University of Essex

School of Mathematics, Statistics and Actuarial Science

MA981 DISSERTATION

Tracking the Factors leading to Breast Cancer Staging and Survival: An Exploratory Analysis

Sourav Ghosh Hansda

Supervisor: **Lausen, Berthold**

December 22, 2023
Colchester

Contents

Abstract

The heterogeneity of breast cancer underscores the urgency for accurate cancer staging and overall survival prediction. In this backdrop, the study has explored the clinical and genomic aspects of breast cancer patients based on the landmark TCGA-GDC database. It observes that the age group around 60 years is most vulnerable to breast cancer. The disease is found more prevalent among the whites. In a rigorous framework of logistic regression, the study brings out the lymph nodes examined count as the only clinical variable, which is statistically significant in the final model. Applying Kaplan-Meier Estimator and Log Rank Test, it finds that the groupings of patients in terms of vital status: alive and dead, tumor staging, and her2 status (ie, aggressive tumor development) are meaningful analytical categories for survival analysis. Given the limited utility of clinical data for survival analysis, it considers gene expression data and tackles the computational challenge by applying differential expression analysis and selecting 60 genes based on p-value out of about 60,000 genes without compromising the performance efficiency. It finds that the optimal number of clusters out of 60 genes may be two in consonance with the number of cancer staging being two: invasive and non-invasive, used in the study as a dependent variable. In view of the limitations of logistic regression in explaining cancer staging, the study carries out supervised learning, namely, random forest and neural network on the dataset. It finds that the random forest model works well with the dataset for explaining both cancer staging and overall survival. The results are found robust against resampling and random sampling of genes in place of differential expression analysis. Given the difficulty in interpreting machine learning-based results, the study suggests that future research may be directed at unraveling the black boxes like machine learning models with a view to improving their utility in the medical domain.

CHAPTER 1

Introduction

Breast cancer, the most prevalent malignancy among women worldwide, poses a significant public health concern. Accounting for around 25% of all female cancer diagnoses and 15% of all cancer-related deaths, breast cancer exhibits a heterogeneous nature, characterized by diverse clinical presentations, treatment responses, and prognoses. This heterogeneity underscores the urgency for accurate staging and overall survival prediction. Accurate prognostication enables tailored treatment strategies, optimizing patient outcomes and improving survival rates. The Cancer Genome Atlas (TCGA) project, a landmark initiative of the National Cancer Institute (NCI) and the National Human Genome Research Institute (NHGRI), provides a comprehensive resource for studying cancer at the molecular level. The TCGA database comprises clinical data, including patient demographics, tumor characteristics, and treatment information, as well as genomic data, such as gene expression, DNA copy number, and DNA methylation profiles. This wealth of data offers invaluable insights into the underlying biology of breast cancer and its response to treatment. The TCGA database is hosted at the Genomic Data Commons Data Portal (GDC) and henceforth is referred as TCGA-GDC database. This study aims to leverage the TCGA-GDC database to identify significant predictors of breast cancer staging and overall survival. By employing a battery of techniques in supervised and unsupervised learning, we will explore the complex interplay between clinical and genomic factors that contribute to breast cancer heterogeneity. The rest of the study is organized as follows. Chapter II delineates the conceptual underpinnings of cancer staging and overall survival (OS). Chapter III presents a brief survey of literature, bringing out the learnings and guideposts relevant for the exercise at hand. Chapter IV elucidates the stylized facts and figures on the predictors and response variables based on the TCGA-GDC database. Chapter V discusses different dimensions of the battery of statistical methodologies employed to decode and decipher the big data set. Chapter VI applies the statistical techniques on the dataset and presents and interprets the results. It compares and, contrasts the results in terms of a comparative study of the methodologies employed on the dataset. The scope for refinement of the study in view of its limitations is explored in Chapter VII towards providing a direction for future research. The concluding observations from the study are provided as a summing up in Chapter VIII.

CHAPTER 2

Conceptual Underpinnings of Cancer Staging and Overall Survival

Cancer staging, which is a crucial prognostic factor, is the classification of (breast) cancer depending on the size and spread in respect of tumor, node, and metastasis (TNM). When such classification is carried out after (before) surgery of tumor, it's known as pathologic (clinical) staging. Pathologic staging is often considered more useful as it uses additional information contained inside the tumor. If, however, surgery is preceded by other treatments like chemo/radiation therapy, pathologic staging could be misleading. TCGA-GDC database provides data on the pathologic stage of breast cancer patients in line with the American Joint Committee on Cancer under the head: `ajcc_pathologic_stage`. According to Rocky Mountain Cancer Centers (2023) and DePolo (2023), breast cancer can be categorized into five major stages[3][6]. While stages I through IV may be considered invasive, stage 0 may be treated as non-invasive or in situ. However, the head: `ajcc_pathologic_stage` in the TCGA database has I to IV stages. As cancer is small and confined to one area in stage I, we take it as the non-invasive stage. As cancer becomes larger and/or extends to other parts of body, it moves to stages II, III and IV, which are taken as invasive stages in the current study. Depending on the invasive (as also its sub-categories) or non-invasive stage of the cancer, best treatments are designed for the patient. Overall survival (OS), a la Gobbini et al (2018), is defined as the duration of time from the date of initial diagnosis to the date of death from any cause, or the date of last follow-up for censored patients[10].

Often, it’s presented in per cent of total number of cancer patients for a given number of years/days of survival. As the number of years/days of survival is raised, OS rate goes down. OS rates can vary depending on the cancer stage. Patients diagnosed with stage I or II breast cancers are expected to have higher overall survival rates than people diagnosed with stage III or IV breast cancers.

CHAPTER 3

A Brief Survey of Literature

Breast cancers with virtually identical TNM characteristics may exhibit highly contrasting behaviors due to divergent molecular profiles. Therefore, predictive models need to incorporate histopathological and molecular predictors to explain better the heterogeneity of the complex group of diseases, called cancer (Giacomelli et al, 2023)[9]. Cancer research (on breast) using survival analysis may be classified into two types: binary classification and risk regression. As part of binary classification approach, short-survival group is distinguished from long-survival group using a duration of survival, say 5 years. In risk regression approach, each patient is assigned a risk score. Using various classification methods like gradient boosting, random forest, artificial neural network, and support vector machine, Zhao et al (2018) attempted to predict 5-year breast cancer survival based on gene expression data and clinical/pathological factors[23]. As different classification methods produced similar accuracy and area under the curve (AUC), they highlighted the role of quality of the data itself. Similarly, Gevaert et al (2006) applied Bayesian networks and obtained higher AUC with the use of integrated dataset than the standalone clinical or gene expression dataset[8]. Using multiple kernel learning, Sun et al (2018) integrated genomic data and pathological imaging data, resulting in marginally higher AUC than the standalone genomic data[22]. In risk regression framework, Cox-proportional hazards model has often been used for survival analysis of breast cancer, wherein predictors are multiplicatively related to hazards like the event of death. With the introduction of machine learning like neural network, linear relationship in the Cox-proportional hazards model is replaced by a non-linear relationship

(Huang et al, 2019)[11]. Bismeijer et al (2018) used functional sparse-factor analysis to integrate different types of molecular data and identify dominant biological processes active in breast and lung tumors[2]. In view of missing data in the TCGA-GDC dataset, Sherafatian (2018) attempted to identify minimal set of biomarkers to classify cancer status and different subtypes of cancer[21]. Rendleman et al (2019) applied imputation techniques for missing data and also dimensionality reduction techniques to bring down computational cost while ensuring good performance[19].

CHAPTER 4

Preliminary Data Analysis – A Clinical Fact Sheet

To start with, we consider clinical data from the TCGA-GDC database and focus on the female category as breast cancer is overwhelmingly a disease faced by females. The age group of around 60 years is found to have the maximum number of patients who are diagnosed with breast cancer, followed by patients of about 50 years (Figure 4.1).

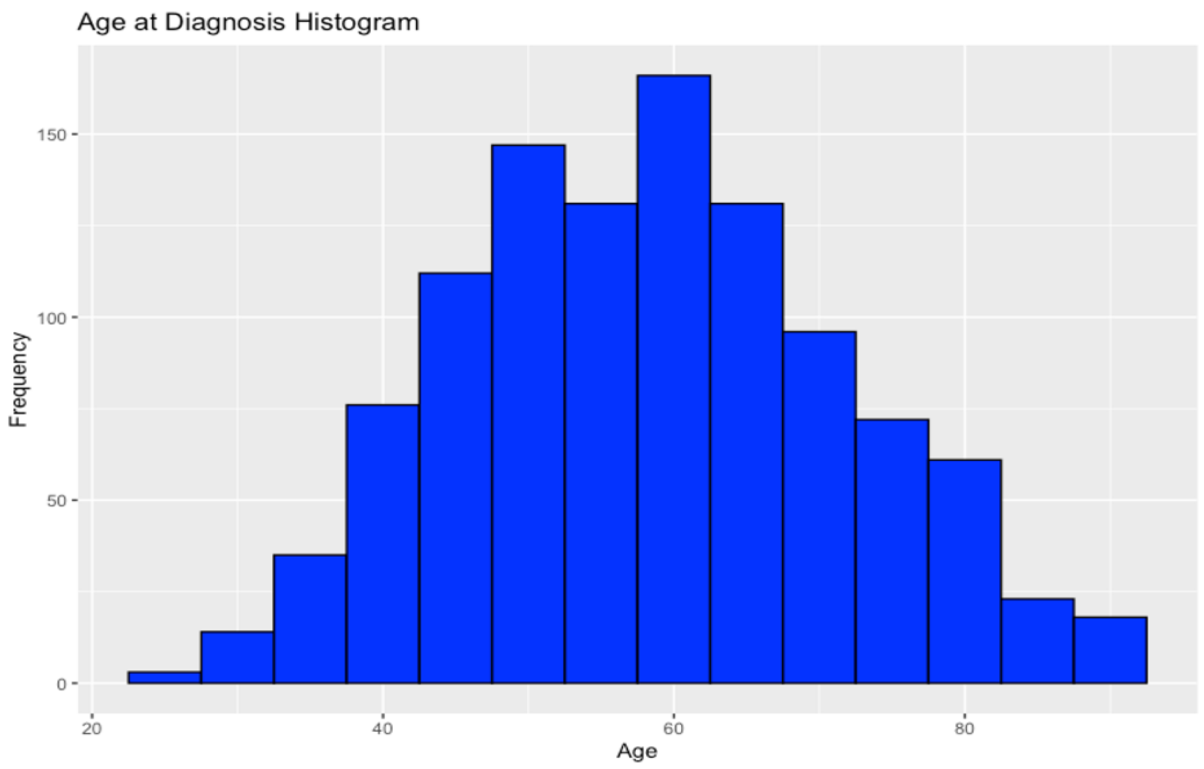


Figure 4.1: Age at Diagnosis

Breast cancer patients are predominantly white, followed by Black/African American and Asian in that order (Figure 4.2).

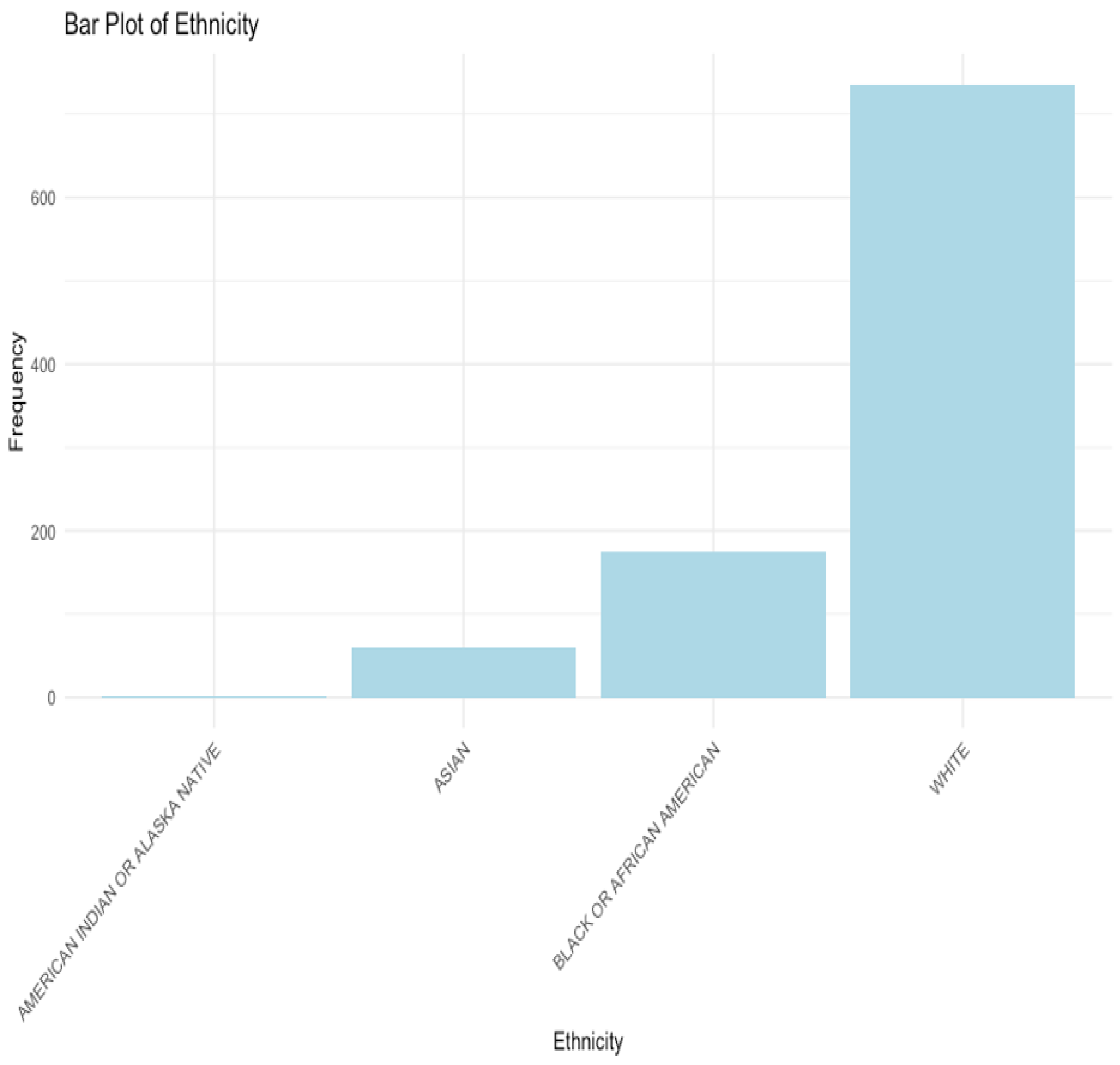


Figure 4.2: Ethnicity

The histological type of cancer provides details on the microscopic appearance of tumor. Based on this criterion, infiltrating carcinoma is found to have the highest frequency and is followed by mucinous carcinoma (Figure 4.3).

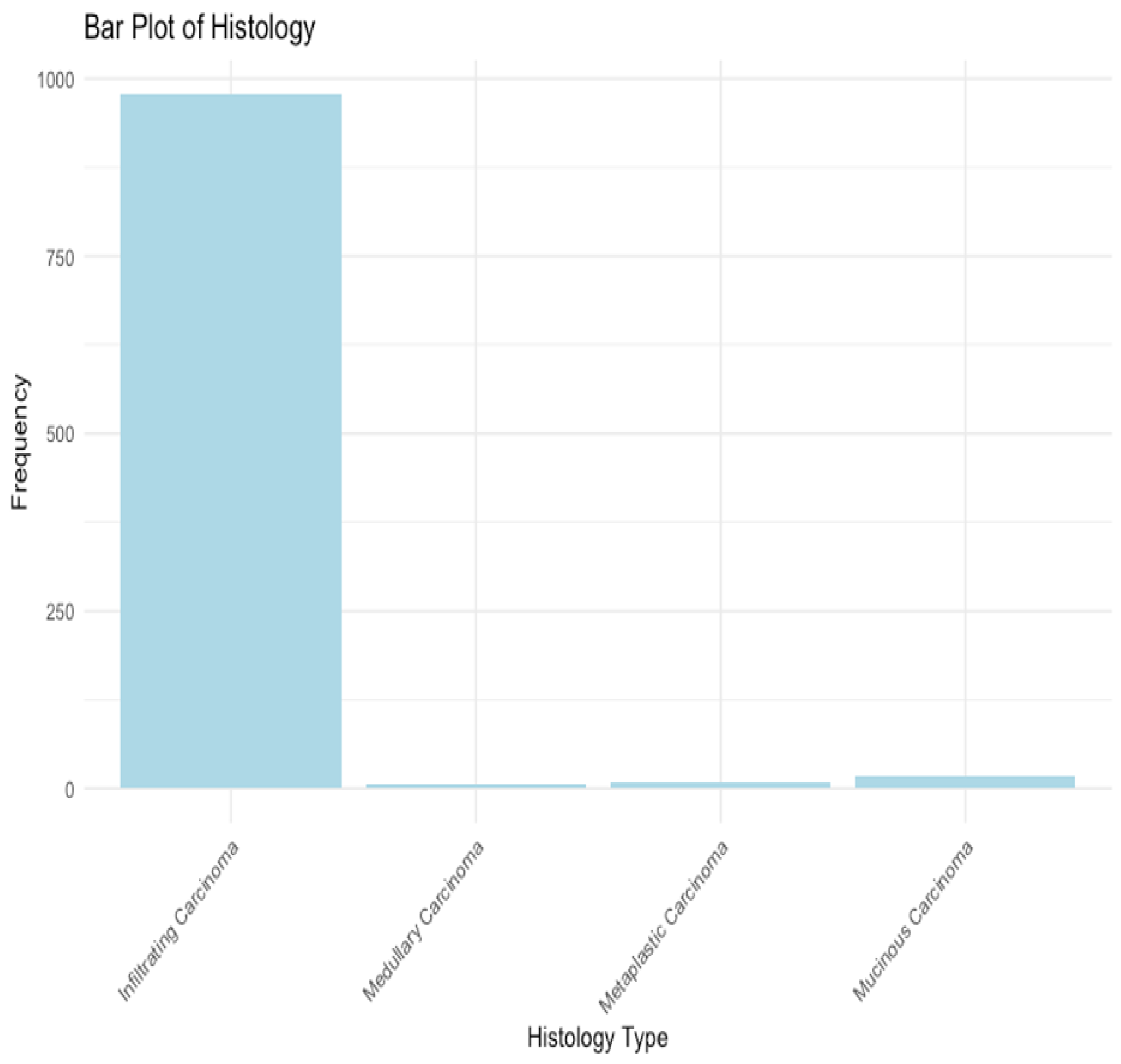


Figure 4.3: Histology Types

As per the pathologic tumor stage, the majority of patients is in stage II, for whom surgery alone may not be curative (Figure 4.4). The next stage in terms of number of patients turns out III, wherein the tumor size and/or its spread is even larger.

