Domanda: Suppose now that the prototype gave good results and training must be scaled to consider all the machines in all the kitchens associated to phase 1, but reduced to a list of arepa types given in a text file (whose location is provided as additional input to your program). Briefly describe how your code would change (if you think it will) and why.

Risposta:

Per come è stato impostato lo script non sarebbe necessario utilizzare un file di testo per esplicitare il tipo di arepa ma potrebbe essere gestito tramite il file di configurazione.

Nel caso si volesse necessariamente caricare il tipo di arepa da un file di testo allora servirebbe aggiungere un file path per la posizione del file all'interno delle configurazioni e, invece di utilizzare la lista di tipi di arepa presente nelle configurazioni, andare ad utilizzare il caricamento del file di testo e la conseguente conversione di quanto caricato in lista.

Domanda: Training is complete now and it is time to put the model in production! Machines maintenance actions are decided once a day at the beginning of the day. Sketch a high level architecture of the system. How would you present the results to the users? Which are the main components and how would they interact? What about CI/CD? Assume that there is one model for each arepa type that take as input the cooking metrics from the last 7 days.

Risposta:

Assumendo di avere un modelli che prende in input i dati delle cooking metrics degli ultimi 7 giorni, è necessario sicuramente che il progetto dello script implementato nella challenge venga schedulato.

Per la sua schedulazione si può pensare di utilizzare un **code repository**, ad esempio Gitlab, implementando e schedulando una pipeline di CI/CD per eseguire, ad esempio, tutte le mattine alle 7:00 a.m. considerando fornendo come input allo script python l'intervallo temporale rolling degli ultimi 7 giorni.

Questa pipeline si occuperà di:

- Recuperare l'immagine più recente del progetto registrata in un container repository (Gitlab, ad esempio, ne mette uno a disposizione off-the-shelf, configurabile)
- Eseguire il progetto montando un volume di input in modo tale che vengano forniti al progetto i dataset necessari per la sua esecuzione; così facendo lo script potrà poi scrivere – su uno specifico volume di output – i dati che a sua volta saranno necessari ai modelli (uno per tipologia di arepa) per effettuare la predizione

In questo possiamo presupporre che Gitlab possa comunicare con un server dove è presente Docker, oppure con uno o più server componenti un cluster Kubernetes.

Il modello poi riceverà in input tali dati (output dello script) e produrrà l'output, a prescindere da che tipo di output sia (una mail di alerting per una certa soglia raggiunta, dei parametri da presentare in una dashboard, etc.).

Si può anche pensare, in un'evoluzione del tutto, che il modello sia embeddato all'interno di un'applicazione e che questa esponga un endpoint che, se invocato, restituisca il risultato della predizione.

La sovrastruttura che può essere scelta per la gestione del modello esula dalla risposta alla domanda.

In un'ottica di CI anche dei modelli è possibile pensare di utilizzare un **model repository** (e.g. MLFlow) con cui valutare le performance dei modelli nel tempo, ipotizzando di effettuare re-training a intervalli regolari (e.g. ogni 2 mesi, la granularità dipende sia dal tipo di modello che dal dataset considerato).

In questo modo, è possibile considerare dataset con profondità temporale maggiore dei 7 giorni ed allenare il modello per riconoscere quali valori di metriche utilizzare per una migliore maintanance predittiva.

Le performance dei modelli re-trainati possono essere quindi confrontate con quelle del modello attualmente in produzione (ad esempio MLFlow ne consente la consultazione) e si può prevedere la schedulazione di una pipeline di promozione in produzione del modello più performante sul dataset più recente utilizzato per il re-training.
Ovviamente per questa "promozione" del modello in produzione è necessario considerare l'incapsulamento del modello come venga effettuato, quindi se eseguito da uno script, piuttosto che esposto tramite endpoint, piuttosto che, ancora, embeddato in una webapp.