R Notebook

Data Visualization - Brain Cancer Model analysis Final Report and Code

By Hashem Fawzy

Hi. Welcome to my analysis on the following Brain cancer dataset.

We will create a model to analyze the following subtypes, cleanse the model of various errors such as missing data, duplicates, etc. Perform data pre-processing, and finally train and test a model of the following dataset.

Before starting, you will find that the following analysis can be easily replicated on most datasets related to tumors and other various diseases involving genes and disease subtypes.

The following analysis is done on R, with the a data on **cBioPortal** on Lower grade Glioma, AKA Brain Cancer.

Source of the file:
https://www.cbioportal.org/study/summary?id=lgg_tcga_pan_can_atlas_2018.

For our analysis, we will perform a foundation level analysis on the lower grade Glioma patient data. Our data will include 3 seperate .txt files that can be downloaded from the source. We will utilize logistic regression on our model, in order to understand and predict the models accuracy and complexity correctly.

1. **data_clinical_patient.txt.** This txt file contains data on the patients studied and logged in the brain cancer dataset. Our main focus will be on the Patient ID and Subtype

2. **data_clinical_sample.txt.** This next file contains information of the various hugo symbols associated. We will later merge this with our dataset.

3. **data_mrna_seq_v2_rsem.txt**. This last file gives the gene expression data of the various hugo symbols. This will be highly important, and needed in order to analyze whether predicting a brain cancer subtype is possible based on the information given and processed later from here.

Our **methodologies** for getting the dataset ready will be as follows:

First, unpackaging the dataset to explain the various features and sections we will be focusing on.

Then, Cleaning the data of various missing and duplicate values, as well as merging the subsections and gene expressions together, to then finally test and train our multinomial logistic model.

```
# Specify global settings
knitr::opts_chunk$set(echo = TRUE, warning = FALSE, message = FALSE)

# Start fresh by clearing R environment
rm(list = ls())




# Load required libraries here
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
library(readr)
library(ggplot2)
library(janitor)
```

```
## Warning: package 'janitor' was built under R version 4.3.2
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(broom)
library(caret) # For working with training/test data
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
```

```
## The following object is masked from 'package:purrr':
##
##     lift

library(nnet)
library(cowplot)

## Warning: package 'cowplot' was built under R version 4.3.2

##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp

library(scales) # For generating color scheme

##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor

library(RColorBrewer) # For generating color scheme
#Colors Here
hex <- hue_pal()(3)
gg_red <- hex[1]
gg_green <- hex[2]
gg_blue <- hex[3]
gg_orange <- brewer.pal(n = 11, name = "PuOr")[4]
gg_purple <- "#C77CFF"

dcp <- read_tsv("D:/School/University/Semester 8/Data Visualization and
Mining/lgg_tcga_pan_can_atlas_2018/data_clinical_patient.txt",skip = 4)

## Rows: 514 Columns: 38
## ── Column specification
────────────────────────────────────────────────────────

## Delimiter: "\t"
## chr (23): PATIENT_ID, SUBTYPE, CANCER_TYPE_ACRONYM, OTHER_PATIENT_ID, SEX,
E...
## dbl  (8): AGE, DAYS_LAST_FOLLOWUP, DAYS_TO_BIRTH,
DAYS_TO_INITIAL_PATHOLOGIC...
## lgl  (7): AJCC_PATHOLOGIC_TUMOR_STAGE, AJCC_STAGING_EDITION, PATH_M_STAGE,
P...
```

```
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this
message.

#Note: Skip 4 is included as the first 3 rows explain the values of the
dataset and messes it up.


head(dcp,10)

## # A tibble: 10 × 38
##    PATIENT_ID   SUBTYPE         CANCER_TYPE_ACRONYM OTHER_PATIENT_ID   AGE
SEX
##    <chr>        <chr>           <chr>               <chr>            <dbl>
<chr>
##  1 TCGA-CS-4938 LGG_IDHmut-non… LGG                 334f715e-08dc-4…    31
Fema…
##  2 TCGA-CS-4941 LGG_IDHwt       LGG                 fc222f23-b3b2-4…    67
Male
##  3 TCGA-CS-4942 LGG_IDHmut-non… LGG                 230f5fa7-aa36-4…    44
Fema…
##  4 TCGA-CS-4943 LGG_IDHmut-non… LGG                 952dfd5d-e65a-4…    37
Male
##  5 TCGA-CS-4944 LGG_IDHmut-non… LGG                 64cd17eb-c778-4…    50
Male
##  6 TCGA-CS-5390 LGG_IDHmut-cod… LGG                 c6cf2b8e-40ed-4…    47
Fema…
##  7 TCGA-CS-5393 LGG_IDHmut-non… LGG                 e8d3d888-e5fc-4…    39
Male
##  8 TCGA-CS-5394 LGG_IDHmut-non… LGG                 97bf8065-3e7d-4…    40
Male
##  9 TCGA-CS-5395 LGG_IDHwt       LGG                 f86fa219-34e9-4…    43
Male
## 10 TCGA-CS-5396 LGG_IDHmut-cod… LGG                 b6c2c9bd-625b-4…    53
Fema…
## # ℹ 32 more variables: AJCC_PATHOLOGIC_TUMOR_STAGE <lgl>,
## #   AJCC_STAGING_EDITION <lgl>, DAYS_LAST_FOLLOWUP <dbl>, DAYS_TO_BIRTH
<dbl>,
## #   DAYS_TO_INITIAL_PATHOLOGIC_DIAGNOSIS <dbl>, ETHNICITY <chr>,
## #   FORM_COMPLETION_DATE <chr>, HISTORY_NEOADJUVANT_TRTYN <chr>, ICD_10
<chr>,
## #   ICD_O_3_HISTOLOGY <chr>, ICD_O_3_SITE <chr>,
## #   INFORMED_CONSENT_VERIFIED <chr>,
## #   NEW_TUMOR_EVENT_AFTER_INITIAL_TREATMENT <chr>, PATH_M_STAGE <lgl>, …
```

PT 1: Information on the dataset

Name of the Dataset:

Brain Lower Grade Glioma (TCA, Pancancer Atlas)

> File name: data_clinical_patient.txt - A dataset listing the various patients and their subtypes, as well as more information about their disease and biological information

List of observations: 514 List of observations with Mrna data: 514

List of columns: 38

The following samples that contain a 12 character description(EG: ABCD-EF-GHIJ-00) show the expression profiles for this dataset. Each of these hold integer values, which explain the expression levels for the various samples on each patient (Shown in patient_sample.txt).

The following list of data on the various subtypes for the various cancer patients listed is as follows:

```
data_mrna_seq_v2_rsem <- read_tsv("D:/School/University/Semester 8/Data
Visualization and
Mining/lgg_tcga_pan_can_atlas_2018/data_mrna_seq_v2_rsem.txt")
head(data_mrna_seq_v2_rsem,10)
```

```
## # A tibble: 10 × 516
##    Hugo_Symbol Entrez_Gene_Id `TCGA-CS-4938-01` `TCGA-CS-4941-01`
##    <chr>                <dbl>             <dbl>             <dbl>
##  1 <NA>             100130426             0                  0
##  2 <NA>             100133144             8.71              36.4
##  3 UBE2Q2P2         100134869            22.8               21.2
##  4 HMGB1P1              10357           269.               157.
##  5 <NA>                 10431           846.               390.
##  6 <NA>                136542             0                  0
##  7 <NA>                155060           183.               325.
##  8 RNU12-2P             26823             0.420              1.73
##  9 SSX9P               280660             0                  0
## 10 <NA>                317712             0                  0
## # ℹ 512 more variables: `TCGA-CS-4942-01` <dbl>, `TCGA-CS-4943-01` <dbl>,
## #   `TCGA-CS-4944-01` <dbl>, `TCGA-CS-5390-01` <dbl>, `TCGA-CS-5393-01`
<dbl>,
## #   `TCGA-CS-5394-01` <dbl>, `TCGA-CS-5395-01` <dbl>, `TCGA-CS-5396-01`
<dbl>,
## #   `TCGA-CS-5397-01` <dbl>, `TCGA-CS-6186-01` <dbl>, `TCGA-CS-6188-01`
<dbl>,
## #   `TCGA-CS-6290-01` <dbl>, `TCGA-CS-6665-01` <dbl>, `TCGA-CS-6666-01`
<dbl>,
## #   `TCGA-CS-6667-01` <dbl>, `TCGA-CS-6668-01` <dbl>, `TCGA-CS-6669-01`
<dbl>,
## #   `TCGA-CS-6670-01` <dbl>, `TCGA-DB-5270-01` <dbl>, …
```