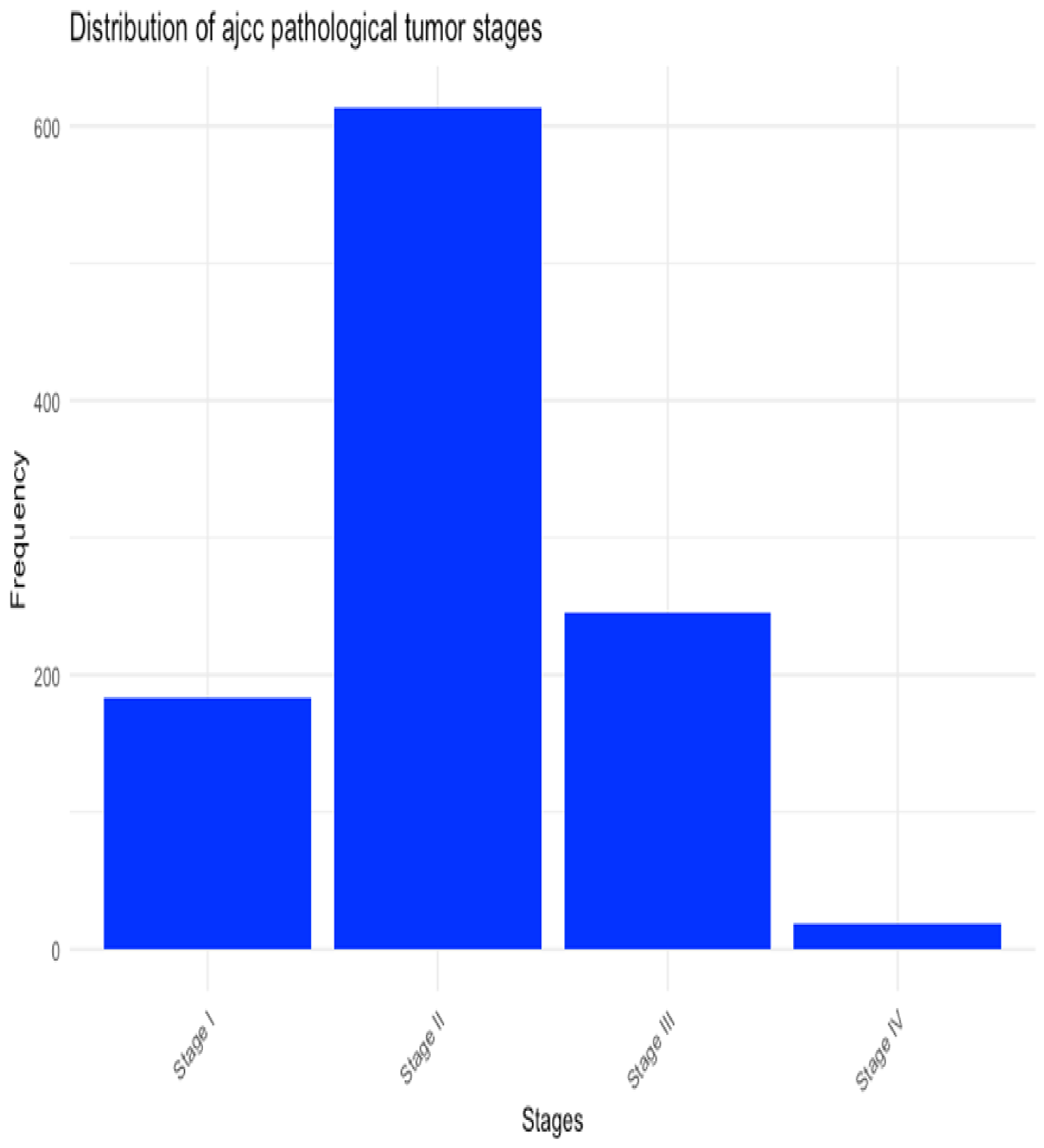


Figure 4.4: Stages

The dominant therapy in use against breast cancer has been chemotherapy, followed by hormone therapy (Table 4.1 and Figure 4.5). While breast cancer patients have generally tried various types of therapies, patents in stage II account for the maximum frequency under each therapy.

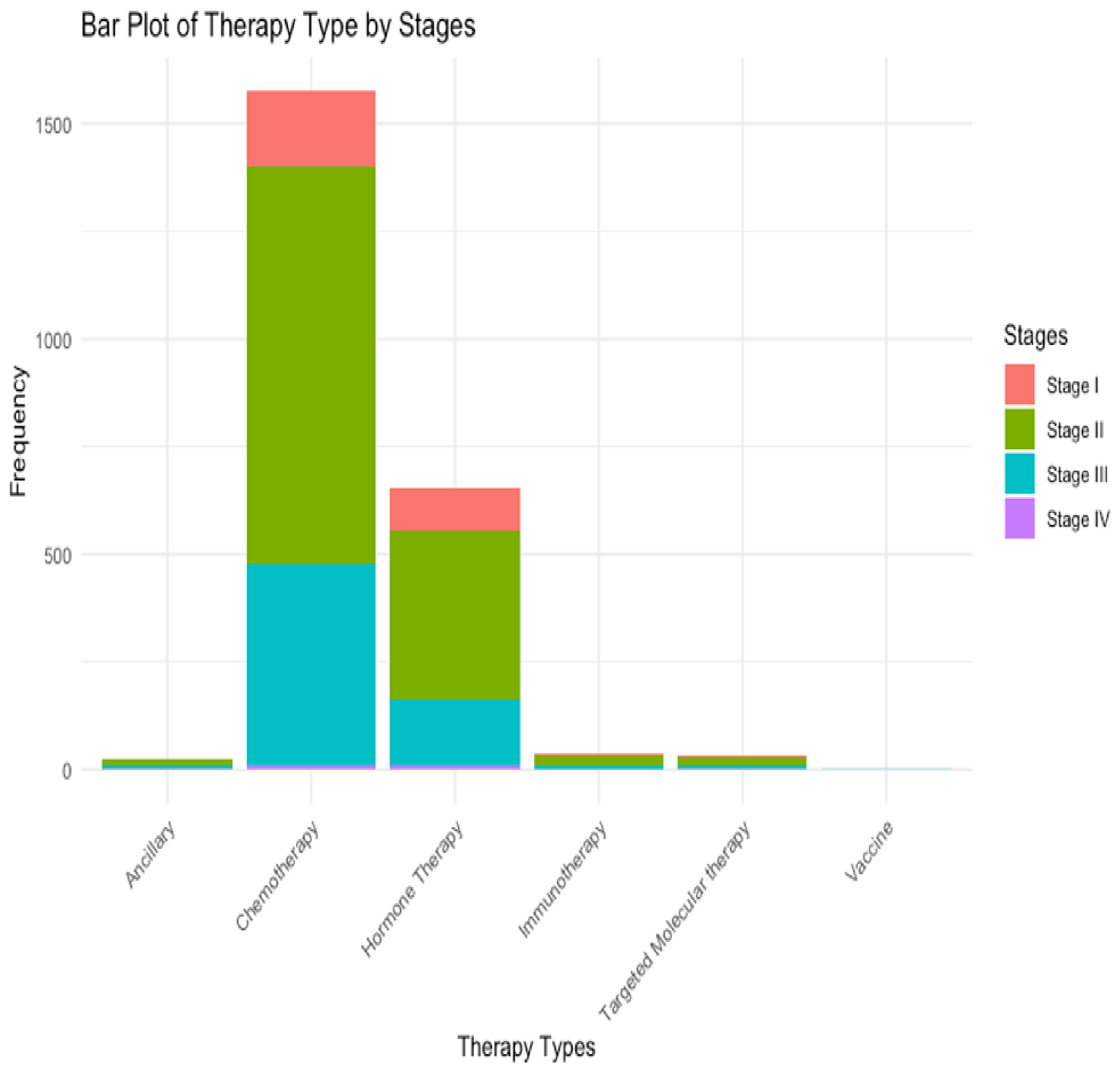


Figure 4.5: Stages in different Therapies

Table 4.1: Stages in different Therapies

Stage	**Ancillary**	**Chemotherapy**	**Hormone therapy**	**Immunotherapy**	**Targeted Molecular therapy**	**Vaccine**
Stage I	1	176	100	6	3	0
Stage II	12	920	390	19	18	0
Stage III	9	467	153	12	9	1
Stage IV	1	12	12	0	1	0

Source: TCGA database.

CHAPTER 5

Variables and Methodology

Terminologies used in the clinical dataset, data description and percentage missing values for 107 entries are presented in Appendix. Throughout the study, cancer staging and overall survival as explained in Chapter II are taken as response variables. Clearly, cancer staging can have binary outcome: invasive or non-invasive whereas overall survival is a continuous variable. After carefully examining the suitability of all clinical variables based on criteria such as clinical importance, absence of multicollinearity and percentage missing values, we have selected the following nine as predictor variables: `age_at_diagnosis` , `menopause_status`, `margin_status` , `lymph_nodes_examined_count` , `er_status_by_ihc` , `pr_status_by_ihc` , `her2_status_by_ihc`,`history_other_malignancy` and `lymph_nodes_examined`. At a disaggregated level, the set of nine predictors boils down to a granular set of 24 clinical variables. Summary statistics of the select set of clinical predictors are presented in Appendix. To start with, we have attempted to fit a logistic regression on the clinical dataset, using cancer staging as the dependent categorical variable. Logistic regression estimates the probability of an event occurring based on a given set of independent variables. The coefficient in this model is commonly estimated through maximum likelihood estimation. After carrying out the logistic regression based on 24 independent variables at the disaggregated level, we perform a backward variable selection and drop variables, one at a time, if such removal helps improve the model's Akaike Information Criterion (AIC). For survival analysis, Kaplan-Meier Estimator, which is a non-parametric statistic, has been used to estimate the survival function in terms of percentage of patients surviving for a certain amount of time after diagnosis/treatment. The estimator is versatile enough to accommodate the event of drop-out of patients, data truncation or missing values. We have also used log rank test, which compares the survival curves of two or more groups. The null hypothesis for a log rank test, which uses a chi-square statistic, is that the groups have the same survival time. These statistics are useful when the variables are categorical or take small number of values. With the induction of gene expression data combined with clinical data, the data dimension increases manifold, given more than 1000 breast cancer patients and over 60,000 genes per patient. In view of the computational challenges in terms of cost/time involved in handling the big data set, we decide to explore and select a set of manageable number of genes, albeit without compromising the quality of analysis. Towards this direction, we first carry out normalization of the data and then perform differential expression (DE) analysis, following Maié and Manolov (2000)[14]. This brings out the genes that show differences in expression level between conditions or are associated with given predictors or responses. This is carried out with the help of a computational pipeline, called `limma_pipeline` in R, which returns the top (set at 60 in current study) differentially expressed genes sorted by p-value. Clustering partitions a dataset into clusters or groups such that data points within a cluster or group are more similar to each other than to those in other clusters. For clustering, we apply K-Means clustering, which partitions data into K clusters based on mean of data points. Deciding the optimal number of clusters, however, remains a challenge. A popular method for optimal cluster count is the Elbow method, which runs K-means clustering for a range of values of K and plots the within-cluster sum of squared distances (WCSS) between each point and the centroid of that cluster. This is also known as scree plot in clustering and Elbow of the plot is the point where the rate of decrease in WCSS slows down, implying that addition of more clusters does not improve the model's fit significantly. The optimal number of clusters can also be based on Silhouette score, which measures in a range of [-1 to +1] how similar an object is to its own cluster compared to other clusters. Scores close to 1 (-1) indicate high (low) similarity, implying well defined clusters. For K-Means clustering, Silhouette score is calculated for different values of K and we choose that K that produces the maximum Silhouette score. We now proceed with the set of 60 select genes and the two response variables to run a random forest model on the dataset for more than 1000 breast cancer patients. Random forest is a supervised non-linear machine learning algorithm built through an ensemble of decision trees. It builds and combines multiple decision trees, trained on different parts of the same training set, with the goal of overcoming over-fitting problem of individual decision tree as was encountered in logistic regression. We also attempt to fit a neural network model on the 60 select genes and the two response variables. Neural network allows complex nonlinear relationships between the predictors (bottom layer) and the response variable (top layer). There would be intermediate/hidden layers in between the way human brain functions. The technique being extremely versatile has the ability to fit any functional relationship in the data. Finally, we have compared and contrasted the model performance of random forest and neural network, using the criteria like accuracy and precision.

CHAPTER 6

Results and Interpretation

6.1 Logistic Regression

The results of logistic regression when we consider the most disaggregated model are presented in Appendix. It may be noted that `lymph_nodes_examined_count`, and the disaggregated categories of `her2_status_by_ihc`: positive, negative and equivocal, are the only four clinical variables, which are statistically significant. In medical domain, `lymph_nodes_examined_count` is taken as indicative of possible presence of a malignant process and its localization or generalization. It is found statistically significant at 0.001 level of significance. Further, a positive `her2_status_by_ihc` implies that breast cancer tests positive for a protein, namely, human epidermal growth factor receptor 2 (HER2), which promotes the growth of cancer cells. Thus, a positive her2 status poses greater risk of relapse and metastasis than a negative her2 status. If her2 status is not clear, it is called equivocal. Though her2 status variables turn out statistically significant, the level of significance remains far higher than that for lymph nodes examined count. As only four variables are found statistically significant out of 24 independent variables, possible overfitting of the model is dealt with by applying a backward variable selection process based on AIC (Appendix). AIC makes a balance between the aspects of model complexity and goodness of fit and, helps retain variable/s that contribute more to model performance. Thus, lymph nodes examined count has turned out to be the single independent variable, which is statistically significant at a very low level of significance of 0.001 and hence, sufficient to predict the binary cancer staging. The log odds of binary cancer staging is estimated to increase by 0.16941 with one unit increase in lymph nodes examined count. As AIC decreases with backward selection to 616.2 from 645.32 in the original disaggregated model, the single-variable reduced model appears to throw up a better fit to the data than the initial model. As all the clinical variables barring one are not statistically significant in the reduced (final) model, logistic regression framework based on clinical variables seems hardly satisfactory for explaining cancer staging.

6.2 Kaplan-Meier Estimator & Log Rank Test

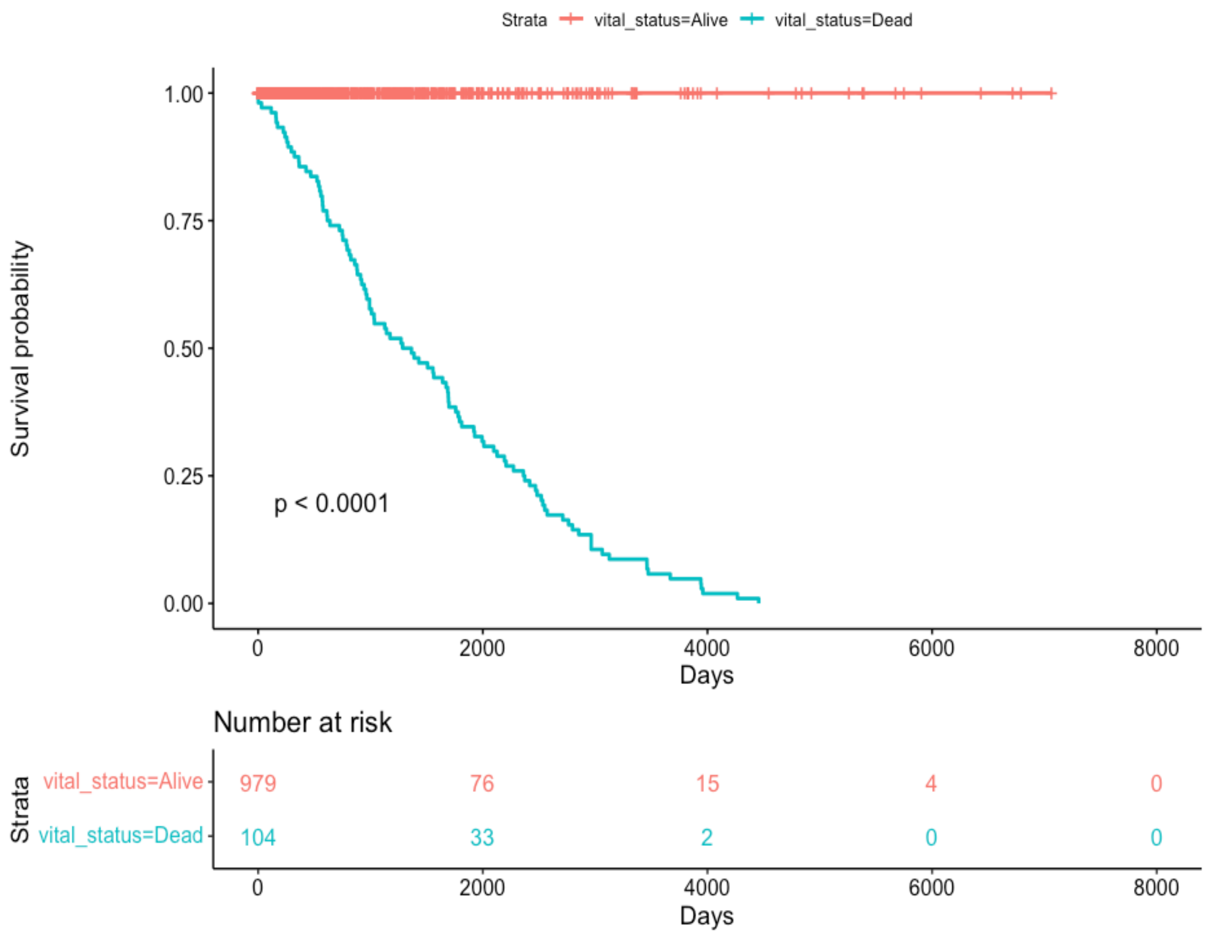
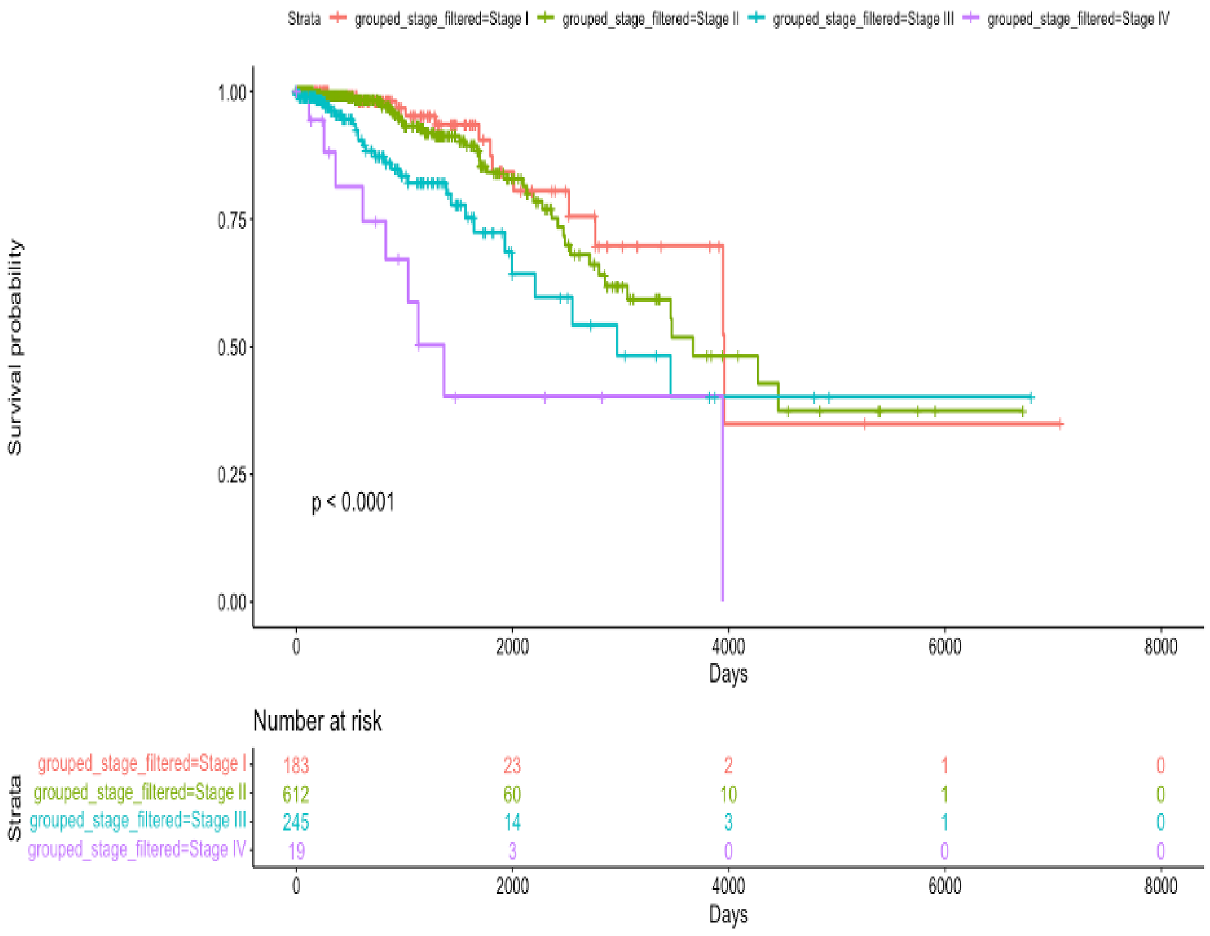


