An Assessment of Machine Learning Classifiers for Breast Cancer Subtype Prediction

Jenea Adams, Daniel Hui, Jacob Leiby, Vivek Sriram

1. Abstract

Breast cancer is the most common cancer in women worldwide and causes over 500,000 deaths per year. Breast cancers can be categorized into immunohistochemistry subtypes based on cell surface proteins. However, IHC subtyping is prone to error. Recent studies have shown that gene expression data from tumors can be used to categorize the subtypes more effectively and consistently. Here, we perform IHC subtype classification with gene expression data from TCGA BRCA cohort using regularized logistic regression, random forests, support vector machines, gradient boosting, and neural networks. We found that L1 regularized logistic regression shows the best performance.

2. Motivation

Breast cancer is the most common cancer in women worldwide with a lifetime prevalence of one in eight, and causes more than 500,000 deaths worldwide per year [https://www.bigagainstbreastcancer.org/why-research]. Strategies for effective treatment may differ greatly depending on the subtype of the cancer determined by immunohistochemistry (IHC).

The cancer IHC subtypes include: luminal A (ER- and/or PR-positive, HER2-negative), luminal B (ER- and/or PR-positive, HER2-positive), HER2-enriched (HER2 positive, ER- and PR-negative), and triple-negative (ER-, PR-, and HER2-negative).[1] The subtypes are traditionally determined using IHC-based methods, however recent studies have shown that there may be discordance in subtype classifications due to intratumoral heterogeneity and/or measurement inaccuracies in subtype profilers.[2] Inaccuracies may also arise when a receptor expression level is on the boundary of positive and negative.

With the decreasing cost of high-throughput mRNA sequencing technologies, prediction of IHC subtypes using RNA expression profiles of the tumor will become more clinically relevant. In

this project we aim to predict IHC cancer subtypes with gene expression data using multiple machine learning techniques to compare performance.

3. Dataset

The initial dataset included 1,068 breast cancer patients with quantified gene expression for 20,532 genes using mRNA sequencing. The expression data was quantified using RSEM and normalized relative to the 75th percentile with a x1000 adjustment factor prior to acquiring it. The clinical data provides ER, PR, and HER2 status (positive/negative) and was used to create the outcome variable of 4 unique breast cancer IHC subtypes explained above. Data were acquired from The Cancer Genome Atlas (TCGA) program BRCA cohort [https://www.cancer.gov/tcga] on November 13, 2020 via FireBrowser [http://firebrowse.org/].

4. Related Work

Employing machine learning methods for breast cancer subtype classification is a well-supported area of research in current literature. As previously mentioned, Yoon et al revealed the competence of machine learning methods for gene-expression based breast cancer subtype classification as mitigation to discrepancies in measurement or experimental biomarker profiles.[2] While our study does not include non-parametric approaches, Amrane et al compare non-parametric k-nearest neighbors and Naive Bayes classifiers for the binary classification of benign or malignant tumors in the Wisconsin Breast cancer Database with favorable accuracies.[3] Yue et al expands on the usage of this particular dataset from the perspective of several machine learning approaches, including support vector machines, decision trees, and varying neural network architectures.[4] Each of the what the aforementioned papers have in common is a sample size of < 700 and binary classification outcomes, much smaller than what Cascianelli et al greatly improves on with 4,731 breast cancer gene expression profiles, multi-class outcomes, and more closely mirrors the analysis presented in our study.[5] This study employs a multiclass decision forest with bagging, a multiclass decision jungle, multiclass logistic regression with Lasso and Ridge regularizations, a fully-connected feed-forward neural network, and finally a multiclass support vector machine with Lasso regularization. The authors showed their multiclass logistic regression model performed the best on their test set after 10-fold cross validation for classifying the breast cancer subtypes of a TCGA dataset. Collectively, these studies provide a crucial framework for further exploring machine learning approaches to larger datasets.

