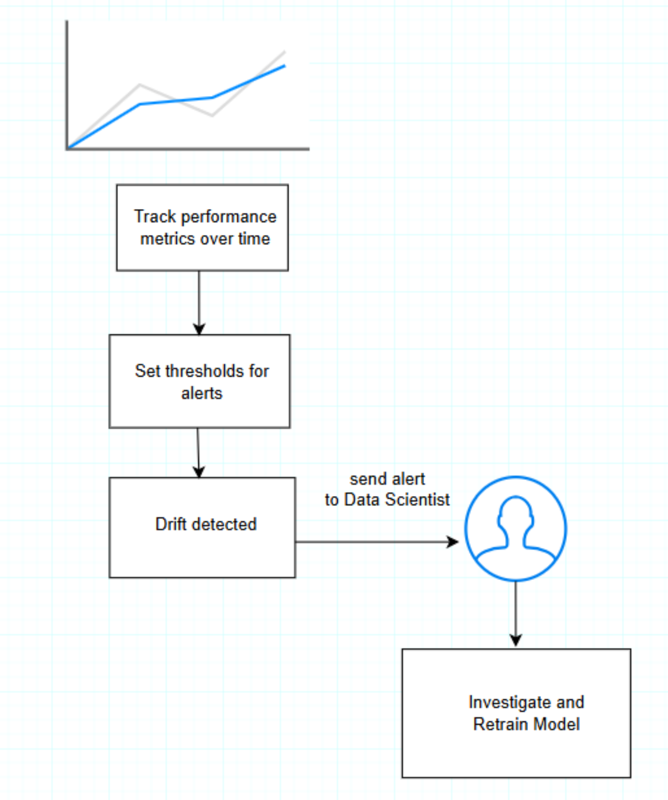
Tracking Model Drift

Model drift can be defined as a situation when a model's performance degrades due to changes in data distribution. When we deploy an ML model into production, it ingests real-world data and this data can differ from that which the models have been trained on, resulting in a potential dip in model performance.

For the field of Automatic Speech Recognition (ASR), the Wav2Vec2 model was trained on datasets that are biased towards UK/American accents. If we use the model to ingest datasets from Singapore, we would likely see a data drift due to the different training context.

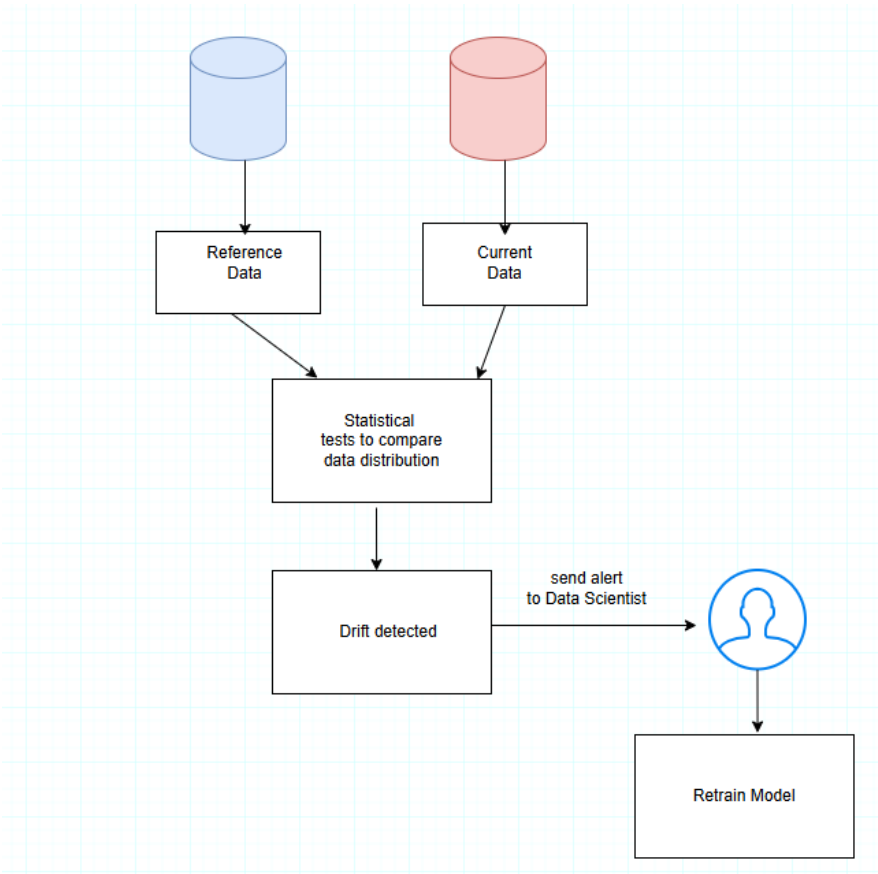
The motivation for planning around model drift is because it can potentially cause a decline in model performance, thereby affecting the reliability of the model. There are two main categories of model drift, and I will outline potential pipelines to monitor for these occurences:

i. Concept drift: when the relationship between the input and target variables change; we will need to monitor performance metrics of the model because if there is a concept drift, we are very likely to see worsening performance.



- Continuously track performance metrics of a model (i.e. RMSE, Accuracy, Precision, F1-score) over time. Compare the live performance metrics against historical baselines and acceptable thresholds.
- If the metrics degrades below these thresholds, alerts will be sent to the ML team to relook at retraining (i.e. if a relationship between y and x changes from linear to non-linear, that will mean using a different model altogether!). In the example of ASR, that could mean a Word Error Rate falling below baseline to be cause of an alert to the ML team.

ii. Data drift: when the distribution of the input change, this would involve comparing the distributions of the training data against the live data



- The model drift pipeline will ingest from the following data sources: Reference Data (the data used to train the model), Current Data (live production data being fed into the model)
- We can employ the following techniques to detect statistical drift: Kolmogorov-Smirnov (KS) test or Chi-squared test. We can set a threshold, typically a p-value at a predefined significance level. When the results of the statistical tests fall below the p-value, the system would then send an alert to the ML/Data Science team for retraining.

Considering the two main causes of model drifts, tracking them would involve monitoring performance metrics and changes in data distribution. I would explore employing the following tech stack to facilitate the process:

i. EvidentlyAI – open-source library which automates drift detection and performance monitoring
ii. MLFlow – for versioning, comparing models, tracking experiments

References:

Diagrams were created in draw.io

https://www.evidentlyai.com/ml-in-production/data-drift

https://c3.ai/glossary/data-science/model-drift/#:~:text=What%20is%20Model%20Drift%3F,the%20target%20(dependent)%20variable%20change