Challenge 1: Let’s order some Pizzas

Qué se aprende: Usar el servicio de Amazon Lex para crear y configurar interfaces conversacionales para aplicaciones. Crear y configura chatbots sofisticados, con lenguaje natural, en aplicaciones nuevas y existentes.

Escenario: Francesca’s Pizza es una pequeña pizzería familiar que solo sirve 4 tipos de pizza. Los pedidos se realizan a través de una aplicación de chatbot alojada en Amazon Lex. Aunque el restaurante contaba con una base leal de clientes, durante el confinamiento por el Covid-19 el negocio sufrió grandes pérdidas. Esto llevó a que el restaurante fuera adquirido por la cadena universal de pizzerías «Pizza Hot». «Pizza Hot» planea ampliar el menú añadiendo 3 tipos más de pizza junto con 2 opciones de masa, además de realizar un cambio de marca en el nombre del local. Como su Programador Principal, deberías poder configurar el bot para cambiar el nombre de Francesca’s Pizza a Pizza Hot, agregar los nuevos tipos de pizza y permitir a los clientes seleccionar el tipo de masa.
La dirección espera que la base de clientes crezca gracias a la mayor variedad de pizzas, además de mantener a los clientes fieles que ya utilizan la aplicación.

Tarea 1 : Cambiar Francesca’s Pizza por Pizza Hot

- Abrimos la consola de Amazon Lex en la región que tenga creado el bot.
- Pinchamos en el nombre del bot.
- Se nos abre un menú a la izquierda. Desplegamos el lenguaje y pinchamos en «Intenciones».
- Hay una intención que se llama «OrderPizza». Pinchamos en ella.
- Bajamos hasta las ranuras, desplegamos la ranura inicial y modificamos el texto.

Tarea 2: Agregar nuevos tipos de pizza

- Abrimos la consola de Amazon Lex en la región que tenga creado el bot.
- Pinchamos en el nombre del bot.
- Se nos abre un menú a la izquierda. Desplegamos el lenguaje y pinchamos en «Tipos de ranura».
- Hay un tipo de ranura que se llema «PizzaTypes». Pinchamos en ella.
- Bajamos hasta los valores de tipos de ranuras y creamos los tres valores de pizzas nuevas:
  – Hawaiian Pizza
  – Meatball Pizza
  – Farm House Pizza
- Abrimos la consola de Amazon Lex en la región que tenga creado el bot.
- Pinchamos en el nombre del bot.
- Se nos abre un menú a la izquierda. Desplegamos el lenguaje y pinchamos en «Intenciones».
- Hay una intención que se llama «OrderPizza». Pinchamos en ella.
- Bajamos hasta las ranuras, desplegamos la ranura «WhichPizza, que llama al tipo de ranura PizzaTypes y, en el campo «Solicitudes», agregamos las pizzas al texto.
- Le damos al botón de «Guardar intención».

Tarea 3: Incluir nuevas opciones de masa.

- Abrimos la consola de Amazon Lex en la región que tenga creado el bot.
- Pinchamos en el nombre del bot.
- Se nos abre un menú a la izquierda. Desplegamos el lenguaje y pinchamos en «Tipos de ranura».
- Hacemos click sobre el botón «Agregar tipo de ranura» y luego en «Agregar tipo de ranura en blanco».

- El el campo «Valores de tipo de ranura» creamos los valores:
  – Thin
  – Regular
- Le damos al botón de «Guardar tipo de ranura».
- Abrimos la consola de Amazon Lex en la región que tenga creado el bot.
- Pinchamos en el nombre del bot.
- Se nos abre un menú a la izquierda. Desplegamos el lenguaje y pinchamos en «Intenciones».
- Hay una intención que se llama «OrderPizza». Pinchamos en ella.
- Bajamos hasta «Ranuras – opcional».
- Agregamos una ranura con:
  – Nombre: WhichCrust
  – Tipo de ranura: CrustTypes
  – Solicitudes: Which crust would you like to order, Thin or regular?
- Guardamos la intención.

