ADS508 Data Science With Cloud Computing

Design Document

GitHub Repository: https://github.com/darrencheninfo/508Group2

Authors: Arjun Venkatesh, Darren Chen, Vinh Dao

Company Name: AdvanceHC Solution

Company Industry: Healthcare AI

Company Size: Startup

Abstract:

AdvanceHC Solution is developing an AI-powered diagnostic tool to detect breast cancer at an early stage using advanced machine learning models. The challenge we face is that our product is still in the testing phase, making it difficult to gain traction with healthcare providers and regulatory bodies.

Problem Statement:

Breast cancer remains one of the leading causes of mortality among women worldwide. Early detection significantly improves survival rates, yet many current screening methods rely on manual radiology analysis, which can be time-consuming, prone to human error, and inaccessible in certain regions. Our company, AdvanceHC Solution, aims to develop an AI-powered tool that can detect breast cancer at an advanced level with higher accuracy and efficiency. However, as a startup, we face a major challenge: our AI model is still in the trial phase, and hospitals and healthcare providers are hesitant to adopt a product that has not been fully validated.

To overcome this, we need a robust dataset and validation process that proves the reliability of our model. Additionally, we must address regulatory concerns, secure partnerships with medical institutions, and ensure our AI system remains explainable and ethical in decision-making.

Goals:

1. Develop and test a machine learning model that can detect breast cancer from mammogram images with high accuracy.
2. Validate our AI model against existing radiologist diagnoses to prove its reliability.
3. Store and process data securely while adhering to healthcare privacy regulations.
4. Build trust with healthcare providers by demonstrating transparent AI decision-making.
5. Create a strategy for commercializing the product, focusing on partnerships with hospitals and research institutions.

Non-Goals:

1. We are not developing an AI system that replaces doctors but rather one that assists them in diagnosis.
2. The focus is on breast cancer detection only, not treatment recommendations.
3. We are not handling direct patient interactions or building a patient-facing application.
4. No real-time diagnosis during the testing phase—this is a research-oriented project.
5. Regulatory approvals (e.g., FDA, CE marking) are outside the scope of this phase.

Data Sources:

- **Primary Source:** Kaggle breast cancer imaging datasets.
    - https://www.kaggle.com/datasets/piotrgrabo/breastcancerproteomes/data
    - (*Breast Cancer Proteomes*, n.d.)
- **Data Storage:** AWS S3 for secure storage and easy access. (*AWS Educate - Cloud Skills for Education- AWS*, n.d.)
- **Risks:** Data quality and potential bias in training sets (e.g., underrepresentation of certain demographics).

Data Exploration:

- Check for missing or corrupted images and remove any anomalies.
- Analyze dataset distribution (age, ethnicity, cancer stages) to avoid bias.
- Identify key image features and metadata (e.g., tumor size, shape, density).
- Ensure ethical considerations are met in data usage.

Storage & Tools:

- Data ingestion and exploration using **AWS SageMaker Studio Notebooks**.
- Code repository stored in **GitHub** for collaboration.

Data Preparation:

- **Data Scrubbing:** Remove duplicate or low-quality images.
- **Feature Selection:** Focus on image characteristics and exclude irrelevant metadata.

- **Feature Creation:** Augment data (e.g., rotation, contrast enhancement) to improve generalization.
- **Feature Transformation:** Normalize pixel values and apply image preprocessing techniques.
- **Balancing:** Address class imbalance using oversampling techniques.
- **Splitting:** 80% training, 10% validation, 10% testing.

Model Training:

- **Training Approach:** Use AWS SageMaker built-in algorithms and fine-tune pre-trained CNN models (e.g., ResNet, EfficientNet).
- **Parameters:** Learning rate, batch size, number of epochs optimized using hyperparameter tuning.
- **Instance Size:** GPU-based SageMaker instances for efficient model training.
- **Evaluation:** Performance measured using accuracy, precision, recall, and AUC-ROC score.

Measuring Impact:

- **Accuracy Improvement:** Increase detection accuracy beyond 90% compared to human radiologists.
- **Reduction in Diagnostic Time:** Decrease time required for diagnosis by at least 50%.

Security Checklist, Privacy, and Risks:

- **PHI Data Handling:** No direct PHI storage; images anonymized before processing.

