HARVARD MEDICAL SCHOOL | BLAVATNIK INSTITUTE BIOMEDICAL INFORMATICS

Identification of Prognostic Pathogenic Genes and Clinical Trends in Individuals with Glioblastoma

Author: Kevin Liu

2022-10-20

Contents

Background

Glioblastoma (GBM) is a common cancer of the central nervous system and affected individuals has shown poor prognosis despite the currently available treatment options. [1] Furthermore, immune-related genes have been shown to play a prominent role in tumor gene expression. [1] Previously, Liang et al. identified 24 immune genes related to the prognosis of GBM using genes from the ImmPort database; of which, the CCL1, LPA, and SH2D1B genes were identified as prognostic pathogenic genes of GBM based on their calculated hazard ratios. [1-2] Additionally, it was found that CCL1 is expressed at higher levels in males relative to females while BMP1 and OSMR are expressed at higher levels in those with ages greater than 50 years old. [1]

In this analysis, we hypothesize that the differential gene expression of prognostic pathogenic genes between normal tissue samples and tumor tissue samples will be the main driving factor influencing our principal components (PCs). Therefore, this analysis aims to identify the prognostic pathogenic genes using PCA, examine the gene expression patterns and interrelationships of both samples from normal tissue and GBM tissue via hierarchical clustering, and replicate the clinical findings of differential gene expression based on age group and sex by Liang et al.

Since the authors' analyses were based on the TCGA-GBM dataset, which is readily available to the public, we will attempt to explore the utility of PCA weights in the screening of prognostic pathogenic genes among the 24 immune genes in GBM instead of the usage of a hazard ratio. The usage of PCA is justified here as we would anticipate that genes with the highest relevance to the etiology of GBM to demonstrate the highest relevance to the PCs that explain the most variance in our RNA-seq read count data between each of the individuals.

Subsequently, we will attempt utilize hierarchical clustering to explore the 24 immune gene expression patterns in terms of how each of the genes are related to one another and whether sample subtypes can be clustered based on the normalized RNA-seq read counts of the 24 immune genes for each sample donor.

Finally, we will attempt to replicate the clinical findings between select genes of interest and their associations with age group and sex identified by Liang et al.

References

[1] Liang P, Chai Y, Zhao H, Wang G. Predictive Analyses of Prognostic-Related Immune Genes and Immune Infiltrates for Glioblastoma. Diagnostics (Basel). 2020;10(3):177. Published 2020 Mar 24. doi: 10.3390/diagnostics10030177

[2] Bhattacharya S, Dunn P, Thomas CG, et al. ImmPort, toward repurposing of open access immunological assay data for translational and clinical research. Sci Data. 2018;5:180015. Published 2018 Feb 27. doi:10.1038/sdata.2018.15

Data Wrangling

We will begin by loading all libraries we expect to use and reading in our data. Here, we read in the list of bar codes, generate a query (i.e., search) for the TCGA-GBM gene expression data set involving samples from either the primary tumor or normal solid tissue (i.e., normal tissue around the tumor), download the data to our local work directory, and read in the downloaded data.

```
# load in all libraries we expect to use
library(tidyverse)
library(SummarizedExperiment)
library(TCGAbiolinks)
library(pheatmap)

# read in barcodes
barcode_list = read.table("GBM_barcodes.txt")$V1

# generate a query to search for the TCGA-GBM dataset
query_GBM = GDCquery(
  project = "TCGA-GBM",
  data.category = "Transcriptome Profiling",
  data.type = "Gene Expression Quantification",
  experimental.strategy = "RNA-Seq",
  workflow.type = "STAR - Counts",
  sample.type = c("Primary Tumor", "Solid Tissue Normal"),
  barcode = barcode_list)

# download the data using our query
GDCdownload(query_GBM, method = "api", files.per.chunk = 10)

# store the data as a SummarizedExperiment object
GBM_data = GDCprepare(query_GBM)
```

```
## |                                                                    |   0%                                  |=
```

Here, we will extract the read counts and sample metadata from our SummarizedExperiment object, manipulate the row names of the sample metadata dataframe to use "." instead of "-" as separators for easy merging with matrices in the future, create a new column to store the sample subtypes (summary table shown below) and the age groups (">50" and "<=50"), and fill in the NA's with the vital status column with "Unknown"'s.

Subsequently, we will extract the gene metadata for future use.

```
# extract RNA-seq read counts table as a matrix
GBMMatrix = SummarizedExperiment::assay(GBM_data)

# extract sample metadata as a dataframe
sample_metadata = data.frame(colData(GBM_data))

# convert "-" in row names of sample metadata to "."
row.names(sample_metadata) = row.names(sample_metadata) %>%
  str_replace_all("-", ".")

# create new sample subtype column and fill in NAs with "Unknown"
sample_metadata$sample_subtype = sample_metadata$paper_Transcriptome.Subtype %>%
  as.character() %>%
  str_replace_na("Unknown")
```

```r
# for the each row, replace the sample subtype data to "Solid Tissue Normal" if
# it is found within the sample type column
sample_metadata$sample_subtype[sample_metadata$sample_type == "Solid Tissue Normal"] =
  "Solid Tissue Normal"

# replace all NAs and "Not Reported" values in the vital status column with "Unknown"
sample_metadata$vital_status = sample_metadata$vital_status %>%
  str_replace_na("Unknown") %>%
  str_replace("^Not.*", "Unknown")

# create a column of age groups, either >50yo or <=50yo; if NA, replace with "Unknown"
sample_metadata = sample_metadata %>%
  mutate(age_group = case_when(age_at_index >  50 ~ ">50",
                               age_at_index <= 50 ~ "<=50",
                               TRUE               ~ "Unknown"))

# extract gene metadata and store as dataframe
gene_metadata = data.frame(rowData(GBM_data))
```

Liang et al. identified 24 immune genes related to GBM prognosis. We will create a vector of the gene names and filter the gene metadata to keep only the rows that have a matching gene name.

```r
# 24 immune genes related to GBM prognosis
immune_genes = c("CD1D", "CXCL13", "CCL5", "BMP1", "CCL1", "DEFA3", "MMP9",
                 "NOD2", "PLTP", "LPA", "FABP5", "CHIT1", "PTX3", "LILRB3",
                 "FCGR2B", "FPR2", "AREG", "IL24", "IL32", "MDK", "TNFSF14",
                 "IL1R2", "OSMR", "SH2D1B")

# keep only rows of genes that have the 24 genes we want in gene metadata
gene_metadata = gene_metadata %>% filter(gene_name %in% immune_genes)
```

Next, we will normalize the read counts and filter the normalized counts table to keep only those 24 genes that we want to analyze.

```r
# normalize counts to 1M
GBMMatrix_norm = apply(GBMMatrix, 2, function(x){log2(x/sum(x)*1000000+1)})

# filter to keep only immune genes related to GBM prognosis in the counts matrix
GBMMatrix_norm_filter = GBMMatrix_norm %>%
  data.frame() %>%
  rownames_to_column("gene_id") %>%
  filter(gene_id %in% row.names(gene_metadata)) %>%
  column_to_rownames("gene_id") %>%
  as.matrix()
```

