants”
            - Store the restaurants you scrape, as unstructured data (one thing you will notice is that some restaurants might have more or less fields than others, which makes DynamoDB ideal for storing this data)
            - With each item you store, make sure to attach a key to the object named “insertedAtTimestamp” with the value of the time and date of when you inserted the particular record
        - Note: you can perform this scraping from your computer or from your AWS account -- your pick.

2. Pick 100 restaurants that you like from the 10,000 you scraped at Step 1.
    - For each restaurant that you find, log the following information (ex. store it in a spreadsheet/CSV file)
        - RestaurantId
        - Cuisine
            - Ex. “Japanese”
        - Rating
            - Ex. 4.3
        - NumberOfReviews
            - Ex. 537
        - Neighborhood
            - Ex. “Soho”
        - Recommended (this is binary value: 0/1. Set it to 1 here for restaurants you like and would like to recommend)
            - 1 (this value is important, it signifies that you would recommend this restaurant)

3. Pick 100 restaurants that you do NOT like from the 10,000 you scraped at Step 1.
    - For each restaurant that you find, log the following information (ex. store it in a spreadsheet)
        - RestaurantId
        - Cuisine
            - Ex. “Japanese”
        - Rating
            - Ex. 4.3
        - NumberOfReviews
            - Ex. 537
        - Neighborhood
            - Ex. “Soho”
        - Recommended (this is binary value: 0/1. Set it to 0 here for restaurants you don’t like and wouldn’t recommend)
            - 0 (this value is important, it signifies that you would NOT recommend this restaurant)

4. Compile the data you put together in steps 2 and 3 into a CSV file (FILE_1).
    - Your file should have a format similar to the following:

      RestaurantId,Cuisine,Rating,NumberOfReviews,Neighborhood,Recommended
      “ABC123”,”Italian”,4.3,534,”Soho”,1
      “DEF456”,”French”,3.75,21,”Hell’s Kitchen”,0
      …

5. Use Amazon ML to build a restaurant prediction engine
    - Upload the CSV (FILE_1) from step 4 above, as the ground truth of the model. These 200 (100 with recommended column value as 1 and 100 with recommended column value as 0) restaurants will form the initial set of training data for your model.
    - Download and convert the DynamoDB restaurant data (10,000 restaurants’ information - the 200 restaurants that form your training data) into a file (FILE_2) with the same CSV format as the file (FILE_1) from step 4. Make sure to filter out those 200 restaurants as your test/validation set should be different from your training data set.
    - Run the model against FILE_2 and see what restaurants the model predicted for you. You can conceptualise your prediction engine/ML model to be such that it received restaurant information

(RestaurantId,Cuisine,Rating,NumberOfReviews,Neighborhood) as input features and outputs/predicts ‘recommended’ column’s value.

- Add more training data to FILE_1 as you see fit until you’re satisfied with the predictions. You can use restaurants from FILE_2, set their ‘recommended’ value (0/1) as you did in steps 2 and 3, to augment your training data.

6. Once you are satisfied with the results of your ML prediction engine, request the larger Yelp data set (FILE_3) from the TAs (make a private piazza post to Instructors) and run the model against this data set. You can consider this data to be the one in production on which your model is used in real life scenarios.
    - We will also include a fourth file (FILE_4), that has the same restaurants as in FILE_3, but with the complete set of metadata.
    - Make sure to upload FILE_4 contents to DynamoDB.

7. Create an ElasticSearch cluster using the AWS ElasticSearch Service.
    - Create an ElasticSearch index called “predictions”
    - Create an ElasticSearch type under the index “predictions” called “Prediction”
    - Store the prediction results from step 6 in ElasticSearch under the “predictions” index, where each prediction has a “Prediction” data type. This data type will be of composite type stored as JSON in ElasticSearch. https://www.elastic.co/guide/en/elasticsearch/guide/current/mapping.html

8. Update your Lex chatbot (LF2 restaurant suggestion function) from assignment 2, to query ElasticSearch for recommended restaurants.
    - Use the DynamoDB table “yelp-restaurants” (which you created from FILE_4) to fetch more information about restaurants (such as restaurant name, address, etc.), since the restaurant predictions from your model will have only a small subset of fields from each restaurant.

In summary, you will be using FILE_1 to train your Amazon ML prediction engine, run it on FILE_2 to file-tune its performance and possibly increase the training data. Finally you will run your model on FILE_3 (CSV file), generate restaurant recommendations (the ones for which your model outputted 1 as the value of ‘recommended’ column) and store these recommendations into ElasticSearch. Now based on the conversation with customer, your LEX chatbot will pick 4-5 parameters based on which you will have to

search through the restaurant data in ElasticSearch and make final recommendations suiting customer's specific needs. At this point, you would also need to query DynamoDB table made from FILE_4 (has same restaurants as in FILE_3 but more information/metadata) to find more information about the restaurants you want to recommend to your customers.

Example Interaction

User: Hello
Bot: Hi there, how can I help?
User: I need some restaurant suggestions.
Bot: Great. I can help you with that. What city or city area are you looking to dine in?
User: Manhattan
Bot: Got it, Manhattan. What cuisine would you like to try?
User: Japanese
Bot: Ok, how many people are in your party?
User: Two
Bot: A few more to go. What date?
User: Today
Bot: What time?
User: 7 pm, please
Bot: Great. Lastly, I need your email address so I can send you my findings.
User: john.doe@columbia.edu
Bot: You're all set. Expect my recommendations shortly! Have a good day.
User: Thank you!
Bot: You're welcome.

(a few minutes later)

User gets the following email:

"Hello! Here are my Japanese restaurant suggestions for 2 people, for today at 7 pm: 1. Sushi Nakazawa, located at 23 Commerce St, 2. Jin Ramen, located at 3183 Broadway, 3. Nikko, located at 1280 Amsterdam Ave. Enjoy your meal!"