- **PII Data Handling:** Ensure compliance with HIPAA/GDPR for data privacy.
- **Bias Consideration:** Regular audits for dataset fairness and transparency in AI decision-making.
- **Ethical Concerns:** AI should not make independent decisions but assist radiologists.

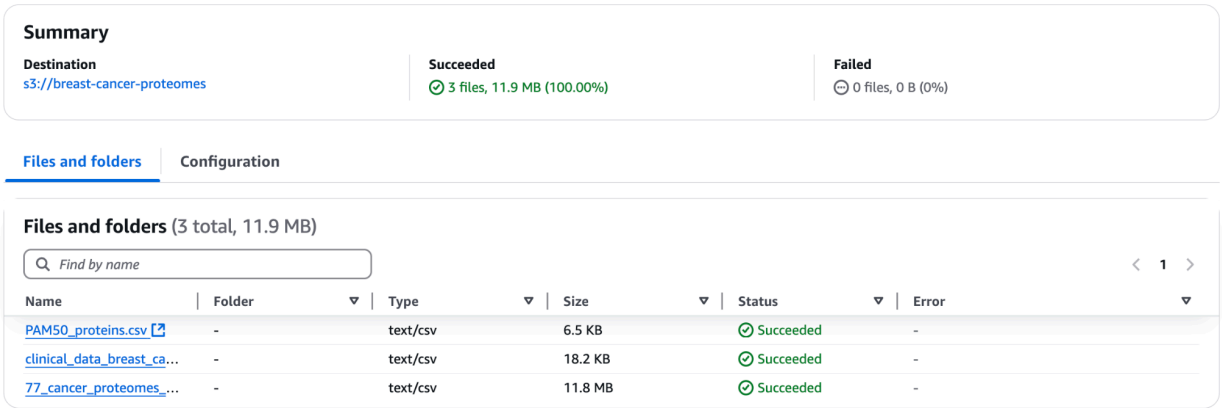
Future Enhancements:

1. Expand dataset sources to include real-world hospital imaging data.
2. Integrate explainable AI techniques to provide justifications for model decisions.
3. Develop a real-time AI-assisted diagnostic tool for clinical use.

Data Exploration

1. Where will you store your data, and how will you ingest it?

- **Storage:** Data is stored in an S3 bucket named `s3://breast-cancer-proteomes/raw/`.



-

- **Ingestion:**
    - Manually uploaded the Kaggle dataset (`77_cancer_proteomes_CPTAC_traq.csv`) to the S3 bucket via the AWS Console.
    - Using a SageMaker Studio Notebook, the data is copied to the local instance using `boto3` and `pandas` for exploration.

2. Tools for Ingestion & Exploration:

- **AWS Services:** S3, SageMaker Studio.
- **Libraries:** `boto3` (AWS SDK), `pandas` (data manipulation), `matplotlib`/`seaborn` (visualization).

3. Key Exploration Tasks:

- **Data Quality:**
    - Identify missing values and null data (e.g., protein expression ratios marked as NaN).
    - Validate data types (e.g., ensure `log2_ratio` columns are numeric).
- **Bias & Ethical Checks:**
    - Assess sample diversity (e.g., age, ethnicity metadata if available).
    - Verify if data represents diverse cancer subtypes (e.g., HER2+, ER+).
- **Key Fields:**

