American University of Beirut

Faculty of Arts and Sciences

Department of Computer Science

Breast Cancer Classification using Machine Learning Models

CMPS 261: Final Project Report

Maurice Salameh - 202211850
Osama Iskandarani - 202205878
Iyad Al Arab - 202203169

April 2024

Table of Contents

0.1 Abstract

This project aims to address the classification of breast cancer types among women, focusing on distinguishing between four distinct types of breast cancer. The dataset comprises 1496 instances, each characterized by 688 features, including clinical and genomic data about patients with breast cancer.

This report outlines the initial data processing procedures undertaken, elucidating the rationale behind these steps. The primary objective is to ensure that the dataset is appropriately prepared for subsequent analysis.

The workflow adopted in this project follows a structured approach. Initially, emphasis is placed on data preprocessing to address issues such as imbalanced data, feature scaling, and categorical encoding. This step is crucial to enhance the quality and reliability of the dataset.

Subsequently, various machine learning classifiers are trained using the preprocessed training data. Evaluation metrics such as accuracy, f-score, and recall are utilized to assess the efficacy of each classifier. The final stage of the workflow involves testing the chosen classifier using cross-validation. This testing methodology ensures the robustness and generalization capability of the selected model.

By adhering to this systematic approach, the project endeavors to develop a robust and accurate classifier for the classification of breast cancer types.

0.2 Introduction

Breast cancer is a disease in which abnormal breast cells grow out of control and form tumors. If left unchecked, the tumors can spread throughout the body and become fatal (World Health Organization, March 2024). It stands as the most prevalent form of cancer among women worldwide, affecting a substantial number of individuals annually, with 2.1 million cases recorded each year. In 2022, there were 2.3 million women diagnosed with breast cancer and 670,000 deaths globally (W.H.O). Till now, it has led to the highest number of cancer-related fatalities among women, underscoring the urgent need for accurate classification methodologies to facilitate early detection and targeted treatment strategies.

"Symptoms of breast cancer can include a breast lump or thickening, often without pain change in size, shape or appearance of the breast dimpling, redness, pitting, bloody fluid from the nipple or other changes in the skin"(W.H.O). Invasive types of cancers can spread to nearby lymph nodes or other organs, which makes them life-threatening. Therefore, treatments for breast cancer depend on the subtype of cancer and how much it has spread outside of the breast to lymph nodes (stages II or III) or other parts of the body (stage IV), and it includes removing the breast tumor, radiation therapy, and medications.

Hence, the problem at hand is breast cancer classification: given data from a patient already afflicted with breast cancer, the primary objective is to classify the type of breast cancer by identifying potential biomarkers specific to each cancer type such as cellularity, tumor size, hormones-receptors and genes.

This report discusses the approach to solving this classification problem, including the model architecture, training procedure, and evaluation metrics. It discusses the performance of the model and provides insights into the classification problem based on our results

0.3 Dataset

The initial dataset has the following distribution:

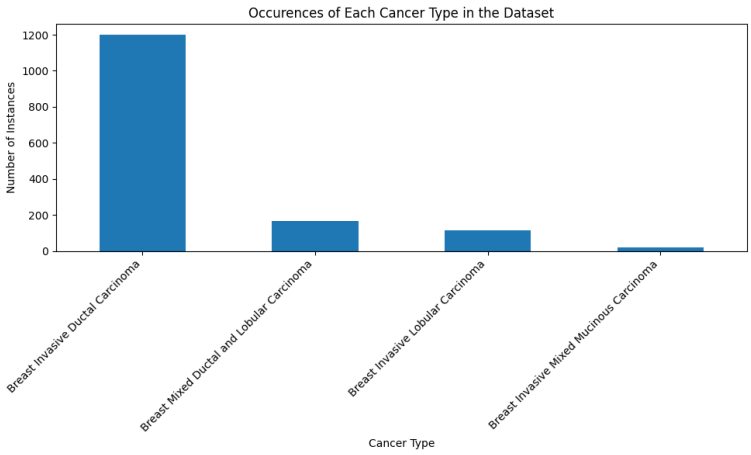


Figure 1: data distribution according to cancer type

Most breast cancers are invasive, meaning the cancer has spread from the original site to other areas, like nearby breast tissue, lymph nodes, or elsewhere in the body; this explains the dominance of the invasive type in the dataset.