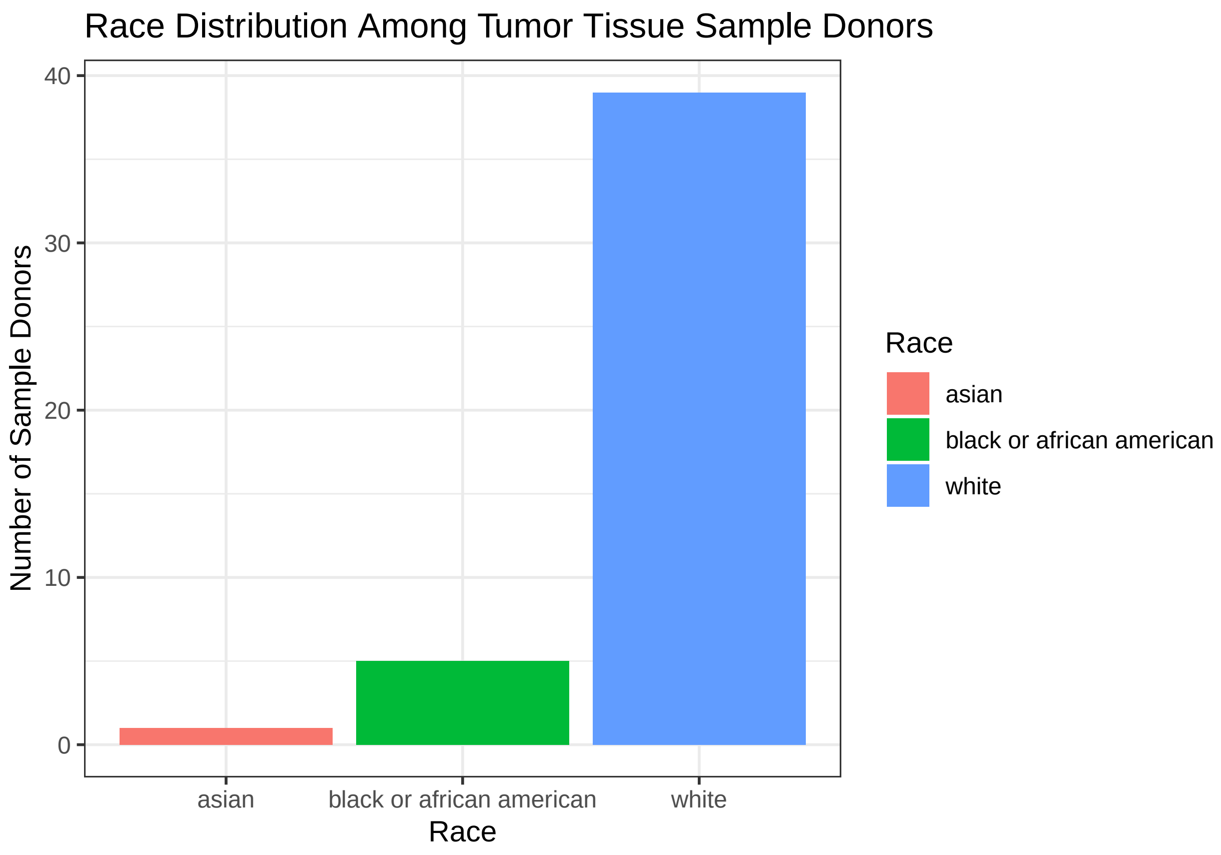
Exploratory Analysis of Sample Donor Demographics

Prior to conducting our main analysis, we will conduct an exploratory analysis on the sample donor demographics to identify any potential sources of bias within our data.

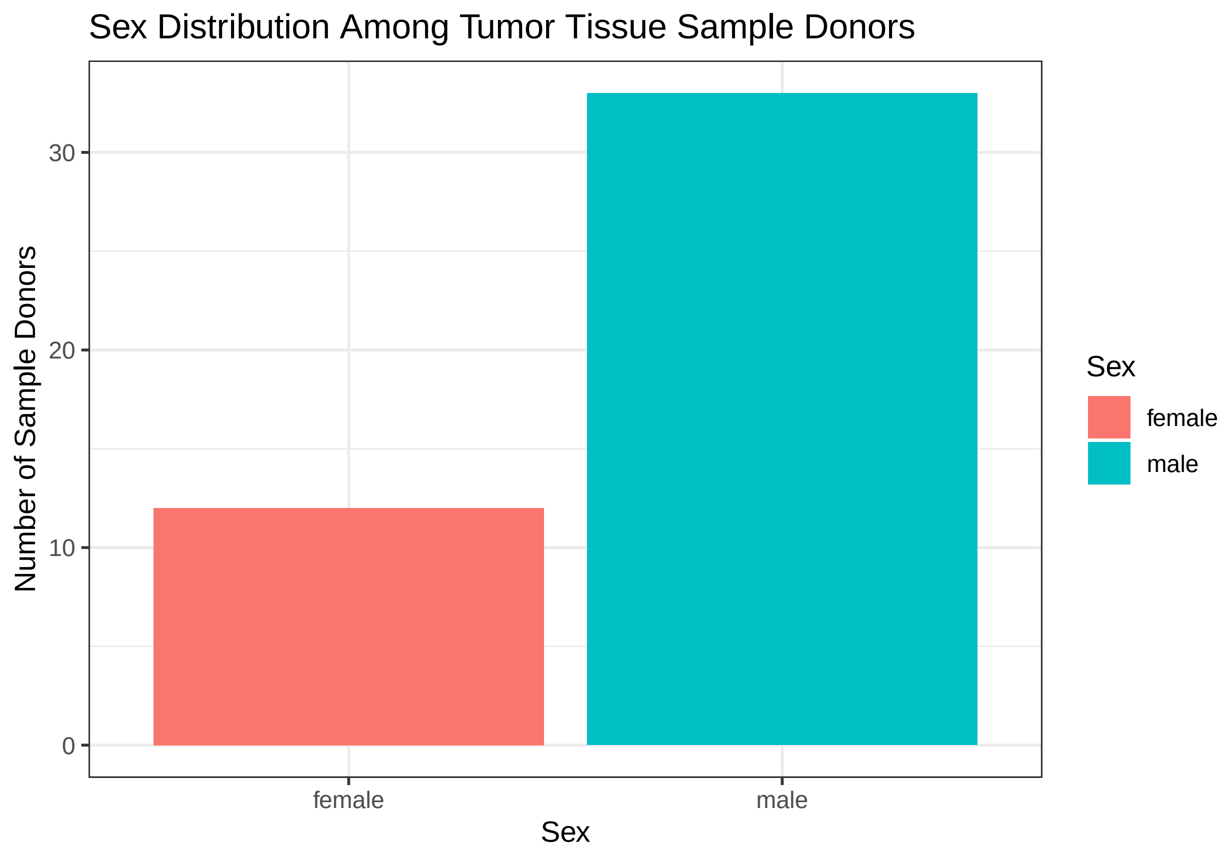
Here, we construct bar plots to illustrate the distribution of tumor tissue sample donors in terms of race, sex, age group, and the number of sample donors for all tissue sample types per tissue source site.

We additionally illustrate the numerical distribution of age at index, age at diagnosis, and survival time in years since indexing for tumor tissue sample donors by sample subtype.

