Project: MLOps – Advanced NLP - Case Study

Shu-Ren Chang

Problem Statement

Let's take a look at the problem statement below. You are working as a data scientist for a healthtech company called BeHealthy. It is a late-stage startup. Be Healthy provides a web-based platform for doctors to list their services and manage patient interactions. It provides various services for patients, such as booking interactions with doctors and ordering medicines online. (Similar to 1mg, PharmEasy) Here, doctors can easily manage appointments, track past medical records and reports and give e-prescriptions.
We can assume that BeHealthy has enrolled thousands of doctors across the major cities in India. You can also assume that millions of patients are using BeHealthy's services; hence, a lot of free-text clinical notes would be present in the e-prescriptions. BeHealthy would want to convert these free-text clinical notes to structured information for analytics purposes. You have seen such a problem in your previous courses on NLP. You had created a CRF model to recognise the Named Entity in the medical data set.

Please find the notebook and dataset attached here.
For example:
Clinical Note: "The patient was a 62-year-old man with squamous cell lung cancer, which was first successfully treated by a combination of radiation therapy and chemotherapy."
Disease: Lung Cancer
Treatment: Radiation Therapy and Chemotherapy

Before solving any business problem, we must always keep in mind why a business needs to resolve the problem. In many cases, data scientists directly jump to the solution, using data sets like Kaggle competitions. But in the real world, you should be able to justify the business need, define KPIs and then design an optimal solution.
Now, let's take a look at a business goal:
Currently, if you need to extract diseases and treatments from free text, a trained person with clinical knowledge must manually look at the clinical notes and then pull this information.
A data entry team would manually look at the clinical text and extract information about diseases and their treatment data. A data validation team would validate the extracted information. This process is prone to errors, and as the data increases, the data-entry team's size would need to be scaled up. Automating this process would result in reduced man-hours. The data-entry team would not be required. The data validation team would still need to perform validation on extracted data. It would also significantly reduce manual errors.

Q1. System design: Based on the above information, describe the KPI that the business should track.

Answer:

To help BeHealthy track the effectiveness of the automated system for extracting diseases and treatments from free-text clinical data, the following Key Performance Indicators (KPIs) should be monitored:

1. Cost Savings: Are they saving money with the new system, like not having to pay people to manually pull out the information?
2. Time Efficiency: Is the new system faster than the old way of manually pulling out the information? They can find this out by timing both methods and comparing them.
3. Scalability: Can the system handle more and more data without making more mistakes or slowing down?
4. Extraction Accuracy: How good is the system at correctly finding diseases and treatments in the notes? They can look at:
    - Precision: Out of everything the system found, what percentage was right?
    - Recall: Out of everything that was actually there, what percentage did the system find correctly?
5. Validation Errors: What percentage of the information that the system found was marked as wrong by the team checking the data? The lower this number, the better the system is at finding the right information.

By keeping track of these things, BeHealthy can see how well the new system is working and figure out where it might need to be improved.

Q2. System Design: Your company has decided to build an MLOps system. What advantages would you get by opting to build an MLOps system?

Answer:

Setting up a well-established MLOps system can offer the company the following benefits:

1. Automation: MLOps can handle the entire machine learning process automatically. This reduces the time and effort required to create, check, implement, and monitor models.
2. Reproducibility: MLOps keeps track of all versions of data, code, and models. This ensures that any experiment can be duplicated exactly.
3. Regular Updates: MLOps supports continuous integration and deployment, which means models can be updated regularly. This ensures that the system is always using the most effective model.
4. Monitoring and Maintenance: MLOps comes with tools that can monitor how well models are performing and the quality of data. If there are any problems, the team will be alerted, and it's straightforward to retrain the model if necessary.