Challenge 2: Explain model decisions

Qué se aprende: Las habilidades adquiridas pueden utilizarse para acelerar la **implementación y adopción de técnicas predictivas** dentro de los sistemas de soporte a decisiones clínicas (CDSS) en un centro de salud. Este reto tiene como objetivo mostrar cómo se pueden entrenar fácilmente modelos predictivos en Amazon SageMaker para optimizar el proceso de triaje en un centro de salud, y cómo se puede utilizar Amazon Clarify para explicar los resultados del sistema de soporte a decisiones clínicas (CDSS).

Escenario: Un gran activo para cualquier organización sanitaria que atiende casos agudos son sus notas clínicas. En el momento de la admisión en un hospital, se toman notas de ingreso. Varios estudios recientes han demostrado que es posible predecir indicadores clave como diagnósticos, procedimientos, duración de la estancia y mortalidad hospitalaria. Estas predicciones son ahora altamente viables a partir únicamente de las notas de ingreso, gracias al uso de algoritmos de procesamiento de lenguaje natural (NLP).
Los avances en modelos de NLP, como las Representaciones de Codificadores Bidireccionales a partir de Transformadores (BERT, por sus siglas en inglés), han permitido realizar predicciones altamente precisas sobre un corpus de texto, como las notas de ingreso, que anteriormente eran difíciles de aprovechar. La predicción de estos indicadores clínicos es altamente aplicable en un sistema de soporte a decisiones clínicas (CDSS). Sin embargo, para utilizar eficazmente estas nuevas predicciones, aún es necesario explicar cómo estos modelos BERT precisos están generando sus resultados.
Existen varias técnicas para explicar las predicciones de tales modelos. Una de ellas es SHAP (SHapley Additive exPlanations), una técnica agnóstica al modelo para explicar la salida de modelos de aprendizaje automático. SHAP permite explicar las predicciones sin necesidad de comprender el funcionamiento interno del modelo. Además, existen técnicas para visualizar estas explicaciones SHAP en texto, de modo que tanto los médicos como los pacientes puedan tener una visión intuitiva de cómo los algoritmos llegan a sus predicciones.
Con las nuevas incorporaciones a SageMaker Clarify, y el uso de modelos preentrenados de Hugging Face que se pueden implementar fácilmente en SageMaker, tanto el entrenamiento del modelo como su explicabilidad pueden llevarse a cabo fácilmente en AWS.
Para ilustrar un ejemplo de principio a fin, se toma como caso de uso el resultado clínico de mortalidad hospitalaria y se muestra cómo este proceso puede implementarse fácilmente en AWS utilizando un modelo BERT preentrenado de Hugging Face, cuyas predicciones se explicarán con SageMaker Clarify.
Los pasos incluyen desplegar este modelo ajustado en SageMaker. Luego, se integra este modelo en una configuración que permite la explicación en tiempo real de sus predicciones. Para lograr este nivel de explicabilidad, se utiliza SageMaker Clarify.

Tarea 1: Descargar el modelo

- Abrimos la consola de AWS y pinchamos sobre «Amazon SageMaker AI».
- En el panel de la izquierda seleccionamos «Notebook» >> «Notebook instances».
- Aparece una instancia de notebook con el nombre ExplainPatientPrioritization. La marcamos con el check circular de la izquierda y le damos a «Abrir JupyterLab».
- Se nos abre el laboratorio y en el panel de la izquierda vemos una notebook llamada «explain-patient-prioritization.ipynb». Si

hacemos doble-click sobre él, se nos despliega la información sobre ese notebook, con los códigos a ejecutar.