An Examination of the type of Brain cancer tumors (and percentages) is as follows in a suitable pie chart:

A table is also below, if the viewer prefers it.

```
data_clinical_sample <- read_tsv("D:/School/University/Semester 8/Data
Visualization and
Mining/lgg_tcga_pan_can_atlas_2018/data_clinical_sample.txt",skip=4)
head(data_clinical_sample)
```

```
## # A tibble: 6 × 18
##   PATIENT_ID SAMPLE_ID ONCOTREE_CODE CANCER_TYPE CANCER_TYPE_DETAILED
TUMOR_TYPE
##   <chr>      <chr>     <chr>         <chr>       <chr>
<chr>
## 1 TCGA-CS-4… TCGA-CS-… DIFG          Glioma      Astrocytoma
Astrocyto…
## 2 TCGA-CS-4… TCGA-CS-… DIFG          Glioma      Astrocytoma
Astrocyto…
## 3 TCGA-CS-4… TCGA-CS-… DIFG          Glioma      Astrocytoma
Astrocyto…
## 4 TCGA-CS-4… TCGA-CS-… DIFG          Glioma      Astrocytoma
Astrocyto…
## 5 TCGA-CS-4… TCGA-CS-… DIFG          Glioma      Astrocytoma
Astrocyto…
## 6 TCGA-CS-5… TCGA-CS-… ODG           Glioma      Oligodendroglioma
Oligodend…
## # ℹ 12 more variables: GRADE <chr>,
## #   TISSUE_PROSPECTIVE_COLLECTION_INDICATOR <chr>,
## #   TISSUE_RETROSPECTIVE_COLLECTION_INDICATOR <chr>,
## #   TISSUE_SOURCE_SITE_CODE <chr>, TUMOR_TISSUE_SITE <chr>,
## #   ANEUPLOIDY_SCORE <dbl>, SAMPLE_TYPE <chr>, MSI_SCORE_MANTIS <dbl>,
## #   MSI_SENSOR_SCORE <dbl>, SOMATIC_STATUS <chr>, TMB_NONSYNONYMOUS <dbl>,
## #   TISSUE_SOURCE_SITE <chr>
```

```
cat<- table(data_clinical_sample$CANCER_TYPE_DETAILED)
pie(cat,
    col = hcl.colors(length(cat), "BluYl"))
```

Brain Cancer Subtypes and Frequencies

Brain Cancer Type	Frequency (# of Patients)	Percentage(%)
Astrocytoma	194	%
Oligodendroglioma	189	%
Oligoastrocytoma	130	%
Low-Grade Glioma (NOS)	1	0.2%
Total	514	100%

Brain Cancer Type	Frequency (# of Patients)	Treatment Option
Astrocytoma	194	Radiotherapy
Oligodendroglioma	189	Surgery
Oligoastrocytoma	130	Surgery
Low-Grade Glioma (NOS)	1	radiotherapy

As you can see, Lower Grade Glioma happens in very rare circumstances, and brain cancer solutions tend to involve Radiotherapy and Surgery as a treatement. We will later remove the Low-Grade Glioma in the late part of our Assignment. But in general, the 3 other types tend to have a close triple-split on patients associated with each type.

sources:

LGG: https://www.mountsinai.org/care/neurosurgery/services/brain-tumors/what-are/low-grade-gliomas#:~:text=Treatment%20Available,are%20looking%20for%20something%20else.

Astrocytoma: https://www.thebraintumourcharity.org/brain-tumour-diagnosis-treatment/types-of-brain-tumour-adult/astrocytoma/

Oligoastrocytoma: https://www.moffitt.org/cancers/brain-cancer/diagnosis/types/oligoastrocytoma/#:~:text=Treatment%20may%20include%20surgery%2C%20chemotherapy,health%2C%20age%20and%20personal%20preferences.

Oligodendroglioma:https://www.mayoclinic.org/diseases-conditions/oligodendroglioma/cdc-20350152#:~:text=Oligodendroglioma%20treatments%20include%3A,without%20harming%20healthy%20brain%20tissue.

Pt: 2 Data Cleaning and Merging the Samples & MRNA data

We will clean the rsem sequence, as the genes will be useful to analyze in a dataset. We will also clean the sample data in the same way too.

Here is another look at the dataset before cleaning it

```
head(data_mrna_seq_v2_rsem,10)
```

```
## # A tibble: 10 × 516
##    Hugo_Symbol Entrez_Gene_Id `TCGA-CS-4938-01` `TCGA-CS-4941-01`
##    <chr>                <dbl>             <dbl>              <dbl>
##  1 <NA>             100130426             0                   0
##  2 <NA>             100133144             8.71               36.4
##  3 UBE2Q2P2         100134869            22.8                21.2
##  4 HMGB1P1              10357           269.                157.
##  5 <NA>                 10431           846.                390.
```

```
##  6 <NA>                136542            0                  0
##  7 <NA>                155060          183.               325.
##  8 RNU12-2P             26823            0.420              1.73
##  9 SSX9P               280660            0                  0
## 10 <NA>                317712            0                  0
## # ℹ 512 more variables: `TCGA-CS-4942-01` <dbl>, `TCGA-CS-4943-01` <dbl>,
## #   `TCGA-CS-4944-01` <dbl>, `TCGA-CS-5390-01` <dbl>, `TCGA-CS-5393-01`
<dbl>,
## #   `TCGA-CS-5394-01` <dbl>, `TCGA-CS-5395-01` <dbl>, `TCGA-CS-5396-01`
<dbl>,
## #   `TCGA-CS-5397-01` <dbl>, `TCGA-CS-6186-01` <dbl>, `TCGA-CS-6188-01`
<dbl>,
## #   `TCGA-CS-6290-01` <dbl>, `TCGA-CS-6665-01` <dbl>, `TCGA-CS-6666-01`
<dbl>,
## #   `TCGA-CS-6667-01` <dbl>, `TCGA-CS-6668-01` <dbl>, `TCGA-CS-6669-01`
<dbl>,
## #   `TCGA-CS-6670-01` <dbl>, `TCGA-DB-5270-01` <dbl>, …
```

NA values could already be seen at the beginning of this dataset. this means we should be cleaning it.