5. Problem Formulation

We framed our project into a classification task – given gene expression data, we wanted to group breast cancer samples into non-overlapping IHC subtypes. Our data included information on 20,532 variables for 1,068 observations (which came normalized and batch-corrected) prior to pre-processing. One of each pair of columns corresponding to correlated genes (Pearson R-squared >.95) were removed, as highly correlated features may be mostly redundant and the standard errors of their coefficients will be greatly increased for logistic regression. Low-variance genes were dropped from the data, as small differences within a low variance feature may be greatly inflated after scaling. Finally we scaled all features to mean 0 and variance 1, so that features would be given equal weighting for methods and regularization schemes that are not scale invariant. From here, we implemented five different machine learning models in an effort to classify each observation – logistic regression, random forest, support vector machine, gradient boosting, and neural networks. Logistic regression was chosen because it was a solid baseline model for classification; however it can only model linear relationships (excluding nonlinear transformations to input features), may be sensitive to outliers, and multicollinearity can greatly affect interpretation of coefficients. Random forest was chosen because it is a versatile algorithm that allows users to ignore features that are unimportant, and it adds additional randomness to the classifier without the need for hyperparameter tuning. Support vector machines were used because they are able to implement nonlinear kernels for the calculation of boundaries with varying topologies. Gradient boosting was implemented because it is a strong ensemble method that could be compared to random forests. Neural networks were used because they can learn non-linear and complex relationships, and also generalize to infer unseen relationships. We chose test accuracy as the metric for evaluation of each of our models because it gave us a simple way to specifically determine specifically how often each class of models predicted the label of an observation correctly.

6. Methods

We decided to evaluate five different methods (logistic regression, random forests, support vector machines, gradient boosting, and neural networks) in terms of their average accuracy performance in classifying breast cancer subtypes.

6.1 Data Pre-Processing

After assigning the IHC subtype categories, 924 samples remained due to missing clinical data. Genes were first filtered on whether expression was measured in at least half of the samples, leaving 17,642 genes. One of each pair of genes with Pearson R-squared greater than .95 (N=2), and genes with variance less than .1 were removed (N=16), leaving 17,624 genes for analysis. Expression values were scaled to mean 0 and variance 1.

5-fold cross-validation was applied to analyze the performance of each machine learning method. The data were split into 70% training, 15% validation, 15% testing datasets. The validation data was used to optimize hyperparameters and the testing data used for final performance measure. Each of the data sets in every fold are the same for each method.

To first explore and visualize the data we used the dimensionality reduction techniques principal component analysis (PCA) and uniform manifold approximation and projection (UMAP) on the processed dataset.

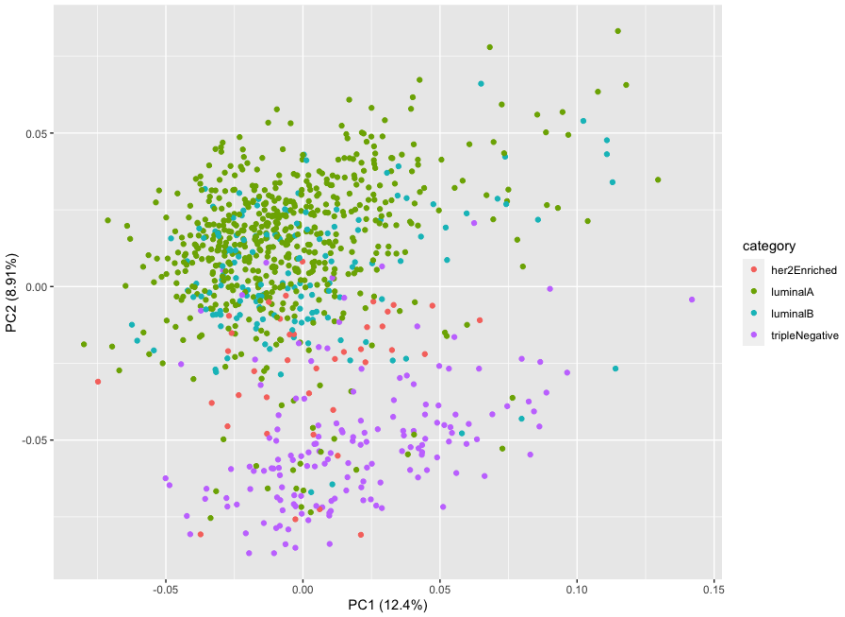

Figure 6.1.1. PCA projection of the gene expression data.