- Vamos ejecutando una a una las «setup cells», hasta que llegamos a la parte de descargar el modelo. Ahí hacemos pausa.
- Accedemos al buscador de Hugging Face en https://huggingface.co/models. En el buscador, escribimos: «bigbird mortality» y tomamos nota del repositorio del modelo.
- Volvemos al notebook y metemos el texto del repositorio en el código del cell: 'mnaylor/bigbird-base-mimic-mortality'. Ejecutamos esa cell.
- Luego, para validar la tarea, insertamos el texto «mnaylor/bigbird-base-mimic-mortality» (sin las comillas) en el campo correspondiente de la tarea.

Tarea 2: Desplegar el modelo de Hugging Face en SageMaker

- Ejecutamos todas las demás hasta llegar a la parte de código que tiene:
  sagemaker_client = <<INPUT MISSING CODE>>
  donde pondremos:
  sagemaker_client = boto3.client(«sagemaker»)
  Esto es porque la variable sagemaker_client debe contener una instancia del cliente de SageMaker proporcionado por boto3.
- Para validar la tarea, escribimos el texto «boto3.client» (sin las comillas) en el campo correspondiente de la tarea.

Tarea 3: Crear el punto de enlace (Endpoint)

- Seguimos ejecutando hasta que encontramos «sagemaker_client.». Ahí ponemos:
  sagemaker_client.create_endpoint_config(
  Esto es porque create_endpoint_config es la función de boto3.client('sagemaker') que te permite definir cómo se desplegará el modelo, qué instancia usará y cómo se explicarán sus predicciones (en este caso, con Clarify).Esta llamada define la configuración del endpoint, pero aún no lo lanza (eso se hace luego con create_endpoint).
- Seguimos hasta la siguiente cell donde nos encontramos que falta el código para crear el endpoint. En este caso ponemos:
  sagemaker_client.create_endpoint(
  create_endpoint lanza el endpoint en SageMaker usando la configuración (EndpointConfigName) que definimos con create_endpoint_config. Después de esto, el endpoint quedará desplegado y listo para recibir inferencias en tiempo real.
- Para validar la tarea escribimos «sagemaker_client.create_endpoint» (sin las comillas), en el campo correcto de la tarea.

Tarea 4: Probar el punto de enlace con explicaciones

- En el siguiente bloque de código que nos falta deberemos poner:
  response = sagemaker_runtime_client.invoke_endpoint(
- Para validar la tarea escribimos «sagemaker_runtime_client.invoke_endpoint» (sin las comillas), en el campo correcto de la tarea.

Tarea 5: Explicar las predicciones para una nota de ingreso no aguda

- Ejecutamos todo el código de las notas hasta que llegamos a la parte que dice: **Explain the predictions for a non-acute admission note**.
- Entonces, ejecutamos esa cell (la de non-acute admission) y el modelo nos dará el resultado. Tomamos nota de la primera oración de la respuesta, incluido el punto final. Esta es «Patient is a 25-year-old male with a chief complaint of acute chest pain.» (sin las comillas).
- Ejecutamos luego la cell de acute admission y el modelo nos dará el resultado. Tomamos nota de la primera oración de la respuesta, incluido el punto final. Esta es «Patient is a 72-year-old female with a chief complaint of severe sepsis and septic shock.» (sin las comillas).
- Para validar la tarea, metemos ambas oraciones en el campo de texto correspondiente. El texto a meter (sin las comillas), es este:

> «Patient is a 25-year-old male with a chief complaint of acute chest pain. Patient is a 72-year-old female with a chief complaint of severe sepsis and septic shock.»

Challenge 3: Automating E-commerce Product Categorization with Amazon Rekognition and AWS Lambda

Introducción:

Este reto se centra en la creación de un sistema automatizado de categorización de productos para una plataforma de comercio electrónico utilizando servicios de AWS. Los participantes desarrollarán una solución sin servidor que procese y clasifique imágenes de productos a medida que se suben.