Data Type Checking

```
var_types <- sapply(data_mrna_seq_v2_rsem, class)

cat(paste0("Hugo gene symbols type: ", var_types["Hugo_Symbol"], "\n"))

## Hugo gene symbols type: character

p <- (length(var_types) - 1) #Cancer gene expression numbers
if (all(var_types[2 : length(var_types)] == "numeric") == TRUE) {
  cat(paste0("All ", p, " expression profiles are of type numeric\n. No data
types need to be changed."))
} else {
  cat("This dataset may have contradicting datatypes")
}

## All 515 expression profiles are of type numeric
## . No data types need to be changed.
```

Yielded positive results, showing our dataset is clean. Onto the next

Cleaning Missing Values

```
missing_hg <- sum(is.na(data_mrna_seq_v2_rsem$Hugo_Symbol))
cat(paste0("at least missing_hg missing Hugo symbols","\n"))

## at least missing_hg missing Hugo symbols

#dropping NA values
data_mrna_seq_v2_rsem <- data_mrna_seq_v2_rsem %>%
drop_na(any_of("Hugo_Symbol"))
```

```
number_of_missing_mrna_exp_values <- sum(is.na(data_mrna_seq_v2_rsem %>%
select(3 : last_col())))
cat(paste0("Missing mRNA expression values: ",
number_of_missing_mrna_exp_values, "\n"))

## Missing mRNA expression values: 0
```

As you see, 13 Hugo Symbols were empty. those were deleted

Cleaning Duplicate Sets

```
cat("Mysterious duplicate genes\n")

## Mysterious duplicate genes

hugo_gene_symbols <- data_mrna_seq_v2_rsem$Hugo_Symbol
duplicate_hugo_gene_symbols <-
hugo_gene_symbols[duplicated(hugo_gene_symbols)]
duplicate_hugo_gene_symbols

## [1] "FGF13"      "ELMOD1"     "NKAIN3"     "PALM2AKAP2" "QSOX1"
## [6] "SNAP47"     "TMEM8B"

data_mrna_seq_v2_rsem %>% get_dupes(Hugo_Symbol)

## # A tibble: 14 × 517
##    Hugo_Symbol dupe_count Entrez_Gene_Id `TCGA-CS-4938-01` `TCGA-CS-4941-
01`
##    <chr>            <int>          <dbl>             <dbl>
<dbl>
##  1 ELMOD1               2          55531            298.             389.
##  2 ELMOD1               2          55531              2.94
0.345
##  3 FGF13                2           2258            138.             133.
##  4 FGF13                2           2258              8.81            30.4
##  5 NKAIN3               2         286183          13190.            1318.
##  6 NKAIN3               2         286183             27.3             10.7
##  7 PALM2AKAP2           2         445815            122.              608.
##  8 PALM2AKAP2           2         445815             68.4             107.
##  9 QSOX1                2         200058             76.8             85.9
## 10 QSOX1                2           5768            789.             1544.
## 11 SNAP47               2         116841             51.1             38.2
## 12 SNAP47               2         116841            915.             1049.
## 13 TMEM8B               2          51754            465.              409.
## 14 TMEM8B               2          51754            958.             1200.
## # ℹ 512 more variables: `TCGA-CS-4942-01` <dbl>, `TCGA-CS-4943-01` <dbl>,
## #   `TCGA-CS-4944-01` <dbl>, `TCGA-CS-5390-01` <dbl>, `TCGA-CS-5393-01`
<dbl>,
## #   `TCGA-CS-5394-01` <dbl>, `TCGA-CS-5395-01` <dbl>, `TCGA-CS-5396-01`
<dbl>,
## #   `TCGA-CS-5397-01` <dbl>, `TCGA-CS-6186-01` <dbl>, `TCGA-CS-6188-01`
<dbl>,
```

```
## #   `TCGA-CS-6290-01` <dbl>, `TCGA-CS-6665-01` <dbl>, `TCGA-CS-6666-01`
<dbl>,
## #   `TCGA-CS-6667-01` <dbl>, `TCGA-CS-6668-01` <dbl>, `TCGA-CS-6669-01`
<dbl>,
## #   `TCGA-CS-6670-01` <dbl>, `TCGA-DB-5270-01` <dbl>, …
```

```
# Drop duplicate genes from the analysis
data_mrna_seq_v2_rsem <- data_mrna_seq_v2_rsem %>% distinct(Hugo_Symbol,
.keep_all = TRUE)
```

Domain Cleaning

```
# Checking for non negative numbers. This isn't allowed in the dataset.
if (any(data_mrna_seq_v2_rsem %>% select(3 : last_col()) >= 0)) {
  cat(paste0("mRNA values are nonnegative\n"))
} else {
  cat(paste0("Negative numbers found!"))
}
```

```
## mRNA values are nonnegative
```

Result - Gene expression cleaning

```
#Final Check
head(data_mrna_seq_v2_rsem,10)
```

```
## # A tibble: 10 × 516
##    Hugo_Symbol Entrez_Gene_Id `TCGA-CS-4938-01` `TCGA-CS-4941-01`
##    <chr>                <dbl>             <dbl>             <dbl>
##  1 UBE2Q2P2         100134869            22.8              21.2
##  2 HMGB1P1              10357           269.              157.
##  3 RNU12-2P             26823             0.420             1.73
##  4 SSX9P               280660             0                 0
##  5 EZHIP               340602             2.10              3.45
##  6 EFCAB8              388795             0.420             0.345
##  7 SRP14P1             390284            12.6              15.2
##  8 TRIM75P             391714             0                 0.345
##  9 SPATA31B1P          404770             0                 0
## 10 REXO1L6P            441362             0                 0
## # ℹ 512 more variables: `TCGA-CS-4942-01` <dbl>, `TCGA-CS-4943-01` <dbl>,
## #   `TCGA-CS-4944-01` <dbl>, `TCGA-CS-5390-01` <dbl>, `TCGA-CS-5393-01`
<dbl>,
## #   `TCGA-CS-5394-01` <dbl>, `TCGA-CS-5395-01` <dbl>, `TCGA-CS-5396-01`
<dbl>,
## #   `TCGA-CS-5397-01` <dbl>, `TCGA-CS-6186-01` <dbl>, `TCGA-CS-6188-01`
<dbl>,
## #   `TCGA-CS-6290-01` <dbl>, `TCGA-CS-6665-01` <dbl>, `TCGA-CS-6666-01`
<dbl>,
## #   `TCGA-CS-6667-01` <dbl>, `TCGA-CS-6668-01` <dbl>, `TCGA-CS-6669-01`
<dbl>,
## #   `TCGA-CS-6670-01` <dbl>, `TCGA-DB-5270-01` <dbl>, …
```

```
#Dropping Entrez_Gene_Id. This is irrelavent to our analysis.

data_mrna_seq_v2_rsem <- data_mrna_seq_v2_rsem %>% select(-Entrez_Gene_Id)
```

For the most part. Only 20 variables have been removed due to this. This means we can be more assured of the dataset's Reliability.

Now is time to repeat this with out sample data

```
head(data_clinical_sample,10)
```

```
## # A tibble: 10 × 18
##    PATIENT_ID   SAMPLE_ID       ONCOTREE_CODE CANCER_TYPE
CANCER_TYPE_DETAILED
##    <chr>        <chr>           <chr>         <chr>       <chr>
##  1 TCGA-CS-4938 TCGA-CS-4938-01 DIFG          Glioma      Astrocytoma
##  2 TCGA-CS-4941 TCGA-CS-4941-01 DIFG          Glioma      Astrocytoma
##  3 TCGA-CS-4942 TCGA-CS-4942-01 DIFG          Glioma      Astrocytoma
##  4 TCGA-CS-4943 TCGA-CS-4943-01 DIFG          Glioma      Astrocytoma
##  5 TCGA-CS-4944 TCGA-CS-4944-01 DIFG          Glioma      Astrocytoma
##  6 TCGA-CS-5390 TCGA-CS-5390-01 ODG           Glioma
Oligodendroglioma
##  7 TCGA-CS-5393 TCGA-CS-5393-01 DIFG          Glioma      Astrocytoma
##  8 TCGA-CS-5394 TCGA-CS-5394-01 DIFG          Glioma      Astrocytoma
##  9 TCGA-CS-5395 TCGA-CS-5395-01 ODG           Glioma
Oligodendroglioma
## 10 TCGA-CS-5396 TCGA-CS-5396-01 ODG           Glioma
Oligodendroglioma
## # ℹ 13 more variables: TUMOR_TYPE <chr>, GRADE <chr>,
## #   TISSUE_PROSPECTIVE_COLLECTION_INDICATOR <chr>,
## #   TISSUE_RETROSPECTIVE_COLLECTION_INDICATOR <chr>,
## #   TISSUE_SOURCE_SITE_CODE <chr>, TUMOR_TISSUE_SITE <chr>,
## #   ANEUPLOIDY_SCORE <dbl>, SAMPLE_TYPE <chr>, MSI_SCORE_MANTIS <dbl>,
## #   MSI_SENSOR_SCORE <dbl>, SOMATIC_STATUS <chr>, TMB_NONSYNONYMOUS <dbl>,
## #   TISSUE_SOURCE_SITE <chr>
```

```
# Only PATIENT_ID and SAMPLE_ID are relevant to our present purpose
sample_bca <- data_clinical_sample %>% select(PATIENT_ID, SAMPLE_ID)

# Collect variable types
var_types <- sapply(data_clinical_sample, class)

# Patient and sample Id type check
cat(paste0("Patient Id data type: ", var_types["PATIENT_ID"], "\n"))

## Patient Id data type: character

cat(paste0("Sample Id data type: ", var_types["SAMPLE_ID"], "\n"))

## Sample Id data type: character
```

