**Data Training**

**Will you be using SageMaker Jumpstart, built-in-algorithms, bring-your-own-script, or bring-your-own container?**  
We chose to use the bring-your-own-script option in SageMaker Studio. This allows us the flexibility to implement custom data preprocessing, handle our clinical and proteomics dataset merging, and train a model that fits our specific problem using our own code. This choice is ideal for research-based projects like ours, where the pipeline involves multiple data sources, complex feature engineering (e.g., combining clinical, PAM50, and proteomic data), and customized model training steps such as balancing with SMOTE. Built-in algorithms do not support these custom preprocessing workflows directly.

**Which algorithm will you be using?**  
We plan to use XGBoost, a gradient boosting algorithm that performs well on structured/tabular data. XGBoost is suitable for our project because it handles feature interactions well, provides built-in handling for missing values, and delivers strong performance with imbalanced data—especially when combined with SMOTE. Its interpretability (e.g., feature importance) also aligns with our goal of building explainable AI tools for healthcare providers.

Which parameters will you be passing?  
Key hyperparameters we plan to tune include:

* max\_depth (e.g., 5–10) to control tree complexity,
* learning\_rate (e.g., 0.01–0.1) for convergence speed,
* n\_estimators (e.g., 100–500) to control the number of trees,
* subsample and colsample\_bytree to reduce overfitting,
* scale\_pos\_weight if imbalance persists after SMOTE.

These parameters impact model complexity, training speed, and generalization. We'll use SageMaker's hyperparameter tuning capabilities to optimize them.

**Which instance size/count will you be using?**  
We will use a ml.m5.xlarge instance (4 vCPU, 16GB RAM) for training. Our dataset is medium-sized after preprocessing, and this instance provides a good balance between cost and performance. We chose one instance for now, but can scale if required. Since we are not using GPU-based models or deep learning, a CPU instance is more cost-efficient.

**How will you evaluate your model?**  
We will evaluate the model using metrics including accuracy, precision, recall, F1-score, and AUC-ROC. In healthcare applications like ours, false negatives (missing cancer cases) are more critical than false positives, so we place particular emphasis on recall and AUC-ROC. These metrics will help ensure our model identifies as many true cancer cases as possible, aligning with our goal of early and reliable breast cancer detection.