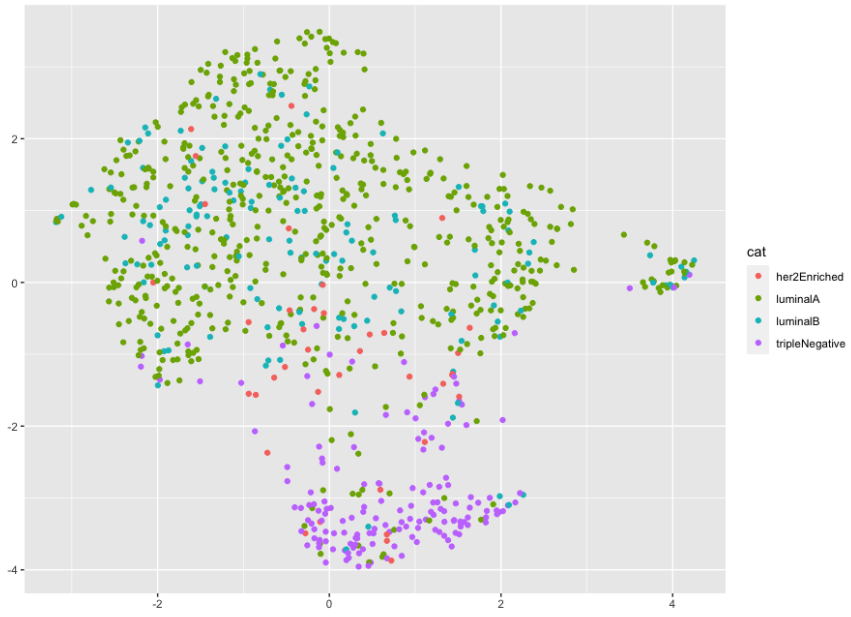

Figure 6.1.2. UMAP projection of the gene expression data.

We can see from these visualizations that the triple negative samples appear to cluster well separately from the other subtypes. Luminal A dominates the other major cluster, with luminal B, and Her2 enriched samples mixed in.

6.2 Logistic Regression

A scikit-learn multi-class logistic regression model was built as a baseline model for accuracy over the 4 classes. Two-dimensional grid search was performed for the regularization penalty and the inverse regularization strength "C" hyperparameters. The potential regularization parameters were L2 or L1, specifying Ridge and Lasso regression, respectively. The "C" parameter represents inverse regularization strength, where higher values yield less regularization.

Grid search for optimal hyperparameter values which yielded the highest validation accuracies was performed, with possible C values of 0.001, 0.01, 0.1, 0.3, 0.5, 1.0, 1.3, 1.5, 3.0, and 5.0. The possible penalty types were L1 and L2. If L1 regularization was applied, the "liblinear" solver was specified to employ coordinate descent. If L2 regularization was applied, the model was evaluated using multinomial loss with a limited-memory Broyden–Fletcher–Goldfarb–Shanno

optimizer ("lbfgs") with a maximum number of iterations of 10,000 as a precaution, and over 4 CPU cores.

All analysis for this method was conducted on a University of Pennsylvania high-performance cluster.

6.3 Random Forest

Random forest modeling was conducted using R package "randomForest" version 4.6-14 and R version 3.6.3. The number of variables used in each tree (parameter `mtry`) was changed from 10, 50, 132 (default, which is sqrt(number of total features)), 500, 1000, and 2500. The number of trees was left at the default value of 500, all other parameters were also left as default -- to alleviate computational burden, models were all trained in parallel on one of University of Pennsylvania's high-performance clusters.