Do the Sample IDs match the Patient ID (EG: TCGA-CS-4938 = TCGA-CS-4938-01)

```
# Check that the Patient Ids and Sample Ids are consistent
n <- nrow(data_clinical_sample)
p_ids <- data_clinical_sample$PATIENT_ID
s_ids <- data_clinical_sample$SAMPLE_ID
bad_rows <- NULL

for (i in 1 : n) {
  patient_id <- p_ids[i]
  sample_id <- s_ids[i]
  sample_id_trunc <- str_sub(sample_id, start = 1, end = -4)

  # if any of the patient ids dont match with the sample ids, then group the
wrong together.
  if (patient_id != sample_id_trunc) {
    replace <- c(replace, i)
  }
}
number_to_replace <- length(replace)

if (number_to_replace == 0) { #If we have no mistakes
  cat("All patients have matching sample ids\n")
} else {
  cat(paste0("There exists", number_to_replace, " or more Patient Ids that do
not match the Sample Ids\n"))
}

## There exists1 or more Patient Ids that do not match the Sample Ids
```

Missing Value Check

```
number_of_missing_p_ids <- sum(is.na(data_clinical_sample$PATIENT_ID))
cat(paste0("Missing Patient Ids: ", number_of_missing_p_ids, "\n"))

## Missing Patient Ids: 0

number_of_missing_s_ids <- sum(is.na(data_clinical_sample$SAMPLE_ID))
cat(paste0("Missing Sample Ids: ", number_of_missing_s_ids, "\n"))

## Missing Sample Ids: 0
```

Both say 0, which means the patient_sample.txt has no missing data

Duplicate Check

```
any(duplicated(sample_bca))

## [1] FALSE
```

Result - No change in sample patients

Nothing was needed to be removed in the sample file, perfect.

One more dataset to clean is the Clinical Patient Data. We will do the same as in the Sample Patient Data

```
# Only PATIENT_ID and SUBTYPE are relevant to our present purpose
pclin_df_clean <- dcp %>% select(PATIENT_ID, SUBTYPE)

number_of_missing_p_ids <- sum(is.na(dcp$PATIENT_ID))
cat(paste0("We have this many missing patient Ids: ",
number_of_missing_p_ids, "\n"))

## We have this many missing patient Ids: 0

number_of_missing_subtypes <- sum(is.na(dcp$SUBTYPE))
cat(paste0("We have this many missing sample Ids: ",
number_of_missing_subtypes, "\n"))

## We have this many missing sample Ids: 7

# Drop patients with missing subtype
pclin_df_clean <- pclin_df_clean %>% drop_na(any_of("SUBTYPE"))

# Collect variable types
var_types <- sapply(dcp, class)

# Patient Id type check
cat(paste0("Patient Id type: ", var_types["PATIENT_ID"], "\n"))

## Patient Id type: character

# Cancer molecular subtype type check
cat(paste0("Cancer molecular subtype type: ", var_types["SUBTYPE"], "\n"))

## Cancer molecular subtype type: character

# Check subtypes are as expected
subtype_tab <- table(dcp$SUBTYPE)
subtype_tab

##
##     LGG_IDHmut-codel LGG_IDHmut-non-codel           LGG_IDHwt
##                  167                  248                  92

#All subtypes have a large part of the involvement in the dataset. So, we
will not remove any subtype.
```

Interestingly, the Codel and Non Codel could be merged together to better form an analysis on the gene expression prediction.

Now, We will merge the Patient Samples with the Patient Data, and write it to a csv file.

```
# Merge patient and sample clinical tibbles on columns of interest
clinical_df <- right_join(pclin_df_clean, sample_bca, by = "PATIENT_ID") %>%
  select(c(SAMPLE_ID, SUBTYPE))
clinical_df
```

```
## # A tibble: 514 × 2
##    SAMPLE_ID       SUBTYPE
##    <chr>           <chr>
##  1 TCGA-CS-4938-01 LGG_IDHmut-non-codel
##  2 TCGA-CS-4941-01 LGG_IDHwt
##  3 TCGA-CS-4942-01 LGG_IDHmut-non-codel
##  4 TCGA-CS-4943-01 LGG_IDHmut-non-codel
##  5 TCGA-CS-4944-01 LGG_IDHmut-non-codel
##  6 TCGA-CS-5390-01 LGG_IDHmut-codel
##  7 TCGA-CS-5393-01 LGG_IDHmut-non-codel
##  8 TCGA-CS-5394-01 LGG_IDHmut-non-codel
##  9 TCGA-CS-5395-01 LGG_IDHwt
## 10 TCGA-CS-5396-01 LGG_IDHmut-codel
## # ℹ 504 more rows
```

```
# Transpose the cleaned mRNA expression data
mrna_df_clean_final <- data_mrna_seq_v2_rsem %>%
     column_to_rownames(var = "Hugo_Symbol") %>%
     as.data.frame()
mrna_df <- as.tibble(t(mrna_df_clean_final), rownames = "SAMPLE_ID")

# Print to console
head(mrna_df, 10)
```

```
## # A tibble: 10 × 20,512
##    SAMPLE_ID      UBE2Q2P2 HMGB1P1 `RNU12-2P` SSX9P EZHIP EFCAB8 SRP14P1
TRIM75P
##    <chr>             <dbl>   <dbl>      <dbl> <dbl> <dbl>  <dbl>   <dbl>
<dbl>
##  1 TCGA-CS-4938-…    22.8     269.      0.420     0  2.10  0.420   12.6
0
##  2 TCGA-CS-4941-…    21.2     157.      1.73      0  3.45  0.345   15.2
0.345
##  3 TCGA-CS-4942-…    11.0     185.      0         0  1.73  0.346   14.9
0
##  4 TCGA-CS-4943-…     5.08    270.      0.326     0  1.30  0       10.4
0.326
##  5 TCGA-CS-4944-…    30.3     216.      0         0  3.03  0       23.2
0
##  6 TCGA-CS-5390-…    27.9     160.      2.56      0  8.76  0.365   10.6
0.365
```

```
##  7 TCGA-CS-5393-…     8.72    198.      0.807     0  2.82  0        9.68
0.404
##  8 TCGA-CS-5394-…    15.4     209.      0.669     0  2.01  0.335    5.69
1.00
##  9 TCGA-CS-5395-…    12.8     255.      0         0  0     0.740    8.89
0.370
## 10 TCGA-CS-5396-…    19.9     130.      0         0  2.76  0.307   10.4
0.307
## # ℹ 20,503 more variables: SPATA31B1P <dbl>, REXO1L6P <dbl>, SDR16C6P
<dbl>,
##   HSPB1P1 <dbl>, PPBPP1 <dbl>, ANKRD20A20P <dbl>, GTPBP6 <dbl>,
##   EFCAB12 <dbl>, A1BG <dbl>, A1CF <dbl>, A2BP1 <dbl>, A2LD1 <dbl>, A2M
<dbl>,
##   A2ML1 <dbl>, A4GALT <dbl>, A4GNT <dbl>, AAA1 <dbl>, AAAS <dbl>, AACS
<dbl>,
##   AACSL <dbl>, AADAC <dbl>, AADACL2 <dbl>, AADACL3 <dbl>, AADACL4 <dbl>,
##   AADAT <dbl>, AAGAB <dbl>, AAK1 <dbl>, AAMP <dbl>, AANAT <dbl>, AARS
<dbl>,
##   AARS2 <dbl>, AARSD1 <dbl>, AASDH <dbl>, AASDHPPT <dbl>, AASS <dbl>, …
```

```
# Merge gene expression data with clinical data
dataset_final <- merge(clinical_df, mrna_df, by = "SAMPLE_ID")

# Drop NA sample subtypes that got added in the joining
dataset_final <- dataset_final %>% drop_na(any_of("SUBTYPE"))

# Write to CSV
write_csv(x = dataset_final, file = "bca-mrna-expression-data-with-cancer-
subtypes.csv")
```