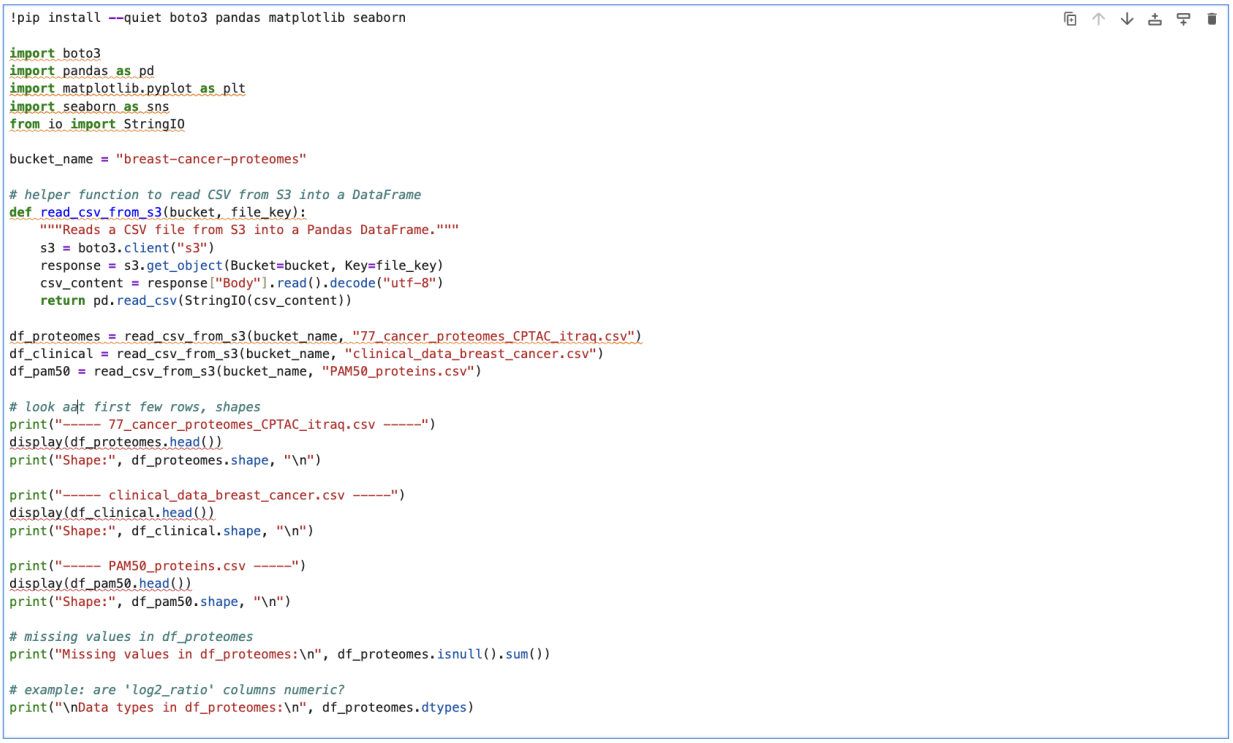
  Identify key fields

- `RefSeq_ID`: Unique protein identifiers.
- `gene_symbol`: Gene associations for proteins.
- `log2_ratio`: Proteomic expression levels across samples.

4. GitHub Code Link:

[GitHub Repository: https://github.com/darrencheninfo/508Group2](https://github.com/darrencheninfo/508Group2)

5. Screen Captures from S3-Local Data Exploration Work

- 

```
!pip install --quiet boto3 pandas matplotlib seaborn

import boto3
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from io import StringIO

bucket_name = "breast-cancer-proteomes"

# helper function to read CSV from S3 into a DataFrame
def read_csv_from_s3(bucket, file_key):
    """Reads a CSV file from S3 into a Pandas DataFrame."""
    s3 = boto3.client("s3")
    response = s3.get_object(Bucket=bucket, Key=file_key)
    csv_content = response["Body"].read().decode("utf-8")
    return pd.read_csv(StringIO(csv_content))

df_proteomes = read_csv_from_s3(bucket_name, "77_cancer_proteomes_CPTAC_itraq.csv")
df_clinical = read_csv_from_s3(bucket_name, "clinical_data_breast_cancer.csv")
df_pam50 = read_csv_from_s3(bucket_name, "PAM50_proteins.csv")

# look aat first few rows, shapes
print("----- 77_cancer_proteomes_CPTAC_itraq.csv -----")
display(df_proteomes.head())
print("Shape:", df_proteomes.shape, "\n")

print("----- clinical_data_breast_cancer.csv -----")
display(df_clinical.head())
print("Shape:", df_clinical.shape, "\n")

print("----- PAM50_proteins.csv -----")
display(df_pam50.head())
print("Shape:", df_pam50.shape, "\n")

# missing values in df_proteomes
print("Missing values in df_proteomes:\n", df_proteomes.isnull().sum())

# example: are 'log2_ratio' columns numeric?
print("\nData types in df_proteomes:\n", df_proteomes.dtypes)
```

```
------ 77_cancer_proteomes_CPTAC_itraq.csv ------
```