6.4 SVM

The scikit-learn "SVC" function was used to implement support vector machine classification on our data. A two-dimensional grid search was performed for two hyperparameters – the "kernel" and "C."

The kernel is a function that defines the transformation used on the input data, allowing users to determine a reasonable hyperplane even if the data are not initially linearly separable. "C" is the regularization parameter, which allows us to decide how strict we want to be with our hyperplane – the higher the value of C, the fewer misclassifications are permitted for our separating plane.

Through a trial-and-error approach that balanced a spread of options for hyperparameters, reasonable runtime of classifiers, and sufficient specificity in terms of accuracy variation, the following values were chosen for grid search of C – 0.005, 0.01, 0.1, 1, 1.25, 1.5, 1.75, and 2. For kernel options, we tested the linear kernel, the third-degree polynomial kernel, the gaussian radial basis function (RBF) kernel, and the sigmoid kernel. Default values were used for all other parameters.

6.5 Gradient Boosting

The scikit-learn “GradientBoostingClassifier” function was used to implement gradient boosting on our data. Due to the extended runtime of this function given our data, only default settings were used, and no hyperparameter search was performed.

6.6 Neural Net

The neural network used in this study was a multilayer perceptron. The model was implemented using PyTorch. The loss function used was cross entropy loss (combining log softmax and negative log-likelihood in the PyTorch implementation), and the parameters were optimized using the Adam optimizer. Adam is an adaptive learning rate optimization algorithm. In this experiment, the initial learning rate and the weight decay (L2-penalty) on the parameters.

A subset of the data was used to first optimize the architecture of the model. After testing different architectures over a range of hidden layers and hiddens nodes in each layer, the final architecture for the neural net was two hidden layers each containing 200 hidden nodes, with an output layer of four nodes. Dropout between layers was tested, however did not show improvement of model performance and was not included in the final model. Analyzing the training and validation loss curves over many (1000+) epochs showed that after roughly 100 epochs, validation loss did not improve. Therefore, the overall models were trained for 100 epochs in both hyperparameter optimization and the final retraining of the model. Finally, through trial-and-error, grids were selected for the initial learning rate and weight decay for Adam, LR: [5e-4, 1e-4, 5e-5, 1e-5]; WD: [0.15, 0.175, 0.2, 0.225]. Optimal hyperparameter pairs were selected using the training and validation data, where the optimal parameters were those associated with the smallest validation loss, and then used to retrain a final model which was then used for the test performance measurement.

7. Results

7.1 Logistic Regression

	Chosen Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy
Split 1	Penalty = L1; C = 0.10	0.9876	0.9137	0.9568
Split 2	Penalty = L1; C = 0.10	0.9876	0.8777	0.9290
Split 3	Penalty = L1; C = 0.30	0.9861	0.9496	0.8921
Split 4	Penalty = L1; C = 0.50	0.9861	0.9137	0.9065
Split 5	Penalty = L1; C = 0.30	0.9845	0.9209	0.8993
	Mean	0.9864	0.9151	0.9167
	Std Dev	0.0013	0.0256	0.0263

Table 7.1.1: Logistic Regression accuracy results after hyperparameter grid search

Table 7.1.1 demonstrates accuracy results from the logistic regression model in the training, validation, and testing splits. Hyperparameter grid search revealed that L1 regularization performed the best in all 5 cross validation folds, with varying, but relatively similar values of C for each fold. Subsequently, these values for regularization and their associated strengths yield a mean validation accuracy of 91.5% and a mean testing accuracy of 91.75 (SD = 0.0256, 0.0263, respectively). Overall the model performed quite well with an inverse regularization strength parameter value between 0.10 and and 0.50, which is lower and more strict than scikit-learn's default parameter values of C=1.0 and L2 penalty.