The most common type of breast cancer — accounting for roughly 70% to 80% of all cases — is called invasive ductal carcinoma (IDC). Invasive lobular carcinoma (ILC) is the second most common type, accounting for roughly 5% to 10% of all breast cancers. The substantial variation in the incidence rates of each subtype of the disease, a frequent observation in medical contexts, is manifested as an imbalance in the distribution of the given dataset.

Another primary challenge within the dataset is the presence of both categorical and non-categorical features. Addressing this issue we opted to re-label the categorical data in-place (encoding in the same column without creating new features) using LabelEncoder(), and used pd.getDummies() (createds a new feature for each unique value in the original feature) to encode the rest non-categorical features.

Notably, the dataset comprises numerous non-categorical values across its features. Employing get_dummies, which is similar to one-hot encoding would substantially amplify the dimensionality of the feature space approx. 15 times.

Additionally, To understand the difference between categorical and non-categorical features, We

must observe certain features in the dataset. "Cellularity" is a prominent example of a categorical feature that is denoted by terms like "low" "medium", and "high" which embody a hierarchical structure. This better captured by label encoding, However, other columns are non-categorical features such as the gene mutations that have substantially more options that are unique to each mutation location in the cell

Thus, throughout the pre-processing part, we begin by dropping the duplicates from our data Pandas data frame to ensure that the data isn't distorted by repeated records.

Then, we fill in any missing values in numeric columns with their respective mean value to maintains data integrity and improves model performance by providing complete data.

Later on, we go into re-labeling our columns in order to be able to train the data models. We do so via the LabelEncoder function for the categorical columns and the get_dummies method for the non-categorical columns. The cancer types are encoded as the following:

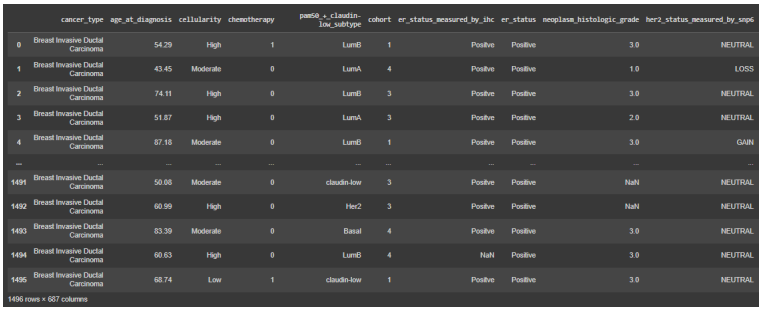
'Breast Invasive Ductal Carcinoma': 0
'Breast Invasive Lobular Carcinoma': 1
'Breast Mixed Ductal and Lobular Carcinoma': 2
'Breast Invasive Mixed Mucinous Carcinoma': 3

After that, we worked on removing the anomalies / outliers using the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) clustering algorithm which can identify outliers as noise points based on the density of data points.

We then split the data, 80 per cent to the train data and 20 per cent to the test data, using the train_test_split method, separating the dataframes into x_train, x_test, y_train, y_test, of which we will scale our x_train and x_test data using the MinMaxScaler function by fitting using the x_train. Reasons for using this scaler function include data normalization, distribution preservation, and outliers sensitivity.

Furthermore, we had a huge data imbalance between the different cancer types, where most of the data referred to one major class type. To address this issue, an up-sampling technique known as Synthetic Minority Over-sampling Technique (SMOTE) is employed to rectify the data imbalance. This approach has demonstrated superiority over down-sampling methods, particularly considering the constraints imposed by the dataset's size (1496 instances). Hence, prioritizing data balance over data loss is preferable within this context.

For visualization purposes, we utilized the Dimensionality Reduction Linear Discriminant Analysis (LDA) algorithm on our scaled data, and visualized the reduced datasets as clusters, each referring to its own cancer type. We used LDA for several reasons, mainly for dimensionality reduction, maximizing class separability, preprocessing for classifiers, and noise reduction. We also used it since it helps visualize high dimensionality data, of which we have tens of thousands of features after re-labeling.