	RefSeq_accession_number	gene_symbol	gene_name	AO-A12D.01TCGA	C8-A131.01TCGA	AO-A12B.01TCGA	BH-A18Q.02TCGA	C8-A130.02TCGA	C8-A138.03TCGA	E2-A154.03TCGA	...	AO-A12B.34TCGA	A
0	NP_958782	PLEC	plectin isoform 1	1.096131	2.609943	-0.659828	0.195341	-0.494060	2.765081	0.862659	...	-0.963904	
1	NP_958785	NaN	plectin isoform 1g	1.111370	2.650422	-0.648742	0.215413	-0.503899	2.779709	0.870186	...	-0.938210	
2	NP_958786	PLEC	plectin isoform 1a	1.111370	2.650422	-0.654285	0.215413	-0.500619	2.779709	0.870186	...	-0.943919	
3	NP_000436	NaN	plectin isoform 1c	1.107561	2.646374	-0.632113	0.205377	-0.510459	2.797995	0.866423	...	-0.935355	
4	NP_958781	NaN	plectin isoform 1e	1.115180	2.646374	-0.640428	0.215413	-0.503899	2.787023	0.870186	...	-0.935355	

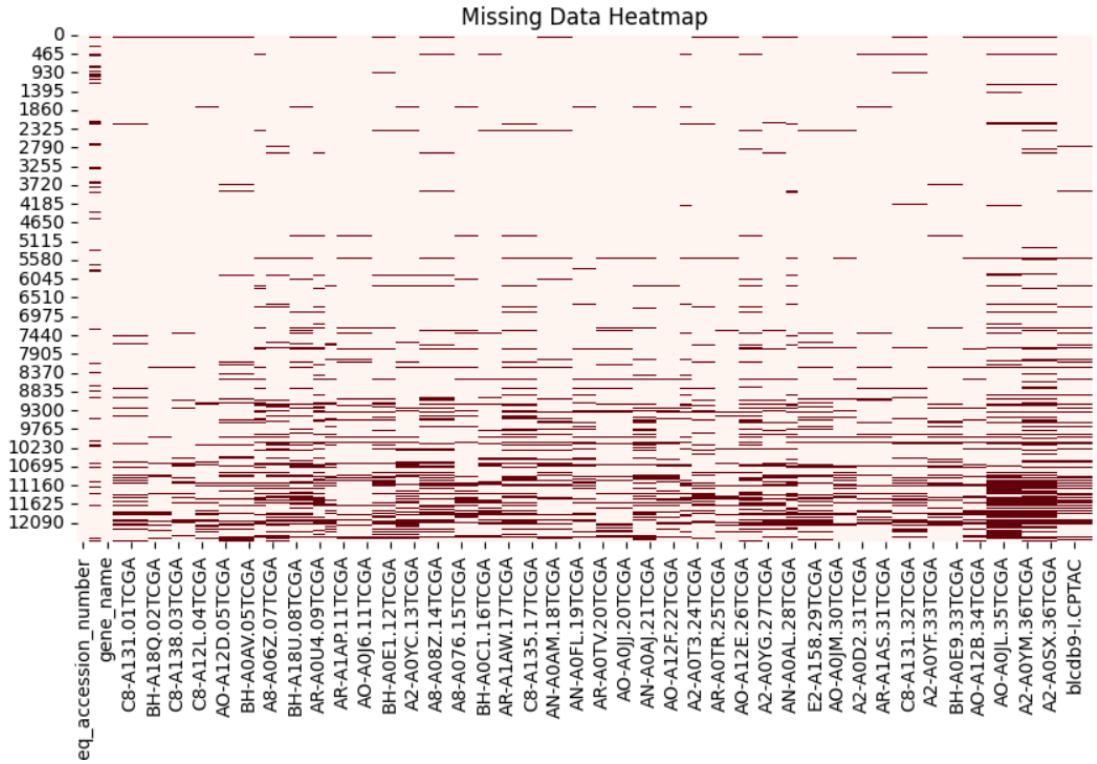
5 rows × 86 columns

```
Shape: (12553, 86)

------ clinical_data_breast_cancer.csv ------
```