Figure 6.1: Vital Status

While the binary vital status: alive and dead appears to be an analytically meaningful category, we check it rigorously using the technique of survival analysis. For the alive, the survival curve remains horizontal all through, implying that the survival probability is unchanged at around 1 irrespective of the survival time, measured in number of days (Figure 6.1). In contrast, the survival curve for the dead has been downward sloping and lies below that for the alive group all through. The apparently higher survival probability of the alive than of the dead is checked for confirmation through the log rank test, which finds that the difference in means between the two curves is statistically significant, p-value being closer to zero. The lower panel of Figure 6.1, which provides the number of patients at risk shows that the number of patients declines gradually for both the groups and in tandem at least up to 4000 days.



Strata	0	2000	4000	6000	8000
grouped_stage_filtered=Stage I	183	23	2	1	0
grouped_stage_filtered=Stage II	612	60	10	1	0
grouped_stage_filtered=Stage III	245	14	3	1	0
grouped_stage_filtered=Stage IV	19	3	0	0	0

Figure 6.2: Stages

Note: 'Filtered' qualifier indicates computation after dropping of missing values.

Patients with tumor in stage IV is found to have the lowest survival percentage all through whereas patients with tumor in stage I have the highest survival percentage at least up to 4000 days (Figure 6.2). In any case, there seems to be significant difference in survival percentage across groups of patients with tumor in different stages. This is also buttressed by very low p-value. The relative number of patients at risk across stages also remain similar over time.

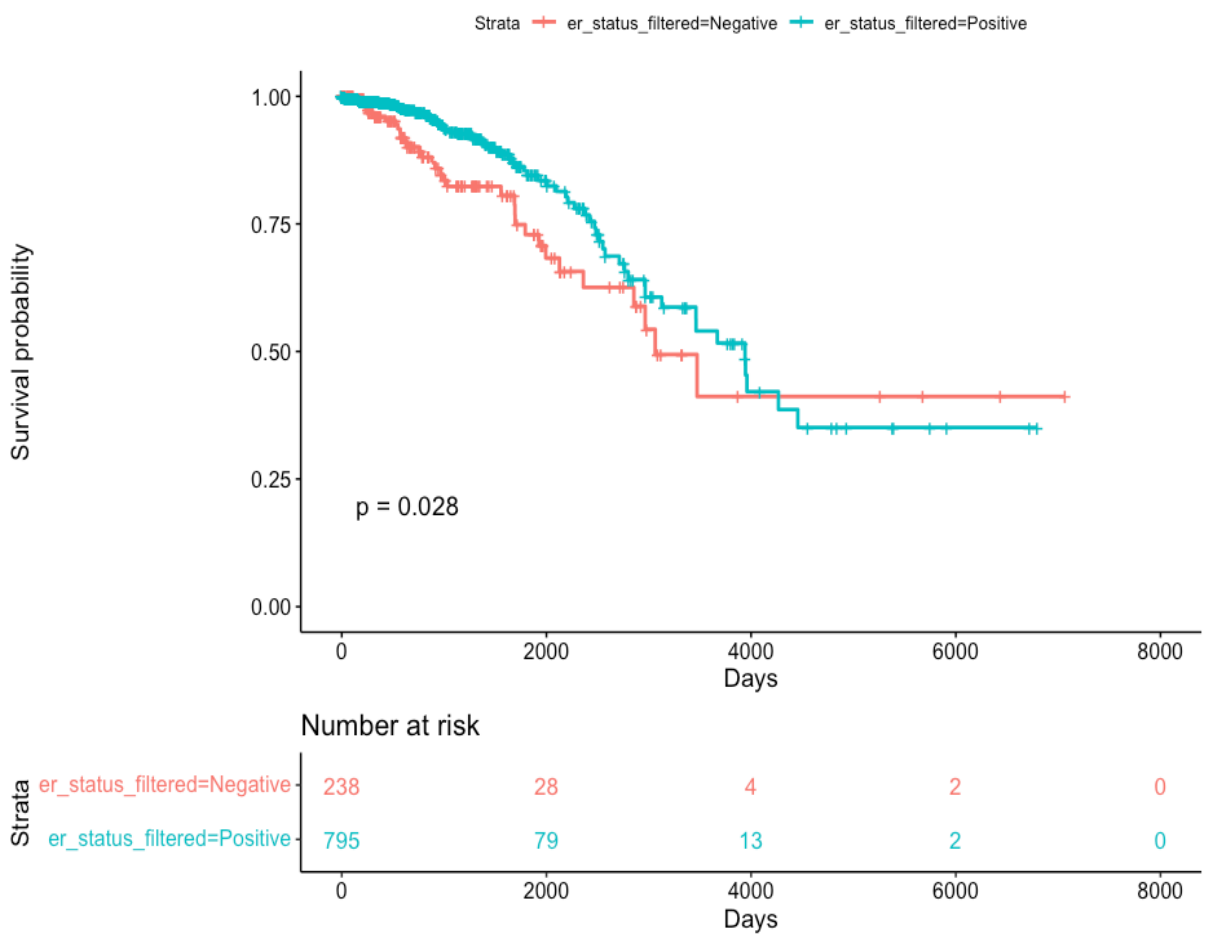


Figure 6.3: ER Status

Note: 'Filtered' qualifier indicates computation after dropping of missing values.

Clinically, patients with positive ER status are expected to respond better to hormone therapies like tamoxifen that block estrogen signaling. Such patients may have higher survival probability up to 4000 days (Figure 6.3). However, the utility of ER status category could not be confirmed by log rank test as we obtain high p-value.

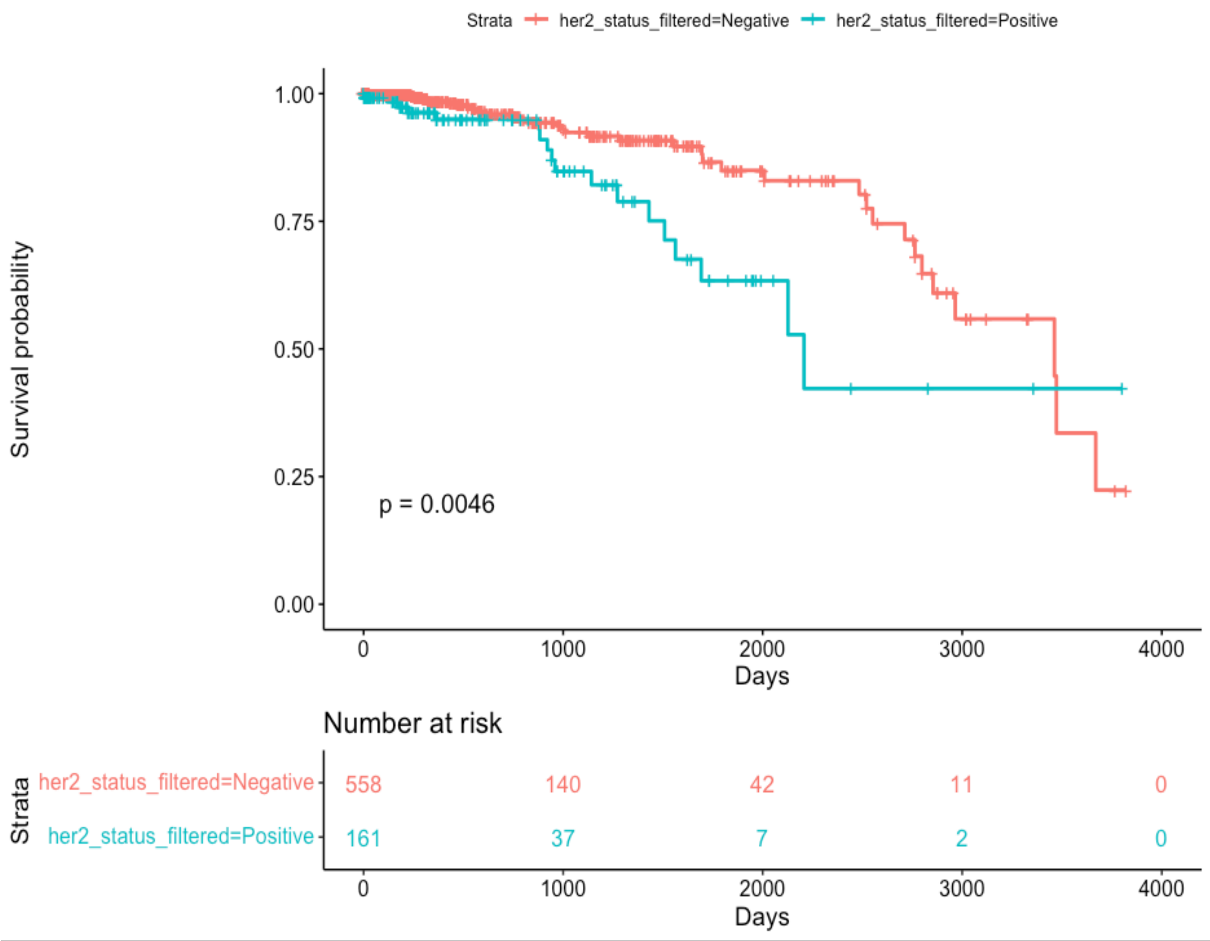


Figure 6.4: her2 Status

Note: ’Filtered’ qualifier indicates computation after dropping of missing values.

Grouping of patients as per her2 status – positive or negative – appears to matter, leading to lower probability of survival for her2 positive group (Figure 6.4). In such group, tumor tends to be more aggressive. This is also supported by log rank test with small p-value.

6.3 Differential Expression Analysis

Ahead of the DE analysis, we drop genes that have small counts of less than 10. Then, we apply trimmed mean of M-values (TMM) method to normalize the data and convert it to produce a similar variance as arrays, based on voom method, which estimates the mean-variance relationship in the data (Figure 6.5). Then, we carry out the differential analysis by comparing primary solid tumor samples with solid tissue normal as the reference group. In the process, we fit a series of linear models, one to each of the probes and rank the most differentially expressed genes based on p-values. Accordingly, we pick up the top-60 genes for further analysis like clustering and machine learning. The first two principal components of the expression matrix of the voom-transformed data are presented in Figure 6.6, which shows that the two sample groups, namely, primary solid tumor and solid tissue normal have clearly distinguishable RNA expression profiles.

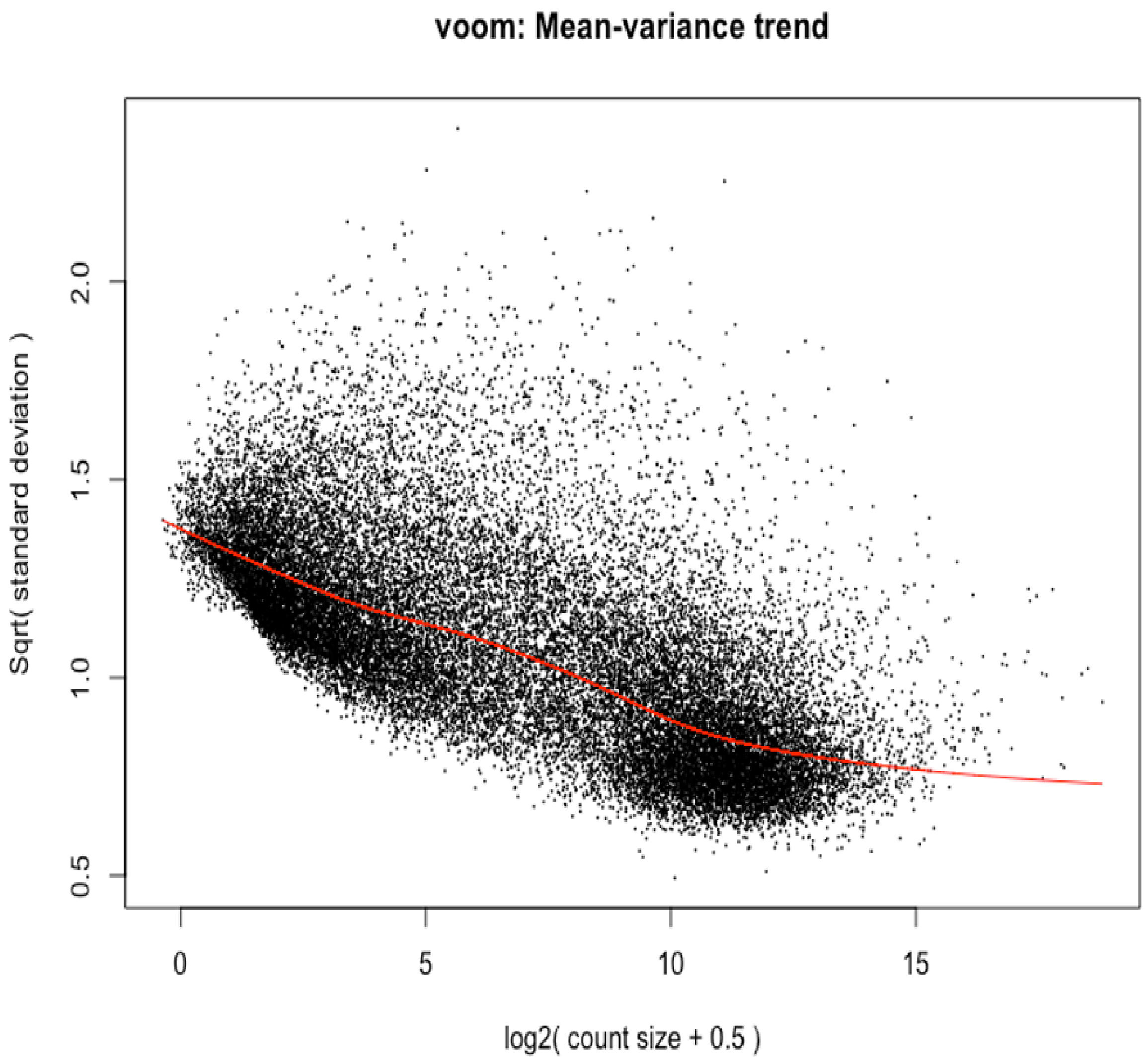


Figure 6.5: Mean Variance Trend

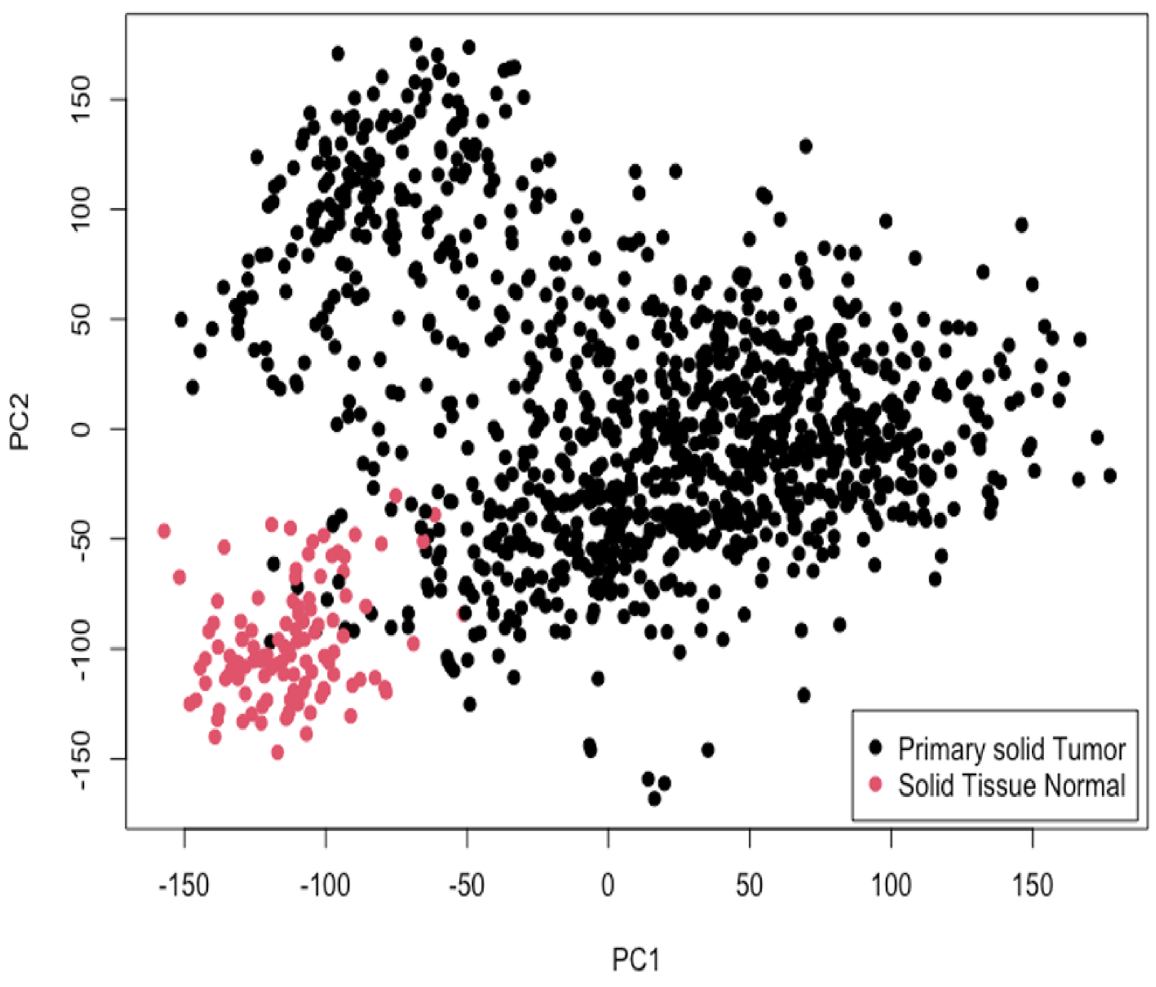


Figure 6.6: PCA

6.4 Clustering

The scree plot of clusters based on 60 identified genes shows that the elbow point is attained when the number of clusters, that is, K takes a value equal to 2 or 3, where adding more clusters does not significantly reduce the sum of squared distances (Figure 6.7).