	cancer_type	age_at_diagnosis	cellularity	chemotherapy	pam50_+_claudin-low_subtype	cohort	er_status_measured_by_ihc	er_status	neoplasm_histologic_grade	her2_status_measured_by_snp6
0	Breast Invasive Ductal Carcinoma	54.29	High	1	LumB	1	Positive	Positive	3.0	NEUTRAL
1	Breast Invasive Ductal Carcinoma	43.45	Moderate	0	LumA	4	Positive	Positive	1.0	LOSS
2	Breast Invasive Ductal Carcinoma	74.11	High	0	LumB	3	Positive	Positive	3.0	NEUTRAL
3	Breast Invasive Ductal Carcinoma	51.87	High	0	LumA	3	Positive	Positive	2.0	NEUTRAL
4	Breast Invasive Ductal Carcinoma	87.18	Moderate	0	LumB	1	Positive	Positive	3.0	GAIN
...	...	...	...	...	...	...	...	...	...	...
1491	Breast Invasive Ductal Carcinoma	50.08	Moderate	0	claudin-low	3	Positive	Positive	NaN	NEUTRAL
1492	Breast Invasive Ductal Carcinoma	60.99	High	0	Her2	3	Positive	Positive	NaN	NEUTRAL
1493	Breast Invasive Ductal Carcinoma	83.39	Moderate	0	Basal	4	Positive	Positive	3.0	NEUTRAL
1494	Breast Invasive Ductal Carcinoma	60.63	High	0	LumB	4	NaN	Positive	3.0	NEUTRAL
1495	Breast Invasive Ductal Carcinoma	68.74	Low	1	claudin-low	1	Positive	Positive	3.0	NEUTRAL

1496 rows × 667 columns

Figure 2: Data Description

0.4 Models

For modeling, we created a pipeline that runs a PCA model which reduces the number of features into n_components of which hyperparameter tuning is utilized for each single model separately to enhance the accuracy and overall performance.

A - Logistic Regression: We tried using Logistic Regression as it is a widely used ML training model for medical applications, and because it supports parameters that avoid overfitting via regularization (e.g., L1 & L2). Testing the prediction on the test data gave an estimated accuracy of 75, recall value of 75, precision of Z, and f-measure of 71.

Parameters used: 'log_reg_C': 10, 'log_regpenalty': 'l2', 'log_regsolver': 'saga', 'pca_n_components' 400, and are used to optimize logistic regression by balancing regularization strength (C), using L2 regularization (penalty), selecting the saga solver for efficiency with large datasets, and reducing dimensionality to 400 components with PCA to improve performance.

Overall, the results were insufficient and unstable for predictions.