	Complete TCGA ID	Gender	Age at Initial Pathologic Diagnosis	ER Status	PR Status	HER2 Final Status	Tumor	Tumor-T1 Coded	Node	Node-Coded	...	PAM50 mRNA	SigClust Unsupervised mRNA	SigClust Intrinsic mRNA	miRNA Clusters	methylation Clusters	RPPA Clusters	CN Clusters	Integ Cl PA
0	TCGA-A2-A0T2	FEMALE	66	Negative	Negative	Negative	T3	T_Other	N3	Positive	...	Basal-like	0	-13	3	5	Basal	3	
1	TCGA-A2-A0CM	FEMALE	40	Negative	Negative	Negative	T2	T_Other	N0	Negative	...	Basal-like	-12	-13	4	4	Basal	4	
2	TCGA-BH-A18V	FEMALE	48	Negative	Negative	Negative	T2	T_Other	N1	Positive	...	Basal-like	-12	-13	5	5	Basal	1	
3	TCGA-BH-A18Q	FEMALE	56	Negative	Negative	Negative	T2	T_Other	N1	Positive	...	Basal-like	-12	-13	5	5	Basal	1	
4	TCGA-BH-A0E0	FEMALE	38	Negative	Negative	Negative	T3	T_Other	N3	Positive	...	Basal-like	0	-13	5	5	Basal	1	

5 rows × 29 columns



----- PAM50_proteins.csv -----

	GeneSymbol	RefSeqProteinID	Species	Gene Name
0	MIA	NP_006524	Homo sapiens	melanoma inhibitory activity
1	FGFR4	NP_002002	Homo sapiens	fibroblast growth factor receptor 4
2	FGFR4	NP_998812	Homo sapiens	fibroblast growth factor receptor 4
3	FGFR4	NP_075252	Homo sapiens	fibroblast growth factor receptor 4
4	GPR160	NP_055188	Homo sapiens	G protein-coupled receptor 160

```
Shape: (100, 4)

Missing values in df_proteomes:
 RefSeq_accession_number       0
gene_symbol                 1780
gene_name                      0
A0-A12D.01TCGA              1219
C8-A131.01TCGA              1218
                            ...
BH-A0C7.36TCGA              2860
A2-A0SX.36TCGA              2856
263d3f-I.CPTAC              1641
blcdb9-I.CPTAC              1668
c4155b-C.CPTAC              1640
Length: 86, dtype: int64

Data types in df_proteomes:
 RefSeq_accession_number     object
gene_symbol                  object
gene_name                    object
A0-A12D.01TCGA              float64
C8-A131.01TCGA              float64
                             ...
BH-A0C7.36TCGA              float64
A2-A0SX.36TCGA              float64
263d3f-I.CPTAC              float64
blcdb9-I.CPTAC              float64
c4155b-C.CPTAC              float64
Length: 86, dtype: object
```

Measuring Impact

Two metrics tied to project goals:

1. **Sub-classification Accuracy:** Increase from 75% to 90% by leveraging proteomic biomarkers.
2. **Missing Data Reduction:** Decrease missing protein values by 50% through imputation techniques.

Security Checklist, Privacy, and Other Risks

Question	Response
PHI Data?	No – Dataset contains anonymized proteomic profiles without patient identifiers.
PII Data?	No – No names, addresses, or demographic details are included.
Track User Behavior?	No – This project does not involve user tracking.
Credit Card Data?	No – Not applicable to healthcare proteomic analysis.
S3 Buckets Used	s3://breast-cancer-proteomes/raw/ (read-only), s3://breast-cancer-proteomes/processed/ (write).
Data Bias	Potential bias in sample selection (e.g., underrepresentation of rare subtypes).

Ethical Concerns	Ensure findings are not misused for non-inclusive treatment plans.

Data Preparation

We combined the proteomics (in wide format), clinical, and PAM50 datasets to form a single, coherent DataFrame. We melted the proteomics data to a long format so each (Gene, Sample) pair became a row, then matched clinical records via a standardized “Patient_ID” field. This was crucial to ensure each proteomics measurement was properly aligned with its corresponding patient information.

Data Scrubbing

We removed duplicates to eliminate redundant rows. We imputed missing numeric columns with the mean and missing categorical columns with the most frequent value. This ensures we don’t discard large parts of the dataset solely because some cells are incomplete, thus preserving more information for modeling.

Feature Selection

We specifically included the “Expression” column (the main proteomic measurement) and certain clinical features (e.g., “Age at Initial Pathologic Diagnosis,” “ER Status,” “PR Status,” “HER2 Final Status,” and “AJCC Stage”) known to be relevant in breast cancer analyses. We

excluded or ignored columns that do not contribute meaningful signals to the model or that duplicate other information.

Feature Creation

The most significant feature creation step was melting the wide-format proteomics file into a single “Expression” column. If the data contained tumor grade categories (e.g., “G1,” “G2,” etc.), we would bucket these into “High,” “Low,” or “Unknown” to simplify analyses. Merging these data sources and shaping them into a final dataset required carefully mapping each sample to the correct patient ID.