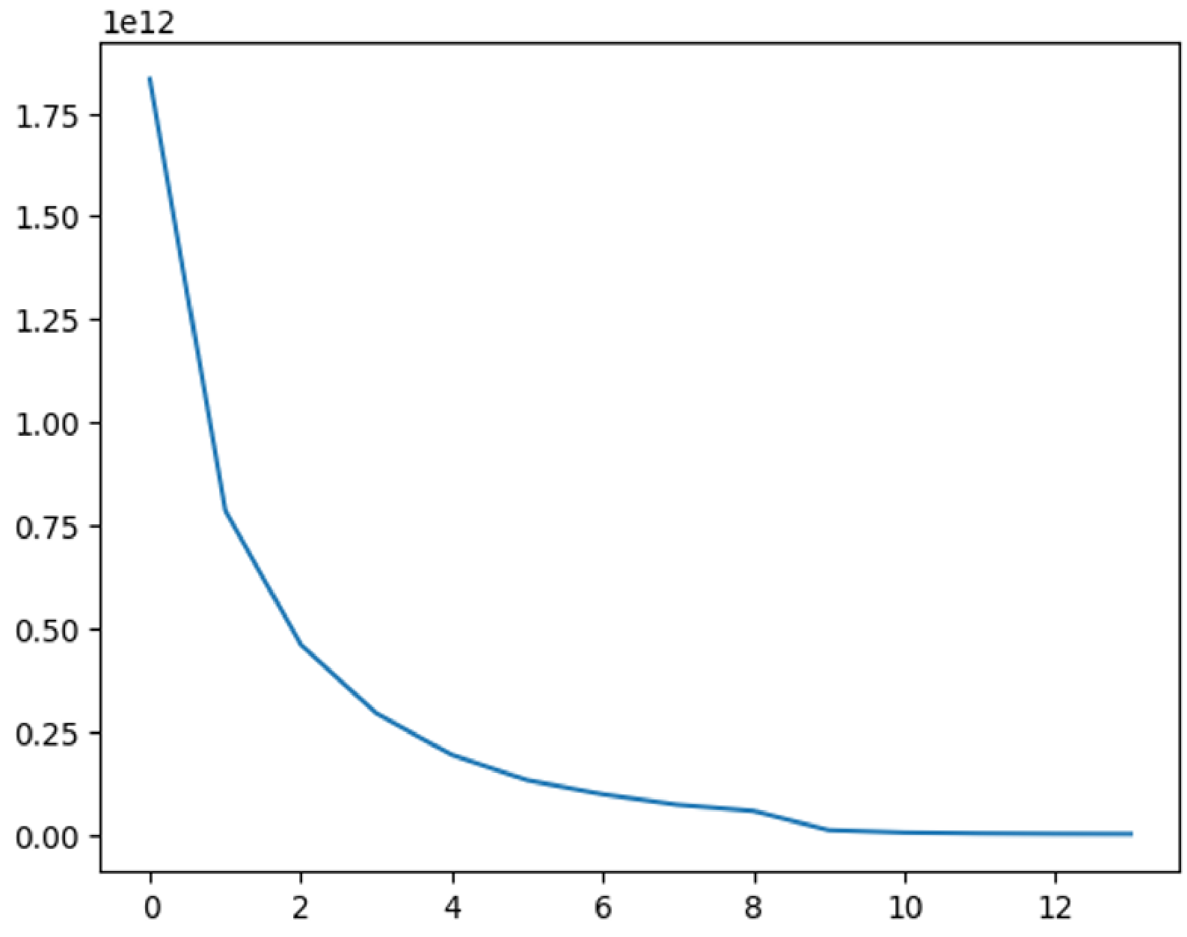


Figure 6.7: Elbow Method

This is also confirmed by Silhouette score, which reaches its maximum of 0.989 when the number of clusters is set at 2 (Table 6.1). As the number of clusters increases from 2 to 3 and so on, the Silhouette score decreases slowly, signifying a gradual moderation in the degree of similarity within a group.

Table 6.1: Cluster and Silhouette Score

Number of clusters	Score
2	0.989
3	0.987
4	0.969
5	0.942
6	0.934
7	0.906
8	0.896

We may, therefore, fix the optimal number of clusters at 2 for the set of 60 genes. This is also consistent with the binary cancer staging considered in the study: invasive and non-invasive.

6.5 Random Forest

In the random forest output for overall survival as the dependent variable, R-squared score calculated for the training data (using 75% of the dataset) turns out low at 0.0715, indicating that the model might not be a good fit. However, a low Test RMSE for Best Model at 0.1300 suggests a reasonable predictive performance: predictions deviate by approximately 0.1300 units from actual values. Test RMSE increases marginally to 0.1309 if 60% data are used for training and 40% for testing, implying stability in results on resampling. When cancer staging is the dependent variable, the test accuracy stands at a high of 0.8324, far exceeding the 'rule of thumb' for a minimum accuracy of 0.70 for model acceptance. ROC curve, which indicates the trade-off between true positive rate and false positive rate at each threshold setting remains throughout above the diagonal line. However, AUC of 0.6288 suggests that the model's ability to discriminate between invasive and non-invasive cases is somewhat better than random chance (AUC=0.5) (Figure 6.8). Overall, the model stands out in predicting cancer staging. The test accuracy increases slightly to 0.8339 if 60% data are used for training and 40% for testing, implying stability in results on resampling.

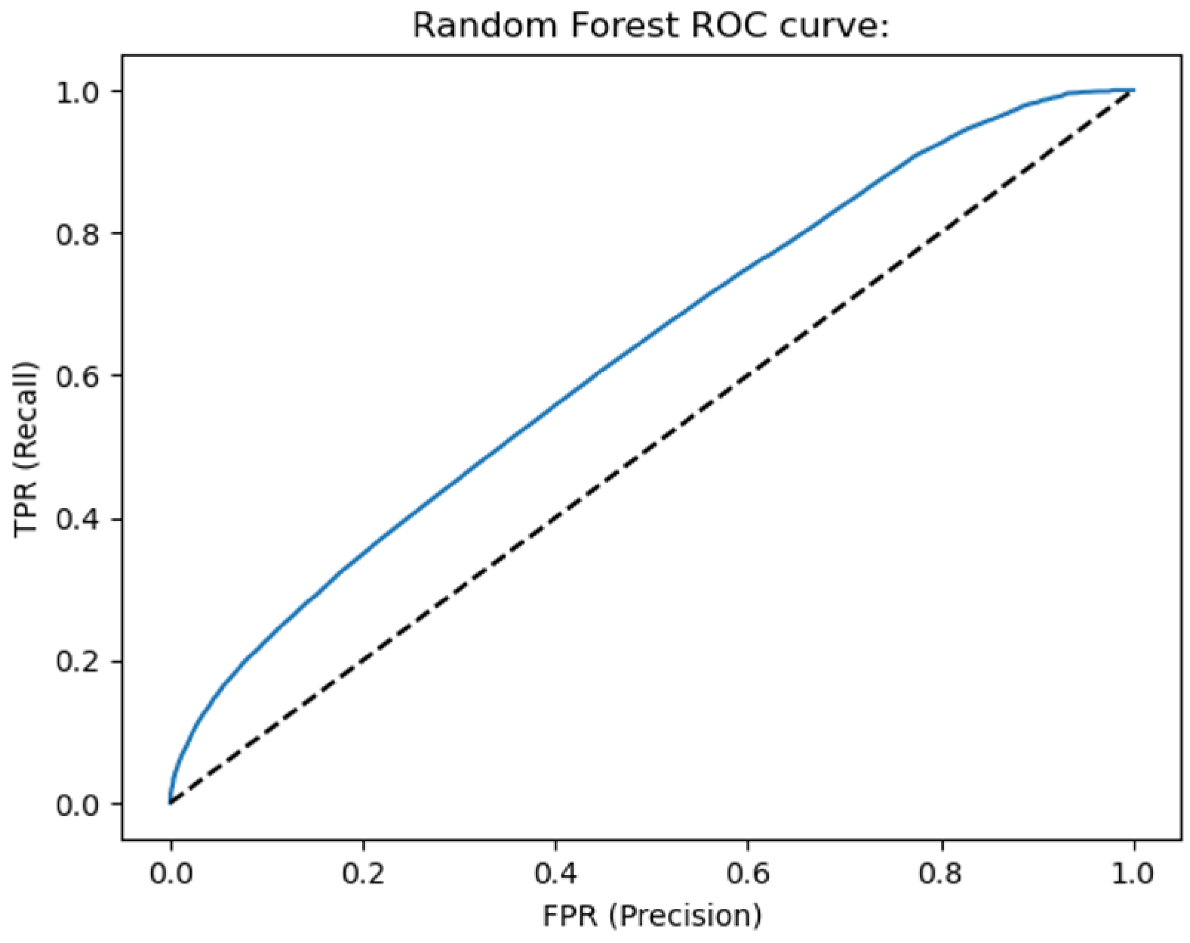


Figure 6.8: ROC

6.6 Neural Network

When cancer staging is used as the target variable, we obtain a very low test accuracy of the model at 0.1676, using 75% and 25% of data for training and testing, respectively. Threshold optimization for classification also does not improve the model's performance. We do not recommend the model in its current form for predicting cancer staging. Test accuracy of neural network model with overall survival as the target variable turns out high at 0.7528. If we set the training at 60% of the dataset, test accuracy of the model decreases marginally to 0.1661 for cancer staging while test accuracy declines fractionally to 0.7515 for overall survival, indicating relative stability in results upon resampling.

6.7 Alternative Methodology for Gene Selection & Results

We check the robustness of results based on the DE analysis by employing an alternative methodology as under. We first check the presence of all the genes with respect to patient by doing a groupby. As the gene names are found equally spread out across patients, we sort the genes by case count in descending order, shuffle the data, reset the index and then take the first 60 genes to bring in randomness into the selection of genes. The rest of the exercise – clustering, fitting a random forest model and a neural network model, and comparing the model performance – is repeated with the new-found set of 60 genes.

6.8 Clustering

The scree plot of clusters based on 60 identified genes is presented in Figure 6.9, wherein the elbow point is attained at K=2 or 3. If we add more clusters, it may not significantly reduce the sum of squared distances. This is in a way confirmed by Silhouette score, which attains its maximum at K=2. We may, therefore, fix the optimal number of clusters at 2 for the set of 60 genes based on random sampling as has been the case based on the DE analysis.

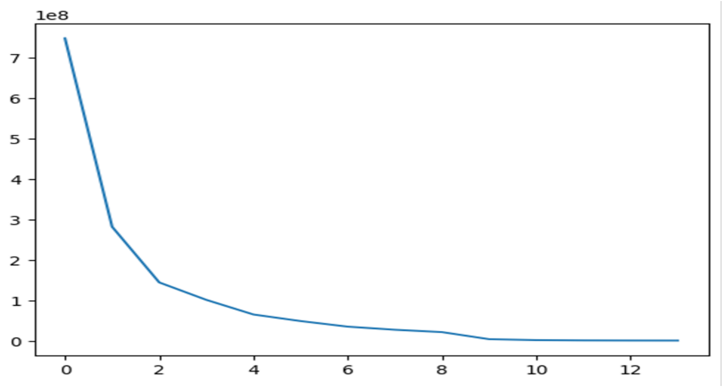


Figure 6.9: Elbow Method

6.9 Random Forest

When we take overall survival as the dependent variable, R-squared score calculated for the training data (using 75% of the dataset) turns out low at 0.2220, even though it remains higher than that based on the DE analysis at 0.0715. Also, Test RMSE for Best Model at 0.0997 turns out lower than that of 0.1300 under the DE analysis. This suggests a reasonable predictive performance of the model wherein genes are picked up based on random sampling. Test RMSE increases marginally to 0.1097 if 60% data are used for training and 40% for testing, implying stability in results on resampling. When we consider cancer staging as the dependent variable, the test accuracy stands at a high of 0.8523, which is marginally higher than that at 0.8324 under the DE analysis. ROC curve based on the random sampling approach also remains throughout above the diagonal line. Moreover, AUC of 0.7236 turns out better than that at 0.6288 under the DE analysis (Figure 6.10). Overall, the model stands out in predicting cancer staging. The test accuracy decreases slightly to 0.8507 if 60% data are used for training and 40% for testing, implying stability in results on resampling.

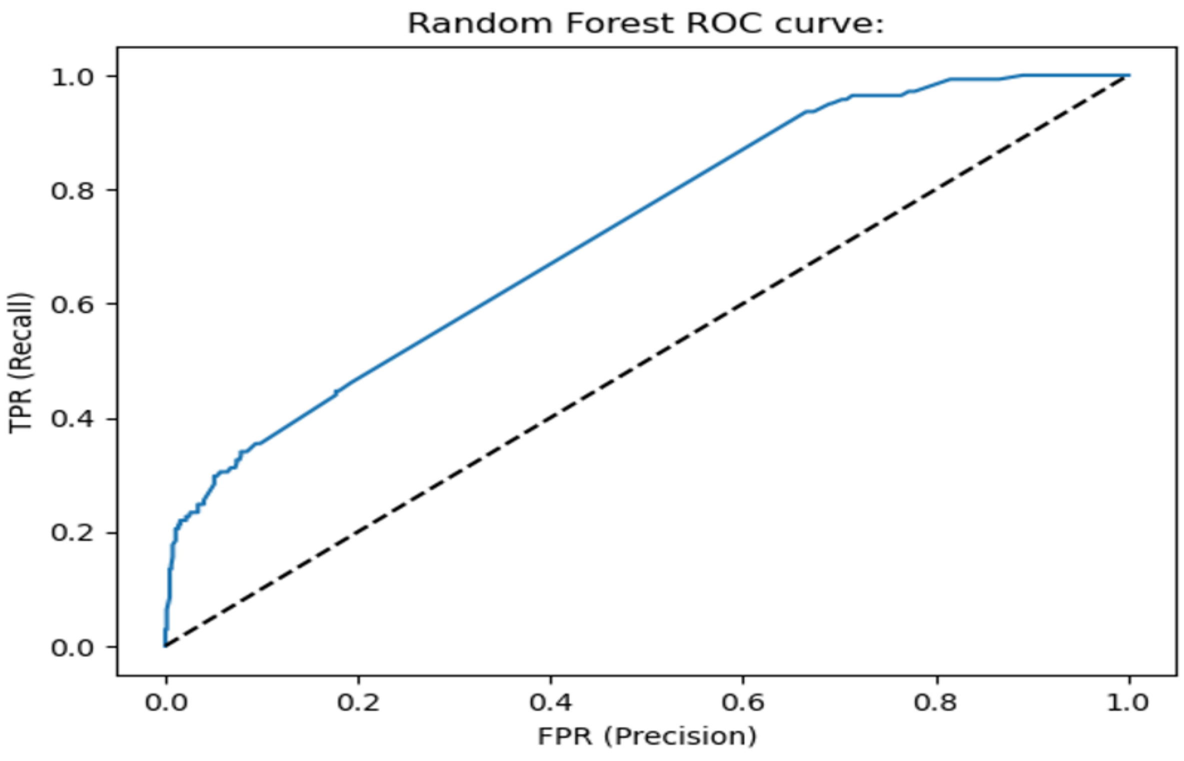


Figure 6.10: ROC

6.10 Neural Network

When we take cancer staging as the target variable, we obtain a very low test accuracy at 0.1477, which stands out marginally lower than that at 0.1676 under the DE analysis, using 75% and 25% of data for training and testing, respectively. Threshold optimization for classification also does not improve the model's performance. We do not recommend the model in its current form for predicting cancer staging. Test accuracy of neural network model with overall survival as the target variable turns out higher at 0.7817 than at 0.7528 under the DE analysis. If we set the training at 60% of the dataset, test accuracy of the neural network model increases marginally to 0.1493 for cancer staging while test accuracy declines fractionally to 0.7735 for overall survival, indicating relative stability in results upon resampling. Random forest model seems to fit the given dataset well both for explaining cancer staging and overall survival regardless of the selection process of genes – either based on the DE analysis or the random sampling of genes. Both the selection processes of genes – one based on the DE analysis and the other based on random sampling – tend to produce very similar results based on random forest or neural network even though composition of the set of 60 genes is different depending on the selection process. Further, there is relative stability in results with respect to the size of training and testing datasets irrespective of the choice of model – random forest or neural network.

CHAPTER 7

Limitations of the Study & Scope for Future Research

Gene expression data has attracted lots of attention in biomarker research. Various statistical and machine learning models are used to identify the biomarkers related to breast cancer. The models used in our study have displayed a degree of robustness upon resampling by varying the size of training and testing datasets. Results have also remained invariant to the two gene selection processes used in the study. However, the models can display low stability of feature selection in view of high dimensionality of gene data and low sample size of patients (Kim et al, 2020)[13]. Indeed, neural network model did not perform well in the study for cancer staging and need to be taken up for further investigation and improvements. Machine learning models also pose challenges in terms of interpretation and are often dubbed as black boxes. Further, notwithstanding high prediction accuracy, they may or may not be clinically relevant. The models have also assumed away inter-dependence of genes, if any. Some of these issues could be taken up for exploration as part of future research.