	Her2 enriched	Luminal A	Luminal B	Triple Negative
Split 1	1.000	0.960	0.950	1.000
Split 2	1.000	0.960	0.850	0.960
Split 3	0.000	0.930	0.720	0.880
Split 4	1.000	0.920	0.760	0.920
Split 5	1.000	0.920	0.760	0.920
Mean	0.800	0.938	0.808	0.936

Table 7.1.2: Logistic Regression accuracy by class for the final model

Further analysis of the per-class accuracy of the final model on the test set revealed favorable performance. The Her2 enriched class, which was the smallest sample size of each split, had the lowest performance at 80% with the highest performing class being Luminal A. This result certainly supports preliminary observations of the data in the dimensionality reduction

representations in Figure 6.1.1 and Figure 6.1.2 -- Luminal A and B were more similar and clustered together, whereas the Her2 enriched class did not have a particularly distinct clustering behavior. With similar gene expression signatures, Luminal A (the most samples in each split) and B present themselves as more similar and harder to classify, which may explain Luminal B's second-lowest accuracy score. Lastly, the Triple Negative class did, in fact, exhibit distinct clustering behavior apart from the other three classes, which may explain its second-highest accuracy score. Overall, each of the classes demonstrated favorable accuracy performance compared to the other methods in our analysis.

7.2 Random Forest

We observed a monotonic increase in the mean validation set accuracy as the number of variables in each tree increased. This increase continued well above the default value (sqrt(total number of features), 132 for our dataset), implying that for this data the default parameters may not be suitable if maximal accuracy is desired (at the expense of longer runtimes).

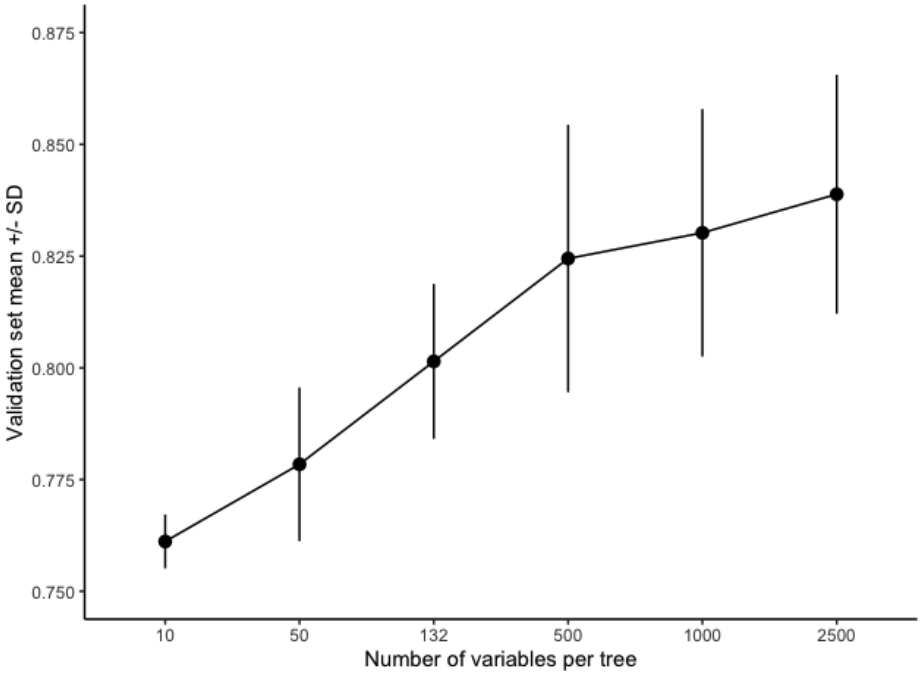

Figure 7.2.1: Number of variables per tree (`mtry` parameter) as a function of mean validation set accuracy (+/- 1 SD). Note 132 variables per tree is the default (sqrt(total number of features)).

After seeing that `mtry` of 2,500 performed the best in the validation sets across all splits, we observed that performance in the test set had similar performance, with only a .006 decrease in mean accuracy compared to the validation set. Despite the training accuracies all being 1, which may imply the models are overfit, they still seemed to generalize well to data they were not trained on.