Feature Transformation

We used label encoding for the target if it was categorical text. For other categorical predictors, we performed one-hot encoding so the model can handle them as numeric indicators. We scaled numerical features (e.g., “Expression”) to have zero mean and unit variance with StandardScaler. This normalization helps many machine learning algorithms converge and improves overall consistency.

How We Balance the Dataset

We employed SMOTE (Synthetic Minority Over-sampling Technique) to address any class imbalance in the “Target” column (e.g., “OS event”). The target column, "OS event", was found to be imbalanced, meaning one class had more records than the other. To correct this, SMOTE (Synthetic Minority Over-sampling Technique) was applied only to the training dataset after splitting. This method generates synthetic examples for the minority class, making both classes

more even in the training set. Balancing only the training set helps the model learn from both classes equally without causing data leakage.

How We Split the Dataset

We split the balanced dataset into training (70%), validation (about 20%), and test (about 10%). First, we designated 70% of the data as training, then divided the remaining 30% so that roughly two-thirds became the validation set and one-third became the test set. We used stratified splitting to preserve the distribution of the target variable, making sure each subset remains representative. A random state was set to ensure reproducibility.

Data Training

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.

Additionally, we're incorporating data validation checks for our preprocessing script. This includes verifying merged records against expected totals, tracking any missing or unexpected values, and ensuring consistent data splits. Because we are using bring-your-own-script, we can seamlessly integrate these checks alongside the SMOTE step. We store each script version in a Git repository, guaranteeing both reproducibility and traceability.

Algorithm

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.

We evaluated other potential algorithms (random forest, logistic regression, etc) and found that XGBoost consistently provides strong performance for tabular data and is easier to explain in the healthcare context.

Parameters

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.

To efficiently manage the tuning process, we will use SageMaker’s Automated Model Tuning jobs, experimenting with a range of values for max_depth, learning_rate, n_estimators, and subsample. Each run’s parameter settings and performance metrics (AUC-ROC, F1-score) will be tracked in CloudWatch logs and a simple data table. This tracking ensures we can revisit, compare, and reproduce specific configurations if needed.

Instance size/count

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.

We will monitor CPU utilization and memory usage with Amazon CloudWatch to confirm that ml.m5.xlarge is neither over- nor under-powered for our training set. If training jobs consistently saturate CPU or memory, or if training times become prohibitively long, we will investigate scaling to ml.m5.2xlarge or distributing training across multiple instances. Conversely, if resource usage remains low, we may reduce to ml.m5.large for cost savings.

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.

Beyond these standard metrics, we will evaluate different probability thresholds to optimize recall in line with clinical priorities. For example, if a slightly higher false-positive rate is acceptable in early cancer detection, we may select a threshold lower than 0.5 to ensure fewer high-risk patients go undetected. We will also look at precision-recall curves to confirm our model's effectiveness in dealing with class imbalance and ensuring that we prioritize minimizing false negatives.

Future Enhancements

1. **Include Additional Patient Information**

One potential improvement involves incorporating a wider range of patient data—such as genetic history, family medical background, and lifestyle factors like diet, exercise, or smoking habits. These features could provide the model with more context and lead to more accurate predictions. The current version primarily uses clinical and proteomic markers but expanding the dataset to include more personal health indicators would help create a more robust and well-rounded prediction tool.

In addition to the existing clinical and proteomic data, incorporating broader patient details such as socioeconomic background, environmental factors, and real-time health metrics from

wearable devices could enhance the model's precision. By integrating these diverse data sources, the model would be better equipped to detect subtle risk factors and adapt its predictions to the complex, multifaceted nature of breast cancer, ultimately supporting more personalized care.

2. **Enable Real-Time Prediction Capability**

Another meaningful enhancement would be transforming the model pipeline to support real-time predictions. While batch processing is sufficient for offline testing and analysis, real-time functionality would be much more practical in a clinical setting. Deploying the model through a cloud-based API—using tools like AWS SageMaker or FastAPI—would allow healthcare professionals to receive predictions instantly during patient consultations, improving workflow efficiency and clinical decision-making.

