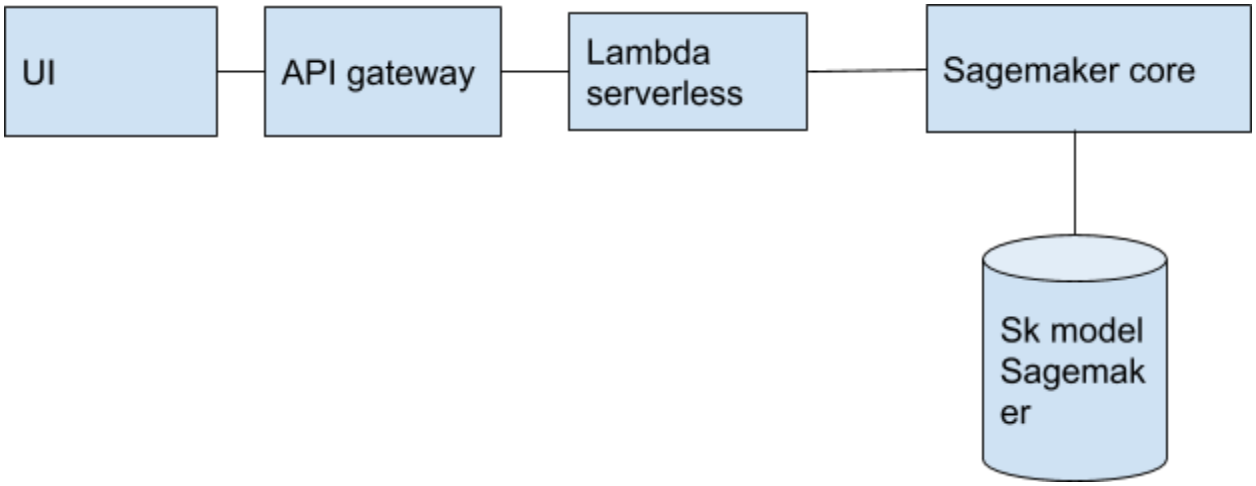

Design

UI:

For the UI, please refer to the index.html. As explained in the screencast, I wasn't able to complete this fully. But, I have a partial set of code. I am facing some issues because of ajax. I will try to fix it sooner.

API gateway:

Successfully, created an API gateway with the post request with CORS enabled. As mentioned in the video, I could get the response. In the design, API gateway mediates between UI and lambda. It redirects the request to lambda to get the response

Lambda:
Created a simple lambda service (lambda_function.py) that gets a response from sagemaker and sends it back to the UI. **The request type is a csv string (\n separated strings). The output is an application/json.**

Sagemaker and model_script: (Aws_sagemaker.py and model_script.py)

Model_script creates a sklearn stochastic gradient descent model with some hypertuning and provides prediction and confidence. I completely restructured from a normal sagemaker script template to facilitate my model building.

After this, sagemaker model is built and deployed as an endpoint. And, that endpoint is used in lambda to get the response.

Please note that my code wouldn't contain many comments, but please consider this document as an explanation for all my code.

Model experimentation and ML problem:

Since the problem is document classification with a set of words like features, the first step was to understand what kind of relations could be found between features and target.

Firstly, we need to devise a proper document indexing technique that gives you intuitive feature values. I started with Tfidvectorizer with a few hyperparameters and a linear svm classifier. Since after finding features, svm could locate the high dimensional features and associate it to a target value.

However, when I deployed in sagemaker real time, I faced several issues with min_df and max_df words. So, I tried a non-lienar sgd classifier and count vectorizer (not effective). It gave decent results, but it worked well at production level. Since, this application has more features, more dimensionality reduction and non-linear classifiers have to be studied.

I experimented with different models before going for the sgd classifier. In experimentation/document_classification.py, we could see my experimentation with different models. Utilized precision, recall and f measure to choose the model. **Sgd performed equally well with respect to different variables.**

As for the UI, I am facing an internal service error when I send a request to the lambda service. Even though, I uploaded a partial working UI to S3 bucket, due to its incompleteness, I couldn't share the S3 url. I started off the Ui part just before midnight so I am unable to complete.

But, I am happy to announce I got the web service working. Given a few more hours, I could get the UI working, but the deadline is over.

Please check the url: https://fku4xsr83b.execute-api.us-east-2.amazonaws.com/test/predict

I request you to consider all my contributions and efforts irrespective of an incomplete UI.

Gaps and other incomplete works

As per the document, CI/CD operations are pending. I planned to use AWS pipeline.

Testing (Pytest) is pending. I understand that testing is important in ML applications particularly for dimensions check.

Code refactoring and commenting are pending.

More safety checks (Try/Catch) have to be deployed.