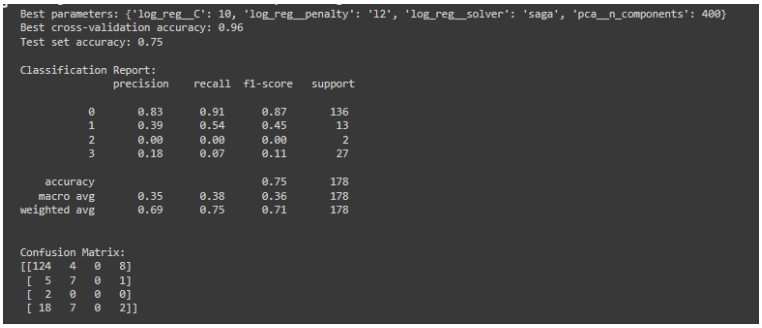


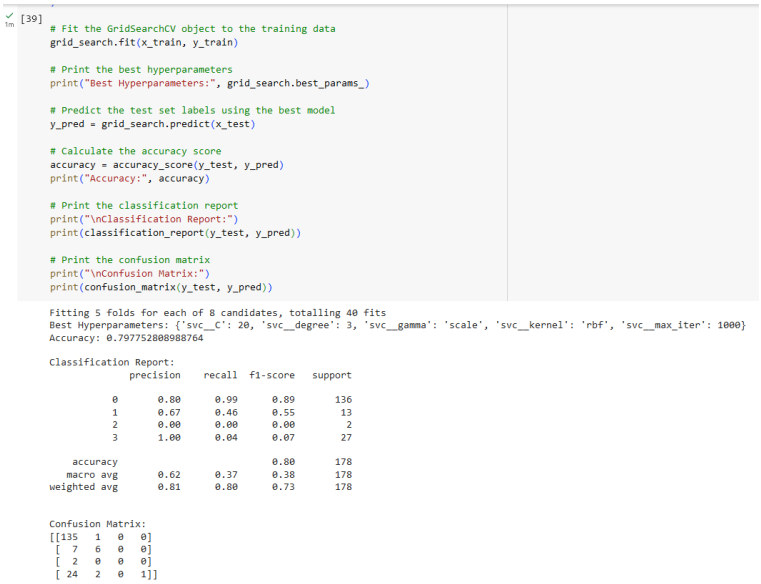
Figure 3: Logistic Regression Results

B- Support Vector Machines Tuned GridSearch: We tried using Support Vector Machines as SVMs are well-suited for datasets with a large number of features compared to the number of samples, and because SVMs can handle nonlinear data by applying kernel functions that implicitly map data to higher-dimensional spaces. Testing the prediction on the test data gave an estimated accuracy of 79.7, recall value of 80, precision of 81, and f-measure of 73.

Parameters used: 'pca_n_components': 250, 'svcC': 10, 'svcdegree': 2, 'svcgamma': 'scale', 'svckernel': 'rbf', 'svc_max_iter': 500 and are used to optimize the SVM model

by reducing data to 250 principal components (PCA), enhancing generalization via a regularization parameter (C) of 10, allowing a polynomial kernel with a degree of 2 and a radial basis function kernel (rbf), using the 'scale' gamma to adjust the influence of each support vector, and capping the iterations for convergence efficiency.

Overall, the results were the second most accurate as we had the second highest values (sometimes the highest as Neural Networks is sometimes inconsistent and can get a bit lower than 80% while SVMs are consistent).
Note: We managed to get an accuracy of 82.7% but that was when we scaled after re-sampling, whereas we ended up doing the vice versa for logical organization reasons, leaving with an accuracy of 79.7%).



```
[39]
# Fit the GridSearchCV object to the training data
grid_search.fit(x_train, y_train)

# Print the best hyperparameters
print("Best Hyperparameters:", grid_search.best_params_)

# Predict the test set labels using the best model
y_pred = grid_search.predict(x_test)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Print the confusion matrix
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

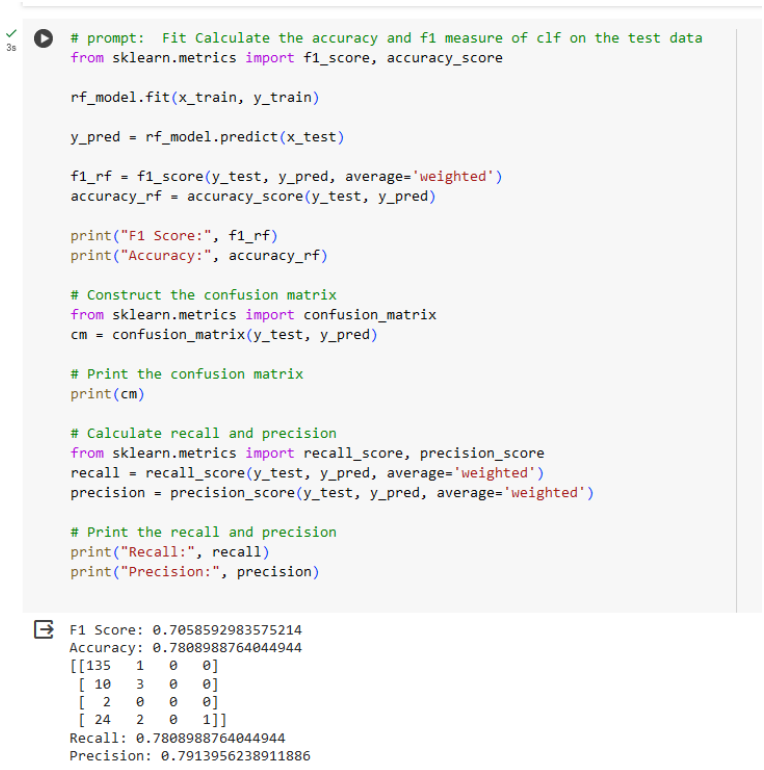
```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
Best Hyperparameters: {'svc__C': 20, 'svc__degree': 3, 'svc__gamma': 'scale', 'svc__kernel': 'rbf', 'svc__max_iter': 1000}
Accuracy: 0.797752808988764

Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.99      0.89       136
           1       0.67      0.46      0.55        13
           2       0.00      0.00      0.00         2
           3       1.00      0.04      0.07        27

    accuracy                           0.80       178
   macro avg       0.62      0.37      0.38       178
weighted avg       0.81      0.80      0.73       178