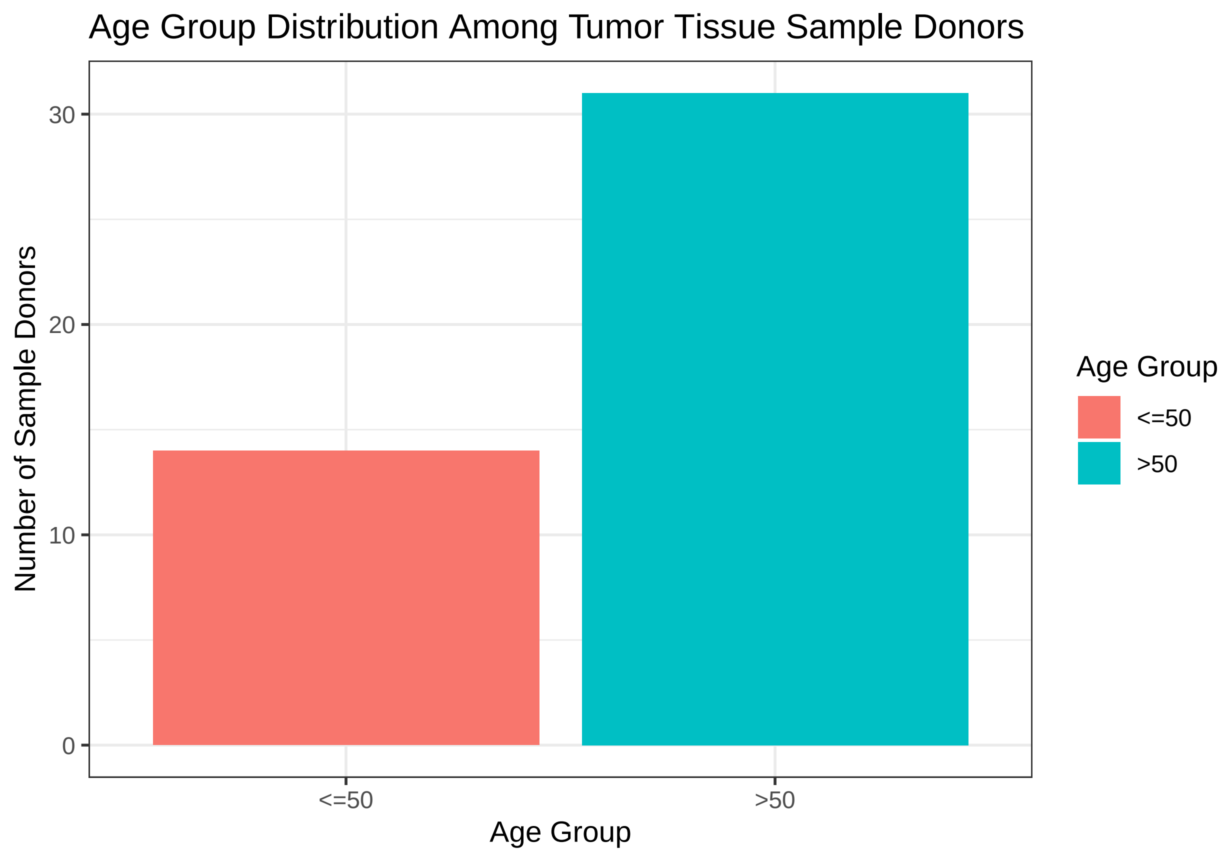
```r
# define vector of categorical demographic variables
demographic_vars = c("race", "gender", "paper_Tissue.source.site", "age_group", "sample_subtype")

# define vector of numerical demographic variables
num_demographic_vars = c("age_at_index", "paper_Survival..months.", "paper_Age..years.at.diagnosis.")

# plot bar plot among tumor tissue smaple donors in terms of race
sample_metadata %>%
  filter(!is.na(race)) %>% # filter out rows with NAs (solid tissue normal samples)
  ggplot() +
  geom_bar(aes(x = race, fill = race)) +
  theme_bw() +
  labs(x = "Race", y = "Number of Sample Donors", fill = "Race",
       title = "Race Distribution Among Tumor Tissue Sample Donors")
```



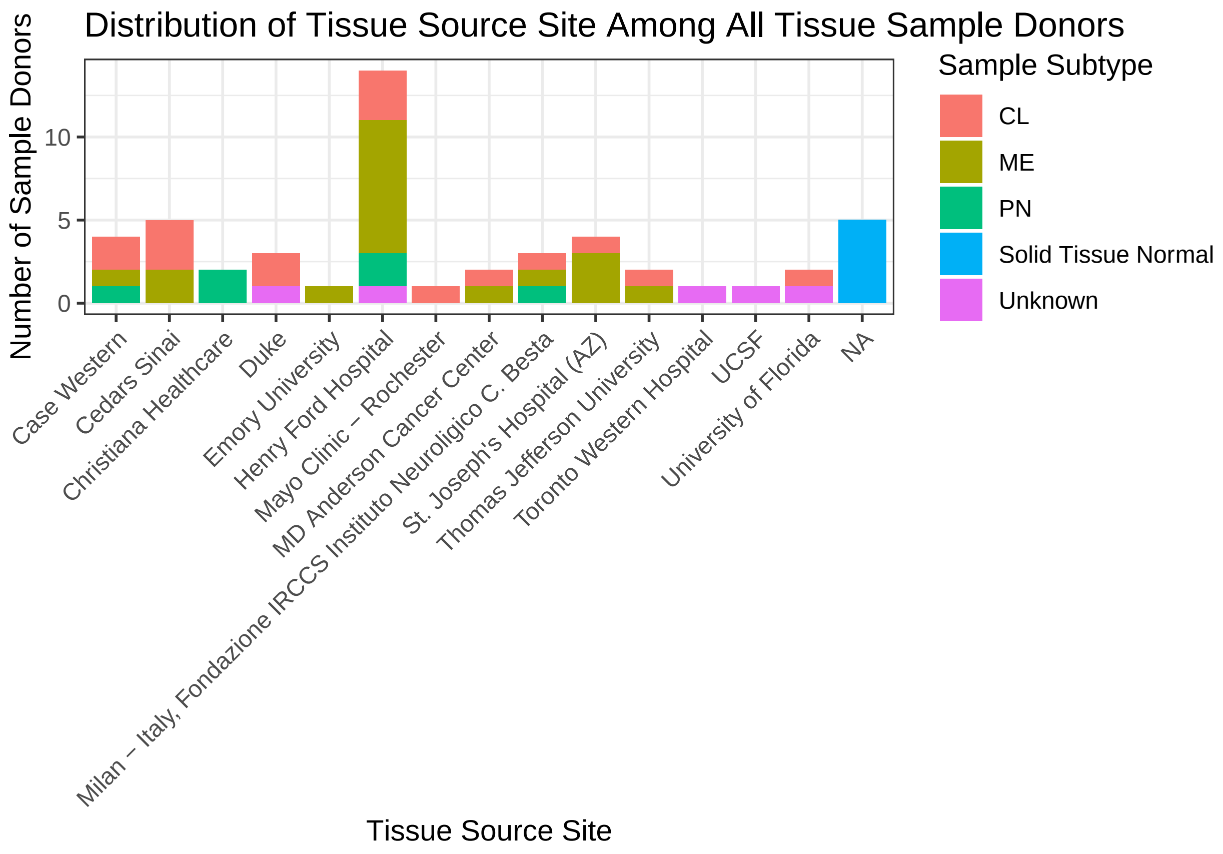
```r
# plot bar plot among tumor tissue sample donors in terms of gender
sample_metadata %>%
  filter(!is.na(gender)) %>% # filter out gender with NAs (solid tissue normal samples)
  ggplot() +
  geom_bar(aes(x = gender, fill = gender)) +
  theme_bw() +
  labs(x = "Sex", y = "Number of Sample Donors", fill = "Sex",
       title = "Sex Distribution Among Tumor Tissue Sample Donors")
```