Q3. System design: You must create an ML system that has the features of a complete production stack, from experiment tracking to automated model deployment and monitoring. For the given problem, create an ML system design (diagram). The MLOps tools that you want to use are up to your judgement. You can use open-source tools, managed service tools or a hybrid. You can use draw.io or any other convenient tool to create the architecture. You can refer to the module on "Designing Machine Learning Systems" for your understanding how to create an MLOps Architecture. You can upload the design/diagram as an image in the PDF.

Answer:

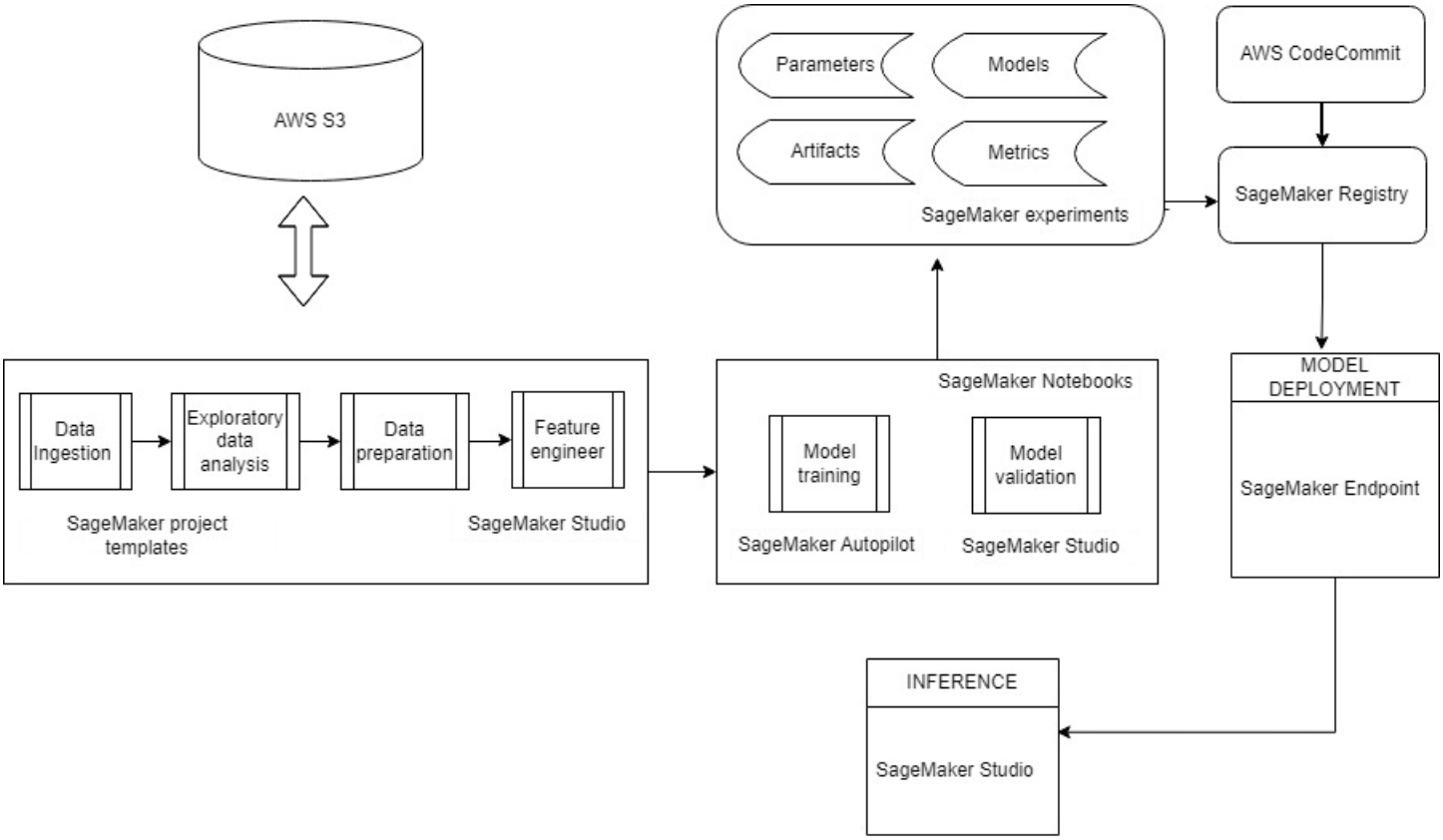

Figure 1: Development Environment in MLOps

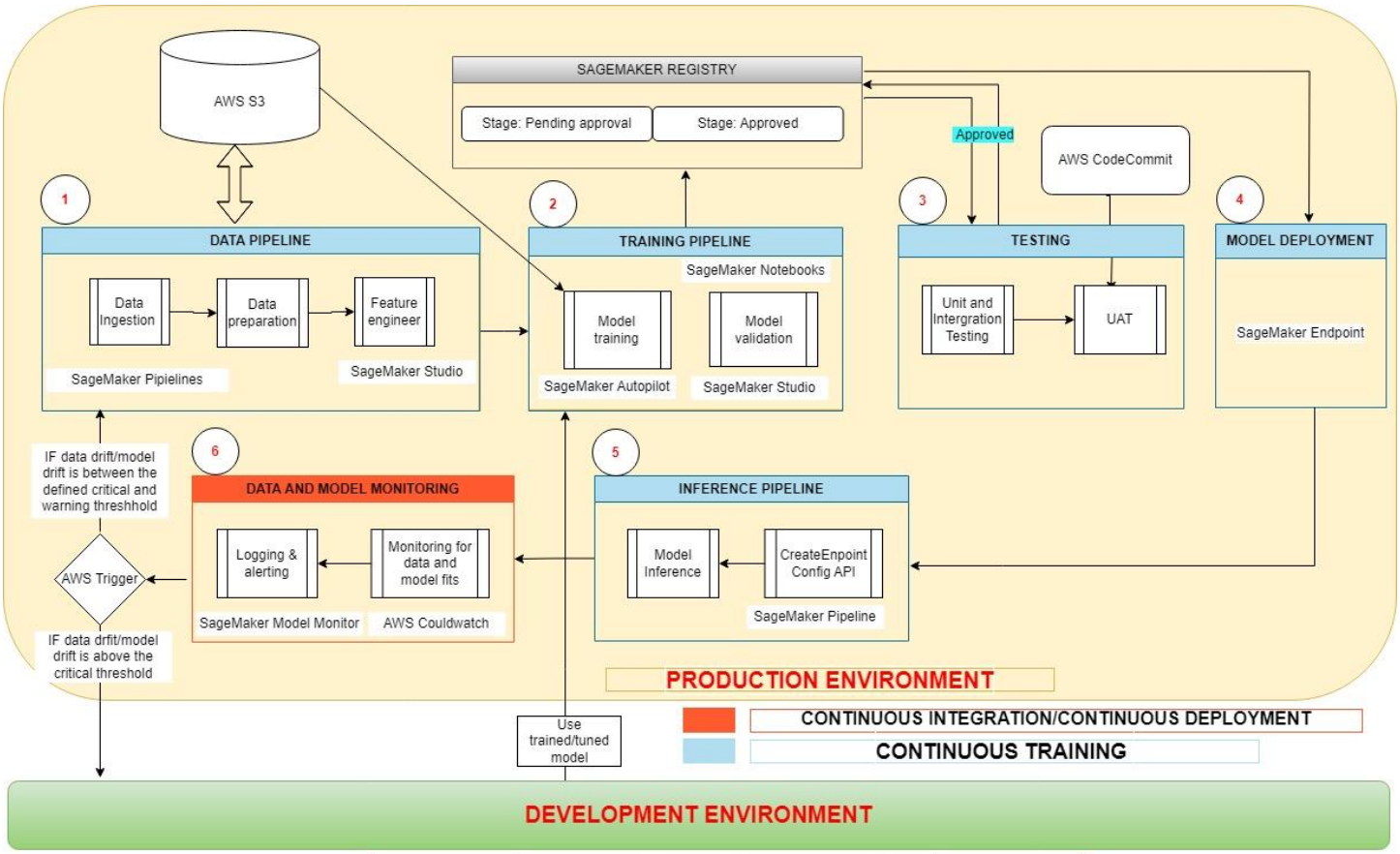

Figure 2: Production Environment in MLOps

Q4. System design: After creating the architecture, please specify your reasons for choosing the specific tools you chose for the use case.

