Personalizing Radiotherapy Protocols in Brain Cancer Treatment Through Patient Clinical and Genetic Data with Machine Learning.

Atalor Polycarp Ehiz

2024-11-27

# Import Data

my\_data <- read.csv("brain-cancer-dataset.csv")

# **1. Introduction**

This report provides a comprehensive analysis of a brain cancer dataset, with a particular focus on the clinical and genetic data of children diagnosed with brain cancer. The goal is to prepare the dataset for machine learning models that predict:

* **OS.Status (Overall Survival Status):** Whether a patient survived.
* **DFS.Status (Disease-Free Survival Status):** Whether a patient remained disease-free.

Here’s a breakdown of the steps taken:

# **2. Description of the Dataset**

### 2.1**Dataset Overview**

* **Number of Records:** 218 patient records.
* **Number of Features:** 61 features, including clinical and genetic data.
* **Missing Values:** Yes (handled during preprocessing)

### 2.2 **Variable Types**

* **Categorical Variables:** Features like Cancer.Type, Treatment, and Tumor.Type.
* **Numerical Variables:** Age-related data, mutation counts, and other clinical measurements.

# **3. Preprocessing Steps**

Before analyzing the data, it’s crucial to ensure the dataset is clean and suitable for analysis. This section explains how missing values, inconsistent column names, and redundant features were handled.

### **3.1 Handling Missing Values**

Missing values can disrupt the quality of a dataset. In this case:

* **Numerical Variables:** Missing values were replaced with the median to preserve the data’s distribution.
* **Categorical Variables:** Missing values were replaced with “Unknown” for consistency.

# Handle Missing Values  
my\_data <- my\_data %>%  
 mutate(across(where(is.numeric), ~ ifelse(is.na(.), median(., na.rm = TRUE), .))) %>%  
 mutate(across(where(is.character), ~ ifelse(is.na(.), "Unknown", .))) %>%  
 mutate(across(where(is.factor), ~ ifelse(is.na(.), as.character(levels(.)[1]), .)))  
  
# Verify no missing values remain  
if (sum(is.na(my\_data)) > 0) {  
 stop("Unresolved missing values in the dataset.")  
}

### **3.2 Feature Cleaning**

* Column names were trimmed and standardized for compatibility with R functions.
* All categorical variables were converted into factors to help with later analysis and modeling.

# Standardize Column Names  
colnames(my\_data) <- trimws(colnames(my\_data))  
colnames(my\_data) <- make.names(colnames(my\_data), unique = TRUE)  
  
# Convert Categorical Variables to Factors  
my\_data <- my\_data %>% mutate(across(where(is.character), as.factor))

### **3.3 Feature Filtering**

* **Variance Filtering:** Features that had very low variance (less than 1% variation across data) were removed, as they are unlikely to be useful in predictive models.
* **Correlation Filtering:** Features that were highly correlated (greater than 0.8 correlation) were removed to reduce redundancy. This step helps avoid issues like multicollinearity, which can confuse models.

# Filter Low-Variance Features  
variances <- apply(my\_data[, sapply(my\_data, is.numeric)], 2, var, na.rm = TRUE)  
data\_filtered <- my\_data[, variances > 0.01]  
  
# Remove Highly Correlated Features  
cor\_matrix <- cor(data\_filtered[, sapply(data\_filtered, is.numeric)], use = "complete.obs")  
high\_corr <- findCorrelation(cor\_matrix, cutoff = 0.8)  
data\_filtered <- data\_filtered[, -high\_corr]

### **3.4 Feature Selection**

For predictive modeling, there is need to select the most relevant features. Chi-Square tests were used to identify categorical variables that are significantly associated with the target variables OS.Status and DFS.Status.

The following categorical variables were selected for analysis, These variables represent distinct groups or categories and will be used to explore their associations with patient prognosis.