We have merged the files together, however, we are not done yet. there is still some preprocessing we need to do.

Part 3: Preprocessing

```
mydata <- read_csv("D:/School/University/Semester 8/Data Visualization and
Mining/bca-mrna-expression-data-with-cancer-subtypes.csv")
head(mydata,10)
```

```
## # A tibble: 10 × 20,513
##    SAMPLE_ID      SUBTYPE UBE2Q2P2 HMGB1P1 `RNU12-2P` SSX9P EZHIP EFCAB8
SRP14P1
##    <chr>          <chr>      <dbl>   <dbl>      <dbl> <dbl> <dbl>  <dbl>
<dbl>
##  1 TCGA-CS-4938-… LGG_ID…    22.8     269.      0.420     0  2.10  0.420
12.6
##  2 TCGA-CS-4941-… LGG_ID…    21.2     157.      1.73      0  3.45  0.345
15.2
##  3 TCGA-CS-4942-… LGG_ID…    11.0     185.      0         0  1.73  0.346
14.9
##  4 TCGA-CS-4943-… LGG_ID…     5.08    270.      0.326     0  1.30  0
```

```
10.4
##  5 TCGA-CS-4944-… LGG_ID…    30.3     216.      0         0  3.03  0
23.2
##  6 TCGA-CS-5390-… LGG_ID…    27.9     160.      2.56      0  8.76  0.365
10.6
##  7 TCGA-CS-5393-… LGG_ID…     8.72    198.      0.807     0  2.82  0
9.68
##  8 TCGA-CS-5394-… LGG_ID…    15.4     209.      0.669     0  2.01  0.335
5.69
##  9 TCGA-CS-5395-… LGG_ID…    12.8     255.      0         0  0     0.740
8.89
## 10 TCGA-CS-5396-… LGG_ID…    19.9     130.      0         0  2.76  0.307
10.4
## # ℹ 20,504 more variables: TRIM75P <dbl>, SPATA31B1P <dbl>, REXO1L6P
<dbl>,
##   SDR16C6P <dbl>, HSPB1P1 <dbl>, PPBPP1 <dbl>, ANKRD20A20P <dbl>,
##   GTPBP6 <dbl>, EFCAB12 <dbl>, A1BG <dbl>, A1CF <dbl>, A2BP1 <dbl>,
##   A2LD1 <dbl>, A2M <dbl>, A2ML1 <dbl>, A4GALT <dbl>, A4GNT <dbl>, AAA1
<dbl>,
##   AAAS <dbl>, AACS <dbl>, AACSL <dbl>, AADAC <dbl>, AADACL2 <dbl>,
##   AADACL3 <dbl>, AADACL4 <dbl>, AADAT <dbl>, AAGAB <dbl>, AAK1 <dbl>,
##   AAMP <dbl>, AANAT <dbl>, AARS <dbl>, AARS2 <dbl>, AARSD1 <dbl>, …

mydata %>% group_by(SUBTYPE) %>%
  summarise(MEAN_ERBB2_EXP = mean(ERBB2), .groups = 'drop')

## # A tibble: 3 × 2
##   SUBTYPE              MEAN_ERBB2_EXP
##   <chr>                         <dbl>
## 1 LGG_IDHmut-codel               699.
## 2 LGG_IDHmut-non-codel           597.
## 3 LGG_IDHwt                     1586.

ggplot(mydata, aes(x = ERBB2)) +
  geom_histogram(color=gg_blue,fill = gg_red) +
  ggtitle("mRNA expression over the frequency of values") +
  xlab("Expression level") +
  ylab("Frequency of Values") +
  theme_minimal()
```

Interestingly there gleams to be an understanding that with the current Gene expression data, the frequency tends to lie in the 0-1000 range. with a right skewed angle. We will later log transform the dataset to get a better understanding, however.

```
ggplot(mydata, aes(x = log2(ERBB2 + 1))) +
  geom_histogram(color=gg_red,fill = gg_blue) +
  ggtitle("log2 transformed mRNA expression values over the frequency level
          ") +
  xlab("Expression level") +
```

```
  ylab("Frequency") +
  theme_minimal()
```

Interestingly, the Log2 transformed data has now been symmetrical as all of the values have transformed to better fit the ggplot. Now, the log 2 transformed data lies in the areas from 8.5-10.

