Problem Statement:
Model drift occurs when there is a general degradation in machine learning model performance. There are two broad reasons for this:

1. **Data drift:** this occurs when the underlying distribution of data being fed to the model changes, meaning that the distribution of data in the model training phase is different from the deployment phase
2. **Concept drift:** this occurs when the fundamental relationship between the input and target change, meaning that learned relationships in the training phase no longer hold in the deployment phase

In the context of a speech-to-text model, data drift could occur if input audio comes from a different distribution - for instance, the trained model might be trained on audio recorded from particular devices or environments whereas audio in deployment might be recorded on poorer quality devices or have background noise. Here, the fundamental relationship between audio and text has not changed, but the distribution of data has. Model drift might occur if the model is deployed to serve different audio profiles from that in the training data, for instance a region with a different accent from what was used in training. Equally possible is where new words introduced into a certain lexicon might lead to a different relationship between audio and text. Here, the fundamental relationship between certain spoken sounds and text might have shifted.

Monitoring Pipeline:
Conceptually, both data and model drift lead to a degradation of model performance metrics. However, given a degradation in performance, we cannot immediately conclude if it is due to concept or data drift.

Data Drift:
Appropriately tracking data drift would mean tracking the underlying distribution of input audio. High-level metrics might be tracking the sampling rate or the level of distortion in input audio. A more model-specific metric in this instance might be to track the distribution of embeddings from the wave2vec model being served along some dimensions - if underlying data has not shifted then the distributions along each of these dimensions should remain relatively stable.
A general pipeline would be as follows:

- At prediction, save audio and model embeddings to some database. Data warehouses like BigQuery might enable easier analysis downstream.
- At fixed intervals, perform analysis on distribution of input data and model embeddings. This could be done through scheduled queries (available on BigQuery to evaluate views) or through a serverless function, which might run an evaluation and send an alert if data drift is detected.

If using managed services, services like Vertex AI provide input monitoring and alerting tools as part of a general suite of MLOps functionality.

Concept Drift:
The challenge for concept drift here is that there is no label for transcribed audio in production, so it would be difficult to track model accuracy. If one were deploying a model to predict trades in the market, for example, accuracy can be measured against actual trades that happened.

Tracking concept drift might hence require periodically acquiring samples of labeled audio data from a representative sample of the population. One way would be to ask users if their audio is correctly labeled, along with what they actually said.

Pipeline:

- If the speech-to-text model is user-facing, integrate a data pipeline with the frontend that allows users to indicate if predictions are correct or not, along with the correct label.
- If not user-facing or not possible, then find a different way to acquire accurately labeled data.
- Save correctly labeled data to cloud storage or database after preprocessing.
- Schedule a serverless function to call model API to perform transcriptions and calculate model performance against this validation data.