Temas cubiertos:

- Almacenamiento en la nube y manejo de imágenes
- Computación sin servidor y arquitectura orientada a eventos
- Clasificación de imágenes mediante aprendizaje automático
- Gestión de bases de datos NoSQL

Conocimientos técnicos previos necesarios:

- Comprensión básica de servicios de AWS (S3, Lambda, Rekognition, DynamoDB)
- Familiaridad con arquitecturas serverless (sin servidor)
- Conocimientos básicos de programación (preferiblemente en Python o Node.js)
- Conocimiento de APIs RESTful y formato JSON

Categoría:

- Computación en la nube

Dificultad:

- Intermedia

Resultados de aprendizaje:

Al completar este reto, los participantes aprenderán a:

- Configurar un bucket S3 para almacenar imágenes de productos
- Implementar una función Lambda de AWS activada por eventos de S3
- Utilizar Amazon Rekognition para la clasificación automática de imágenes
- Almacenar y recuperar datos usando Amazon DynamoDB
- Crear una solución serverless de extremo a extremo para el procesamiento y la categorización de imágenes

Componentes involucrados:

- **Amazon S3**: Almacenar todas las imágenes de productos
- **Amazon Rekognition**: Clasificar imágenes en categorías predefinidas

- **AWS Lambda**: Procesar imágenes subidas y almacenar los resultados de clasificación
- **Amazon DynamoDB**: Almacenar información de productos y sus categorías

Tarea 1: Troubleshooting AWS Lambda Triggers for Automated E-commerce Image Categorization

Antecedentes: Acme Online, una empresa de comercio electrónico, está implementando una solución automatizada para categorizar las imágenes de sus productos utilizando servicios de AWS. Este sistema tiene como objetivo reemplazar su proceso manual de categorización, el cual consume mucho tiempo y es propenso a inconsistencias.

Tarea: Tu empresa de comercio electrónico, Acme Online, ha configurado los componentes iniciales de un sistema automatizado para la categorización de imágenes de productos utilizando servicios de AWS. Los componentes clave de esta solución incluyen:

- Un bucket de Amazon S3 (con prefijo «product-categorization») para almacenar todas las imágenes de productos.
- Una función AWS Lambda llamada «product-categorization-function» diseñada para procesar y categorizar imágenes.
- Amazon Rekognition para la clasificación de imágenes.
- Amazon DynamoDB para almacenar los resultados de la categorización.

Sin embargo, el sistema no está funcionando como se espera. Los usuarios están subiendo archivos al bucket de Amazon S3, pero la función Lambda no se está activando automáticamente. Como ingeniero, debes asegurarte de que cuando se suban archivos a S3, la función Lambda se active automáticamente para procesar estas imágenes.

Servicios a utilizar

- Amazon S3
- AWS Lambda

Inventario

- La función Lambda llamada product-categorization-function ya está creada.
- Un bucket S3 con el prefijo product-categorization está configurado para recibir imágenes de productos.

Flujo de trabajo:

1. El usuario sube una imagen al bucket S3
2. El bucket S3 debería activar la función Lambda (pero actualmente no ocurre)
3. La función Lambda debería procesar la imagen
4. Amazon Rekognition categoriza el artículo
5. La tabla DynamoDB se completa con un nuevo ítem junto con los detalles de la categorización

Nota: Tienes permisos para modificar la configuración del bucket de S3 y las configuraciones de la función Lambda, pero no puedes modificar el código de Lambda ni las políticas IAM.

Primeros pasos: Revisa la configuración del bucket S3 (con prefijo product-categorization) y la configuración de la función Lambda. Identifica por qué la función Lambda no se activa cuando se suben nuevos archivos al bucket. Implementa una solución para garantizar que la función Lambda se active automáticamente al subir archivos a S3. Prueba la solución actualizada subiendo nuevas imágenes de productos al bucket S3 y verifica que la función Lambda se active.

Validación de la tarea: La tarea se completará automáticamente una vez que encuentres una solución. Puedes comprobar tu progreso haciendo clic en el botón «Check my progress» en la pantalla de detalles del reto.

Tarea 2: x

.

Tarea 3: x

.