Split	Training Accuracy	Validation Accuracy	Testing Accuracy
1	1	0.863	0.827
2	1	0.799	0.856
3	1	0.863	0.813
4	1	0.835	0.827
5	1	0.835	0.842
Mean	1	0.839	0.833
SD	0	0.0267	0.0164

Table 7.2.1: Each split's training, validation, and testing accuracy for the hyperparameters with maximal validation accuracy (all were with `mtry` of 2500).

Using the final models on the test set, we observed that the Luminal A and Triple Negative had the highest classification accuracies (.968 and .974, respectively), which were well above the HER-2 enriched and Luminal B accuracies (.700 and .709, respectively). It may be that, since Luminal A has many more instances it is easier to classify -- as for Triple Negative, it is clearly separated from the other classes per the PCA and UMAP plots.

Split	HER-2 enriched	Luminal A	Luminal B	Triple Negative
1	1.00 (6/6)	.955 (84/88)	.773 (17/22)	1.00 (23/23)
2	.500 (3/6)	.989 (86/87)	.909 (20/22)	1.00 (24/24)

3	.667 (4/6)	.977 (86/88)	.773 (17/22)	1.00 (23/23)
4	.833 (5/6)	.977 (86/88)	.636 (14/22)	.957 (22/23)
5	.500 (3/6)	.943 (83/88)	.455 (10/22)	.913 (21/23)
Total accuracy	.700 (21/30)	.968 (425/439)	.709 (78/110)	.974 (113/116)

Table 7.2.2: Per-class accuracy in the test set of the models that performed best on the validation set (all had `mtry` values of 2,500).

7.3 SVM

We see that varying our kernel choice and the value of our regularization yields different results depending on the split of our data. We end up with a mean testing accuracy of 0.797 and a standard deviation for this accuracy of 0.0156. In general, it appears that the use of the sigmoid kernel seems most appropriate for these data.

	Chosen parameters	Training Accuracy	Validation Accuracy	Testing Accuracy
Split 1	Kernel = Sigmoid; C = 1	0.831	0.806	0.806
Split 2	Kernel = Sigmoid; C = 1.5	0.864	0.791	0.806
Split 3	Kernel = RBF; C = 1.75	0.983	0.842	0.770
Split 4	Kernel = Sigmoid; C = 1.25	0.844	0.813	0.806
Split 5	Kernel = Linear; C = 0.005	1.00	0.799	0.799
		Mean = 0.904	Mean = 0.810	Mean = 0.797
		Std Dev = 0.0806	Std Dev = 0.0194	Std Dev = 0.0156

Table 7.3.1: SVM accuracy results after hyperparameter grid search

We also note that depending on the number of observations we have in each subclass of breast cancer, the accuracy of the SVM classifier differs drastically. For the Luminal A category where each split has roughly 88 observations, our total testing accuracy across all splits comes out to 0.936, while for the Her-2 enriched category, which has just six observations per split, the total testing accuracy comes out to 0.267.

	HER-2 enriched	Luminal A	Luminal B	Triple Negative	All classes
Split 1 Test Accuracy	0.500 (3/6)	0.965 (83/88)	0.273 (6/22)	0.870 (20/23)	0.806 (112/139)
Split 2 Test Accuracy	0.167 (1/6)	0.920 (80/87)	0.500 (11/22)	0.833 (20/24)	0.806 (112/139)
Split 3 Test Accuracy	0.000 (0/6)	0.943 (83/88)	0.136 (3/22)	0.913 (21/23)	0.770 (107/139)
Split 4 Test Accuracy	0.333 (2/6)	0.966 (85/88)	0.227 (5/22)	0.870 (20/23)	0.806 (112/139)
Split 5 Test Accuracy	0.333 (2/6)	0.909 (80/88)	0.364 (8/22)	0.913 (21/23)	0.799 (111/139)
Total Accuracy	0.267 (8/30)	0.936 (411/439)	0.300 (33/110)	0.879 (102/116)	0.797 (554/695)

Table 7.3.2: SVM test accuracies by class after hyperparameter grid search

7.4 Gradient Boosting

We end up with a mean testing accuracy of 0.809 and a standard deviation for these accuracies of 0.0225.

