Machine Learning Identification of Triple Negative Breast Cancers using Expression of Genes with Minimal Data Leakage

David Chen, Ph.D.

Project repo: `github.com/ydavidchen/machine-learning-id-of-tnbc`

May 28, 2021

1 Definition

1.1 Project Overview

Triple Negative Breast Cancer is the most aggressive breast cancer subtype. Such cancers have limited treatment options because they do not express ER (*ESR1* gene), PR (*PGR* gene), and HER2 (*ERBB2* gene) hormone receptors that can be targeted by endocrine drugs, and hence the "triple-negative" annotation. Early detection of the *triple-negative* cancer subtype is crucial to making timely clinical decisions.

Determining whether a cancer has is triple-negative requires effort from a domain-specific expert: a medically trained pathologist. The process requires additional time and human resource. More importantly, the assessment procedure by clinicians might not be consistent and reliable. In cancer diagnosis and other image , it is well-known that machine learning algorithms often outcompete human domain experts.

Machine learning has been increasingly used in healthcare. In cancer machine learning has been used to diagnose cancer and subtype diseases. The input for such classifiers can be very flexible, that is, the features can be derived from clinical information, images, or genomic profiles. Here, I am interested in leveraging machine learning to subtype all breast cancers into triple-negative (class 1) and non-triple negative (class 0).

Related works (refs.[1] and [2]) attempted to address the breast cancer subtyping problems using machine learning. Ref.[1] uses a regression approach to individually estimate ER, PR, and HER2 receptor status, which was not a direct approach to the problem and would require three separate classifiers followed by manual data wrangling to assign a triple-negative class label. Ref. [2] recently achieved impressive accuracy but rather low precision and recall. This result is not surprising because of the severe class imbalance: the triple-negative subtype comprise about only $10 - 15\%$ of all breast cancers.

In this project, I use a machine-learning approach to determine whether a breast cancer specimen has a triple-negative clinical diagnosis using its gene expression profile. Gene expression studies have become increasingly popular, and the high-dimensional nature of data represents the next generation of biomedicine. It is very crucial that the effort in machine learning and AI be under way to utilize gene expression data for scientifically and clinically relevant tasks, such

as identifying the triple-negative cancer subtype.

1.2 Problem Statement

The *primary goal* of this project is to build a binary classifier that identifies breast cancer samples (rows) that are "triple negative" (i.e. Label=1) using mRNA data of m genes (columns).

The **input** of the classifier is the gene expression of a biological sample from a woman diagnosed with a non-metastatic breast canccer. The original gene expression data has a high dimension ($\in \mathbb{R}^{m\times 1}, m > 20,000$), but will be reduced by L1/LASSO feature selection.

The **output** will be a $\mathbb{R}^{1\times n}$ array of class labels predicted by fitting the *trained model* (either the benchmark or the target model) to the gene features of an observation.

2 Methodology

2.1 Data Preprocessing

The Cancer Genome Atlas (TCGA) is a population-scale cancer study with over $1,100$ breast cancer patients. The majority (>90%) of these patients have mRNA data (features) and clinical data that contains columns needed to define the class label (see sections below) [3]. The dataset is freely available to the public for download from multiple Internet repositories. For this project, the mRNA features and clinical data (used to infer class labels 1 vs. 0) were downloaded from cBioPortal (`https://www.cbioportal.org/`.

2.1.1 Data Observations

The specific dataset is the TCGA Breast Cancer dataset. The original dataset has approximately $n = 1,100$ patients or rows. Observations with the following characteristics were excluded:

- were not from a female subject
- lacked gene expression data (features)
- had metastatic (non-primary) cancer
- lacked sufficient inforation to define class labels (discussed below)

2.1.2 Class Labels for Observations

Class labels are defined based on whether an observation has a "triple-negative" clinical diagnosis. That is, whether a pathologist evaluated a biospecimen, under the microscope, had detectable proteins of ER, PR, and HER2 receptors. The records are publicly available as a clinical meta data.

However, not all observations underwent the ER/PR/HER2 scoring process. Sometimes the observations were skipped, while other times they might be omitted due to data entry issues. However, the class label may still be definable.

The rules for defining the class label were as follows:

- If all 3 receptors' statuses are missing in an observation, we canot work with that observation which must be excluded.

- Observations with (2 missing + 1 negative), (1 missing + 2 negative) cannot be assigned a definitive abel and were also excluded
- If at least 1 receptor is positive, the class label would be 0 (i.e. not triple-negative).
- Scoring columns with values such as "equivocal" or "intermediate" are considered positive.