CHAPTER 8

A Summing Up

Breast cancer accounts for around 25% of all female cancer diagnoses and 15% of all cancer-related deaths. It exhibits a heterogeneous nature, characterized by diverse clinical presentations, treatment responses, and prognoses. This heterogeneity underscores the urgency for accurate cancer staging and overall survival prediction. In this backdrop, the study has explored the clinical and genomic aspects of breast cancer patients based on the landmark TCGA-GDC database. It observes that the age group around 60 years is most vulnerable to breast cancer. The disease, which is known as the emperor of maladies (Mukherjee, 2010) is found more prevalent among the whites[16]. Infiltrating carcinoma is the histological type, often found amongst breast cancer patients. The majority of patients in the study belong to the invasive stage II of breast cancer, for whom surgery alone may not be curative. Most of the patients have gone for chemotherapy as part of treatment. In a rigorous framework of logistic regression, the study initially attempts to link cancer staging to a number of relevant clinical variables. It brings out the lymph nodes examined count as the only clinical variable, which is statistically significant in the final equation. Based on Kaplan-Meier Estimator and Log Rank Test, groupings of patients in terms of vital status: alive and dead, tumor staging, and her2 status (ie, aggressive tumor development) are found to be meaningful analytical categories for survival analysis while the grouping based on ER status (ie, sensitivity to hormones) is not so. Given the limited utility of clinical data for survival analysis, we consider the gene expression data. With the induction of gene data, the dimensionality of dataset increases. We handle the computational challenge by applying differential expression analysis and selecting 60 genes based on p-value out of about 60,000 genes without compromising the performance efficiency. Then we carry out an unsupervised learning by clustering the 60 genes and find that the optimal number of clusters may be two. This is also consistent with the number of cancer staging being two: invasive and non-invasive, used in the study as a dependent variable. In view of the limitations of logistic regression in explaining cancer staging, we employ supervised learning, namely, random forest and neural network on the dataset. We find that the random forest model works out well with the dataset for explaining both cancer staging and overall survival. The results of the models remain robust against resampling, passing the litmus test of stability for assessment of results. They also remain largely invariant when we use random sampling for selection of genes in place of differential expression analysis. We are, however, aware that machine learning techniques often present black boxes which are not as interpretable as regression models rooted in explicit mathematical formulations. In this context, we suggest that future research can focus on unraveling the black boxes with a view to improving their utility in the medical domain.

APPENDIX A

Definitions and Data Description

1. **Estrogen Receptor Status (ER status)**

   - ER status refers to whether the breast cancer cells have receptors for the hormone estrogen. This is typically measured by immunohistochemistry (IHC) on tumor samples.
   - If more than 1% of tumor cells stain positive, the cancer is deemed ER positive. This occurs in about 70% of breast cancers. ER+ cancers tend to respond better to hormone therapies like tamoxifen that block estrogen signaling. (1)
   - Example: A pathology report states the patient's breast cancer is ER positive, with 95% of tumor cells staining positive for ER by IHC. This indicates an ER+ tumor that will likely respond well to hormonal therapy.

2. **HER2 Status**

   - HER2 is a growth-promoting protein. Tumors with increased levels of HER2 (HER2 positive) tend to be more aggressive. (2)
   - HER2 status is measured by IHC and/or fluorescent in situ hybridization (FISH). FISH counts the number of HER2 gene copies within cancer cells.
   - Example: A patient's tumor is scored 3+ on IHC staining, indicating strong, uniform staining in >10% of tumor cells. This confirms HER2 positivity.

Alternatively, FISH showing HER2 gene amplification (HER2:Chr17 ratio > 2) would confirm a HER2+ result.

3. **Lymph Node Involvement**
   - Lymph nodes near the breast are often the first place breast cancer spreads. Assessing if cancer is present helps determine prognosis and need for additional treatments. (3)
   - Example: A pathology report indicates 2 of 15 lymph nodes examined contained metastatic breast carcinoma. This suggests the cancer has started spreading to the lymph nodes. More aggressive adjuvant chemotherapy may be warranted.
4. **Treatment Implications**
   - Early stage cancers are treated with surgery and radiation. Additional treatments like chemotherapy or hormonal therapy are considered for node positive or higher risk cancers to reduce recurrence risk. (4)
   - Example: A large tumor with close surgical margins and cancer in 3 lymph nodes has a higher risk of recurrence. Adjuvant chemotherapy followed by 5 years of hormone therapy is typically recommended.
5. **`form_completion_date`**
   - This field typically indicates the date when a particular form or document related to cancer research or medical records was completed. It's essential for tracking the timing of data collection.
6. **`prospective_collection`**
   - This field is used to indicate whether the data was collected prospectively, meaning it was gathered in real-time as events occurred, or retrospectively, meaning it was collected from historical records.
   - Example: Prospective collection may involve ongoing clinical trials where data is collected as patients receive treatment.
7. **`retrospective_collection`**

- Similar to prospective collection, this field indicates whether the data was collected retrospectively, often from existing medical records or databases.
- Example: Retrospective collection might involve studying the medical records of cancer patients over the past decade to analyze treatment outcomes.

8. **`menopause_status`**
   - This field refers to a patient's menopause status, whether they are pre-menopausal, postmenopausal, or perimenopausal.
   - Example: Understanding menopause status is crucial in breast cancer research, as hormone receptor status can influence treatment decisions.

9. **`history_other_malignancy`**
   - Indicates whether the patient has a history of other malignancies or cancer types apart from the current diagnosis.
   - Example: A patient with a history of lung cancer who is now diagnosed with breast cancer would have a history of other malignancy.

10. **`history_neoadjuvant_treatment`**
    - This field indicates whether the patient received neoadjuvant treatment (treatment before surgery).
    - Example: A breast cancer patient receiving chemotherapy before surgery to shrink the tumor.

11. **`radiation_treatment_adjuvant`**
    - Specifies whether adjuvant radiation therapy was administered after primary treatment.
    - Example: Post-surgery radiation therapy for breast cancer patients to reduce the risk of recurrence.

12. **`pharmaceutical_tx_adjuvant`**
    - Indicates the use of adjuvant pharmaceutical treatments, such as hormone therapy or targeted therapy.

- Example: Adjuvant hormone therapy for hormone receptor-positive breast cancer.

13. **`histologic_diagnosis_other`**
    - If the standard histologic diagnosis doesn't apply, this field captures other specific histologic diagnoses.
    - Example: In rare cases, a cancer type might have unique histological features not covered by standard classifications.
14. **`initial_pathologic_dx_year`**
    - The year in which the initial pathologic diagnosis was made.
    - Example: A patient diagnosed with breast cancer in 2020 would have an `initial_pathologic_dx_year` of 2020.
15. **`method_initial_path_dx`**
    - Describes the method used for the initial pathologic diagnosis, which can include methods like biopsy or imaging.
    - Example: A breast cancer diagnosis may involve a core needle biopsy for pathologic diagnosis.
16. **`method_initial_path_dx_other`**
    - If the method used for the initial pathologic diagnosis is not covered by standard categories, this field allows for specifying other methods.
    - Example: A specialized imaging technique used for diagnosing rare tumors.
17. **`surgical_procedure_first`**
    - Indicates the type of surgical procedure performed as the first treatment.
    - Example: Lumpectomy or mastectomy for breast cancer.
18. **`first_surgical_procedure_other`**
    - If the first surgical procedure is not covered by standard categories, this field allows for specifying other procedures.

- Example: Complex reconstructive surgery following tumor removal.

19. **`margin_status`**

    - Specifies whether surgical margins were clear or involved by tumor cells.
    - Example: Negative margin status indicates clear margins, while positive margin status means tumor cells were found at the edges of the surgical specimen.

20. **`surgery_for_positive_margins`**

    - Indicates whether additional surgery was performed due to positive margin status.
    - Example: Re-excision surgery following positive margin status in breast cancer surgery.

21. **`surgery_for_positive_margins_other`**

    - Allows for specifying other surgical procedures performed in case of positive margins if they don't fit standard categories.
    - Example: A specialized resection technique used for specific tumor types.

22. **`margin_status_re-excision`**

    - Describes the margin status after re-excision surgery.
    - Example: After re-excision, margin status may change from positive to negative.

23. **`axillary_staging_method`**

    - Refers to the method used for staging axillary lymph nodes in breast cancer or similar procedures in other cancers.
    - Example: Sentinel lymph node biopsy as an axillary staging method in breast cancer.

24. **`axillary_staging_method_other`**

- Allows for specifying other methods used for axillary staging if they are not covered by standard categories.
- Example: Rarely used axillary staging techniques.

25. **`micromet_detection_by_ihc`**
    - Indicates whether micrometastases were detected using immunohistochemistry (IHC).
    - Example: Detecting micrometastases in lymph nodes through IHC staining.
26. **`lymph_nodes_examined`**
    - The total number of lymph nodes examined during surgery.
    - Example: If 20 lymph nodes were removed and examined, this field would contain the value 20.
27. **`lymph_nodes_examined_count`**
    - Specifies the count of lymph nodes examined, which can be crucial for staging.
    - Example: If only 10 out of 20 lymph nodes examined were found to be positive for cancer, this field would contain the value 10.
28. **`lymph_nodes_examined_he_count`**
    - This field might represent the count of lymph nodes examined using hematoxylin and eosin (HE) staining.
    - Example: Count of lymph nodes examined using HE staining to determine metastasis.
29. **`lymph_nodes_examined_ihc_count`**
    - Indicates the count of lymph nodes examined using immunohistochemistry (IHC) staining.
    - Example: The number of lymph nodes examined with IHC to detect micrometastases.

30. **`ajcc_staging_edition`**

    - Refers to the edition of the American Joint Committee on Cancer (AJCC) staging system used for cancer staging.
    - Example: AJCC 8th edition for breast cancer staging.

31. **`ajcc_tumor_pathologic_pt`**

    - Represents the pathologic T stage in the AJCC staging system, indicating the primary tumor size.
    - Example: pT2 for a breast tumor larger than 2 cm but not larger than 5 cm.

32. **`ajcc_nodes_pathologic_pn`**

    - Indicates the pathologic N stage in the AJCC staging system, describing the extent of regional lymph node involvement.
    - Example: pN1a for breast cancer with metastasis to movable ipsilateral axillary lymph nodes.

33. **`ajcc_metastasis_pathologic_pm`**

    - Refers to the pathologic M stage in the AJCC staging system, indicating the presence or absence of distant metastases.
    - Example: pM0 for no distant metastases in breast cancer.

34. **`ajcc_pathologic_tumor_stage`**

    - Represents the overall pathologic stage of cancer according to the AJCC staging system.
    - Example: Stage IIA for breast cancer with specific pT and pN stages.

35. **`metastasis_site`**

    - Specifies the site(s) of metastasis in cases where cancer has spread to distant locations.
    - Example: Liver, lung, bone, or brain metastasis in various cancer types.

36. **`metastasis_site_other`**

- Allows for specifying other sites of metastasis not covered by standard categories.
- Example: Rare sites of metastasis that may require special consideration.

37. **`er_status_by_ihc`**
    - Indicates estrogen receptor (ER) status determined by immunohistochemistry (IHC) staining.
    - Example: ER-positive or ER-negative status in breast cancer.
38. **`er_status_ihc_Percent_Positive`**
    - Represents the percentage of ER-positive cells determined by IHC.
    - Example: If 80% of tumor cells are ER-positive, this field would contain the value 80.
39. **`er_positivity_scale_used`**
    - Specifies the scale used to measure ER positivity.
    - Example: The Allred scoring system or H-score.
40. **`er_ihc_score`**
    - Represents the ER IHC score based on staining intensity and percentage of positive cells.
    - Example: An ER IHC score of 7 on the Allred scale.
41. **`er_positivity_scale_other`**
    - Allows for specifying other ER positivity scales if they are used.
    - Example: A novel ER positivity scoring system developed for research purposes.
42. **`er_positivity_method`**
    - Describes the method used to determine ER positivity.
    - Example: Immunohistochemistry (IHC) or reverse transcription-polymerase chain reaction (RT-PCR).

43. **`pr_status_by_ihc`**

    - Indicates progesterone receptor (PR) status determined by immunohistochemistry (IHC).
    - Example: PR-positive or PR-negative status in breast cancer.

44. **`pr_status_ihc_percent_positive`**

    - Represents the percentage of PR-positive cells determined by IHC.
    - Example: If 60% of tumor cells are PR-positive, this field would contain the value 60.

45. **`pr_positivity_scale_used`**

    - Specifies the scale used to measure PR positivity.
    - Example: The Allred scoring system or H-score.

46. **`pr_positivity_ihc_intensity_score`**

    - Represents the PR IHC intensity score, which is part of the PR positivity assessment.
    - Example: An PR IHC intensity score of 3 indicating strong staining.

47. **`pr_positivity_scale_other`**

    - Allows for specifying other PR positivity scales if used.
    - Example: A custom PR positivity scoring system developed for specific research.

48. **`pr_positivity_define_method`**

    - Describes the method used to define PR positivity.
    - Example: Immunohistochemistry (IHC) or gene expression profiling.

49. **`her2_status_by_ihc`**

    - Indicates human epidermal growth factor receptor 2 (HER2) status determined by immunohistochemistry (IHC).

- Example: HER2-positive or HER2-negative status in breast cancer.

50. **`her2_ihc_percent_positive`**
    - Represents the percentage of cells with HER2-positive staining determined by IHC.
    - Example: If 90% of tumor cells show HER2-positive staining, this field would contain the value 90.

51. **`her2_ihc_score`**
    - Represents the HER2 IHC score based on staining intensity and percentage of positive cells.
    - Example: An HER2 IHC score of 3+ indicating strong staining.

52. **`her2_positivity_scale_other`**
    - Allows for specifying other HER2 positivity scales if used.
    - Example: A modified HER2 positivity scoring system for research purposes.

53. **`her2_positivity_method_text`**
    - Describes the method used to assess HER2 positivity.
    - Example: Immunohistochemistry (IHC), fluorescence in situ hybridization (FISH), or gene expression profiling.

54. **`her2_fish_status`**
    - Indicates HER2 status determined by fluorescence in situ hybridization (FISH).
    - Example: HER2 amplified or not amplified status in breast cancer.

55. **`her2_copy_number`**
    - Represents the HER2 copy number determined by FISH.
    - Example: If the HER2 copy number is 5, this field would contain the value 5.

56. **`cent17_copy_number`**

- Represents the cent17 copy number determined by FISH.
- Example: If the cent17 copy number is 2, this field would contain the value 2.

57. **`her2_and_cent17_cells_count`**
    - Indicates the count of cells in which both HER2 and cent17 were counted during FISH analysis.
    - Example: If 100 cells were counted, this field would contain the value 100.

58. **`her2_cent17_ratio`**
    - Represents the ratio of HER2 copy number to cent17 copy number in FISH analysis.
    - Example: If the HER2/cent17 ratio is 2.5, this field would contain the value 2.5.

59. **`her2_and_cent17_scale_other`**
    - Allows for specifying other scales or methods used in HER2 and cent17 copy number analysis.
    - Example: A novel scale developed for specific research purposes.

60. **`her2_fish_method`**
    - Describes the method used for HER2 fluorescence in situ hybridization (FISH) analysis.
    - Example: Standard FISH or alternative FISH techniques.

61. **`new_tumor_event_dx_indicator`**
    - Indicates whether a new tumor event or recurrence was diagnosed.
    - Example: Used to mark whether a patient has experienced a cancer recurrence.

62. **`nte_er_status`**
    - ER status of the new tumor event, if applicable.

- Example: To determine if the recurrent tumor retains its ER-positive or ER-negative status.

63. **nte_er_status_ihc__positive**
    - Indicates whether the ER status of the recurrent tumor is positive.
    - Example: To track changes in ER status between the primary tumor and recurrence.

64. **nte_er_ihc_intensity_score**
    - Represents the intensity score for ER staining in the recurrent tumor.
    - Example: To assess changes in ER staining intensity in recurrent cancer.

65. **nte_er_positivity_other_scale**
    - Allows for specifying alternative scales or methods used to assess ER positivity in the recurrent tumor.
    - Example: A modified scoring system for ER positivity in recurrent cancer research.

66. **nte_er_positivity_define_method**
    - Describes the method used to define ER positivity in the recurrent tumor.
    - Example: Immunohistochemistry (IHC) or gene expression profiling in recurrent cancer analysis.

67. **nte_pr_status_by_ihc**
    - PR status of the new tumor event, if applicable.
    - Example: To determine if the recurrent tumor retains its PR-positive or PR-negative status.

68. **nte_pr_status_ihc__positive**
    - Indicates whether the PR status of the recurrent tumor is positive.
    - Example: To track changes in PR status between the primary tumor and recurrence.

69. **`nte_pr_ihc__score`**

    - Represents the intensity score for PR staining in the recurrent tumor.
    - Example: To assess changes in PR staining intensity in recurrent cancer.

70. **`nte_pr_positivity_other_scale`**

    - Allows for specifying alternative scales or methods used to assess PR positivity in the recurrent tumor.
    - Example: A modified scoring system for PR positivity in recurrent cancer research.

71. **`nte_pr_positivity_define_method`**

    - Describes the method used to define PR positivity in the recurrent tumor.
    - Example: Immunohistochemistry (IHC) or gene expression profiling in recurrent cancer analysis.

72. **`nte_her2_status`**

    - HER2 status of the new tumor event, if applicable.
    - Example: To determine if the recurrent tumor retains its HER2-positive or HER2-negative status.

73. **`nte_her2_status_ihc__positive`**

    - Indicates whether the HER2 status of the recurrent tumor is positive.
    - Example: To track changes in HER2 status between the primary tumor and recurrence.

74. **`nte_her2_positivity_ihc_score`**

    - Represents the HER2 IHC score in the recurrent tumor.
    - Example: To assess changes in HER2 staining intensity in recurrent cancer.

75. **`nte_her2_positivity_other_scale`**

    - Allows for specifying alternative scales or methods used to assess HER2 positivity in the recurrent tumor.

- Example: A modified scoring system for HER2 positivity in recurrent cancer research.

76. **`nte_her2_positivity_method`**

    - Describes the method used to assess HER2 positivity in the recurrent tumor.
    - Example: Immunohistochemistry (IHC), fluorescence in situ hybridization (FISH), or gene expression profiling in recurrent cancer analysis.

77. **`nte_her2_fish_status`**

    - Indicates HER2 status of the new tumor event determined by fluorescence in situ hybridization (FISH).
    - Example: To determine if the recurrent tumor retains its HER2 amplified or not amplified status.

78. **`nte_her2_signal_number`**

    - Represents the HER2 signal number in FISH analysis of the recurrent tumor.
    - Example: To assess changes in HER2 copy number in recurrent cancer.