	Training Accuracy	Testing Accuracy
Split 1	1.0	0.799
Split 2	1.0	0.842
Split 3	1.0	0.784
Split 4	1.0	0.799
Split 5	1.0	0.820
	Mean = 1.0	Mean = 0.809
	Std Dev = 0.0	Std Dev = 0.0225

Table 7.4.1: GB accuracy results after hyperparameter grid search

Yet again, we can see that the number of instances of each subclass of breast cancer affects the accuracy of our classifier. For the Luminal A category where each split has roughly 88 observations, our total testing accuracy across all splits comes out to 0.938, while for the Her-2

enriched category, which has just six observations per split, the total testing accuracy comes out to 0.233.

	HER-2 enriched	Luminal A	Luminal B	Triple Negative	All classes
Split 1 Test Accuracy	0.667 (4/6)	0.909 (80/88)	0.318 (7/22)	0.870 (20/23)	0.799 (111/139)
Split 2 Test Accuracy	0.000 (0/6)	0.943 (82/87)	0.636 (14/22)	0.875 (21/24)	0.842 (117/139)
Split 3 Test Accuracy	0.167 (1/6)	0.932 (82/88)	0.227 (5/22)	0.913 (21/23)	0.784 (109/139)
Split 4 Test Accuracy	0.167 (1/6)	0.9773 (86/88)	0.227 (5/22)	0.826 (19/22)	0.799 (111/139)
Split 5 Test Accuracy	0.167 (1/6)	0.932 (82/88)	0.500 (11/22)	0.870 (20/23)	0.820 (114/139)
Total	0.233 (7/30)	0.938 (412/439)	0.382 (42/110)	0.863 (101/117)	0.809 (562/695)

Table 7.4.2: GB test accuracies by class after hyperparameter grid search

7.5 Neural Net

The overall performance of this neural net architecture given the hyperparameters is 0.796 with a standard deviation of 0.0174.

	Chosen parameters	Training Accuracy	Validation Accuracy	Testing Accuracy
Split 1	Learning rate: 0.0005; weight decay: 0.225	0.989	0.789	0.791
Split 2	Learning rate: 0.0001; weight decay: 0.175	0.975	0.770	0.806
Split 3	Learning rate: 0.0005; Weight decay: 0.15	0.997	0.834	0.777
Split 4	Learning rate: 0.0001; Weight decay: 0.175	0.989	0.813	0.784
Split 5	Learning rate: 0.0001; Weight decay: 0.175	0.984	0.806	0.820
Mean		0.987	0.802	0.796
Std Dev		0.00807	0.0242	0.0174

Table 7.5.1. Neural net optimal hyperparameters and final model overall accuracies.

In each fold of the data, the Luminal A and Triple negative subtypes have the highest accuracy whereas Her2 enriched and Luminal B subtypes have poor accuracy. Her2 enriched performance is likely due to the fact that there are only six samples in each data split. The Luminal B performance is likely due to the similarity to Luminal A in expression patterns, and the fact that there are four times as many Luminal A samples as there are Luminal B.

	Her2 enriched	Luminal A	Luminal B	Triple negative
Split 1	0.667	0.83	0.454	1.00
Split 2	0.167	0.919	0.454	0.875
Split 3	0.167	0.886	0.273	0.913
Split 4	0.333	0.943	0.182	0.870
Split 5	0.167	0.920	0.410	0.956

Table 7.5.2. Per class test accuracies from the final models.

7.6 Overall Accuracy

Our final test accuracies, by class and in total, are listed in the following table. Best performance in each column is included in bold font.