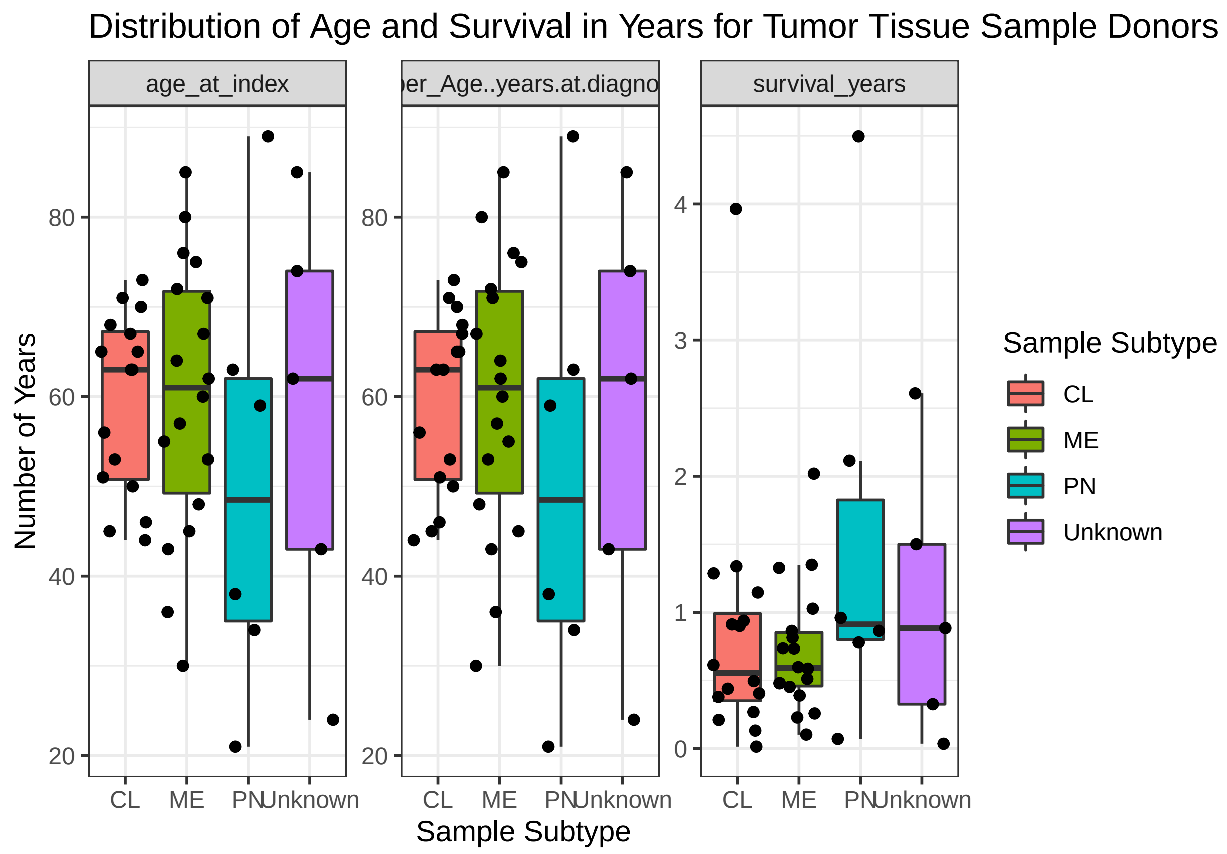
```r
# plot bar plot among tumor tissue sample donors in terms of age groups
sample_metadata %>%
  filter(age_group != "Unknown") %>% # filter out age group with NAs (solid tissue normal)
  ggplot() +
  geom_bar(aes(x = age_group, fill = age_group)) +
  theme_bw() +
  labs(x = "Age Group", y = "Number of Sample Donors", fill = "Age Group",
       title = "Age Group Distribution Among Tumor Tissue Sample Donors")
```



```r
# plot bar plot among all tissue sample donors in terms of tissue source site
sample_metadata %>%
  # count how many sample subtypes per tissue source site
  dplyr::count(paper_Tissue.source.site, sample_subtype, name = "count") %>%
  ggplot() +
  geom_bar(aes(x = paper_Tissue.source.site, y = count, fill = sample_subtype), stat = "identity") +
  theme_bw() +
  labs(x = "Tissue Source Site", y = "Number of Sample Donors", fill = "Sample Subtype",
       title = "Distribution of Tissue Source Site Among All Tissue Sample Donors") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1))
```



```r
# plot box plot among tumor tissue sample donors in terms of age, survival years,
# and age at diagnosis
sample_metadata %>%
  select(sample_subtype, all_of(num_demographic_vars)) %>%
  mutate(survival_years = paper_Survival..months. /12) %>%
  pivot_longer(cols = c(age_at_index, survival_years, paper_Age..years.at.diagnosis.)) %>%
  filter(!is.na(value)) %>% # filter out data with NAs (solid tissue normal)
  ggplot() +
  geom_boxplot(aes(x = sample_subtype, y = value, fill = sample_subtype),
               outlier.shape = NA) + # remove outlier plots to avoid double-plotting
  geom_jitter(aes(x = sample_subtype, y = value)) +
  theme_bw() +
  labs(x = "Sample Subtype", y = "Number of Years", fill = "Sample Subtype",
       title = "Distribution of Age and Survival in Years for Tumor Tissue Sample Donors") +
  facet_wrap(~name, scales = "free")
```



Based on the above plots, it is apparent that our dataset is biased towards having a majority of sample donors who are white, male, and age greater than 50 years. We Additionally observe that Henry Ford Hospital contributed the largest number of samples and that donors with the PN sample subtype are typically younger in terms of age at index and age of diagnosis; these individuals also appear to have a longer survival time in years, which is expected given their young age and earlier diagnosis.

PCA for Identification of Prognostic Pathogenic Genes in GBM

As stated in the project overview, we hypothesize that PCA can be used to distinguish the prognostic pathogenic genes from our 24 candidate immune genes. Here, we will perform the PCA on our normalized and filtered RNA-seq read count matrix and attempt to interpret the PCA weights in hopes that it will inform us of the prognostic pathogenic genes of interest that were identified by Liang et al.

Determine number of PCs needed to explain at least 90% variance

First, we will define a function that computes the number of PCs needed to account for at least 90% variance, which may inform us of the number of PCs to be used in subsequent analyses. We will then compute the PCA object, apply our function to determine the number of PCs needed to account for 90% variance, and inspect the percent variance accounted for by each PC.

```r
# function that indicates how many PCs needed for 90% variance
n_pc = function(pr){
  pr_var_exp = pr$sdev^2/sum(pr$sdev^2)
  var_exp_tot = 0
  pc_count = 0

  for(i in 1:length(pr_var_exp)){
    if(var_exp_tot < 0.9){
      var_exp_tot = var_exp_tot + pr_var_exp[i]
      pc_count = pc_count + 1
      }
    }

  return(pc_count)
}

# create PCA object on transposed counts matrix after scaling and centering the data
count_pca = prcomp(t(GBMMatrix_norm_filter), scale. = TRUE, center = TRUE)

# compute number of PCs needed to account for 90% variance
n_pc(count_pca) # needs 12 PCs to account for 90% variance
```

```
## [1] 12
```

```r
# inspect how much variance each PC explains
count_pca$sdev^2/sum(count_pca$sdev^2)
```