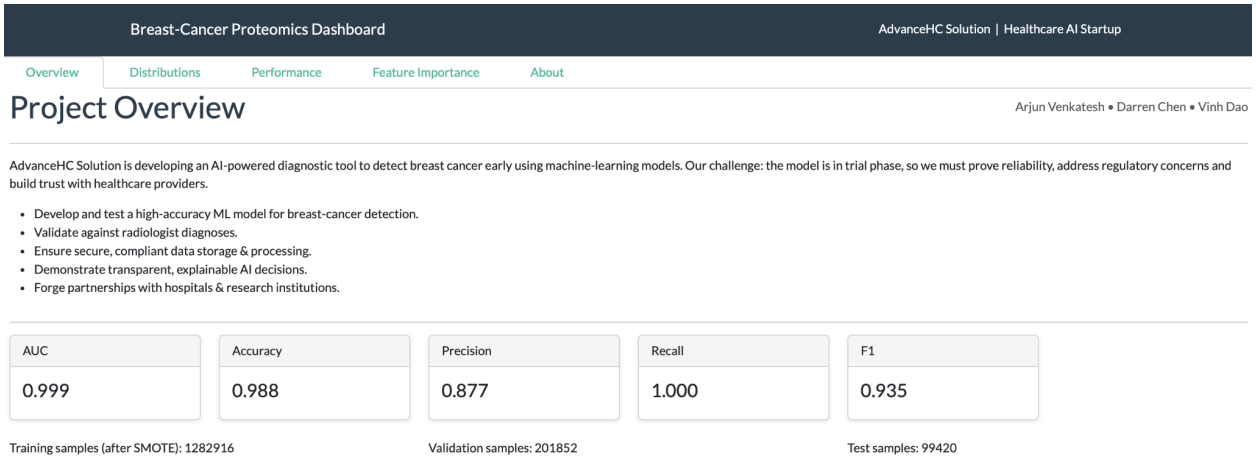
Expanding the model to support real-time predictions would allow it to integrate directly with clinical systems, such as electronic health records, to provide continuous updates. By developing a streamlined API for immediate data processing and prediction, healthcare professionals could receive timely risk assessments during patient visits, thereby improving decision making speed and overall performance.

3. **Integrate Explainable AI (XAI) Tools**

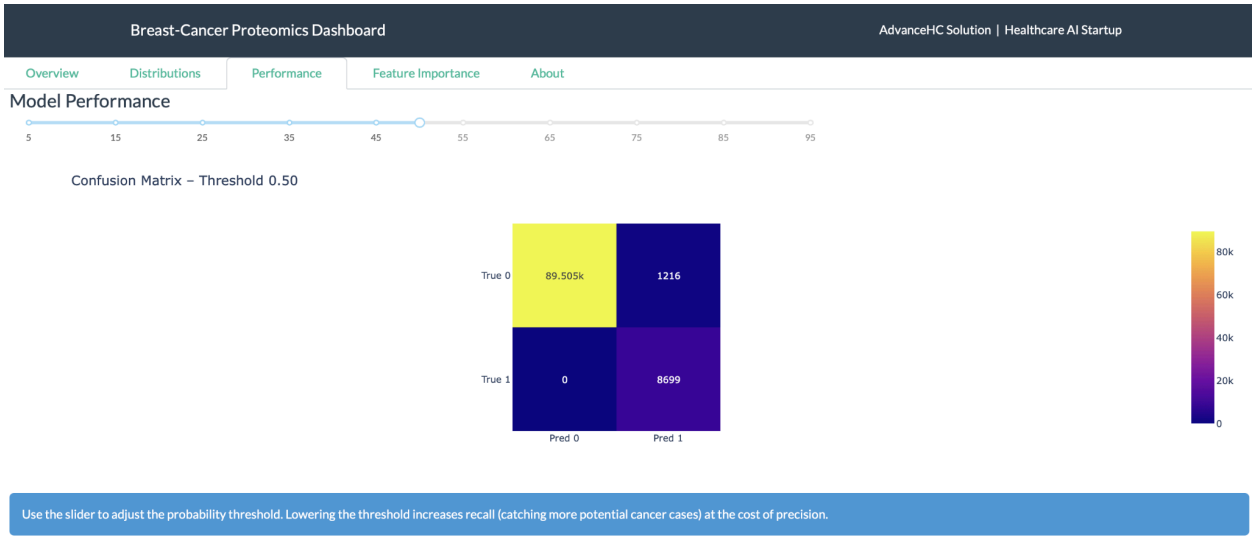
To improve transparency and usability, it would be beneficial to integrate explainability methods such as SHAP (SHapley Additive Explanations) or LIME. These tools can highlight which input features contributed most to prediction, helping medical professionals better understand the model’s decisions. In high-stakes fields like healthcare, providing clear, interpretable output not only supports better decision-making but also builds trust in technology.