	HER-2 enriched	Luminal A	Luminal B	Triple Negative	Mean Total Test Accuracy
Logistic Regression	**0.800**	0.938	**0.808**	0.936	**0.917**
Random Forest	0.700	**0.968**	0.709	**0.974**	0.833
SVM	0.267	0.936	0.300	0.879	0.797
Gradient Boosting	0.233	0.938	0.382	0.863	0.809
Neural Net	0.302	0.900	0.355	0.923	0.796

Table 7.6.1. Summary results across machine learners

8. Conclusion and Discussion

We can see from our results that logistic regression and random forest work the best in terms of testing accuracies out of our classifiers. Notably, random forest has the highest test accuracy for the Luminal A and Triple Negative groups, which have the clearest distributions in the PCA and UMAP plots earlier on. However, overall logistic regression works the best in its total test accuracy, out-performing other classifiers in the HER-2 enriched group as well as the Luminal B group. This result corroborates with the conclusions of Yoon et al. 2020, which found logistic regression outperformed neural networks, SVMs, and random forests in subtyping of breast cancer tissues. One potential reason for why logistic regression outperforms other classifiers is that in truth there may only be a relatively small number of genes responsible for the majority of the differences between classes, and that they also linearly separate the data -- due to the very large number of genes, the other models may be struggling with overfit. It may be possible to

investigate this possibility by looking at the number of genes present in the final L1 regularized logistic regression model, as well as the distribution of their effects. As random forest generally seemed to perform second best, it may also be possible to investigate feature importance and observe how many features seem to be responsible for the majority of discriminatory signal.

From this project we learned two important things: hyperparameter tuning is time intensive and it is difficult to divide data into efficient training, validation, and testing datasets with a small sample size and class imbalance. To overcome this first issue, we found it was often best to spend time through trial-and-error to find appropriate architectures for the models and appropriate hyperparameter grid spaces to search. The second issue was difficult to overcome. We decided that a 5-fold cross validation scheme including training, validation, and testing sets in each fold would best suit our model comparisons with the given data. Ideally, we would have a larger dataset and a separate testing cohort to see how well our models generalize.

To further this project, we could explore how each of the models perform on selected or embedded features instead of using the entire gene expression space as input. This transformation could be done through PCA or feature embedding with an autoencoder. Biologically, we could use our gene expression based IHC subtypes to perform survival analyses and compare them to the traditional IHC subtype classifications to see if sequencing based subtyping is more coherent.

9. Model walk-through

For the model walk-through, refer to the neural net code file included:
CIS520_final_neural_net.ipynb

10. References

1. Sohn, Y. M., Han, K., & Seo, M. (2016). Immunohistochemical Subtypes of Breast Cancer: Correlation with Clinicopathological and Radiological Factors. *Iranian journal of radiology : a quarterly journal published by the Iranian Radiological Society*, *13*(4), e31386. https://doi.org/10.5812/iranjradiol.31386
2. Yoon, S., Won, H. S., Kang, K., Qu, K., Park, W. J., & Koo, Y. H. (2020). Hormone Receptor-Status Prediction in Breast Cancer Using Gene Expression Profiles and Their Macroscopic Landscape. *Cancers*, *12*(5), 1165. 10.3390/cancers12051165

3. M. Amrane, S. Oukid, I. Gagaoua and T. Ensarİ, "Breast cancer classification using machine learning," 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT), Istanbul, 2018, pp. 1-4, doi: https:/doi.org/10.1109/EBBT.2018.8391453.
4. Yue, W.; Wang, Z.; Chen, H.; Payne, A.; Liu, X. Machine Learning with Applications in Breast Cancer Diagnosis and Prognosis. Designs 2018, 2, 13. https://www.mdpi.com/2411-9660/2/2/13
5. Cascianelli, S., Molineris, I., Isella, C. et al. Machine learning for RNA sequencing-based intrinsic subtyping of breast cancer. Sci Rep 10, 14071 (2020). https://doi.org/10.1038/s41598-020-70832-2