# Define categorical columns  
categorical\_columns <- c(  
 "Study.ID", "Patient.ID", "Sample.ID", "Age.Class", "BRAF\_RELA.Status",   
 "BRAF.Status", "BRAF.Status2", "Cancer.Predispositions", "Cancer.Type",   
 "Cancer.Type.Detailed", "Chemotherapy", "Chemotherapy.Agents",   
 "Chemotherapy.Type", "Clinical.Status.at.Collection.Event",   
 "CTNNB1.Status", "DFS.Status", "Ethnicity", "Extent.of.Tumor.Resection",   
 "External.Patient.ID", "Formulation", "H3F3A\_CTNNB1.Status",   
 "Initial.CNS.Tumor.Diagnosis.Related.to.OS", "Initial.Diagnosis.Type",   
 "LGG\_BRAF.Status", "Medical.Conditions", "Multiple.Cancer.Predispositions",   
 "Multiple.Medical.Conditions", "Multiple.Tumor.Locations", "Oncotree.Code",   
 "OS.Status", "Protocol.and.Treatment.Arm", "Race", "Radiation",   
 "Radiation.Site", "Radiation.Type", "Sample.Annotation", "Sample.Origin",   
 "Sex", "Surgery", "Treatment", "Treatment.Changed", "Treatment.Status",   
 "Tumor.Location.Condensed", "Tumor.Tissue.Site", "Tumor.Type", "Updated.Grade"  
)

Significant features were identified for both OS.Status and DFS.Status using a Chi-square test.

# Chi-Square Test for Feature Selection  
chi\_sq\_results\_OS <- lapply(categorical\_columns, function(col) {  
 table\_data <- table(my\_data[[col]], my\_data$OS.Status)  
 if (any(dim(table\_data) == 0)) {  
 return(data.frame(Feature = col, p\_value = NA))  
 }  
 test\_result <- chisq.test(table\_data)  
 data.frame(Feature = col, p\_value = test\_result$p.value)  
})  
  
chi\_sq\_summary\_OS <- do.call(rbind, chi\_sq\_results\_OS)  
significant\_features\_OS <- chi\_sq\_summary\_OS %>% filter(p\_value < 0.05)

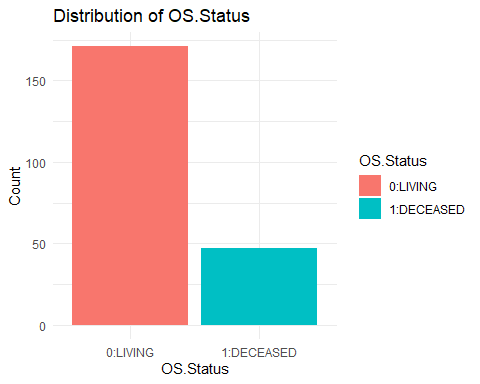
# 4. Key Insights from Exploratory Data Analysis

EDA helps to understand the data and its relationships, providing insights that can guide further analysis and modeling.

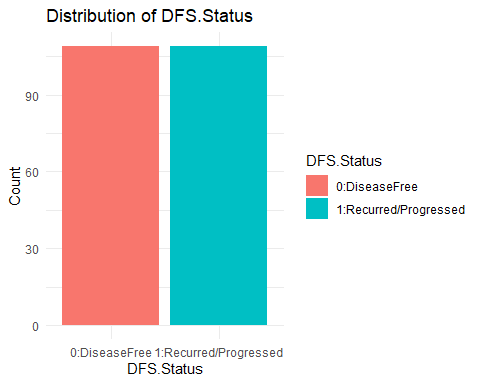
### **4.1 Target Variable Distribution**

The distribution of OS.Status and DFS.Status was checked using bar plots. Both variables showed balanced distributions, which is crucial for developing a reliable machine learning model.

# OS.Status Distribution  
ggplot(my\_data, aes(x = OS.Status, fill = OS.Status)) +  
 geom\_bar() +  
 labs(title = "Distribution of OS.Status", x = "OS.Status", y = "Count") +  
 theme\_minimal()



# DFS.Status Distribution  
ggplot(my\_data, aes(x = DFS.Status, fill = DFS.Status)) +  
 geom\_bar() +  
 labs(title = "Distribution of DFS.Status", x = "DFS.Status", y = "Count") +  
 theme\_minimal()



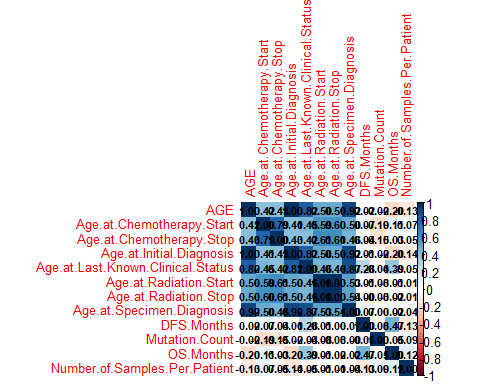
* **Insight:** The OS.Status distribution indicates a balanced dataset for survival analysis.
* **Insight:** The DFS.Status distribution shows similar proportions for patients who remained disease-free and those who relapsed.