Confusion Matrix:
[[135   1   0   0]
 [  7   6   0   0]
 [  2   0   0   0]
 [ 24   2   0   1]]
```

Figure 4: Main Approach SVM Results

C - Random Forests Tuned GridSearch: Similarly, we tried the Random Forest Classifier model as they offer Ensemble Learning, where they combine multiple decision trees to create a more

accurate and robust model. Testing the prediction on the test data gave an estimated accuracy of 78.8, recall value of 78.8, precision of 70.1, and f-measure of 70.5.

Parameters used:'pca_n_components': 400, 'rfmax_depth': None, 'rfmin_samples_leaf': 2, 'rfmin_samples_split': 5, 'rf_n_estimators': 50 and are used to optimize the Random Forest classifier by using PCA to reduce dimensionality to 400 components, limiting overfitting through min_samples_leaf and min_samples_split, and leveraging 50 trees (n_estimators) of unrestricted depth (max_depth: None) for flexible and robust modeling.



```
# prompt:  Fit Calculate the accuracy and f1 measure of clf on the test data
from sklearn.metrics import f1_score, accuracy_score

rf_model.fit(x_train, y_train)

y_pred = rf_model.predict(x_test)

f1_rf = f1_score(y_test, y_pred, average='weighted')
accuracy_rf = accuracy_score(y_test, y_pred)

print("F1 Score:", f1_rf)
print("Accuracy:", accuracy_rf)

# Construct the confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Print the confusion matrix
print(cm)

# Calculate recall and precision
from sklearn.metrics import recall_score, precision_score
recall = recall_score(y_test, y_pred, average='weighted')
precision = precision_score(y_test, y_pred, average='weighted')

