### **1. Baseline Identification**

* **Per-Patient Baseline Calculation**:
  + **Initial Period Analysis**: For each patient, identify an initial time window (e.g., the first day of data - “baseline” or “0”) during which their typical biomarker levels and wearable metrics are recorded. Use this period to establish the patient's baseline.
  + **Statistical Baseline Features**: Compute key statistics such as the percentile, mean, median, variance, and range for each variable (e.g., biomarkers, and wearable device metrics) to define the patient’s baseline profile.
  + **Dynamic Baselines**: In some cases, a patient's baseline might change over time. Consider using a rolling window (e.g., the last 7 days) to update the baseline dynamically, especially for wearable data.
* **Normalization Using Baseline**:
  + Subtract the patient's baseline from the real-time data to normalize each patient's features. For example, if heart rate variability is a feature, use the deviation from the patient’s baseline as a new feature.
  + This approach helps to focus the model on changes relative to the patient’s norm, rather than absolute values, which are highly individualistic.

### **2. Patient Clustering for Baseline Similarity**

* **Clustering to Group Similar Patients**:
  + Before model training, use clustering algorithms (e.g., K-Means, DBSCAN, hierarchical clustering) to group patients based on their baseline characteristics (e.g., biomarker levels, wearable data trends).
  + **Feature Selection for Clustering**: Select features that represent baseline health metrics (e.g., resting heart rate, baseline glucose levels). Standardize these features before clustering.
* **Clustering Algorithms**:
  + **K-Means**: A straightforward clustering algorithm that groups patients based on their feature vectors. Specify the number of clusters (K) based on exploratory data analysis (e.g., using the elbow method).
  + **DBSCAN**: Useful if patients naturally fall into clusters of varying shapes and densities, allowing the detection of outliers.
  + **Hierarchical Clustering**: For cases where the number of clusters is not known a priori a hierarchical relationship among patients is needed.
  + **Time Series Cluster**: DTW - what we implemented
* **Cluster-Specific Models**: Train separate machine learning models for each patient cluster to account for group-specific baseline variations, improving the model's sensitivity to changes indicative of CRS.

### **3. Model Training with Baseline Integration**

* **Feature Engineering**:
  + **Baseline-Adjusted Features**: Incorporate features that measure deviations from the patient’s baseline. For example, instead of raw biomarker values, use “change from baseline” as input features.
  + **Cluster Features**: Add the cluster label as an additional feature, enabling the model to differentiate predictions based on the patient group’s baseline characteristics.
* **Multi-Level Modeling**:
  + **Global Model with Personalized Adjustments**: Train a global model for all patients using both baseline-adjusted features and cluster labels. Introduce interaction terms between features and cluster labels to allow the model to adjust its decision boundary based on the patient's group.
  + **Ensemble of Cluster-Specific Models**: An alternative is to build an ensemble of models, where each sub-model specializes in predicting CRS for a specific patient cluster. During inference, a patient's cluster is first identified, and the corresponding model is used for prediction.
  + **Model selection:** 
    - **Time-Series GAN**
    - **LSTM / RNN**
    - **Simple models:**
      * **Linear regression (baseline?)**
    - **Ensemble method:**
      * **Random forest**
      * **Xgboost**
    - **Should we try to finetune a language model through API？**

### **4. Handling Time-Series Data with Baselines**

* For wearable device data that changes over time:
  + **Sequential Models**: Use Recurrent Neural Networks (RNNs) like LSTM or GRU, feeding in both the current data and the baseline-adjusted data (e.g., deviations from the baseline).
  + **Temporal Convolutional Networks (TCNs)**: TCNs can process temporal data while taking into account patient-specific variations in a more flexible way.
* **Data Augmentation**:
  + **SMOTE (Synthetic Minority Over-sampling Technique)**: Commonly used for augmenting data in classification problems, SMOTE can generate synthetic samples in the feature space. However, for time-series or personalized data like patient records, this may not be directly applicable.
  + **Time-Series Augmentation**: If your data points are time-series in nature (e.g., daily biomarker readings), consider augmenting the data using techniques such as:
    - **Window Slicing**: Create additional data points by slicing the time-series data into overlapping windows (e.g., 5-day windows) to generate more samples for training.
    - **Perturbation**: Add small variations (e.g., random noise) to the existing data points to generate new samples, while ensuring the perturbations are biologically or clinically reasonable.

### **5. Evaluation Metrics Considering Baselines**

* Evaluate the model using metrics that focus on its ability to detect deviations from patient-specific baselines:
  + **Precision and Recall at Deviations**: Specifically, track how well the model detects instances where patient metrics deviate significantly from their baseline.
* **Cluster-Based Evaluation**: Assess the model's performance within each patient cluster to ensure the model generalizes well across different baseline groups.

### **6. Deployment & Dynamic Adaptation**

* During deployment, update each patient's baseline continuously or periodically to adapt to their evolving health state. Use this dynamic baseline to adjust incoming data before making predictions.
* Implement a **Baseline Drift Detector**: Set up a monitoring system to detect shifts in the patient’s baseline (e.g., due to lifestyle changes, treatment interventions) and trigger model retraining if significant changes are observed.