79. **`nte_cent_17_signal_number`**

    - Represents the cent17 signal number in FISH analysis of the recurrent tumor.
    - Example: To assess changes in cent17 copy number in recurrent cancer.

80. **`her2_cent17_counted_cells_count`**

    - Indicates the count of cells in which both HER2 and cent17 were counted during FISH analysis of the recurrent tumor.
    - Example: If 150 cells were counted, this field would contain the value 150.

81. **`nte_cent17_her2_ratio`**

    - Represents the ratio of HER2 copy number to cent17 copy number in FISH analysis of the recurrent tumor.
    - Example: To assess changes in the HER2/cent17 ratio in recurrent cancer.

82. **`nte_cent17_her2_other_scale`**
    - Allows for specifying alternative scales or methods used in HER2 and cent17 copy number analysis of the recurrent tumor.
    - Example: A novel scale developed for specific research purposes.
83. **`nte_her2_fish_define_method`**
    - Describes the method used for HER2 fluorescence in situ hybridization (FISH) analysis in the recurrent tumor.
    - Example: Standard FISH or alternative FISH techniques for recurrent cancer research.
84. **`anatomic_neoplasm_subdivision`**
    - Specifies the anatomical subdivision or location of the neoplasm (tumor).
    - Example: In breast cancer, this could specify whether the tumor is in the upper, lower, inner, or outer quadrant of the breast.
85. **`clinical_M`**
    - Represents the clinical M stage, indicating the presence or absence of distant metastases based on clinical evaluation.
    - Example: cM0 for no clinical evidence of distant metastasis.
86. **`clinical_N`**
    - Represents the clinical N stage, describing the extent of regional lymph node involvement based on clinical evaluation.
    - Example: cN1 for clinical evidence of metastasis to movable ipsilateral axillary lymph nodes.
87. **`clinical_T`**
    - Represents the clinical T stage, indicating the size and extent of the primary tumor based on clinical evaluation.
    - Example: cT2 for a clinically determined tumor size larger than 2 cm but not larger than 5 cm.

88. **clinical_stage**
    - Specifies the clinical stage of cancer based on the combined assessment of clinical T, N, and M stages.
    - Example: Stage IIIB for advanced breast cancer with specific cT, cN, and cM stages.
89. **days_to_initial_pathologic_diagnosis**
    - Indicates the number of days from a relevant reference point to the initial pathologic diagnosis.
    - Example: The number of days from the date of biopsy to the date of pathologic diagnosis.
90. **extranodal_involvement**
    - Specifies whether cancer has spread to extranodal sites outside the lymph nodes.
    - Example: Metastasis to extranodal sites like liver, lung, or bone.
91. **histological_type**
    - Describes the specific histological type of cancer, providing details about the tumor's microscopic appearance.
    - Example: Ductal carcinoma in situ (DCIS) for a type of breast cancer.
92. **icd_10**
    - Refers to the International Classification of Diseases, 10th Revision, code used for coding the diagnosis of cancer.
    - Example: C50.0 for malignant neoplasm of the nipple and areola of the female breast.
93. **icd_o_3_histology**
    - Specifies the histology code from the International Classification of Diseases for Oncology, 3rd Edition.

- Example: 8500/3 for invasive ductal carcinoma of the breast.

94. **`icd_o_3_site`**
    - Specifies the site code from the International Classification of Diseases for Oncology, 3rd Edition.
    - Example: C50.0 for the site of the nipple and areola of the female breast.

95. **`informed_consent_verified`**
    - Indicates whether informed consent for treatment or participation in research has been verified.
    - Example: Verifying that patients have provided informed consent before participating in a clinical trial.

96. **`metastatic_tumor_indicator`**
    - Specifies whether the tumor is metastatic, meaning it has spread to distant sites.
    - Example: Used to distinguish between primary and metastatic tumors.

97. **`site_of_primary_tumor_other`**
    - Allows for specifying the primary tumor site if it doesn't fit standard categories.
    - Example: Rare primary tumor sites not covered by standard classifications.

98. **`stage_other`**
    - Provides additional details or descriptors related to cancer staging if needed.
    - Example: Additional information about the extent of disease not covered by standard staging criteria.

99. **`tissue_source_site`**
    - Indicates the source or location where the tumor tissue was obtained for analysis.
    - Example: A specific hospital or research center where the tumor biopsy or specimen was collected.

APPENDIX B

Codes

B.1 Data analysis of clinical baseline data

[4][1]

```r
setwd("/Users/souravghoshhansda/Library/CloudStorage/
    OneDrive-UniversityofEssex/Dissertation")
library("TCGAbiolinks")
library("glmnet")
library("factoextra")
library("FactoMineR")
library("caret")
library("SummarizedExperiment")
library("ggplot2")
library("RColorBrewer")
library("gProfileR")
library("GenomeInfoDbData")
library("keras")
library("tensorflow")
library("dplyr")
library("DT")
library("MultiAssayExperiment")
library("maftools")
library("ComplexHeatmap")
library("ggplot2")
library("psych")
library("corrplot")
library("survival")
library("glmnet")
library("survminer")
library("writexl")
# get a list of projects
TCGAbiolinks:::getProjectSummary("TCGA-BRCA")
query1<- GDCquery(
  project = "TCGA-BRCA",
  data.category = "Clinical",
  data.type = "Clinical Supplement",
  data.format = "BCR Biotab"
)
output_query1<-getResults(query1)
GDCdownload(query1)
clinical.BCRtab.all <- GDCprepare(query1,
    summarizedExperiment = TRUE)
clinical.BCRtab.all <- na.omit(clinical.BCRtab.all)
names(clinical.BCRtab.all)
#rm(patient_stage)
#patient
patient <- rbind(clinical.BCRtab.all$clinical_patient_brca
    )
patient<-patient[-2, ]
patient<-patient[-1, ]
#checking missing values
values_to_check <- c("[Not Available]", "[Not Applicable]"
    , "[Not Evaluated]","[Unknown]","Indeterminate","
    Equivocal","[Discrepancy]","not amplified")
occurrences <- sapply(values_to_check, function(value) sum
    (patient == value, na.rm = TRUE))
# Display the results
result_table <- data.frame(Value = values_to_check,
    Occurrences = occurrences)
print(result_table)
values_to_check <- c("[Not Available]", "[Not Applicable]"
    , "[Not Evaluated]","[Unknown]","Indeterminate","
    Equivocal","[Discrepancy]","not amplified")
calculate_percentage <- function(column) {
  total_rows <- length(column)
  count <- sum(column %in% values_to_check)
  percentage <- round((count / total_rows) * 100, 0)
  return(percentage)
}
percentage_results <- sapply(patient, calculate_percentage
    )
result_table <- data.frame(Column = names(percentage_
    results), Percentage = as.integer(percentage_results))
print(result_table)
#readr::write_csv(result_table, file = "/Users/
    souravghoshhansda/Library/CloudStorage/OneDrive-
    UniversityofEssex/Dissertation/complete/missing_values_
    percentage.csv")
patient <- na.omit(patient)
dim(patient)
patient <- patient[patient$gender != "MALE", ]
patient <- patient %>%
  mutate(
    grouped_stage = case_when(
      ajcc_pathologic_tumor_stage %in% c('Stage I', 'Stage
          IA', 'Stage IB') ~ 'Stage I',
      ajcc_pathologic_tumor_stage %in% c('Stage II','Stage
          IIA', 'Stage IIB') ~ 'Stage II',
      ajcc_pathologic_tumor_stage %in% c('Stage III','
         Stage IIIA', 'Stage IIIB', 'Stage IIIC') ~ 'Stage
          III',
      ajcc_pathologic_tumor_stage == 'Stage IV' ~ 'Stage
         IV',
      TRUE ~ ajcc_pathologic_tumor_stage
    )
  )
any(colnames(patient) %in% c("vital_status","last_contact_
   days_to","death_days_to"))
which(colnames(patient) %in% c("vital_status","last_
   contact_days_to","death_days_to"))
patient[,c(14,15,16)]
patient$deceased<-ifelse(patient$vital_status=="Alive",
   FALSE,TRUE)
table(patient$deceased)
table(patient$vital_status)
patient$overall_survival<-ifelse(patient$vital_status=="
   Alive",patient$last_contact_days_to,  patient$death_
   days_to)
sum(is.na(patient$overall_survival))
class(patient$overall_survival)
max(patient$overall_survival)
table(patient$ajcc_pathologic_tumor_stage)
patient$cancer_staging <- ifelse(
    patient$ajcc_pathologic_tumor_stage %in% c("Stage II",
        "Stage IIA", "Stage IIB", "Stage III", "Stage IIIA
       ", "Stage IIIB", "Stage IIIC", "Stage IV"),
    "Invasive",
    ifelse(patient$ajcc_pathologic_tumor_stage %in% c("
       Stage I", "Stage IA", "Stage IB"),
           "Non-Invasive",
           NA)
  )
table(patient$cancer_staging)
```

```r
patient$binary_cancer_staging <- ifelse(patient$cancer_staging == "Invasive", 1, 0)
table(patient$binary_cancer_staging)
#readr::write_csv(patient, file = "/Users/souravghoshhansda/Library/CloudStorage/OneDrive-UniversityofEssex/Dissertation/complete/patient_female.csv")
table(patient$grouped_stage)
#grouped_stage_patient_female
ggplot(data = patient[!(patient$grouped_stage %in% c("[Not Available]", "[Discrepancy]","Stage X")), ], aes(x = grouped_stage)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of ajcc pathological tumor stages", x = "Stages", y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
#Race_patient_female
table(patient$race)
ggplot(data = patient[!(patient$race %in% c("[Not Available]", "[Not Evaluated]")), ],
      aes(x = race)) +
  geom_bar(fill = "lightblue") +
  labs(title = "Bar Plot of Ethnicity", x = "Ethnicity", y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
#histology_grouped_patient_female
table(patient$histological_type)
patient <- patient %>%
  mutate(
    histology_grouped = case_when(
      histological_type %in% c('Infiltrating Carcinoma NOS', 'Infiltrating Ductal Carcinoma', 'Infiltrating
        Lobular Carcinoma') ~ 'Infiltrating Carcinoma',
      TRUE ~ histological_type
    )
  )
table(patient$histology_grouped)
ggplot(data = patient[!(patient$histology_grouped %in% c("
    [Not Available]", "Mixed Histology (please specify)","
    Other, specify")), ]
, aes(x = histology_grouped)) +
  geom_bar(fill = "lightblue") +
  labs(title = "Bar Plot of Histology", x = "Histology
      Type", y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
#age_at_diagnosis_patient_female
class(patient$age_at_diagnosis)
patient$age_at_diagnosis <- as.numeric(patient$age_at_
    diagnosis)
ggplot(patient, aes(x = age_at_diagnosis)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "
      black") +
  labs(title = "Age at Diagnosis Histogram", x = "Age", y
      = "Frequency")
female_selected <- c("bcr_patient_uuid", "bcr_patient_
    barcode", "grouped_stage")
patient_female_filtered <- patient[, female_selected]
#drug
drug<-rbind(clinical.BCRtab.all$clinical_drug_brca)
drug<-drug[-2, ]
drug<-drug[-1, ]
drug <- na.omit(drug)
names(drug)
drug_selected<-c("bcr_patient_uuid", "bcr_patient_barcode"
    , "pharmaceutical_therapy_type")
```

```r
drug_filtered <- drug[, drug_selected]
#combination_patient_drug
patient_drug <- inner_join(patient_female_filtered, drug_
    filtered, by = c("bcr_patient_uuid", "bcr_patient_
    barcode"))
table(patient_drug$grouped_stage)
names(patient_drug)
dim(patient_drug)
table(patient_drug$pharmaceutical_therapy_type)
condition <- !(patient_drug$pharmaceutical_therapy_type %
    in% c("[Not Available]", "[Discrepancy]","Other,
    specify in notes","Chemotherapy|Hormone Therapy")) &
  !(patient_drug$grouped_stage %in% c("[Not Available]", "
      [Discrepancy]","Stage X"))
filtered_patient_drug <- patient_drug[condition, ]
my_table<-table(filtered_patient_drug$grouped_stage,
    filtered_patient_drug$pharmaceutical_therapy_type)
#csv_file_path <-"/Users/souravghoshhansda/Library/
    CloudStorage/OneDrive-UniversityofEssex/Dissertation/
    complete/therapy_stage.csv"
#write.table(my_table, file = csv_file_path, sep = ",",
    col.names = NA, quote = FALSE)
cross_table<-as.data.frame(table(filtered_patient_drug$
    grouped_stage, filtered_patient_drug$pharmaceutical_
    therapy_type))
names(cross_table) <- c("Stages", "Pharmaceutical_Therapy_
    Types", "Frequency")
ggplot(cross_table, aes(x = Pharmaceutical_Therapy_Types,
    y = Frequency, fill = Stages)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(title = "Bar Plot of Therapy Type by Stages",
       x = "Therapy Types", y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```r
#survival analysis
max(patient$overall_survival)
sum(is.na(patient$overall_survival))
patient$overall_survival <- as.numeric(patient$overall_
    survival)
survival_formula<-Surv(patient$overall_survival, patient$
    deceased)
#vital_status_survival
fit1 = survfit(survival_formula ~ vital_status, data=
    patient)
print(fit1)
pval1= surv_pvalue(fit1, data=patient)$pval
print(pval1)
ggsurvplot(fit1, data=patient, pval=T, risk.table=T, risk.
    table.col="strata", xlab="Days")
#tumor_stage_survival
patient$grouped_stage_filtered <- factor(patient$grouped_
    stage, levels = c("Stage I", "Stage II", "Stage III", "
    Stage IV"))
fit2 = survfit(survival_formula ~ grouped_stage_filtered,
    data=patient)
print(fit2)
ggsurvplot(fit2, data=patient, pval=T, risk.table=T, risk.
    table.col="strata",xlab="Days")
pval2= surv_pvalue(fit2, data=patient)$pval
print(pval2)
#er_status_by_ihc_survival
patient$er_status_filtered <- factor(patient$er_status_by_
    ihc, exclude = c("[Not Evaluated]", "Indeterminate"))
fit4 = survfit(survival_formula~ er_status_filtered, data=
    patient)
print(fit4)
ggsurvplot(fit4, data=patient, pval=T, risk.table=T, risk.
    table.col="strata",xlab="Days")
```

```r
pval4= surv_pvalue(fit4, data=patient)$pval
print(pval4)
#her2_status_by_ihc_survival
patient$her2_status_filtered <- factor(patient$her2_status
    _by_ihc, exclude = c("[Not Available]", "[Not Evaluated
    ]", "Equivocal", "Indeterminate"))
fit5 = survfit(survival_formula ~ her2_status_filtered,
    data=patient)
print(fit5)
ggsurvplot(fit5, data=patient, pval=T, risk.table=T, risk.
    table.col="strata",xlab="Days")
pval5= surv_pvalue(fit5, data=patient)$pval
print(pval5)
#summary statistics
predictors_response<-patient[, c("age_at_diagnosis", "
    menopause_status","margin_status","lymph_nodes_examined
    _count","er_status_by_ihc","pr_status_by_ihc","history_
    other_malignancy","lymph_nodes_examined","her2_status_
    by_ihc")]
describe(predictors_response)
#write.csv(describe(predictors_response), file = "/Users/
    souravghoshhansda/Library/CloudStorage/OneDrive-
    UniversityofEssex/Dissertation/complete/summary_
    statistics.csv")
#logistic regression
patient$lymph_nodes_examined_count<-as.numeric(patient$
    lymph_nodes_examined_count)
set.seed(123)
trainingRowIndex <- sample(1:nrow(patient), 0.8 * nrow(
    patient))
trainingData <- patient[trainingRowIndex, ]
testData <- patient[-trainingRowIndex, ] %>% na.omit()   #
    Remove NAs for test data
# Fit the initial logistic regression model
```

```
model1 <- glm(binary_cancer_staging ~ age_at_diagnosis +
                menopause_status +
                margin_status +
                lymph_nodes_examined_count +
                er_status_by_ihc +
                pr_status_by_ihc+
                history_other_malignancy+
                lymph_nodes_examined +
                her2_status_by_ihc, data = trainingData,
                   family = "binomial")
summary(model1)
# backward variable selection
backward_model <- step(model1, direction = "backward",
   trace = 0)
summary(backward_model)
#output_text <- capture.output(summary(backward_model))
#csv_file_path <- "/Users/souravghoshhansda/Library/
   CloudStorage/OneDrive-UniversityofEssex/Dissertation/
   complete/backward_model_summary_text.txt"
#writeLines(output_text, con = csv_file_path)
% Code sources:
% - ChatGPT: https://www.chatbot.com/
%TCGABOILINKS:https://bioconductor.org/packages/devel/bioc
   /vignettes/TCGAbiolinks/inst/doc/download_prepare.html
```