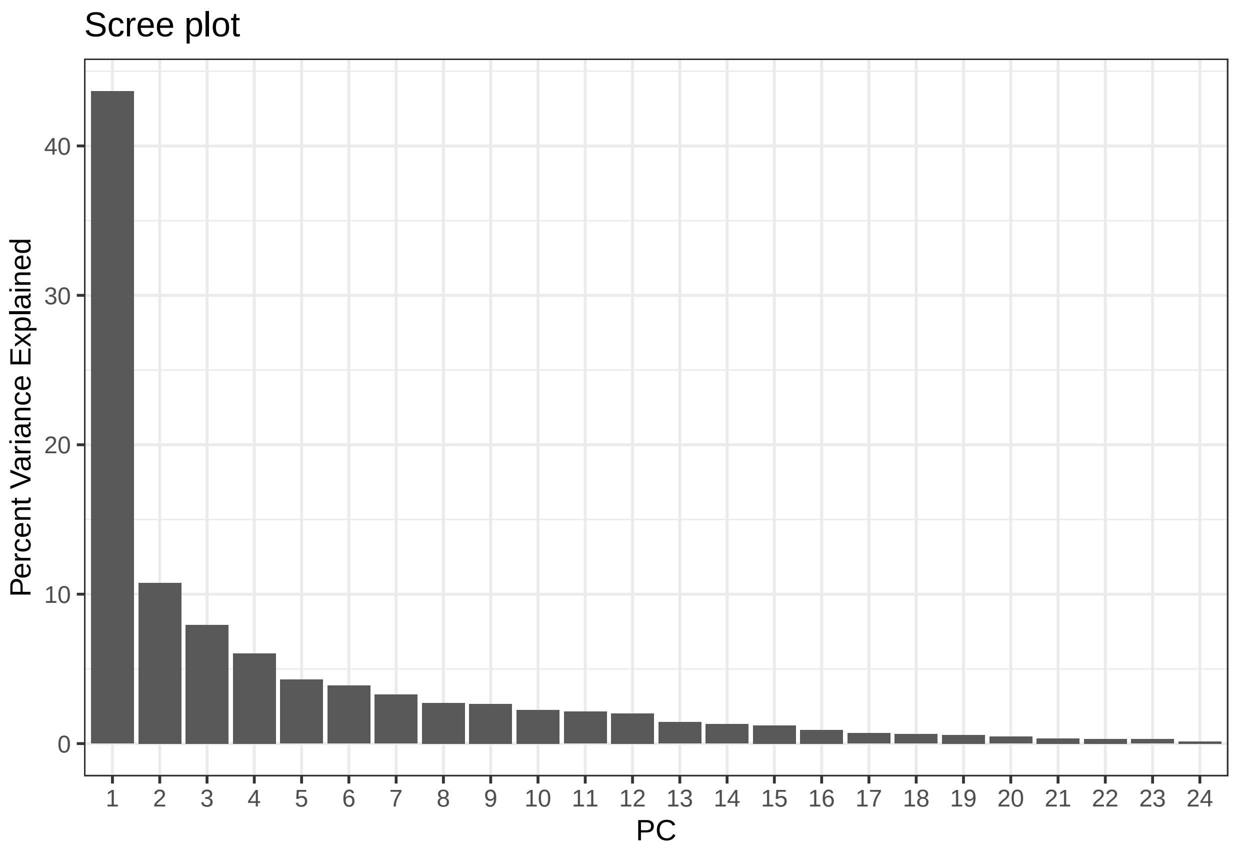
```
##  [1] 0.436595429 0.107658357 0.079536937 0.060410524 0.042822935 0.038803455
##  [7] 0.032756539 0.027324232 0.026520806 0.022574195 0.021318769 0.020245674
## [13] 0.014396656 0.012946540 0.012212993 0.009164719 0.006982328 0.006332617
## [19] 0.005873127 0.004573318 0.003491596 0.003028238 0.002901954 0.001528061
```

As shown above, we will need 12 PCs to account for at least 90% variance. Our first PC accounts for 43.7% of variance and our second PC accounts for 10.8% of variance.

Scree plot of PCA variance explained per PC

To visually inspect the amount of variance explained per PC, we will plot a scree plot as shown below.

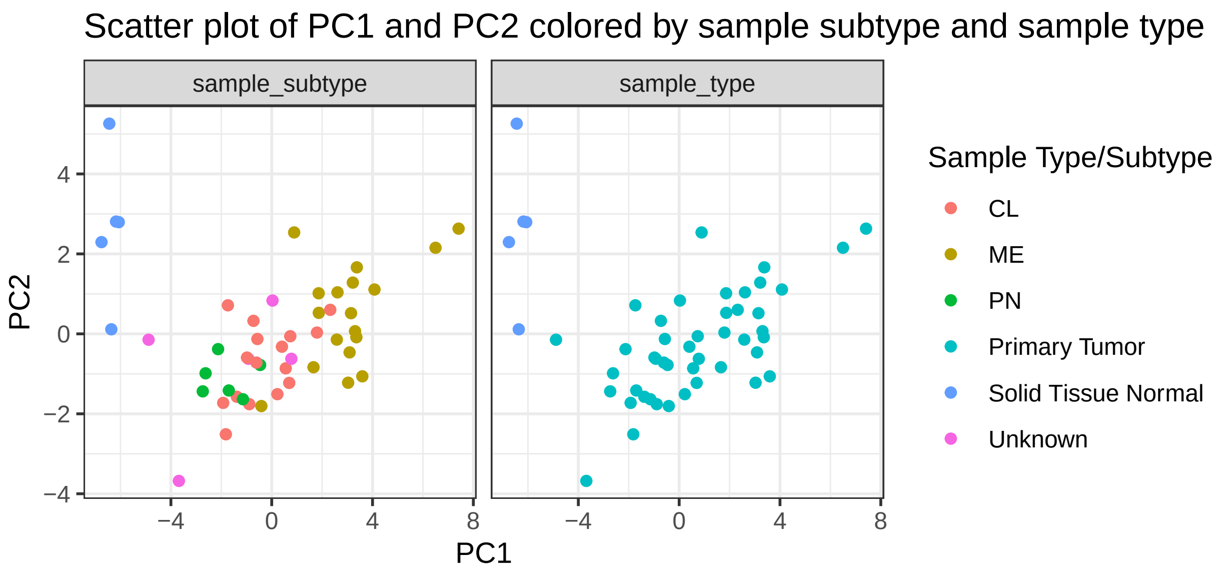
```r
# create a sequentially numbered dataframe of variances explained per PC and plot scree plot
data.frame(PC = seq(1, ncol(count_pca$x)),
           var_explained = (count_pca$sdev^2/sum(count_pca$sdev^2))*100) %>%
  ggplot(aes(x = factor(PC), y = var_explained)) + # make PC a factor so axis ticks match each bar
  geom_bar(stat = "identity") +
  theme_bw() +
  labs(x = "PC", y = "Percent Variance Explained", title = "Scree plot")
```



