Primera Pregunta

El primer paso es definir claramente el problema y recolectar los datos necesarios. El objetivo es clasificar los correos electrónicos en cuatro categorías: Compras cementos, Compras energía, Compras concretos y correos generales o de otra índole. Se utilizarán los correos históricos proporcionados, que suman aproximadamente 22,176 correos, como datos de entrenamiento y prueba para el modelo de clasificación.

Recolectados los datos, el siguiente paso es el preprocesamiento. Realizacion de ETL esto incluye la limpieza del texto para eliminar información irrelevante como firmas, headers y footers, así como la normalización del texto, que implica convertir todo a minúsculas y eliminar puntuación y caracteres especiales. Después, se procede a la tokenización. Se deben eliminar las stop words, es decir, palabras comunes que no aportan mucho significado, y aplicar lematización o stemming para reducir las palabras a sus formas base.

Para representar los textos de manera efectiva, se utilizarán técnicas de extracción de características. Una opción es el TF-IDF. Alternativamente, se pueden usar representaciones semánticas más avanzadas, como embeddings preentrenados de modelos como BERT o GPT, que capturan mejor el contexto y significado de las palabras.

A continuación se deben dividir los datos en conjuntos de entrenamiento y prueba, por ejemplo, usando un 80% de los datos para entrenamiento y un 20% para prueba. Con los datos preparados, se seleccionan y entrenan los modelos de clasificación. Inicialmente, se pueden probar modelos basados en reglas simples para una solución rápida. Para una clasificación más robusta, se pueden utilizar modelos de machine learning como Naive Bayes, Support Vector Machines (SVM), Random Forests o Gradient Boosting. Modelos de deep learning como redes neuronales recurrentes (RNNs) o Transformers como BERT también pueden ser útiles para capturar secuencias y contextos en el texto.

Una vez que los modelos están entrenados, se deben evaluar utilizando métricas como la precisión (accuracy), así como precisión, recall y F1-score para cada categoría. La matriz de confusión también ayudará a entender los errores de clasificación y ajustar el modelo en consecuencia.

Para la implementación, se desarrollará una API de clasificación que permitirá a los analistas enviar correos electrónicos y obtener la clasificación automática. Esta API debe integrarse con el sistema de correo electrónico de la empresa para clasificar los correos a medida que llegan. Además, es importante monitorear el rendimiento del modelo en producción y realizar ajustes según sea necesario, así como reentrenar el modelo periódicamente con nuevos datos para mantener su precisión.

Para llevar a cabo este proyecto, se necesitarán varios recursos humanos, incluyendo científicos de datos para el desarrollo de modelos, ingenieros de software para la

integración y desarrollo de la API, y analistas de negocio para definir los requisitos y evaluar los resultados. En términos de recursos tecnológicos, se requerirán servidores con capacidad de procesamiento adecuada, especialmente si se utilizan modelos complejos, así como herramientas de desarrollo como Python con bibliotecas de NLP (NLTK, SpaCy, Scikit-Learn, TensorFlow o PyTorch).

Segunda Pregunta

Después de seis meses de haber desplegado un modelo de regresión en producción, los usuarios han notado una disminución en la precisión de las predicciones. Este deterioro en el rendimiento puede ser causado por drift, que posee dos formas principales: concept drift y data drift. Concept drift ocurre cuando la relación entre las características de entrada y la variable objetivo cambia con el tiempo. Por otro lado, data drift se refiere a cambios en la distribución de las características de entrada, lo que puede resultar en una discrepancia entre los datos utilizados para entrenar el modelo y los datos actuales en producción.

Para validar si el modelo está sufriendo drift , se debe realizar un análisis exhaustivo. Se debe monitorear el rendimiento del modelo en producción comparando métricas de evaluación como el MSE (Mean Squared Error), MAE (Mean Absolute Error) o R^2 con las métricas obtenidas durante el entrenamiento. Una disminución en estas métricas puede indicar un drift. Además, es importante analizar los datos de entrada actuales en comparación con los datos de entrenamiento. Esto se puede hacer utilizando técnicas estadísticas como pruebas de Kolmogorov-Smirnov o o Shapiro-Wilk para evaluar si hay diferencias significativas en la distribución de los datos. Herramientas de visualización, como histogramas y diagramas de dispersión, también pueden ser útiles para identificar cambios en las características de entrada.

Si se confirma la presencia de este fenómeno, se deben tomar medidas para corregir el problema. Es necesario recopilar datos recientes que reflejen los cambios en la distribución de entrada o en la relación entre las características y la variable objetivo. Con estos datos actualizados, se debe reentrenar el modelo y validar su rendimiento utilizando técnicas de validación cruzada para asegurar que el modelo se ajuste correctamente a las nuevas condiciones. Además, es recomendable realizar una búsqueda de hiperparámetros para optimizar el rendimiento del modelo en el contexto actual. En algunos casos, puede ser útil explorar modelos alternativos que sean más robustos ante cambios en los datos.

Para prevenir futuros problemas se debe implementar un sistema de monitoreo continuo que evalúe automáticamente el rendimiento del modelo y detecte señales de drift en tiempo real. Esto requiere configurar alertas para notificar a los equipos responsables cuando se detecten cambios significativos en el rendimiento o en la distribución de datos. Establecer un ciclo regular de revisión y actualización del modelo garantizará que el modelo se adapte a los cambios en el entorno y en los datos, manteniendo así su precisión y utilidad a lo largo del tiempo.

Tercera Pregunta

Para esto es fundamental diseñar prompts específicos. En lugar de usar preguntas generales, se debe formular las preguntas de manera que claramente dirijan al modelo hacia la información que se necesita. Por ejemplo, si necesita saber la ubicación del usuario, se podría utilizar un promt así: "Para ofrecerte la mejor ayuda, ¿puedes decirme en qué ciudad te encuentras?". Esto facilita el guiar al modelo para que proporcione respuestas relacionadas con el objetivo de la conversación. Se debe asegúrar de que el modelo recuerde la información ya proporcionada y enfoque las preguntas siguientes en obtener la información que falta. También es útil establecer reglas claras para el chatbot. Proporciona instrucciones sobre cómo debe proceder el modelo en diferentes situaciones. Es decir, se puede programar al chatbot para que siempre pida el número de teléfono después del nombre del usuario y que redirija la conversación si el usuario cambia de tema.

Además, se puede implementar un sistema de validación para las respuestas del modelo. Se puede configurar un filtro que verifique si las respuestas del chatbot están alineadas con el objetivo de la conversación y hacer ajustes si es necesario. Esto asegura que las respuestas sean pertinentes y útiles. Si es posible, realiza un ajuste fino del modelo con ejemplos específicos que muestren cómo debe responder a las solicitudes de información. Finalmente, se diseña la interacción de manera que el chatbot refuerce las respuestas correctas y guíe al usuario de manera positiva hacia la información que se necesita.