Log 2 Transformation

```
# Keeping
mydata_log2 <- mydata

# A Base R way: Log transform the expression values
column_offset <- 2 # Keep track of the first two columns of clinical annotations
hugo_gene_symbols <- colnames(mydata_log2)[-c(1 : column_offset)] # Store HUGO gene symbols
gene_count <- length(hugo_gene_symbols) # Number of genes
for (i in (column_offset + 1) : ncol(mydata_log2)) {
  mydata_log2[, i] <- log2(mydata[, i] + 1) # Log2 transform with unit offset
}

# Standardize the log2 transformed mRNA expression values
#This is useful to do in the event you encounter fold changes, and target unregulated genes in the analysis as well as resgulared genes.
data_log2_scaled <- mydata_log2

mu <- mean(mydata_log2 %>% select(where(is.numeric)) %>% as.matrix())
sd <- sd(mydata_log2 %>% select(where(is.numeric)) %>% as.matrix())

# Calculate sample Z-scores (A Base R way):
for(i in 3 : ncol(mydata_log2)) {
  data_log2_scaled[, i] <- (mydata_log2[, i] - mu) / sd
}
data_log2_scaled[1:10, ]
```

```
## # A tibble: 10 × 20,513
##    SAMPLE_ID     SUBTYPE UBE2Q2P2 HMGB1P1 `RNU12-2P` SSX9P  EZHIP EFCAB8
SRP14P1
##    <chr>         <chr>      <dbl>   <dbl>      <dbl> <dbl>  <dbl>  <dbl>
<dbl>
##  1 TCGA-CS-4938… LGG_ID…   -0.482   0.390      -1.49 -1.62 -1.21   -1.49
-0.683
##  2 TCGA-CS-4941… LGG_ID…   -0.507   0.197      -1.26 -1.62 -1.08   -1.51
-0.620
##  3 TCGA-CS-4942… LGG_ID…   -0.727   0.257      -1.62 -1.62 -1.26   -1.51
-0.627
##  4 TCGA-CS-4943… LGG_ID…   -0.971   0.391      -1.52 -1.62 -1.32   -1.62
-0.745
```

```
##  5 TCGA-CS-4944… LGG_ID…   -0.384   0.312      -1.62 -1.62 -1.12   -1.62
-0.476
##  6 TCGA-CS-5390… LGG_ID…   -0.412   0.204      -1.16 -1.62 -0.802  -1.51
-0.740
##  7 TCGA-CS-5393… LGG_ID…   -0.803   0.281      -1.41 -1.62 -1.14   -1.62
-0.769
##  8 TCGA-CS-5394… LGG_ID…   -0.614   0.299      -1.44 -1.62 -1.22   -1.52
-0.938
##  9 TCGA-CS-5395… LGG_ID…   -0.678   0.372      -1.62 -1.62 -1.62   -1.42
-0.797
## 10 TCGA-CS-5396… LGG_ID…   -0.528   0.130      -1.62 -1.62 -1.14   -1.52
-0.745
## # ℹ 20,504 more variables: TRIM75P <dbl>, SPATA31B1P <dbl>, REXO1L6P
<dbl>,
##   SDR16C6P <dbl>, HSPB1P1 <dbl>, PPBPP1 <dbl>, ANKRD20A20P <dbl>,
##   GTPBP6 <dbl>, EFCAB12 <dbl>, A1BG <dbl>, A1CF <dbl>, A2BP1 <dbl>,
##   A2LD1 <dbl>, A2M <dbl>, A2ML1 <dbl>, A4GALT <dbl>, A4GNT <dbl>, AAA1
<dbl>,
##   AAAS <dbl>, AACS <dbl>, AACSL <dbl>, AADAC <dbl>, AADACL2 <dbl>,
##   AADACL3 <dbl>, AADACL4 <dbl>, AADAT <dbl>, AAGAB <dbl>, AAK1 <dbl>,
##   AAMP <dbl>, AANAT <dbl>, AARS <dbl>, AARS2 <dbl>, AARSD1 <dbl>, …
```

```
# Keep top 5000 most variable genes
hugo_gene_symbols <- colnames(data_log2_scaled)[-c(1, 2)]
top_n <- 5000 # keep the top 5000 with highest variance across patient
samples
gexp_mat <- data_log2_scaled %>%
  select(where(is.numeric)) %>%
  as.matrix()
gexp_sds <- apply(gexp_mat, 2, sd)
keep_hugo_gene_symbols <- hugo_gene_symbols[order(gexp_sds, decreasing =
TRUE)[1 : top_n]]
drop_hugo_gene_symbols <- setdiff(hugo_gene_symbols, keep_hugo_gene_symbols)

print("low variance genes:")
```

```
## [1] "low variance genes:"
```

```
length(drop_hugo_gene_symbols)
```

```
## [1] 15511
```

```
# Filter out low variance genes
data_log2_scaled_reduced <- data_log2_scaled %>%
  select(all_of(c("SAMPLE_ID", "SUBTYPE", keep_hugo_gene_symbols)))
data_log2_scaled_reduced[1:10, ]
```

```
## # A tibble: 10 × 5,002
##    SAMPLE_ID    SUBTYPE    XIST RPS4Y1   DDX3Y  KDM5D  USP9Y EIF1AY    UTY
TTTY15
##    <chr>        <chr>     <dbl>  <dbl>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
```

```
<dbl>
##  1 TCGA-CS-493… LGG_ID…  1.31  -1.49  -1.49   -1.62  -1.62  -1.62  -1.62
-1.62
##  2 TCGA-CS-494… LGG_ID… -1.22   1.15   0.932   0.669  0.635  0.428  0.361
0.227
##  3 TCGA-CS-494… LGG_ID…  1.17  -1.51  -1.43   -1.62  -1.51  -1.62  -1.62
-1.62
##  4 TCGA-CS-494… LGG_ID… -0.964  0.553  0.937   0.765  0.501  0.582  0.495
0.663
##  5 TCGA-CS-494… LGG_ID… -0.894  1.49   0.744   0.503  0.520  0.714  0.220
0.265
##  6 TCGA-CS-539… LGG_ID…  1.86  -1.62  -1.51   -1.62  -1.62  -1.62  -1.62
-1.62
##  7 TCGA-CS-539… LGG_ID… -0.783  1.09   0.694   0.533  0.545  0.314  0.314
0.290
##  8 TCGA-CS-539… LGG_ID… -0.975  0.237  0.0671 -0.226 -0.276 -0.477 -0.555
-0.587
##  9 TCGA-CS-539… LGG_ID… -0.703  1.31   1.08    0.806  0.952  0.717  0.638
0.538
## 10 TCGA-CS-539… LGG_ID…  1.50  -1.62  -1.62   -1.62  -1.62  -1.62  -1.62
-1.62
## # ℹ 4,992 more variables: ZFY <dbl>, TSIX <dbl>, CYorf15A <dbl>, GSTM1
<dbl>,
##   CYorf15B <dbl>, GSTT1 <dbl>, LTF <dbl>, CHI3L1 <dbl>, OPALIN <dbl>,
##   SLC17A7 <dbl>, POSTN <dbl>, TMSB4Y <dbl>, NLGN4Y <dbl>, NEFL <dbl>,
##   GJB6 <dbl>, WIF1 <dbl>, KIAA0748 <dbl>, GPR26 <dbl>, DAO <dbl>,
##   SFRP2 <dbl>, TLX1 <dbl>, PRLHR <dbl>, PCDHGA10 <dbl>, HOXA7 <dbl>,
##   GABRA1 <dbl>, VSNL1 <dbl>, KCNS1 <dbl>, PDYN <dbl>, SLC6A7 <dbl>,
##   psiTPTE22 <dbl>, LINC00689 <dbl>, MOXD1 <dbl>, LHX5 <dbl>, HOXA10
<dbl>, …
```

```
write_csv(x = data_log2_scaled_reduced, file ="bca-mrna-expression-data-with-
cancer-subtypes-preprocessed.csv") #You will find this file in your working
directory
```