Scatter plots of PCA rotated data points

We hypothesized that the differential gene expression between normal tissue samples and tumor tissue samples (i.e., Solid Tissue Normal and Primary Tumor) will be the main driving factor influencing our PCs. Here, we will plot the first two PCs as a scatter plot and color them by either sample subtype or sample type to see if the respective data points group together nicely.

```
# merge PCA scores with sample metadata for plotting
sample_metadata_pca = data.frame(count_pca$x) %>%
  merge(sample_metadata, by = "row.names") %>%
  column_to_rownames("Row.names")

# plot PC1 and PC2 as scatter plots, colored and faceted by sample subtype and diagnosis
sample_metadata_pca %>%
  select(sample_type, sample_subtype, PC1, PC2) %>%
  pivot_longer(cols = c(sample_type, sample_subtype)) %>%
  ggplot() +
  geom_point(aes(x = PC1, y = PC2, color = value)) +
  theme_bw() +
  facet_wrap(~name) +
  theme(aspect.ratio = 1) +
  labs(color = "Sample Type/Subtype",
       title = "Scatter plot of PC1 and PC2 colored by sample subtype and sample type")
```

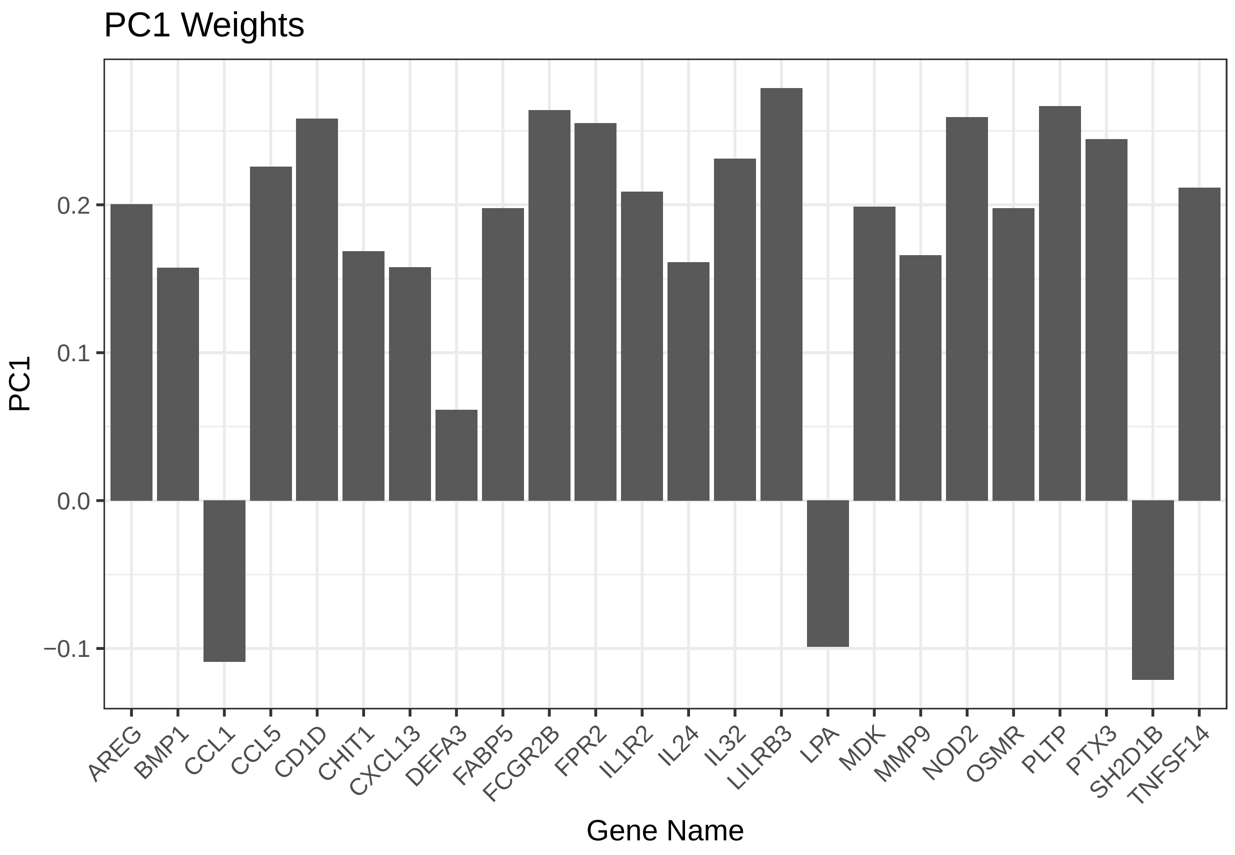


Based on the above plot, it seems like our hypothesis is mostly correct: when colored by sample type (i.e., normal tissue samples and tumor tissue samples), the two groups of data points group together nicely; when colored by sample subtype, the Solid Tissue Normal samples are clearly distinguished from the tumor tissue samples while the subtypes are largely mixed together and cannot be distinguished.

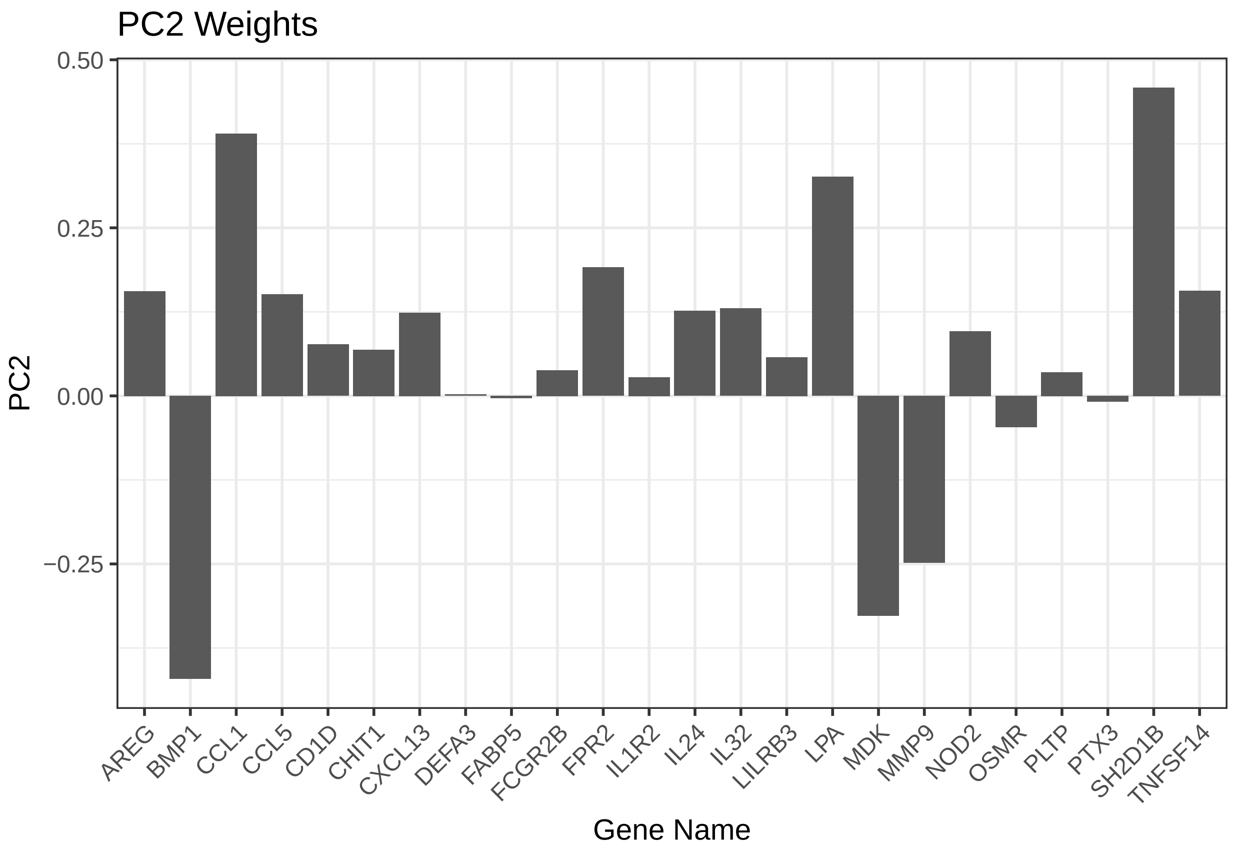
Bar plots of PCA weights to determine gene relevance of PC1 and PC2