Adding straightforward interpretability features, like simplified visual dashboards and analysis reports, can make the model's predictions more accessible to clinicians. By clearly outlining which factors drive each prediction, these tools would not only improve transparency but also offer a simpler feedback mechanism for identifying potential biases, ultimately enhancing performance.

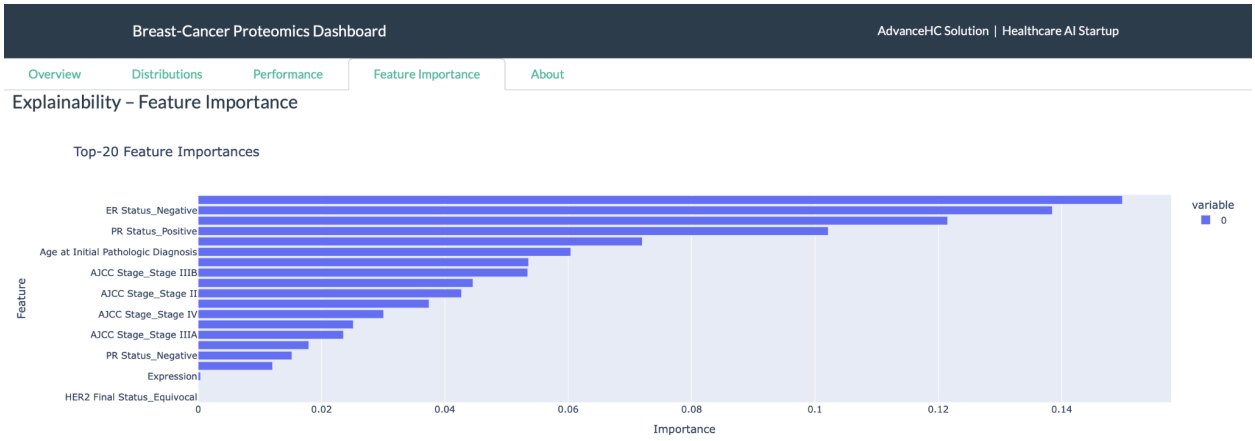
Additional: Dashboard Snapshots

- 

- 


Breast-Cancer Proteomics Dashboard
AdvanceHC Solution | Healthcare AI Startup
Overview
Distributions
Performance
Feature Importance
About
Model Performance
5
15
25
35
45
55
65
75
85
95
Confusion Matrix – Threshold 0.50
True 0
True 1
89.505k
1216
0
8699
Pred 0
Pred 1
80k
60k
40k
20k
0
Use the slider to adjust the probability threshold. Lowering the threshold increases recall (catching more potential cancer cases) at the cost of precision.

-


Breast-Cancer Proteomics Dashboard
AdvanceHC Solution | Healthcare AI Startup
Overview
Distributions
Performance
Feature Importance
About
Explainability – Feature Importance
Top-20 Feature Importances
ER Status_Negative
PR Status_Positive
Age at Initial Pathologic Diagnosis
AJCC Stage_Stage IIIB
AJCC Stage_Stage II
AJCC Stage_Stage IV
AJCC Stage_Stage IIIA
PR Status_Negative
Expression
HER2 Final Status_Equivocal
Feature
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Importance
variable
0
XGBoost's gain-based importance shows which proteins and clinical features most influence predictions, supporting transparent decision-making for clinicians.

-

References:

AWS Educate—Cloud Skills for Education- AWS. (n.d.). Retrieved March 9, 2025, from https://aws.amazon.com/education/awseducate/

Amazon Web Services. (n.d.). *Quickstart — Boto3 documentation*. Boto3. Retrieved March 23, 2025, from https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html#using-boto3

Breast Cancer Proteomes. (n.d.). Retrieved March 9, 2025, from https://www.kaggle.com/datasets/piotrgrabo/breastcancerproteomes

Tjoa, E., & Guan, C. (2020). A survey on explainable artificial intelligence (XAI): Toward medical XAI. *IEEE Transactions on Neural Networks and Learning Systems, 32*(11), 4793–4813. https://doi.org/10.1109/TNNLS.2020.2975744