The final dataset population has $n = 999$ observations. For supervised machine learning, $100(10\%)$ of data was held out for final evaluation, while the remainder of $799(80\%)$ and $100(10\%)$ were used for training and validation, respectively.

2.1.3 Features

The number of columns (or genes) available is approximately $20,000$. Since the data available to download was already standard-normalized, feature values were used as-is without further preprocessing or adjustments.

A small fraction (1%) of genes have missing values due to issues such as technical problems with instrument measurement. I excluded the 1% gene features (columns)(see **Notebook 1** for implementation).

There is a conern regarding the high dimensionality of the feature space. To address this, I used **L1/LASSO feature selection** (implemented in sklearn's SelectKBest). to reduce the feature space down to $m = 33$ (see **Notebook 3** for detail).

Genes *ESR1*, *PGR1*, and *ERBB2* code for the ER, PR, and HER2 hormone receptors used to define the class label, respectively. Even though the gene expression values were NOT what the domain experts used to define ER, PR, and HER2 status (instead clinicans look under the microscope), the three genes removed from the feature space to *minimize data leakage* (see **Notebook 1**).

2.2 Classification and Hyperparameter Tuning

The programming environment of this project was AWS Sagemaker Notebook Instance, Conda Python 3.6. A personal Amazon AWS account was used. Models built including:

- PCA for data exploration (scikit-learn; **Notebook 2**)
- The benchmark SVM model with *scikit-learn SVM* to set a baseline performance (see **Notebook 4**)
- **XGBoost** with random-search hyperparameter tuning. Specifically, the hyperparameters maximum depth, number of boosted rounds, and early stopping rounds were optimized (see **Notebook 5**).

For the supervised models, I have addressed the severe class imbalance (rarity of the positive class) using **inverse class weights** (details can be found in **Notebooks 4 and 5**). The positive class here has a weight of 7.69.

For fair comparison of the target model (XGBoost) to the Benchmark (Gaussian SVM), I ensured variables such as the Test Set and inverse class-balance weight were identical between the modeling attempts.

2.3 Metrics

2.3.1 Benchmark Model

A recent work [2] showed that a Support Vector Machine (SVM) classifier This model achieved accuracy score (unweighted) of 0.9 but an F1 score of 0.67. The rather poor F1 score was expected since the triple-negative classification is rare and the class distribution was imbalanced. As a benchmark model, I built a Gaussian SVM. The optimal set of hyperparameters was selected using random search based on validation-set AUC.

2.3.2 Evaluation Metrics

In training the model, I used the area under the receiver operation curve (AUC) metric on the hold-out validation set (n=100, 10% of the data).

For final model evaluation, I focused on the macro-averaged F1 score, the harmonic mean between precision and recall (whose calculations were implemented in *sklearn.metrics* module).

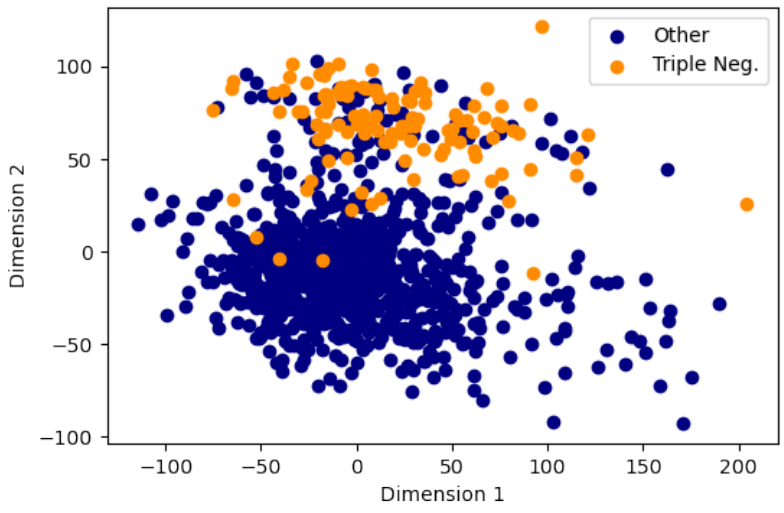
3 Analysis and Results

3.1 Exploratory Data Analysis

In a summary table, I overviewed the distribution of several key variables. Importantly I confirm all 999 observations had *informed consent*, an important criterion for performing any human subject research (**Notebook 1**).

Next, I performed **unsupervised learning** to determine whether there is any discernible patterns in the data features especially in relation to the class labels. To this end, I performed a two-dimensional **Principal Component Analysis (PCA)** and visualized the principal components on a 2D scatter plot, which was then color-coded by class label (**Figure 1**).