Next, we will make bar plots of the PCA weights for PC1 and PC2 to determine the contribution of each gene to the weight of rotation that gives rise to our respective PCA scores. We expect that the prognostic pathogenic genes to have the most extreme values in either PC1 or PC2 as they should contribute the most to each component.

```r
# plot the weights of PC1 as bar plots, using gene names instead of gene codes
data.frame(count_pca$rotation) %>%
  mutate(gene_name = gene_metadata %>% select(gene_name) %>% as_vector()) %>%
  ggplot(aes(x = gene_name, y = PC1)) +
  geom_bar(stat = "identity") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)) +
  labs(x = "Gene Name", title = "PC1 Weights")
```



```r
# plot the weights of PC2 as bar plots, using gene names instead of gene codes
data.frame(count_pca$rotation) %>%
  mutate(gene_name = gene_metadata %>% select(gene_name) %>% as_vector()) %>%
  ggplot(aes(x = gene_name, y = PC2)) +
  geom_bar(stat = "identity") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)) +
  labs(x = "Gene Name", title = "PC2 Weights")
```



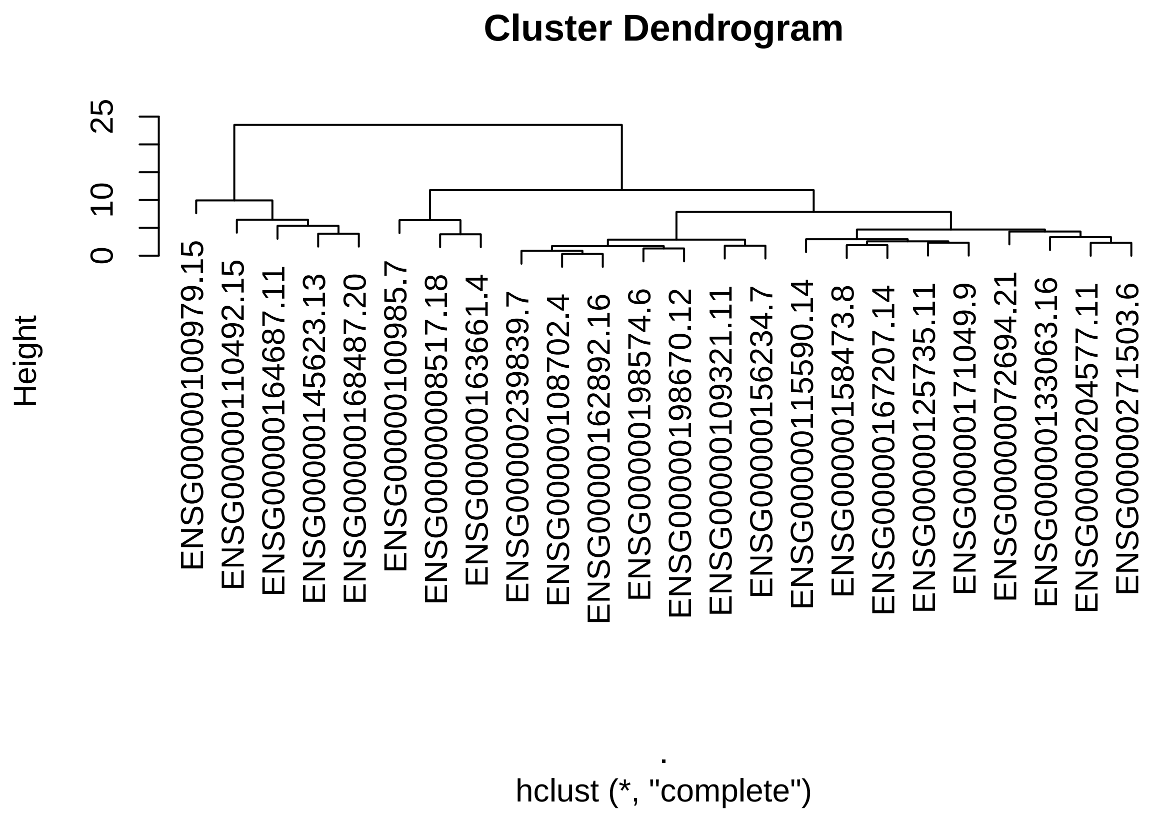
It is evident based on the above two plots that CCL1, LPA, and SH2D1B are consistently giving most extreme weights for PC1 and PC2. This is consistent with the identified prognostic pathogenic genes by Liang et al. and supports our hypothesis.

Hierarchical Clustering of Immune Gene Counts

Next, we will perform hierarchical clustering on all 24 immune genes identified by Liang et al. as well as all samples to see if there are any relationships between genes and between samples based on their gene expression profiles.

We will first cluster on genes across samples using euclidean distance and complete linkage to give the most compact clusters.

```r
set.seed(713) # set seed to 713 for reproducibility of clustering

# create hierarchical clustering object of genes using complete linkage method
# after scaling the counts matrix and calculating the euclidean distances
gene_hclust = GBMMatrix_norm_filter %>%
  scale() %>%
  dist(method = "euclidean") %>%
  hclust(method = "complete")

plot(gene_hclust) # plot the cluster dendrogram
```

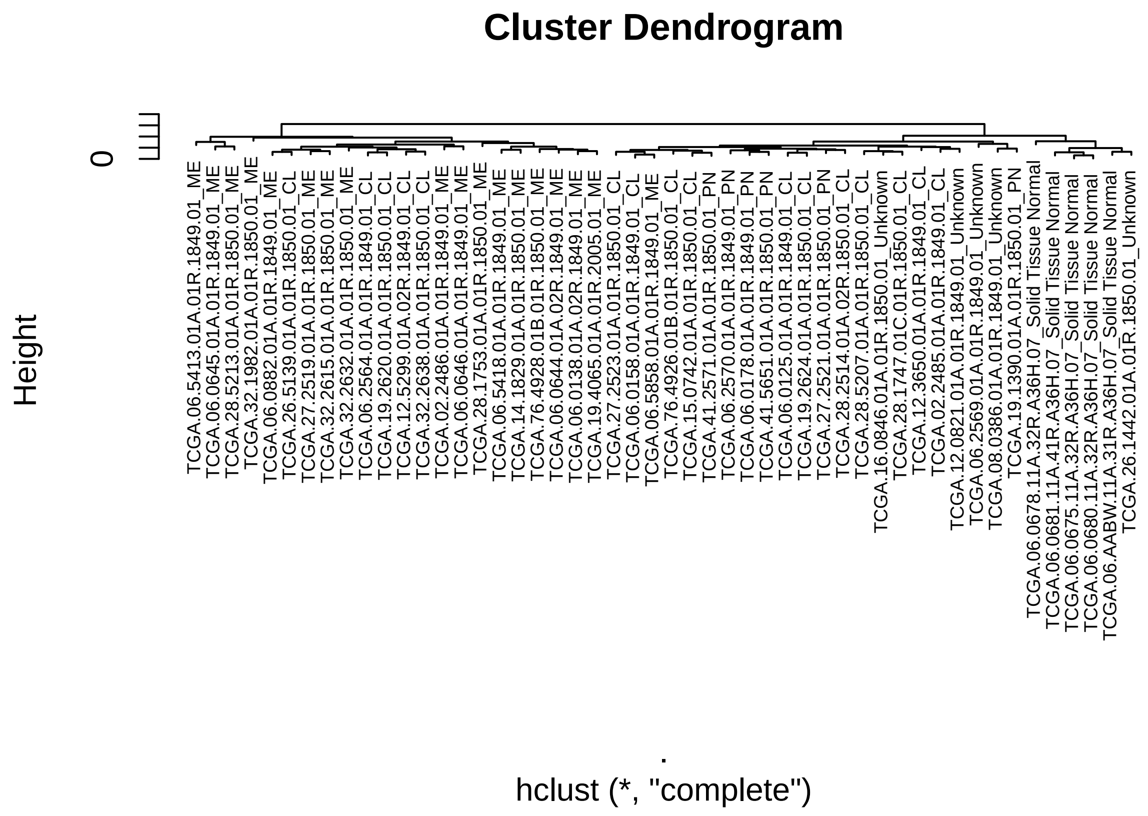


```r
# cut the dendrogram into 5 clusters after visual inspection of dendrogram and
# store as dataframe
gene_clusters = data.frame(clusters = cutree(gene_hclust, k = 5))
```