B.2 Differential Expression Analysis on Gene Data

[14]

```r
setwd("/Users/souravghoshhansda/Library/CloudStorage/
    OneDrive-UniversityofEssex/Dissertation/test56")
#Load required Libraries
library("TCGAbiolinks")
library("limma")
library("edgeR")
library("glmnet")
library("factoextra")
library("FactoMineR")
library("caret")
library("SummarizedExperiment")
library("gplots")
library("survival")
library("survminer")
library("RColorBrewer")
library("gProfileR")
library("gprofiler2")
library("genefilter")
library(neuralnet)
library("dplyr")
library("tidyr")
library("tibble")
library("data.table")
query_TCGA = GDCquery(
  project = "TCGA-BRCA",
  data.category = "Transcriptome Profiling", # parameter
      enforced by GDCquery
  experimental.strategy = "RNA-Seq",
  workflow.type = "STAR - Counts",
  sample.type = c("Primary Tumor","Solid Tissue Normal"),
  data.type = "Gene Expression Quantification",
  access = "open")
brca_res = getResults(query_TCGA) # make results as table
# head(brca_res) # data of the first 6 patients.
colnames(brca_res) # columns present in the table
head(brca_res$sample_type)
summary(factor(brca_res$sample_type))
GDCdownload(query = query_TCGA)
tcga_data = GDCprepare(query_TCGA)
dim(tcga_data)
colnames(colData(tcga_data))
table(tcga_data@colData$vital_status)
table(tcga_data@colData$gender)
table(tcga_data@colData$race)
table(tcga_data$definition)
dim(assay(tcga_data))
head(assay(tcga_data)[,1:10])
head(rowData(tcga_data))
saveRDS(object = tcga_data, file = "tcga_data.RDS",
    compress = FALSE)
# To load the previous data
tcga_data = readRDS(file = "tcga_data.RDS")
clinical_data = colData(tcga_data)
any(colnames(clinical_data) %in% c("vital_status","days_to
    _last_follow_up","days_to_death"))
which(colnames(clinical_data) %in% c("vital_status","days_
    to_last_follow_up","days_to_death"))
clinical_data[,c(28,50,63)]
clinical_data$deceased<-ifelse(clinical_data$vital_status
    =="Alive",FALSE,TRUE)
table(clinical_data$deceased)
table(clinical_data$vital_status)
clinical_data$overall_survival<ifelse(clinical_data$vital_
    status=="Alive",clinical_data$days_to_last_follow_up,
clinical_data$days_to_death)
```

```r
sum(is.na(clinical_data$overall_survival))
class(clinical_data$overall_survival)
max(clinical_data$overall_survival)
table(clinical_data$ajcc_pathologic_stage)
clinical_data$cancer_staging <- ifelse(
  clinical_data$ajcc_pathologic_stage %in% c("Stage II","Stage IIA","Stage IIB","Stage III", "Stage IIIA", "Stage IIIB", "Stage IIIC", "Stage IV"),
  "Invasive",
  ifelse(clinical_data$ajcc_pathologic_stage %in% c("Stage I", "Stage IA", "Stage IB"),
         "Non-Invasive",
         NA)
)
table(clinical_data$cancer_staging)
clinical_data$binary_cancer_staging <- ifelse(clinical_data$cancer_staging == "Invasive", 1, 0)
table(clinical_data$binary_cancer_staging)
names(clinical_data)
group = factor(clinical_data$definition)
group = relevel(group, ref="Solid Tissue Normal")
design = model.matrix(~group)
head(design)
dge = DGEList( # creating a DGEList object
  counts=assay(tcga_data),
  samples=colData(tcga_data),
  genes=as.data.frame(rowData(tcga_data)))
# filtering
keep = filterByExpr(dge,design) # defining which genes to keep
dge = dge[keep,,keep.lib.sizes=FALSE] # filtering the dge object
rm(keep) #  use rm() to remove objects from memory if you don’t need them anymore
```

```r
dge = calcNormFactors(dge,method="TMM")
v = voom(dge,design,plot=TRUE)
fit = lmFit(v, design)
fit = eBayes(fit)
topGenes = topTable(fit, coef=1, sort.by="p")
print(topGenes)
limma_pipeline = function(tcga_data, condition_variable,
   reference_group = NULL) {
design_factor = colData(tcga_data)[, condition_variable,
   drop = TRUE]
  group = factor(design_factor)
  if (!is.null(reference_group)) {
    group = relevel(group, ref = reference_group)
  }
  design = model.matrix(~ group)
  dge = DGEList(counts = assay(tcga_data),
                samples = colData(tcga_data),
                genes = as.data.frame(rowData(tcga_data)))
  # Filtering
  keep = filterByExpr(dge, design)
  dge = dge[keep, , keep.lib.sizes = FALSE]
  rm(keep)
  # Normalization (TMM followed by voom)
  dge = calcNormFactors(dge)
  v = voom(dge, design = design, plot = TRUE)  # Pass the
     design matrix directly
  # Fit model to data given design
  fit = lmFit(v, design)
  fit = eBayes(fit)
  # Show top genes
  topGenes = topTable(fit, coef = ncol(design), number =
     60, sort.by = "p")
  return(
    list(
      voomObj = v,    # normalized data
      fit = fit,      # fitted model and statistics
      topGenes = topGenes  # the 22000 most differentially
          expressed genes
    )
  )
}
limma_res = limma_pipeline(
  tcga_data = tcga_data,
  condition_variable = "definition",
  reference_group = "Solid Tissue Normal"
)
saveRDS(object = limma_res,
        file = "limma_res.RDS",
        compress = FALSE)
limma_res = readRDS(file = "limma_res.RDS")
plot_PCA = function(voomObj, condition_variable){
  group = factor(voomObj$targets[, condition_variable])
  pca = prcomp(t(voomObj$E))
  # Take PC1 and PC2 for the plot
  plot(pca$x[,1:2],col=group, pch=19)
  # include a legend for points
  legend("bottomright", inset=.01, levels(group), pch=19,
     col=1:length(levels(group)))
  return(pca)
}
res_pca = plot_PCA(limma_res$voomObj, "definition")
head(limma_res$topGenes)
geneid <- limma_res$topGenes$gene_id
genename <- limma_res$voomObj$genes[geneid, "gene_name"]
genecounts <- assay(tcga_data)[geneid, ]
length(geneid)
length(genename)
GenesData <- data.frame(
```

```r
  gene_id = geneid,
  gene_name = genename,
  counts = as.vector(genecounts),
  case_id = colnames(genecounts)
  )
#readr::write_csv(GenesData, file = "/Users/
   souravghoshhansda/Library/CloudStorage/OneDrive-
   UniversityofEssex/Dissertation/complete/GenesData.csv")
clinical_data <- as.data.frame(clinical_data)
combined_data_final <- GenesData %>%
  merge(clinical_data, by.x = 'case_id', by.y = 'barcode')
names(combined_data_final)
#readr::write_csv(combined_data_final, file = "/Users/
   souravghoshhansda/Library/CloudStorage/OneDrive-
   UniversityofEssex/Dissertation/complete/combined_data_
   final.csv")
combined_data_final_selected <- c("overall_survival", "
   cancer_staging", "submitter_id", "binary_cancer_staging
   ","gene_name","gene_id","counts")
selected_combined_data_final <- combined_data_final[,
   combined_data_final_selected]
sum(is.na(selected_combined_data_final))
max(selected_combined_data_final$overall_survival)
selected_combined_data_final<-na.omit(selected_combined_
   data_final)
selected_combined_data_final$overall_survival<-selected_
   combined_data_final$overall_survival/8605
#readr::write_csv(selected_combined_data_final, file = "/
   Users/souravghoshhansda/Library/CloudStorage/OneDrive-
   UniversityofEssex/Dissertation/complete/selected_
   combined_data_final_60.csv"
gene_expression_data<-as.data.frame(assay(tcga_data))
gene_metadata <- as.data.frame(rowData(tcga_data))
combined_data <- gene_expression_data %>%
  as.data.frame() %>%
  rownames_to_column(var = 'gene_id') %>%
  gather(key = 'case_id', value = 'counts', -gene_id) %>%
  left_join(., gene_metadata, by = "gene_id")
combined_data$case_id <- gsub('-01.*', '', combined_data$case_id)
combined_data_selected <- c("case_id", "counts", "gene_name")
selected_combined_data <- combined_data[, combined_data_selected]
clinical_data_selected <- c("overall_survival", "cancer_staging", "submitter_id", "binary_cancer_staging")
selected_clinical_data <- clinical_data[, clinical_data_selected]
combined_data_final_60000 <- selected_combined_data %>%
  merge(selected_clinical_data, by.x = 'case_id', by.y = 'submitter_id')
sum(is.na(combined_data_final_60000$overall_survival))
sum(is.na(combined_data_final_60000))
max(combined_data_final_60000$overall_survival)
combined_data_final_60000<-na.omit(combined_data_final_60000)
combined_data_final_60000$overall_survival<-combined_data_final_60000$overall_survival/8605
head(combined_data_final_60000)
#readr::write_csv(combined_data_final_60000, file = "/Users/souravghoshhansda/Library/CloudStorage/OneDrive-UniversityofEssex/Dissertation/complete/combined_data_final_60000.csv")
gene_expression_data<-as.data.frame(assay(tcga_data))
gene_metadata <- as.data.frame(rowData(tcga_data))
combined_data <- gene_expression_data %>%
  as.data.frame() %>%
  rownames_to_column(var = 'gene_id') %>%
  gather(key = 'case_id', value = 'counts', -gene_id) %>%
  left_join(., gene_metadata, by = "gene_id")
combined_data$case_id <- gsub('-01.*', '', combined_data$
    case_id)
combined_data_selected <- c("case_id", "counts", "gene_
    name")
selected_combined_data <- combined_data[, combined_data_
    selected]
clinical_data_selected <- c("overall_survival", "cancer_
    staging", "submitter_id", "binary_cancer_staging")
selected_clinical_data <- clinical_data[, clinical_data_
    selected]
combined_data_final_60000 <- selected_combined_data %>%
  merge(selected_clinical_data, by.x = 'case_id', by.y = '
      submitter_id')
sum(is.na(combined_data_final_60000$overall_survival))
sum(is.na(combined_data_final_60000))
max(combined_data_final_60000$overall_survival)
combined_data_final_60000<-na.omit(combined_data_final_
    60000)
combined_data_final_60000$overall_survival<-combined_data_
    final_60000$overall_survival/8605
head(combined_data_final_60000)
#readr::write_csv(combined_data_final_60000, file = "/
    Users/souravghoshhansda/Library/CloudStorage/OneDrive-
    UniversityofEssex/Dissertation/complete/combined_data_
    final_60000.csv")
% Code sources:
% - ChatGPT: https://www.chatbot.com/
% - :https://www.costalab.org/wp-content/uploads/2020/11/R
    _class_D3.html#3_RNASeq_Normalization
```

[14]

B.3 Gene Selection by Random Sampling

[4]

```python

import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from matplotlib
import pyplot as plt
%matplotlib inline
import warningswarnings.filterwarnings("ignore")
df= pd.read_csv("/content/combined_data_final_60000.csv")
df.head(2)
df.shape
gene_counts = df.groupby('gene_name')['case_id'].nunique()
    .reset_index()gene_counts.columns = ['Gene_Name', 'Case
    _id_Count']

gene_counts = gene_counts.sort_values(by='Case_id_Count',
    ascending=False)

gene_counts

 df_shuffled = gene_counts.sample(frac=1, random_state=42)

 only_60_genes = df_shuffled.iloc[:60]

 gene_names_list = only_60_genes['Gene_Name'].tolist()

 gene_names_list

 filtered_df = df[df['gene_name'].isin(gene_names_list)]
```

```

filtered_df

unique_gene_names = filtered_df['gene_name'].nunique()
print("Number of unique gene names:", unique_gene_names)

filtered_df.to_csv('/content/filtered_df.csv', index=
    False)

import os
print(os.getcwd())

gene_counts = filtered_df.groupby('gene_name')['case_id'
    ].nunique().reset_index()
gene_counts.columns = ['Gene_Name', 'Case_id_Count']
gene_counts = gene_counts.sort_values(by='Case_id_Count',
   ascending=False)

gene_counts.shape

df2 = filtered_df[['counts','gene_name']]
df2.head(2)
% Code sources:
% - ChatGPT: https://www.chatbot.com/
```

B.4 Supervised learning/predictive modelling:Neural Networks

[4][12][15][18]

```
import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from matplotlib import pyplot as plt
%matplotlib inline

df = pd.read_csv("/Users/souravghoshhansda/Library/
    CloudStorage/OneDrive-UniversityofEssex/Dissertation/
    complete/selected_combined_data_final_60.csv")

df.head(2)

sns.countplot(x = df.cancer_staging)

from imblearn.over_sampling import SMOTE #generating
    synthetic samples of the minority class
#from imblearn.under_sampling import RandomUnderSampler
from imblearn.pipeline import Pipeline
from collections import Counter

df.columns

df.info()

df.isnull().sum()

X = df[['counts']]
y1 = pd.get_dummies(df[['cancer_staging']], drop_first =
    True)

y1.value_counts()

from keras.models import Sequential
from keras.layers import Dense

```

```
classifier = Sequential()
classifier.add(Dense(10, activation = 'relu', input_dim =
    30))
classifier.add(Dense(10, activation = 'relu'))
classifier.add(Dense(1,  activation = 'sigmoid'))
classifier.compile(optimizer = 'adam', loss = 'binary_
    crossentropy', metrics = ['accuracy'])
classifier.summary()

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y1,
     test_size = 0.25, random_state = 0)

# Create the ANN model
model = Sequential()
model.add(Dense(10, input_dim=X.shape[1], activation='relu
    '))  # Input layer with 10 neurons
model.add(Dense(8, activation='relu'))  # Hidden layer
    with 8 neurons
model.add(Dense(y1.shape[1], activation='softmax'))  #
    Output layer with number of classes as neurons

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='
    adam', metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, epochs=1, batch_size=5,
    verbose=1, validation_split=0.1)

# Evaluate the model on test data
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test Accuracy: {accuracy:.4f}')

```

```
from sklearn import metrics
from sklearn.metrics import precision_score,recall_score

y_pred = model.predict(X_test)

precision_score(y_test, y_pred)

recall_score(y_test, y_pred)

def make_confusion_matrix(y_actual,y_predict,title):
    fig, ax = plt.subplots(1, 1)

    cm = confusion_matrix(y_actual, y_predict, labels
       =[0,1])
    disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                               display_labels=["No","Yes"
                                  ])
    disp.plot(cmap='Greens',colorbar=True,ax=ax)
    ax.set_title(title)
    plt.tick_params(axis=u'both', which=u'both',length=0)
    #plt.grid(b=None,axis='both',which='both',visible=
       False)
    plt.show()

from sklearn.metrics import confusion_matrix,
   classification_report
from sklearn.metrics import accuracy_score,precision_score
   ,recall_score,f1_score
from statsmodels.stats.outliers_influence import variance_
   inflation_factor
from sklearn import metrics
#AUC ROC curve
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
from sklearn.metrics import precision_recall_curve

from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

def get_metrics_score(model,X_train_df,X_test_df,y_train_pass,y_test_pass,statsklearn,threshold=0.5,flag=True,roc=False):

    # defining an empty list to store train and test results

    score_list=[]

    if statsklearn==0:
        pred_train = model.predict(X_train_df)
        pred_test = model.predict(X_test_df)
    else:
        pred_train = (model.predict(X_train_df)>threshold)
        pred_test = (model.predict(X_test_df)>threshold)


    pred_train = np.round(pred_train)
    pred_test = np.round(pred_test)

    train_acc = accuracy_score(y_train_pass,pred_train)
    test_acc = accuracy_score(y_test_pass,pred_test)

    train_recall = recall_score(y_train_pass,pred_train)
    test_recall = recall_score(y_test_pass,pred_test)

    train_precision = precision_score(y_train_pass,pred_train)
    test_precision = precision_score(y_test_pass,pred_test
       )

    train_f1 = f1_score(y_train_pass,pred_train)
    test_f1 = f1_score(y_test_pass,pred_test)
    #till here - all metrics are assigned variables separately


    score_list.extend((train_acc,test_acc,train_recall,test_recall,train_precision,test_precision,train_f1,test_f1))

    if flag == True:
        print("\x1b[0;30;47m \033[1mMODEL PERFORMANCE\x1b[0m")

        print("\x1b[0;30;47m \033[1mAccuracy   : Train:\x1b[0m",
              round(accuracy_score(y_train_pass,pred_train),3),
              "\x1b[0;30;47m \033[1mTest:\x1b[0m ",
              round(accuracy_score(y_test_pass,pred_test),3))

        print("\x1b[0;30;47m \033[1mRecall     : Train:\x1b[0m"
              ,round(recall_score(y_train_pass,pred_train),3),
              "\x1b[0;30;47m \033[1mTest:\x1b[0m" ,
              round(recall_score(y_test_pass,pred_test),3))

        print("\x1b[0;30;47m \033[1mPrecision  : Train:\x1b[0m",
              round(precision_score(y_train_pass,pred_train),3),
              "\x1b[0;30;47m \033[1mTest:\x1b[0m ",
              round(precision_score(y_test_pass,pred_test),3))

        print("\x1b[0;30;47m \033[1mF1         : Train:\x1b[0m",
              round(f1_score(y_train_pass,pred_train),3),
              "\x1b[0;30;47m \033[1mTest:\x1b[0m",
              round(f1_score(y_test_pass,pred_test),3))

        make_confusion_matrix(y_train_pass,pred_train,"Confusion Matrix for Train")
        make_confusion_matrix(y_test_pass,pred_test,"Confusion Matrix for Test")

    if roc == True:

        print("\x1b[0;30;47m \033[1mROC-AUC Score  :Train:\x1b[0m: ",
              round(roc_auc_score(y_train_pass,pred_train),3),
              "\x1b[0;30;47m \033[1mTest:\x1b[0m: ",
              round(roc_auc_score(y_test_pass,pred_test),3))

    return score_list # returning the list with train and test scores

# # defining empty lists to add train and test results
acc_train = []
acc_test = []
recall_train = []
```

```
recall_test = []
precision_train = []
precision_test = []
f1_train = []
f1_test = []

def add_score_model(score):
     #'''Add scores to list so that we can compare all models score together'''
    acc_train.append(score[0])
    acc_test.append(score[1])
    recall_train.append(score[2])
    recall_test.append(score[3])
    precision_train.append(score[4])
    precision_test.append(score[5])
    f1_train.append(score[6])
    f1_test.append(score[7])

statmodel=1
scores_Sklearn = get_metrics_score(model,X_train, X_test, y_train, y_test,statmodel)

optimal_threshold_curve = 0.3

scores_opt_curve = get_metrics_score(model,X_train, X_test, y_train, y_test,statmodel,threshold=optimal_threshold_curve,roc=True)
add_score_model(scores_opt_curve)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense
from tensorflow.keras.preprocessing import sequence

```

```
y = df[['overall_survival']]

from sklearn.model_selection import train_test_split
X_train1, X_test1, y_train1, y_test1 = train_test_split(X,
     y, test_size = 0.25, random_state = 0)

# Create the ANN model
model = Sequential()
model.add(Dense(10, input_dim=X.shape[1], activation='relu
    '))  # Input layer with 10 neurons
model.add(Dense(8, activation='relu'))  # Hidden layer
    with 8 neurons
model.add(Dense(y.shape[1], activation='softmax'))  #
    Output layer with number of classes as neurons

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='
    adam', metrics=['mse'])

# Train the model
model.fit(X_train1, y_train1, epochs=1, batch_size=5,
    verbose=1, validation_split=0.1)

# Evaluate the model on test data
loss, accuracy = model.evaluate(X_test1, y_test1)
print(f'Test Accuracy: {accuracy:.4f}')
% Code sources:
% - ChatGPT: https://www.chatbot.com/
% - Kanncaa1's deep learning tutorial: https://www.kaggle.
    com/code/kanncaa1/deep-learning-tutorial-for-beginners
% - Machine Learning Mastery tutorial: https://
    machinelearningmastery.com/tutorial-first-neural-
    network-python-keras/
% - Real Python tutorial: https://realpython.com/python-ai
```

```
    -neural-network/
```