Pt 4: Modeling and Testing/Training, using the log2 model

```
data <- read_csv("bca-mrna-expression-data-with-cancer-subtypes-
preprocessed.csv")
head(data)
```

```
## # A tibble: 6 × 5,002
##   SAMPLE_ID      SUBTYPE   XIST RPS4Y1  DDX3Y  KDM5D  USP9Y EIF1AY    UTY
TTTY15
##   <chr>          <chr>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
<dbl>
## 1 TCGA-CS-4938-… LGG_ID…  1.31  -1.49  -1.49  -1.62  -1.62  -1.62  -1.62
-1.62
## 2 TCGA-CS-4941-… LGG_ID… -1.22   1.15   0.932  0.669  0.635  0.428  0.361
0.227
## 3 TCGA-CS-4942-… LGG_ID…  1.17  -1.51  -1.43  -1.62  -1.51  -1.62  -1.62
```

```
-1.62
## 4 TCGA-CS-4943-… LGG_ID… -0.964  0.553  0.937  0.765  0.501  0.582  0.495
0.663
## 5 TCGA-CS-4944-… LGG_ID… -0.894  1.49   0.744  0.503  0.520  0.714  0.220
0.265
## 6 TCGA-CS-5390-… LGG_ID…  1.86  -1.62  -1.51  -1.62  -1.62  -1.62  -1.62
-1.62
## # ℹ 4,992 more variables: ZFY <dbl>, TSIX <dbl>, CYorf15A <dbl>, GSTM1
<dbl>,
## #   CYorf15B <dbl>, GSTT1 <dbl>, LTF <dbl>, CHI3L1 <dbl>, OPALIN <dbl>,
## #   SLC17A7 <dbl>, POSTN <dbl>, TMSB4Y <dbl>, NLGN4Y <dbl>, NEFL <dbl>,
## #   GJB6 <dbl>, WIF1 <dbl>, KIAA0748 <dbl>, GPR26 <dbl>, DAO <dbl>,
## #   SFRP2 <dbl>, TLX1 <dbl>, PRLHR <dbl>, PCDHGA10 <dbl>, HOXA7 <dbl>,
## #   GABRA1 <dbl>, VSNL1 <dbl>, KCNS1 <dbl>, PDYN <dbl>, SLC6A7 <dbl>,
## #   psiTPTE22 <dbl>, LINC00689 <dbl>, MOXD1 <dbl>, LHX5 <dbl>, HOXA10
<dbl>, …
```

```
# Combine the Non-codal and codal subtypes together to make for easier
modeling
data <- data %>%
  mutate(SUBTYPE = recode(SUBTYPE, "LGG_IDHmut-codel" = "LGG_IDHmut",
"LGG_IDHmut-non-codel" = "LGG_IDHmut"))

# Set random seed for reproducibility reason
set.seed(343534) #Random Seed = Student ID

# Create training/test data split. We will do an 80/20 rule, meaning 80% of
the dataset is for training while the rest is testing.
index <- createDataPartition(data$SUBTYPE, p = 0.80, list = FALSE)
train <- data[index,]
test <- data[-index,]

#Principle Component Analysis - Train and test
train_pca_fit <- train %>%
  select(where(is.numeric)) %>% # retain only numeric columns
  prcomp()

#Train
train_pca <- predict(train_pca_fit, train) %>%
  as_tibble() %>%
  add_column(SUBTYPE = train$SUBTYPE, .before = 1)

train_pca
```

```
## # A tibble: 406 × 407
##    SUBTYPE       PC1     PC2    PC3    PC4      PC5    PC6    PC7    PC8
PC9
##    <chr>       <dbl>   <dbl>  <dbl>  <dbl>    <dbl>  <dbl>  <dbl>  <dbl>
<dbl>
##  1 LGG_IDHwt -10.6   -19.6    -2.71  0.311  -6.47    0.837 -1.64   3.17  -
1.45
```

```
##  2 LGG_IDHm…  -2.49    0.443   5.18 -6.39   -3.61   -2.68   3.36  -6.75  -
3.55
##  3 LGG_IDHm…  -6.12   11.1    -4.43 -7.56   -1.60   -9.15   1.22   1.65  -
0.841
##  4 LGG_IDHm…  -2.36   -0.394   8.91  3.33   -0.0501  9.86  -0.989  2.19
5.10
##  5 LGG_IDHm…  10.5     7.09   -7.27  2.54   -4.06   -2.69   5.65  -4.65
1.43
##  6 LGG_IDHm…   3.04   22.2   -12.1   1.40   -8.82    6.96   1.37   0.738
2.42
##  7 LGG_IDHwt  -0.453 -11.9    -2.40 -1.06  -14.3    -3.94  -5.74   3.79
1.80
##  8 LGG_IDHm…  -3.41   10.2   -12.4   2.01   -4.54    7.23  10.5   -5.43
2.57
##  9 LGG_IDHwt   4.32  -21.5    -4.11 -4.06   -4.86   -1.34   0.492 -5.12  -
0.0465
## 10 LGG_IDHwt -15.6   -14.1    -6.06  2.57   -6.23   -1.82   1.07   2.08
4.18
## # ℹ 396 more rows
## # ℹ 397 more variables: PC10 <dbl>, PC11 <dbl>, PC12 <dbl>, PC13 <dbl>,
## #   PC14 <dbl>, PC15 <dbl>, PC16 <dbl>, PC17 <dbl>, PC18 <dbl>, PC19
<dbl>,
## #   PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>, PC24 <dbl>, PC25
<dbl>,
## #   PC26 <dbl>, PC27 <dbl>, PC28 <dbl>, PC29 <dbl>, PC30 <dbl>, PC31
<dbl>,
## #   PC32 <dbl>, PC33 <dbl>, PC34 <dbl>, PC35 <dbl>, PC36 <dbl>, PC37
<dbl>,
## #   PC38 <dbl>, PC39 <dbl>, PC40 <dbl>, PC41 <dbl>, PC42 <dbl>, PC43
<dbl>, …
```

```
#Test
test_pca <- predict(train_pca_fit, test) %>%
  as_tibble() %>%
  add_column(SUBTYPE = test$SUBTYPE, .before = 1)

# Print to console
test_pca
```

```
## # A tibble: 101 × 407
##    SUBTYPE       PC1     PC2    PC3    PC4     PC5    PC6     PC7     PC8
PC9
##    <chr>       <dbl>   <dbl>  <dbl>  <dbl>   <dbl>  <dbl>   <dbl>   <dbl>
<dbl>
##  1 LGG_IDHmut  -8.67  4.90   11.8   -2.40   -7.42   4.41   -4.83  -6.04
1.89
##  2 LGG_IDHmut   5.81  0.0840  7.04  -4.44   -0.663 -3.00    1.86  -0.0516
2.97
##  3 LGG_IDHwt  -17.0  -4.06   -2.20   8.12  -12.8    3.19  -12.3    2.56
7.19
```

```
##  4 LGG_IDHmut  -3.55  7.51    4.51  -3.43    1.15   4.30   -4.10  -4.12
2.03
##  5 LGG_IDHmut -14.2   4.11    6.40  -4.52   -2.10  -1.19    3.27   1.48
0.283
##  6 LGG_IDHmut   2.17  4.98    0.295 -4.33    2.60  -0.320  -0.865  5.20
-5.74
##  7 LGG_IDHmut  16.3  -0.929   2.48   0.620  -2.67   0.872   3.66   2.63
0.314
##  8 LGG_IDHmut   4.79  3.01    3.05   6.87    1.38   1.25    4.03   5.08
-2.94
##  9 LGG_IDHmut   7.30  1.10    1.28  -2.73   -2.15  -7.38   -2.35  -4.92
-4.83
## 10 LGG_IDHmut   4.47 -1.92   -0.829 -8.45    3.16  -0.244  -0.713 -3.39
-0.500
## # ℹ 91 more rows
## # ℹ 397 more variables: PC10 <dbl>, PC11 <dbl>, PC12 <dbl>, PC13 <dbl>,
## #   PC14 <dbl>, PC15 <dbl>, PC16 <dbl>, PC17 <dbl>, PC18 <dbl>, PC19
<dbl>,
## #   PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>, PC24 <dbl>, PC25
<dbl>,
## #   PC26 <dbl>, PC27 <dbl>, PC28 <dbl>, PC29 <dbl>, PC30 <dbl>, PC31
<dbl>,
## #   PC32 <dbl>, PC33 <dbl>, PC34 <dbl>, PC35 <dbl>, PC36 <dbl>, PC37
<dbl>,
## #   PC38 <dbl>, PC39 <dbl>, PC40 <dbl>, PC41 <dbl>, PC42 <dbl>, PC43
<dbl>, …
```

```
# Subtype counts in the training data
table(train$SUBTYPE)
```

```
##
## LGG_IDHmut  LGG_IDHwt
##        332         74
```

```
# Subtype counts in test data
table(test$SUBTYPE)
```

```
##
## LGG_IDHmut  LGG_IDHwt
##         83         18
```

As we are going to take the principle components from the worst outcome, we must choose LGG_IDHwt. there are 74 here in the training set, meaning that if we were to follow a 1/10 rule, we would consider the 7 principle components