Figure 1: Two-dimensional PCA of the entire dataset. Individual data points were color-coded by class label. The code for this analysis is in **Notebook 2**.



As shown in (**Figure 1**), there were two predominant clusters in the dimensionality-reduced dataset. The triple-negative (class 1) data points located primarily in the upper cluster. This

result suggests that gene features at the **global level** are different between the classes. Without supervision, there is already a clearcut distinction between the classes, implying that the two classes could be separated when a *supervised* machine learning approach is used.

3.2 Benchmark Model: Gaussian SVM with Hyperparameter Optimization

The benchmark or baseline model is Gaussian SVM (i.e. RBF kernel). A Validation Set consisting of 10% of data was usedfor Random Search of hyperparameters C and γ (method ref. [4]). The best hyperparameter set was decided using up to 20 candidate models' Validation-Set AUC scores (**Table 1**).

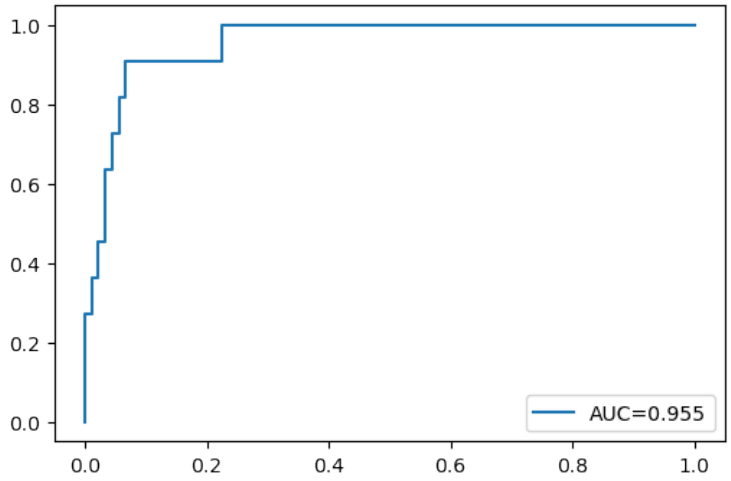
Table 1: Candidate SVM benchmark models built with hyperparameters from Random Search. Ranked by validation-set AUC. See **Notebook 4** for how the results were curated and exported as a CSV.

Candidate no.	Hyperparam C	Hyperparam gamma	Validation AUC
17	125.00	0.0040	0.9070
10	3.60	0.0183	0.8938
14	63.89	0.0264	0.8938
5	6.14	0.0073	0.8927
4	22.11	0.0491	0.8927
7	116.44	0.0284	0.8927
8	48.84	0.0271	0.8927
13	76.33	0.0546	0.8907
9	162.68	0.0449	0.8907
0	4.85	0.0468	0.8907
11	53.58	0.0586	0.8887
19	39.25	0.0436	0.8887
18	158.07	0.0341	0.8887
6	84.28	0.1222	0.8825
15	83.21	0.2210	0.8795
12	44.62	0.1438	0.8784
3	14.71	0.1930	0.8784
2	144.58	0.2097	0.8784
1	9.42	0.3500	0.8754
16	24.74	0.3498	0.8754

The best candidate achieved an validation-set AUC of 0.907 (**Table 1**). This candidate was chosen as the Benchmark for evaluation on the hold-out Test Set.

Applying the chosen benchmark model is applied to the hold-out test set (not exposed to training) yielded a macro-averaged F1 score of 0.81 (**Notebook 4 classification report**). **Figure 3** shows the ROC and AUC of the benchmark model on the Test Set.

Figure 2: ROC of the hyperparameter-optimized Benchmark, Gaussian SVM, applied to the hold-out Test Set.

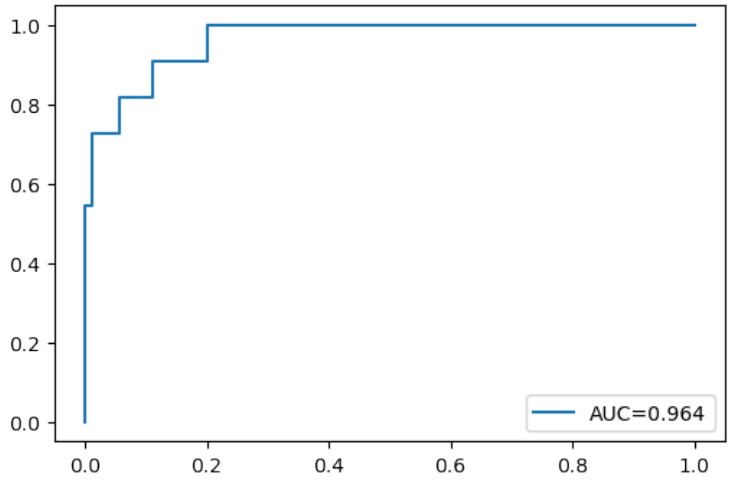


3.3 Target Model: XGBoost with Hyperparameter Tuning

Select the best model from each method/classifier, and apply to the hold-out test set. To prevent data leakage, the hold-out test set will only be used at the very end and not exposed to any classifiers during training. Metrics and plots mentioned in the **Evaluation Metrics** sections will be reported here as well.