Answer:

For BeHealthy, a late-stage startup, the AWS SageMaker, he managed (cloud) services, will be chosen for Machine Learning Operations (MLOps) for several reasons:

The benefits for using AWS SageMaker are as follow:

1. Easy Deployment: It's easier to deploy.
2. Low Learning Curve: It's easier to learn and use.
3. Low Maintenance: It requires less effort to maintain the infrastructure.
4. Pre-configured MLOps Template: It comes with a ready-to-use MLOps template.
5. Integrated MLOps: It provides an end-to-end MLOps solution.
6. Quick Setup: It takes less time to set up.

At this stage of the startup, the vendor can provide quickly solutions to get started quickly, allowing us to focus our limited resources on our core products. However, as the use cases grow, vendor costs might become too high, and it might be more cost-effective to invest in our own solution.

The data doesn't seem to contain any sensitive user information, so it can be safely sent to managed services on the cloud. Also, cloud services offer the scalability required as the business grows.

The benefits for using SageMaker, with its unique characteristics are:

1. With SageMaker, data scientists/developers can easily build and train machine learning models, and then quickly deploy them into a production-ready hosted environment.
2. Amazon SageMaker is a fully managed machine learning service, which provides an integrated Jupyter authoring notebook instance along with easy access to data sources for exploratory analysis, so there is no need to manage servers.
3. Training and hosting will be billed by minutes of usage, with no minimum fees and no upfront commitments.
4. The services also provide common machine learning algorithms that are optimized to run efficiently against extremely large data in a distributed environment.

The benefits and reasons why the following tools within AWS SageMaker were chosen.

1. Amazon SageMaker Notebooks are with enhanced notebook experience with quick-start and easy collaboration.
2. Amazon SageMaker Experiments has a good experiment management system to organize, track, and compare thousands of experiments.
3. Amazon SageMaker Studio provides a fully integrated development environment (IDE) for machine learning.
4. Amazon SageMaker Debugger can execute automatic debugging analysis and alerting.
5. Amazon SageMaker Monitor provides model monitoring to detect deviation in quality and take corrective actions.
6. Amazon SageMaker Autopilot has the functions for automatic generation of machine learning models with full visibility and control.

Q5. Workflow of the solution: You must specify the steps to be taken to build such a system end to end. The steps should mention the tools used in each component and how they are connected with one another to solve the problem. Broadly, the workflow should include the following. Be more comprehensive of each step that is involved here.

- **Data and model experimentation**
- **Automation of data pipeline**
- **Automation of the training pipeline**
- **Automation of inference pipeline**
- **Continuous monitoring pipeline**

The workflow should ALSO explain the actions to be taken under the following conditions.
After you deployed the model, you noticed that there was a sudden increase in the drift due to a shift in data.

1. **What component/pipeline will be triggered if there is any drift detected? What if the drift detected is beyond an acceptable threshold?**
2. **What component/pipeline will be triggered if you have additional annotated data?**

3. **How will you ensure the new data you are getting is in the correct format that the inference pipeline takes?**

Answer:

With the AWS could services, the following solutions could be done with the specific workflow stages:

1. Data and Model Experimentation: Use Amazon SageMaker Notebooks for data exploration and model experimentation. You can also use Amazon SageMaker Experiments to organize, track, and compare your machine learning experiments.
2. Automation of Data Pipeline: Use Amazon S3 for data storage and AWS Glue for ETL operations. This will automate the process of data extraction, transformation, and loading.
3. Automation of the Training Pipeline: Use Amazon SageMaker Autopilot to automatically train and tune your model. It will explore different solutions to find the best model.
4. Automation of Inference Pipeline: Deploy the trained model using Amazon SageMaker. It provides a managed service for deploying machine learning models to production.
5. Continuous Monitoring Pipeline: Use Amazon SageMaker Model Monitor to detect and remediate concept drift. It continuously monitors the model's quality and provides alerts when there are deviations.