We need to have a reference subtype ready for our training dataset. Luckily, there is a subtype with many outcomes which we just combined with 2 of the same: LGG_IDHmut

```
train_pca$SUBTYPE <- relevel(factor(train_pca$SUBTYPE), ref = "LGG_IDHmut")

# Create training sub-dataset consisting of the top 7 principal components
```

```r
top <- 7
train_pca_sub1 <- train_pca[,1 : (top + 1)]

multinom_fit1 <- multinom(SUBTYPE ~ ., data = train_pca_sub1)
```

```
## # weights:  9 (8 variable)
## initial  value 281.417755
## iter  10 value 34.856148
## iter  20 value 12.134647
## iter  30 value 10.927312
## final  value 10.923340
## converged
```

```r
# Print model summary to console
summary(multinom_fit1)
```

```
## Call:
## multinom(formula = SUBTYPE ~ ., data = train_pca_sub1)
##
## Coefficients:
##                 Values Std. Err.
## (Intercept) -9.9818620 4.0232718
## PC1         -0.2209167 0.1006359
## PC2         -1.3168593 0.4963647
## PC3         -0.9638443 0.4211408
## PC4          0.7400206 0.3374684
## PC5         -1.1536813 0.4860487
## PC6         -0.1130933 0.1339013
## PC7         -0.4936171 0.3143921
##
## Residual Deviance: 21.84668
## AIC: 37.84668
```

```r
# Predict test data tumor subtypes
p1 <- predict(multinom_fit1, test_pca)

# Creating a confusion matrix using the table command
confusion_mat1 <- table(p1, test_pca$SUBTYPE)
cat("\nConfusion Matrix (top 7 PCs):\n")
```

```
##
## Confusion Matrix (top 7 PCs):
```

```r
confusion_mat1
```

```
##
## p1           LGG_IDHmut LGG_IDHwt
##   LGG_IDHmut         78         0
##   LGG_IDHwt           5        18
```

```r
# Calculate missclassification rate
accuracy_rate1 <- sum(diag(confusion_mat1)) / sum(confusion_mat1)
missclassification_rate1 <- 1 - accuracy_rate1
cat("\nMisclassification Rate (top 7 PCs):\n")

##
## Misclassification Rate (top 7 PCs):

missclassification_rate1

## [1] 0.04950495

# Fit multinomial logistic regression model to training data
multinom_fit2 <- multinom(SUBTYPE ~ PC1 + PC2, data = train_pca)

## # weights:  4 (3 variable)
## initial  value 281.417755
## iter  10 value 51.974001
## final  value 51.972734
## converged

# Predict test data tumor subtypes
p2 <- predict(multinom_fit2, test_pca)

# Create confusion matrix
confusion_mat2 <- table(p2, test_pca$SUBTYPE)
cat("\nConfusion Matrix (top 2 PCs):\n")

##
## Confusion Matrix (top 2 PCs):

confusion_mat2

##
## p2           LGG_IDHmut LGG_IDHwt
##   LGG_IDHmut         83         3
##   LGG_IDHwt           0        15

# Calculate missclassification rate
accuracy_rate2 <- sum(diag(confusion_mat2)) / sum(confusion_mat2)
missclassification_rate2 <- 1 - accuracy_rate2
cat("\nMisclassification Rate (top 2 PCs):\n")

##
## Misclassification Rate (top 2 PCs):

missclassification_rate2

## [1] 0.02970297

# Test 3, this time using 100 Principle components!
hund <- 100
train_pca_sub3 <- train_pca[, 1 : (hund + 1)]
```

```r
# Fit multinomial logistic regression model to training data
multinom_fit3 <- multinom(SUBTYPE ~ ., data = train_pca_sub3)
```

```
## # weights:  102 (101 variable)
## initial  value 281.417755
## iter  10 value 31.824869
## iter  20 value 17.096907
## iter  30 value 6.427440
## iter  40 value 1.912188
## iter  50 value 0.298368
## iter  60 value 0.004944
## iter  70 value 0.000146
## iter  70 value 0.000083
## iter  70 value 0.000076
## final  value 0.000076
## converged
```

```r
# prediction performed on the datasets
p3 <- predict(multinom_fit3, test_pca)

# Create confusion matrix- useful for understanding subtypes during test
confusion_mat3 <- table(p3, test_pca$SUBTYPE)
cat("\nConfusion Matrix (100 Features/PCs):\n")
```

```
##
## Confusion Matrix (100 Features/PCs):
```

```r
confusion_mat3
```

```
##
## p3           LGG_IDHmut LGG_IDHwt
##   LGG_IDHmut         83         0
##   LGG_IDHwt           0        18
```

```r
# Calculate missclassification rate
accuracy_rate3 <- sum(diag(confusion_mat3)) / sum(confusion_mat3)
missclassification_rate3 <- 1 - accuracy_rate3
cat("\nMisclassification Rate (00 PCs/Features):\n")
```

```
##
## Misclassification Rate (00 PCs/Features):
```

```r
missclassification_rate3
```

```
## [1] 0
```

Executive Summary:

The following logistic model with the 3 different principle components: 7, 2, and 100, have created great missclassification rates and show great potential in the Logistic model

Model 1 has classified 95% of the molecular subtypes of the Brain cancer gene expression dataset correctly. Great potential! but there is more.

Model 2, which uses 2 principle components, has garnered 97% of the molecular subtypes correctly in its training/testing phase. This means that it is better than Model 1 and while it only uses 2 features, only has a 3% potential in finding incorrect values.

Lastly, Model 3, which uses 100, has in a most surprising feat, garnered 100% of the molecular subtypes, No mistakes! This model would be perfect for this data set, although the amount of features it uses needs to be brought into attention.

In the end, for the most potentially great model, you would think it to be **model 3**, as there is seemingly no reason why to pass the model with the highest accuracy and perfect too. However, the number of features a model has is shown to be potentially bad in terms of the model's longevity. Keeping the amount of features minimal is key to collecting a great and useful model outside of its practice datasets.

This is why I believe that **Model 2** is the best in terms of the highest accuracy and lowest features. three percent may be risky, but it shows the model can avoid being overfit.

This Statistical Analysis has ended with the following conclusion to our hypothesis:

Yes, the possibility of predicting the molecular subtype using gene expression data is wholly possible. We have analyzed and done visual inspections on the dataset to clear it of missing data, and examined the subtypes and other factors when preprocessing the data. Finally, we have utilized a great model, the Logistic model, in order to find and predict our testing set correctly.

Results & Conclusion:

Utilizing the Logistic Model, and testing with several different features, we gathered that utilizing 2 features in our model yielded the best accuracy with the lowest number of principle components, thereby avoiding over fitting somewhat. We accomplished our hypothesis with the endgoal of utilizing this model on other different forms of tumors.

This analysis did not come without several challenges, however. One of these challenges, such as the requirement of preprocessed and data fit for the model is required. For certain datasets of the same type of tumor, this woud not be possible with the same model to determine molecular subtype over gene expression. Several acts had to be performed on the patient data, sample data, and rsem data, in order to fit it for the model.

In conclusion, the course that taught me how to perform Data-Visualization and processing, taught me how to perform modelling and preprocessing on datasets such as the ones on this analysis. Follow up work, such as continuing the examination of different tumors, such

as colon cancer or lung cancer could yield results capable of predicting the molecular subtype, similar to what was done here. Further tests could even possibly yield the gene expression, if with careful research and better management of the data. The results of this led myself to understand more about cancer and the various gene expressions that are involved in the mutant disease, further providing a scientific explanation on it.

The results of the model utilizing the multinomial approach are exceptionally promising, boasting an impressive accuracy rate of 97%. This level of accuracy indicates the robustness and effectiveness of the multinomial model in capturing complex patterns and relationships within the data. The high accuracy suggests that the model can reliably classify and predict outcomes with a high degree of confidence, making it a valuable asset in decision-making processes. Such a strong performance underscores the model's capability to handle diverse datasets and underscores its potential for real-world applications where precision and reliability are paramount.

Thank you