For the final selected model (first row of **Table 2**), when applied to the hold-out test set, macro-averaged F1 score is also 0.81, same as the Benchmark (**Notebook 5 classification report**).

Figure 3: ROC of the hyperparameter-tuned XGBoost model candidate applied to the hold-out Test Set.



Importantly, the AUC value was *higher* for the XGBoost model than the benchmark SVM on

Table 2: Candidate XGBoost models built with hyperparameters from Random Search. Ranked by validation-set AUC. See **Notebook 5** for how the results were curated and exported as a CSV.

Candidate	eta	gamma	max_depth	min_child_weight	FinalObjectiveValue
9	0.363	9.778	10	8	0.9784
8	0.228	2.908	10	4	0.9762
7	0.453	1.255	9	4	0.9754
13	0.366	1.883	11	6	0.9747
19	0.225	5.206	7	2	0.9732
15	0.123	6.444	8	2	0.9732
6	0.292	9.079	3	7	0.9725
16	0.375	2.654	11	8	0.9717
12	0.162	4.036	8	6	0.9706
2	0.418	8.904	7	7	0.9688
5	0.341	9.249	9	8	0.9654
4	0.437	4.705	6	8	0.9635
0	0.487	6.377	8	4	0.9621
3	0.330	7.211	8	8	0.9617
10	0.196	4.590	5	3	0.9598
11	0.411	7.301	8	3	0.9591
18	0.430	7.749	10	6	0.9580
17	0.362	4.262	7	4	0.9568
14	0.369	1.313	11	4	0.9427
1	0.221	2.940	11	3	0.9412

the Test Set (which was identical between the XGBoost and SVM modeling runs because the same *random seed* value was set): 0.964 vs. 0.955, respectively.

4 Conclusions

In this healthcare-related project, I used expression profiles of genes to predict triple-negative breast cancer subtype, a classification that currently requires input of a human domain expert with clinical training. The high-dimensional nature of the feature space makes the problem challenging. However, principal components analysis (PCA) suggests the overall goal is promising to solve with machine learning. An L1/LASSO feature selection approach reduced the feature space to only 33 genes.

The project involved trickly label definition. The class label requires three metadata columns with assessments from a human domain expert. As we have seen in this dataset, the metadata columns often have missing values which can make training examples scarce and complicate the class label definition process.

The latest work in the subject [2] did not exclude the ER, PR, and HER2 gene expression, which is correlated with their respective protein expression and clinical diagnosis, thus suffer from the data leakage problem. Here, I addressed the issue by excluding these three genes from the feature space, thereby making the problem more challenging yet interesting, and the machine-learning solutions presented here thus have greater value.

Whenscikit-learn Gaussian SVM was used as the benchmark, I was able to achieve the generally high classification performance reported in liberature [2]. However, using the presumably more robust, complex, and ensemble learning-based XGBoost implemented in AWS Sagemaker, **I achieved comparable F1 score and suprior AUC values than the benchmark**. It is important to note, however, that the optimization objective I chose was *maximizing the validation AUC*. In theory, I might be able to get higher F1 score (but suffer from lower AUC) had I choose F1 score as the objective metric. Nevertheless, my work here highlights the value of XGBoost, given that the benchmark/baseline model was already well-performing (AUC>0.95).

References

1. Brian D Lehmann, Joshua A Bauer, Xi Chen, Melinda E Sanders, A Bapsi Chakravarthy, Yu Shyr, Jennifer A Pietenpol, et al. Identification of human triple-negative breast cancer subtypes and preclinical models for selection of targeted therapies. *The Journal of clinical investigation*, 121(7):2750–2767, 2011.

2. Jiande Wu and Chindo Hicks. Breast cancer type classification using machine learning. *Journal of Personalized Medicine*, 11(2):61, 2021.

3. Cancer Genome Atlas Network et al. Comprehensive molecular portraits of human breast tumours. *Nature*, 490(7418):61, 2012.

4. User afsharov. Using hold-out-set for validation in randomizedsearchcv in scikit-learn? 2020.