The following are the breakdown for major steps in the process of building a machine learning (ML) system. The process involves several steps and uses various tools to ensure the system works efficiently.

Development Environment: This is where we experiment with different models to find the one that best solves our problem using the available data.

Production Environment: Once we've identified the best model in the development environment, we implement it in the production environment.

The development of the model in the development environment involves several steps:

1. Exploratory Data Analysis: We examine the data to understand its patterns and gain insights. We use visualization tools like Matplotlib and Seaborn to help us see these patterns.
2. Data Preparation and Feature Engineering: After understanding the data, we clean it, preprocess it, and engineer features from it. This step is crucial as the features we extract will be used to build the model.
3. Model Training and Tuning: We start building the model using the features we've extracted. We experiment with multiple models or algorithms to find the best one. Tools like Sagemaker AutoPilot and Sagemaker Experiments and Trials are used for rapid prototyping and experiment tracking.
4. Model Validation: We evaluate the performance of all the models we've experimented with and choose the best one. If none meet our standards, we go back and experiment until we find a satisfactory model.
5. Model Serving/Endpoint: After finalizing the best-performing model, we create an endpoint for it. This endpoint acts as a simulator of the production environment.

In the production environment, we work with multiple pipelines. A pipeline is a way to break down the ML workflow into independent, reusable parts. These parts can then be put together to perform tasks like feature creation, model training, prediction serving, etc.

Automated Data and Training Pipeline: This component focuses on automating the model training process. It converts the code developed in notebooks to Python scripts and runs whenever there's a change in the live data. This ensures the continuous delivery of existing deployed models after they're re-trained on the newly transformed data stored in the feature store.

Testing: We test the different methods used in data preparation, feature extraction, and model validation to ensure all components are working as expected. If the model passes all these tests, it can be moved to production for making predictions.

Inference Pipeline: Once the model/code passes all the tests, we deploy the model for serving predictions.

Data and Model Monitoring: Continuous monitoring of the deployed model is essential to track its performance and ensure it remains effective. If there are any changes in the live data, the monitoring system signals what action needs to be taken: model experimentation or model retraining. Tools like Sagemaker Model Monitor and AWS Cloudwatch are used for this purpose.

Answers for the certain conditions

1. **What component/pipeline will be triggered if there is any drift detected? What if the drift detected is beyond an acceptable threshold?**

   Answers to Question 1: If drift is detected, the data pipeline and training pipeline would be triggered to retrain the model with the new data. If the drift is beyond an acceptable threshold, a human review of the changes might be necessary.

   Situation status: Drift Detection: If there is any drift detected, the Amazon SageMaker Model Monitor will be triggered. If the drift is beyond an acceptable threshold, you might need to retrain your model with new data.

2. **What component/pipeline will be triggered if you have additional annotated data?**

   Answers to Question 2: If additional annotated data is available, the data pipeline and training pipeline would be triggered to incorporate the new data into the model.

   Situation status: Additional Annotated Data: If you have additional annotated data, you can trigger the training pipeline to retrain the model and improve its performance.

3. **How will you ensure the new data you are getting is in the correct format that the inference pipeline takes?**

   Answers to Question 3: Implement data validation checks in the data pipeline to ensure the new data matches the expected format. This could involve checking data types, value ranges, missing values, etc.

Situation status: Data Format Validation: To ensure the new data is in the correct format, you can use AWS Glue or AWS Lambda to validate the data schema before it enters the inference pipeline.