B.5 Supervised learning/predictive modelling:Random Forest

[4][7][20][17][5]

```
import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from matplotlib import pyplot as plt
%matplotlib inline

df = pd.read_csv("/Users/souravghoshhansda/Library/
    CloudStorage/OneDrive-UniversityofEssex/Dissertation/
    complete/selected_combined_data_final_60.csv")

df.head(2)

df.columns

df.info()

df.isnull().sum()

X = df[['counts']]
y = df[['overall_survival']]

y

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
   test_size = 0.25, random_state = 0)

from sklearn.ensemble import RandomForestRegressor
RForest = RandomForestRegressor(n_estimators=300)
RForest.fit(X_train,y_train)
print(RForest.score(X_train,y_train))

from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV, train_
   test_split
from sklearn.metrics import mean_squared_error

rf_regressor = RandomForestRegressor()

# Define the parameters for grid search
parameters = {
    'n_estimators': [100, 200, 300],  # Number of trees in
        the forest
    'max_depth': [None, 5, 10, 15],  # Maximum depth of a
       tree
    'min_samples_split': [2, 5, 10]  # Minimum number of
       samples required to split a node
}

# Perform Grid Search with cross-validation
grid_search = GridSearchCV(estimator=rf_regressor, param_
   grid=parameters,
 scoring='neg_mean_squared_error', cv=5, n_jobs=-1)
grid_search.fit(X_train, y_train)

# Get the best parameters and best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_
```

```python

print("Best Parameters:", best_params)
print("Best Negative Mean Squared Error:", best_score)

# Evaluate the model with the best parameters on the test
    set
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)
test_rmse = mean_squared_error(y_test, y_pred, squared=
    False)
print(f"Test RMSE with Best Model: {test_rmse:.4f}")

y1 = pd.get_dummies(df[['cancer_staging']], drop_first =
    True)

X = df[['counts']]

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y1,
     test_size = 0.25, random_state = 0)

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

# Define the Random Forest classifier
rf_classifier = RandomForestClassifier()

# Define the parameters for grid search
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10, 15],
    'min_samples_split': [2, 5, 10]
}
```

```python

# Perform Grid Search with cross-validation
grid_search = GridSearchCV(estimator=rf_classifier, param_grid=param_grid,
                           scoring='accuracy', cv=5, n_jobs=-1)
grid_search.fit(X_train, y_train)

# Get the best parameters and best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_

print("Best Parameters:", best_params)
print("Best Accuracy Score:", best_score)

# Evaluate the model with the best parameters on the test set
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)
test_accuracy = accuracy_score(y_test, y_pred)
print(f"Test Accuracy with Best Model: {test_accuracy:.4f}")

from sklearn import metrics
from sklearn.metrics import precision_score,recall_score

precision_score(y_test, y_pred)

precision_score(y_test, y_pred)

recall_score(y_test, y_pred)

recall_score(y_test, y_pred)

```

```
y_train.value_counts()

y_train.value_counts()

from sklearn.metrics import  roc_auc_score,precision_
    recall_curve, auc, roc_curve

y_score = best_model.predict_proba(X_train)[:,1]
fpr, tpr, _ = roc_curve(y_train, y_score)

plt.title('Random Forest ROC curve:')
plt.xlabel('FPR (Precision)')
plt.ylabel('TPR (Recall)')

plt.plot(fpr,tpr)
plt.plot((0,1), ls='dashed',color='black')
plt.show()
print ('Area under curve (AUC): ', auc(fpr,tpr))

% Code sources:
% - GeeksforGeeks Random Forest Classifier tutorial: https
    ://www.geeksforgeeks.org/random-forest-classifier-using
    -scikit-learn/
% - Scikit-learn RandomForestClassifier documentation:
    https://scikit-learn.org/stable/modules/generated/
    sklearn.ensemble.RandomForestClassifier.html
% - Kaggle Tutorial by prashant111: https://www.kaggle.com
    /code/prashant111/random-forest-classifier-tutorial
% - DataCamp:https://www.datacamp.com/tutorial/random-
    forests-classifier-python
```

B.6 Unsupervised learning/clustering of Gene expression:Elbow method and Silhouette Score

[4]

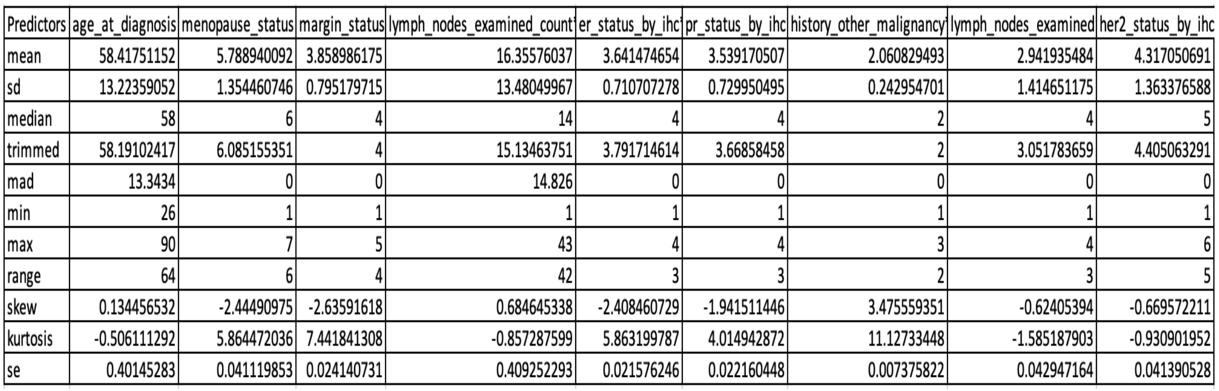
```
import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from matplotlib import pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
df2= pd.read_csv("/content/selected_combined_data_final_
    60.csv")
df2.head()
df2.shape
df2 = df2[['counts','gene_name']]
df2.head(2)
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import dendrogram
from scipy.cluster.hierarchy import cut_tree
class MultiColumnLabelEncoder:

    def __init__(self, columns=None):
        self.columns = columns # array of column names to
            encode


    def fit(self, X, y=None):
        self.encoders = {}
        columns = X.columns if self.columns is None else self.columns
        for col in columns:
            self.encoders[col] = LabelEncoder().fit(X[col])
        return self


    def transform(self, X):
        output = X.copy()
        columns = X.columns if self.columns is None else self.columns
        for col in columns:
            output[col] = self.encoders[col].transform(X[col])
        return output


    def fit_transform(self, X, y=None):
        return self.fit(X,y).transform(X)


    def inverse_transform(self, X):
        output = X.copy()
        columns = X.columns if self.columns is None else self.columns
        for col in columns:
            output[col] = self.encoders[col].inverse_transform(X[col])
        return output
        from sklearn.preprocessing import LabelEncoder
        multi = MultiColumnLabelEncoder(columns=[’gene_name’])

```

```
df2 = multi.fit_transform(df2)
#from sklearn.preprocessing import LabelEncoder
#le = LabelEncoder()
#df2.gene_name = le.fit_transform(df2.gene_name)
df2
kmeans = KMeans(n_clusters=5, max_iter=50)
kmeans.fit(df2)
ssd = []
range_n_clusters = [2, 3, 4, 5, 6, 7, 8, 9
    ,10,20,30,40,50,59]
for num_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=num_clusters, max_iter=50)
    kmeans.fit(df2)

    ssd.append(kmeans.inertia_)

# plot the SSDs for each n_clusters
plt.plot(ssd)
# Silhouette analysis
range_n_clusters = [2, 3, 4, 5, 6, 7, 8]

for num_clusters in range_n_clusters:

    # intialise kmeans
    kmeans = KMeans(n_clusters=num_clusters, max_iter=50)
    kmeans.fit(df2)

    cluster_labels = kmeans.labels_

    # silhouette score
    silhouette_avg = silhouette_score(df2, cluster_labels)
    print("For n_clusters={0}, the silhouette score is {1}
        ".format(num_clusters, silhouette_avg))
```

APPENDIX C

Outputs

C.1 Descriptive Statistics of clinical predictors



Predictors	age_at_diagnosis	menopause_status	margin_status	lymph_nodes_examined_count	er_status_by_ihc	pr_status_by_ihc	history_other_malignancy	lymph_nodes_examined	her2_status_by_ihc
mean	58.41751152	5.788940092	3.858986175	16.35576037	3.641474654	3.539170507	2.060829493	2.941935484	4.317050691
sd	13.22359052	1.354460746	0.795179715	13.48049967	0.710707278	0.729950495	0.242954701	1.414651175	1.363376588
median	58	6	4	14	4	4	2	4	5
trimmed	58.19102417	6.085155351	4	15.13463751	3.791714614	3.66858458	2	3.051783659	4.405063291
mad	13.3434	0	0	14.826	0	0	0	0	0
min	26	1	1	1	1	1	1	1	1
max	90	7	5	43	4	4	3	4	6
range	64	6	4	42	3	3	2	3	5
skew	0.134456532	-2.44490975	-2.63591618	0.684645338	-2.408460729	-1.941511446	3.475559351	-0.62405394	-0.669572211
kurtosis	-0.506111292	5.864472036	7.441841308	-0.857287599	5.863199787	4.014942872	11.12733448	-1.585187903	-0.930901952
se	0.40145283	0.041119853	0.024140731	0.409252293	0.021576246	0.022160448	0.007375822	0.042947164	0.041390528

Figure C.1: Descriptive Statistics

C.2 Logistic Regression

```
Call:
glm(formula = binary_cancer_staging ~ age_at_diagnosis+
menopause_status + margin_status + lymph_nodes_examined_count
+ er_status_by_ihc + pr_status_by_ihc + history_other_malignancy
+ lymph_nodes_examined + her2_status_by_ihc, family = "binomial",
data = trainingData)

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                    -0.76       1.42   -0.54     0.590
age_at_diagnosis               -0.01       0.01   -1.03    0.304
menopause_status[Not Evaluated]15.59    1278.34    0.01     0.990
menopause_status[Unknown]      -0.92       0.92   -0.99     0.322
menopause_statusIndeterminate  -0.78       0.87   -0.90     0.367
menopause_statusPeri           -0.51       0.69   -0.74     0.458
menopause_statusPost           -0.25       0.50   -0.51     0.611
menopause_statusPre            -0.40       0.55   -0.74     0.459
margin_status[Unknown]         15.54    1417.95    0.01     0.991
margin_statusClose             -0.35       0.78   -0.45     0.652
margin_statusNegative           0.24       0.56    0.43     0.664
margin_statusPositive           0.34       0.70    0.48     0.628
lymph_nodes_examined_count      0.17       0.02    7.93  2.26e-15***
er_status_by_ihcNegative        14.70    2399.54    0.01     0.995
er_status_by_ihcPositive        14.70    2399.54    0.01     0.995
pr_status_by_ihcIndeterminate  -0.19    2675.00    0.00     0.999
pr_status_by_ihcNegative       -14.62    2399.54   -0.01     0.995
pr_status_by_ihcPositive       -14.93    2399.54   -0.01     0.995
history_other_malignancyYes     0.18       0.45    0.40     0.687
lymph_nodes_examinedYES         0.10       0.25    0.39     0.699
her2_status_by_ihc[Not Evaluated]1.41       0.92    1.54     0.125
```

```
her2_status_by_ihcEquivocal      2.01       0.90     2.24    0.025*
her2_status_by_ihcIndeterminate  1.99       1.40     1.42    0.156
her2_status_by_ihcNegative       1.72       0.87     1.98    0.047*
her2_status_by_ihcPositive       1.91       0.91     2.11    0.035*

---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 723.68  on 753  degrees of freedom
Residual deviance: 595.32  on 729  degrees of freedom
(114 observations deleted due to missingness)
AIC: 645.32

Number of Fisher Scoring iterations: 15
```

C.3 Backward Variable Selection

```
Call:
glm(formula = binary_cancer_staging ~ lymph_nodes_examined_count,
    family = "binomial", data = trainingData)

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                 0.20019    0.14993   1.335    0.182
lymph_nodes_examined_count  0.16941    0.02075   8.165 3.21e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 723.68  on 753  degrees of freedom
Residual deviance: 612.20  on 752  degrees of freedom
  (114 observations deleted due to missingness)
AIC: 616.2

Number of Fisher Scoring iterations: 6
```

C.4 Missing Values

Table C.1: Missing Values

Predictors	**Percentage**
`birth_days_to`	1
gender	0
`menopause_status`	8
race	9
ethnicity	16
`history_other_malignancy`	0
`history_neoadjuvant_treatment`	0
`tumor_status`	11
`vital_status`	0
`last_contact_days_to`	9
`death_days_to`	91
`radiation_treatment_adjuvant`	85
`pharmaceutical_tx_adjuvant`	85
`histologic_diagnosis_other`	93
`initial_pathologic_dx_year`	0
`age_at_diagnosis`	0
`method_initial_path_dx`	9
`method_initial_path_dx_other`	94
`surgical_procedure_first`	5

Table C.2: Missing Values

Predictors	Percentage
first_surgical_procedure_other	75
margin_status	6
surgery_for_positive_margins	95
surgery_for_positive_margins_other	99
margin_status_reexcision	95
axillary_staging_method	20
axillary_staging_method_other	98
micromet_detection_by_ihc	33
lymph_nodes_examined	34
lymph_nodes_examined_count	11
lymph_nodes_examined_he_count	15
lymph_nodes_examined_ihc_count	69
ajcc_staging_edition	13
ajcc_tumor_pathologic_pt	0
ajcc_nodes_pathologic_pn	0
ajcc_metastasis_pathologic_pm	0
ajcc_pathologic_tumor_stage	1
metastasis_site	98
metastasis_site_other	99
er_status_by_ihc	5
er_status_ihc_Percent_Positive	57
er_positivity_scale_used	87
er_ihc_score	80

Table C.3: Missing Values

Predictors	Percentage
er_positivity_scale_other	78
er_positivity_method	80
pr_status_by_ihc	5
pr_status_ihc_percent_positive	61
pr_positivity_scale_used	87
pr_positivity_ihc_intensity_score	81
pr_positivity_scale_other	80
pr_positivity_define_method	81
her2_status_by_ihc	34
her2_ihc_percent_positive	81
her2_ihc_score	43
her2_positivity_scale_other	99
her2_positivity_method_text	92
her2_fish_status	63
her2_copy_number	90
cent17_copy_number	90
her2_and_cent17_cells_count	91
her2_cent17_ratio	79
her2_and_cent17_scale_other	100
her2_fish_method	95
new_tumor_event_dx_indicator	82
nte_er_status	99
nte_er_status_ihc__positive	99

Table C.4: Missing Values

Predictors	**Percentage**
nte_er_ihc_intensity_score	100
nte_er_positivity_other_scale	100
nte_er_positivity_define_method	100
nte_pr_status_by_ihc	99
nte_pr_status_ihc__positive	100
nte_pr_ihc_intensity_score	100
nte_pr_positivity_other_scale	100
nte_pr_positivity_define_method	100
nte_her2_status	99
nte_her2_status_ihc__positive	100
nte_her2_positivity_ihc_score	100
nte_her2_positivity_other_scale	100
nte_her2_positivity_method	100
nte_her2_fish_status	100
nte_her2_signal_number	100
nte_cent_17_signal_number	100
her2_cent17_counted_cells_count	100
nte_cent_17_her2_ratio	100
nte_cent17_her2_other_scale	100
nte_her2_fish_define_method	100
anatomic_neoplasm_subdivision	0
clinical_M	100

Table C.5: Missing Values

Predictors	**Percentage**
clinical_N	100
clinical_T	100
clinical_stage	100
days_to_initial_pathologic_diagnosis	0
days_to_patient_progression_free	100
days_to_tumor_progression	100
disease_code	100
extranodal_involvement	100
histological_type	0
icd_10	0
icd_o_3_histology	0
icd_o_3_site	0
informed_consent_verified	0
metastatic_tumor_indicator	64
patient_id	0
project_code	100
site_of_primary_tumor_other	100
stage_other	100
tissue_source_site	0
tumor_tissue_site	0

Bibliography

[1] TCGAbiolinks package on bioconductor. Accessed on December 18, 2023.

[2] T. Bismeijer, S. Canisius, and L.F.A. Wessels. Molecular characterization of breast and lung tumors by integration of multiple data types with functional sparse-factor analysis. *PLoS Computational Biology*, 14:1–28, 2018.

[3] Rocky Mountain Cancer Centers. What's the difference between invasive and non-invasive breast cancers?, 2023. Accessed on December 8, 2023.

[4] ChatGPT. Chatgpt. https://www.chatbot.com/.

[5] DataCamp. DataCamp Tutorial, Year. Accessed on Date.

[6] Jamie DePolo. Breast cancer stages, 2023. Accessed on December 8, 2023.

[7] GeeksforGeeks. GeeksforGeeks Random Forest Classifier Tutorial, Year. Accessed on Date.

[8] O. Gevaert, F.D. Smet, D. Timmerman, Y. Moreau, and B.D. Moor. Predicting the prognosis of breast cancer by integrating clinical and microarray data with bayesian networks. *Bioinformatics*, 22(14):e184–90, 2006.

[9] L. Giacomelli, R. Sacco, S. Papa, and B.I. Carr. Understanding the drawbacks of the current tumor staging systems: How to improve? *Cancers*, 15(4):1242, 2023.

[10] E. Gobbini et al. Time trends of overall survival among metastatic breast cancer patients in the real-life esme cohort. *European Journal of Cancer*, 96, June 2018.

[11] Z. Huang, X. Zhan, S. Xiang, et al. Salmon: survival analysis learning with multi-omics neural networks on breast cancer. *Frontiers in Genetics*, 10, 2019.

[12] Kanncaa1. Kanncaa1's Deep Learning Tutorial, Year. Accessed on Date.

[13] A.A. Kim, S.R. Zaim, and V. Subbian. Assessing reproducibility and veracity across machine learning techniques in biomedicine: A case study using tcga data. *International Journal of Medical Informatics*, 141, September 2020.

[14] Tiago Maié and Martin Manolov. Analysis of cancer genome atlas in r. Accessed on December 19, 2023.

[15] Machine Learning Mastery. Machine Learning Mastery Tutorial, Year. Accessed on Date.

[16] Siddhartha Mukherjee. *The Emperor of All Maladies: A Biography of Cancer*. Scribner, 2010.

[17] Prashant111. Kaggle Tutorial by prashant111, Year. Accessed on Date.

[18] Real Python. Real Python Tutorial, Year. Accessed on Date.

[19] M.C. Rendleman, J.M. Buatti, T.A. Braun, et al. Machine learning with the tcga-hnsc dataset: Improving usability by addressing inconsistency, sparsity, and high-dimensionality. *BMC Bioinformatics*, 20:1–9, 2019.

[20] Scikit-Learn. Scikit-Learn RandomForestClassifier Documentation, Year. Accessed on Date.

[21] M. Sherafatian. Tree-based machine learning algorithms identified a minimal set of mirna biomarkers for breast cancer diagnosis and molecular subtyping. *Gene*, 677:111–118, 2018.

[22] D. Sun, A. Li, B. Tang, and M. Wang. Integrating genomic data and pathological images to effectively predict breast cancer clinical outcome. *Computer Methods and Programs in Biomedicine*, 161:45–53, 2018.

[23] M. Zhao, Y. Tang, H. Kim, and K. Hasegawa. Machine learning with k-means dimensional reduction for predicting survival outcomes in patients with breast cancer. *Cancer Informatics*, 17:1176935118810215, 2018.