### 4.2 Correlation Analysis

A correlation matrix was created to visualize relationships between numerical variables. This helped in understanding how variables relate to each other and where correlations are strong enough to potentially affect model performance.

A heatmap was used to visualize correlations among numerical variables. Highly correlated features were excluded.

# Correlation Matrix Visualization  
corrplot(cor\_matrix, method = "color", tl.cex = 0.8, number.cex = 0.7, addCoef.col = "black")



* **Insight:** Some numerical features showed high correlations and were excluded to prevent multicollinearity.

### 4.3 Selected Features

To summarize the features selected for OS.Status and DFS.Status, we present: - A Bar Plot showing the total number of features selected for each target variable. - A Table listing the features categorized as unique to each target variable or common to both.

#### Features for OS.Status & DFS.Status

**Chi-Square Test for Feature Significance**

The chi-square tests were performed to evaluate the relationship between categorical features and the target variables, OS.Status and DFS.Status. For each feature in the categorical\_columns, contingency tables were created between the feature and the target variable. The resulting p-values were computed to assess the statistical significance of each feature in relation to the target variables.

# Chi-square test for OS.Status and DFS.Status  
chi\_sq\_results\_OS <- lapply(categorical\_columns, function(col) {  
 table\_data <- table(my\_data[[col]], my\_data$OS.Status)  
 if (any(dim(table\_data) == 0)) {  
 return(data.frame(Feature = col, p\_value = NA))  
 }  
 test\_result <- chisq.test(table\_data)  
 data.frame(Feature = col, p\_value = test\_result$p.value)  
})  
  
chi\_sq\_summary\_OS <- do.call(rbind, chi\_sq\_results\_OS)  
  
chi\_sq\_results\_DFS <- lapply(categorical\_columns, function(col) {  
 table\_data <- table(my\_data[[col]], my\_data$DFS.Status)  
 if (any(dim(table\_data) == 0)) {  
 return(data.frame(Feature = col, p\_value = NA))  
 }  
 test\_result <- chisq.test(table\_data)  
 data.frame(Feature = col, p\_value = test\_result$p.value)  
})  
  
chi\_sq\_summary\_DFS <- do.call(rbind, chi\_sq\_results\_DFS)

**Filtering Significant Features**

Features with p-values less than 0.05 were considered statistically significant and were selected for further analysis. These significant features were extracted from the chi-square test results for both OS.Status and DFS.Status. The dataset was then subset to include only these significant features, resulting in two datasets: data\_OS for OS.Status and data\_DFS for DFS.Status.

# Filter significant features for OS.Status and DFS.Status  
significant\_features\_OS <- chi\_sq\_summary\_OS %>% filter(p\_value < 0.05)  
significant\_features\_DFS <- chi\_sq\_summary\_DFS %>% filter(p\_value < 0.05)  
  
# Subset data for significant features  
data\_OS <- my\_data %>% select(all\_of(significant\_features\_OS$Feature), OS.Status)  
data\_DFS <- my\_data %>% select(all\_of(significant\_features\_DFS$Feature), DFS.Status)

**Feature Count Summary**

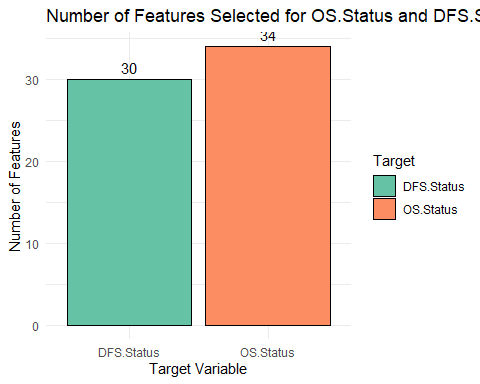
The selected features for OS.Status and DFS.Status were extracted, excluding the target variable itself. A summary was created to report the number of significant features for each target variable, as well as the number of common features between OS.Status and DFS.Status.