# Print the recall and precision
print("Recall:", recall)
print("Precision:", precision)
```

```
F1 Score: 0.7058592983575214
Accuracy: 0.7808988764044944
[[135   1   0   0]
 [ 10   3   0   0]
 [  2   0   0   0]
 [ 24   2   0   1]]
Recall: 0.7808988764044944
Precision: 0.7913956238911886
```

Figure 5: Main Approach Random Forest Results

D - Neural Networks: Finally, we chose and ended up using a Neural Networks model thanks to its ability to learn complex relationships through non-linear relationships, deep learning capabilities, and feature learning. This aligns well with our complex data of tens of thousands of features (after re-labeling).Testing the prediction on the test data gave an estimated accuracy of 80.6, recall value of 80.6, precision of 76.2, and f-measure of 75.7. For the parameters, they are inconsistent due to the dropout rate.

Overall, the results were the most accurate as we had the highest values for each of the above predictions. This implies that our test results from using Model B (Support Vector Machines) and Model D (Neural Networks) are logical as the results are similar (79.7% by 80.6%).

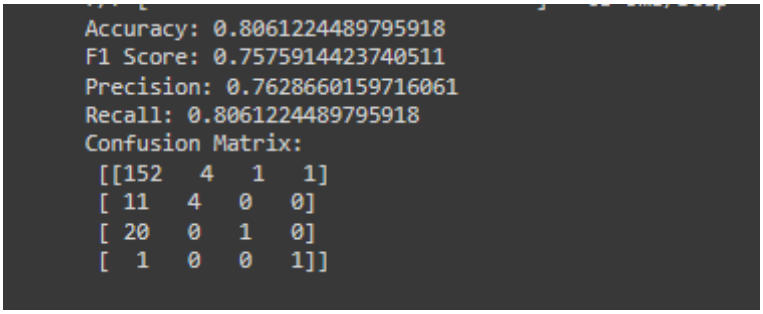


Figure 6: Main Approach Neural Networks Results

0.5 Results

Learning Curve:

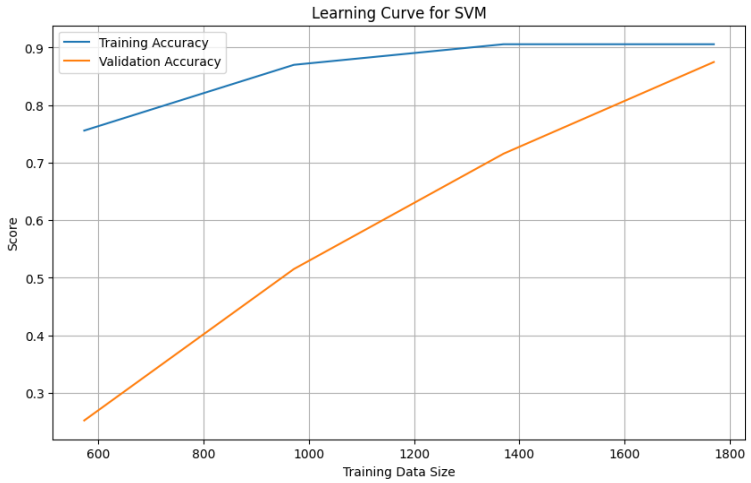


Figure 7: SVM Learning Curve

LDA:

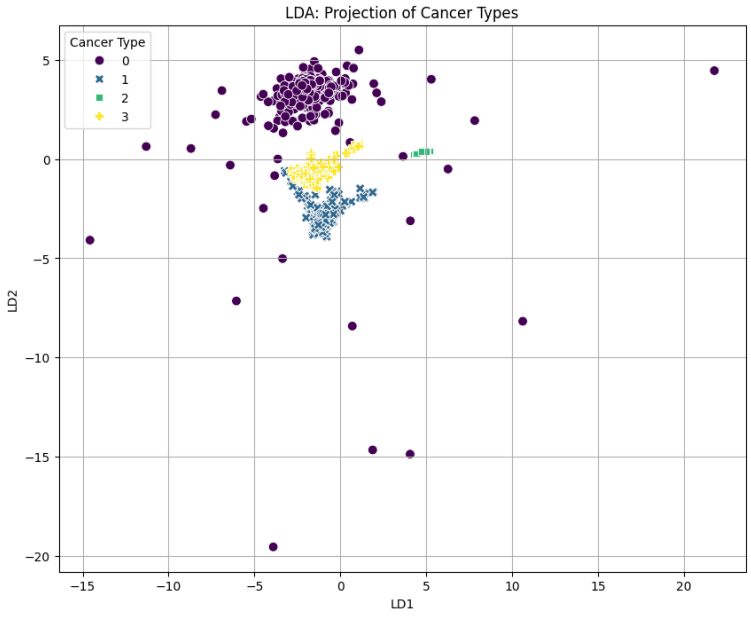


Figure 8: LDA Plot on Train Data

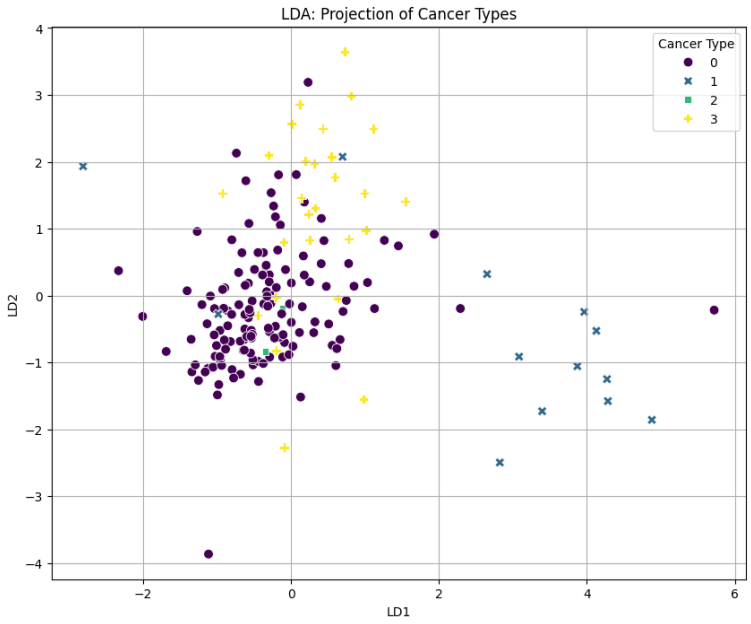


Figure 9: LDA Plot on Test Data

Approach One: Through this approach, we committed to re-labeling, removing anomalies, splitting, scaling, balancing, and reducing the data as mentioned above. We followed this approach as it was organized in a logical manner in accordance to our research, and because we got the highest accuracy, precision, and recall percentages, approximating 80 per cent, mainly with the Support Vector Machines and Neural Networks models.