# Extract selected features  
selected\_features\_OS <- colnames(data\_OS)[colnames(data\_OS) != "OS.Status"]  
selected\_features\_DFS <- colnames(data\_DFS)[colnames(data\_DFS) != "DFS.Status"]  
  
# Create a summary of feature counts  
feature\_counts <- data.frame(  
 Target = c("OS.Status", "DFS.Status"),  
 Features = c(length(selected\_features\_OS), length(selected\_features\_DFS)),  
 Common = length(intersect(selected\_features\_OS, selected\_features\_DFS))  
)

**Visualization of Selected Features**

A bar plot was generated to visualize the number of significant features selected for OS.Status and DFS.Status. The plot was created using the geom\_bar() function to display the counts of features, with labels added through geom\_text(). The plot includes a clear title, axis labels, and a minimal theme, utilizing a color palette for better visual clarity.

# Create a bar plot  
ggplot(feature\_counts, aes(x = Target, y = Features, fill = Target)) +  
 geom\_bar(stat = "identity", position = "dodge", color = "black") +  
 geom\_text(aes(label = Features), vjust = -0.5, size = 4) +  
 labs(  
 title = "Number of Features Selected for OS.Status and DFS.Status",  
 x = "Target Variable",  
 y = "Number of Features"  
 ) +  
 theme\_minimal() +  
 scale\_fill\_brewer(palette = "Set2")



* **Union of Features:**

To explore the relationship between the selected features for OS.Status and DFS.Status, the union of the features was computed. A new table was created to categorize each feature based on its presence in either or both target variables.

The features were categorized as follows:

“Both”: Features present in both OS.Status and DFS.Status. “OS.Status”: Features that are exclusive to OS.Status. “DFS.Status”: Features that are exclusive to DFS.Status. The categorized features were then displayed in a table for better visualization and clarity.

# Categorize features  
feature\_categories <- data.frame(  
 Feature = union(selected\_features\_OS, selected\_features\_DFS),  
 Category = ifelse(  
 union(selected\_features\_OS, selected\_features\_DFS) %in% intersect(selected\_features\_OS, selected\_features\_DFS), "Both",  
 ifelse(union(selected\_features\_OS, selected\_features\_DFS) %in% selected\_features\_OS, "OS.Status", "DFS.Status")  
 )  
)  
  
# Display the table  
knitr::kable(  
 feature\_categories,  
 caption = "Feature Categories by Target Variable",  
 col.names = c("Feature Name", "Category")  
)

Feature Categories by Target Variable

| Feature Name | Category |
| --- | --- |
| Study.ID | OS.Status |
| Age.Class | OS.Status |
| BRAF\_RELA.Status | Both |
| BRAF.Status | Both |
| BRAF.Status2 | Both |
| Cancer.Predispositions | Both |
| Cancer.Type | Both |
| Cancer.Type.Detailed | Both |
| Chemotherapy | Both |
| Chemotherapy.Agents | OS.Status |
| Chemotherapy.Type | Both |
| Clinical.Status.at.Collection.Event | Both |
| DFS.Status | OS.Status |
| Extent.of.Tumor.Resection | Both |
| Formulation | OS.Status |
| H3F3A\_CTNNB1.Status | Both |
| Initial.CNS.Tumor.Diagnosis.Related.to.OS | Both |
| Initial.Diagnosis.Type | Both |
| LGG\_BRAF.Status | Both |
| Medical.Conditions | OS.Status |
| Multiple.Cancer.Predispositions | Both |
| Multiple.Medical.Conditions | OS.Status |
| Multiple.Tumor.Locations | Both |
| Oncotree.Code | Both |
| Radiation | Both |
| Sample.Annotation | Both |
| Sample.Origin | Both |
| Surgery | Both |
| Treatment | Both |
| Treatment.Status | Both |
| Tumor.Location.Condensed | Both |
| Tumor.Tissue.Site | OS.Status |
| Tumor.Type | Both |
| Updated.Grade | Both |
| OS.Status | DFS.Status |
| Race | DFS.Status |
| Radiation.Type | DFS.Status |
| Treatment.Changed | DFS.Status |

# Summary

* Preprocessing: Missing values were handled, column names standardized, and irrelevant features removed.
* Feature Selection: Chi-Square tests helped identify the most significant features for predicting OS.Status and DFS.Status.
* Exploratory Data Analysis: Revealed balanced target variable distributions and correlations among features. ## End