Approach Two: Through this approach, we used both the correlation matrix, the SHAP (SHapley Additive exPlanations), and manual probability analysis to both understand feature relationships and redundancy, and interpret and explain machine learning models by showing feature importance for predictions. Through that, we chose 40 genes, and three mutation columns (pik3a_mut, tp53_mut, muc16_mut). Overall, the results were not satisfactory where the accuracy and precision were in the range of high 50s to mid 60s in per cent. Thus, we discontinued this approach and concluded that we need the rest of the genes and mutation columns for training our models for cancer type prediction.

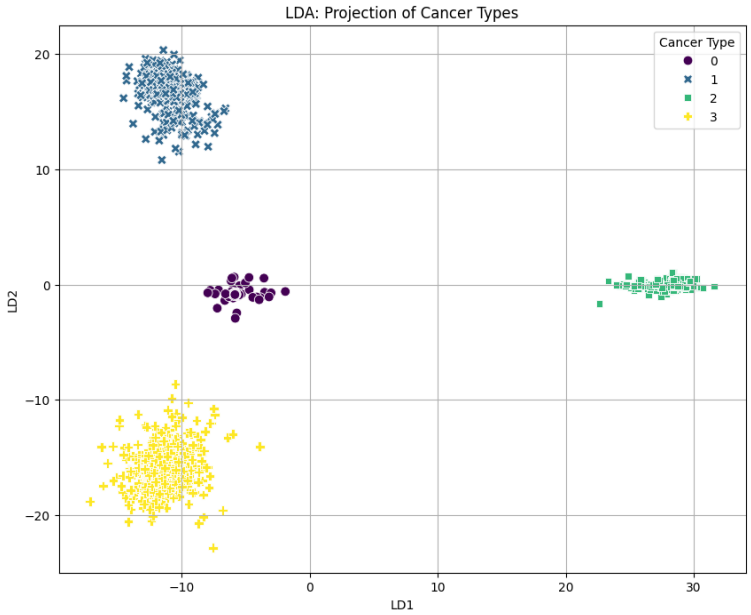


Figure 10: LDA Plot on Train Data (Clusters)

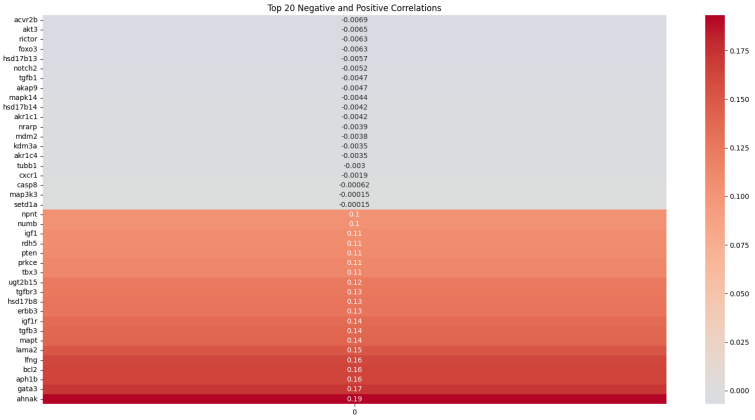


Figure 11: Genomic Correlation Heatmap (Approach Two, Discontinued)

0.6 Conclusion

Overall, we saw that keeping all the data is necessary, especially that the data is initially low (approximately a 1496). We made sure to utilize various pre-processing methodologies that were either learned in class or discovered through thorough research (e.g., DB Scan, LDA, etc.). Our results of the main and final approach showed similar values for accuracy, precision, and recall, with the SVM and Neural Network models having the highest percentages of 79.7% and 80.6% per cent respectively. We also made sure to visualize the data in accordance to the cancer type to help us check for outliers and data inconsistency. However, there is a sign of overfitting, as our cross-validation accuracy is in the 90s per cent while validation test accuracy is nearly 80 per cent. Thus, more data is needed.

0.7 References

https://www.cancercenter.com/cancer-types/breast-cancer/types

https://www.who.int/news-room/fact-sheets/detail/breast-cancer

https://www.frontiersin.org/journals/genetics/articles/10.3389/fgene.2022.958780/full

https://davidbaranger.com/2018/04/09/improving-genetic-prediction-data-cleaning-meta-an

https://ai.plainenglish.io/top-10-python-libraries-for-handling-imbalanced-data-in-ml-c

https://towardsdatascience.com/using-shap-values-to-explain-how-your-machine-learning-m

https://www.nlm.nih.gov/

https://medium.com/analytics-vidhya/how-to-remove-outliers-for-machine-learning-24620c4