Here, we decided to cut the dendrogram into 5 clusters of genes as these clusters (visually) convey the most relevance between genes without leaving too many single genes in their own clusters.

We will repeat the above process for samples across genes. This time, we are attempting to cluster samples of the same subtype together since we expect that similar subtypes should have similar gene expression profiles. For plotting purposes, this will be done by concatenating together the sample ID with each sample's subtype.

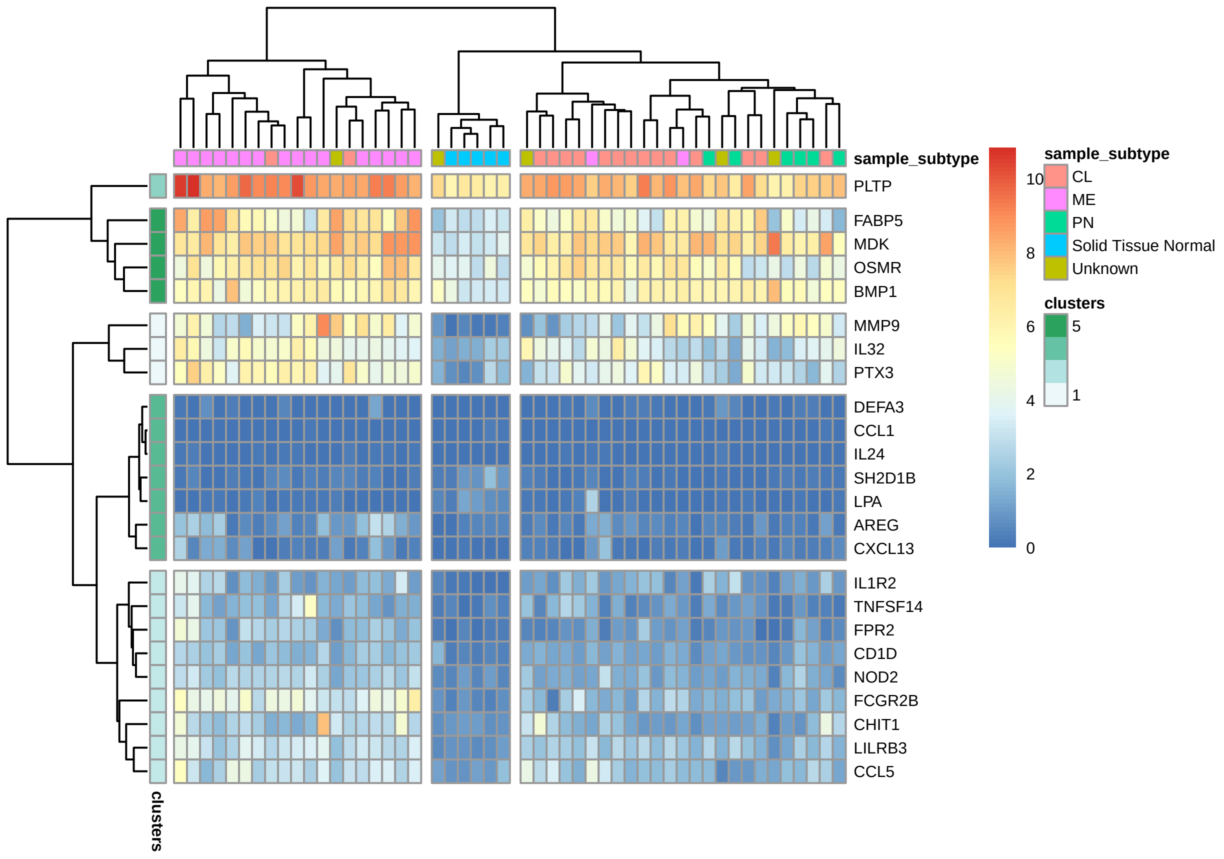
```r
set.seed(713) # set seed to 713 for reproducibility of clustering

# create a hierarchical clustering dendrogram of individuals and their subtypes
GBMMatrix_norm_filter %>%
  t() %>% # transpose the matrix so sample IDs are rows for merging with sample metadata
  merge(sample_metadata %>% select(sample_subtype), by = "row.names") %>%
  mutate(Row.names = str_c(Row.names, "_", sample_subtype)) %>% # concat ID and subtypes
  column_to_rownames("Row.names") %>% # make the concatenated string the new row names
  select(-sample_subtype) %>% # remove the sample subtype from the dataframe
  as.matrix() %>% # convert counts dataframe to matrix
  scale() %>% # scale the data
  dist(method = "euclidean") %>% # compute euclidean distances for clustering
  hclust(method = "complete") %>% # hierarchical clustering using complete linkage
  plot(cex = 0.6) # plot the dendrogram
```



Given the above dendrogram, we will cut the dendrogram into 3 clusters: (mostly) ME subtype, (mostly) Solid Tissue Normal subtype, and all remaining subtypes that are mixed together. We did not perform the cutting step here since we will create a heatmap of the clusters in the next step.

To better visualize the gene expression profiles of the 24 immune genes relevant to the prognosis of GBM, we will create a heatmap that incorporates hierarchical clustering based on the number of clusters we have determined in the above steps both for genes across samples (rows) and samples across genes (columns).

```
# create a heatmap of gene expression levels and perform hierarchical clustering
# on both individuals (columns) and genes (rows)
pheatmap(GBMMatrix_norm_filter,
         clustering_method = "complete", # use complete linkage method
         clustering_distance_rows = "euclidean", # use euclidean distance
         clustering_distance_cols = "euclidean", # use euclidean distance
         fontsize = 6,
         cluster_cols = TRUE, cluster_rows = TRUE, # hclust on both rows and columns
         cutree_cols = 3, cutree_rows = 5, # cut into number of clusters
         labels_row = gene_metadata %>% select(gene_name) %>% as_vector(), # show gene names
         annotation_col = sample_metadata %>% select(sample_subtype), # indicate subtype
         annotation_row = gene_clusters, # indicate gene clusters
         show_colnames = FALSE) # don't show sample names
```

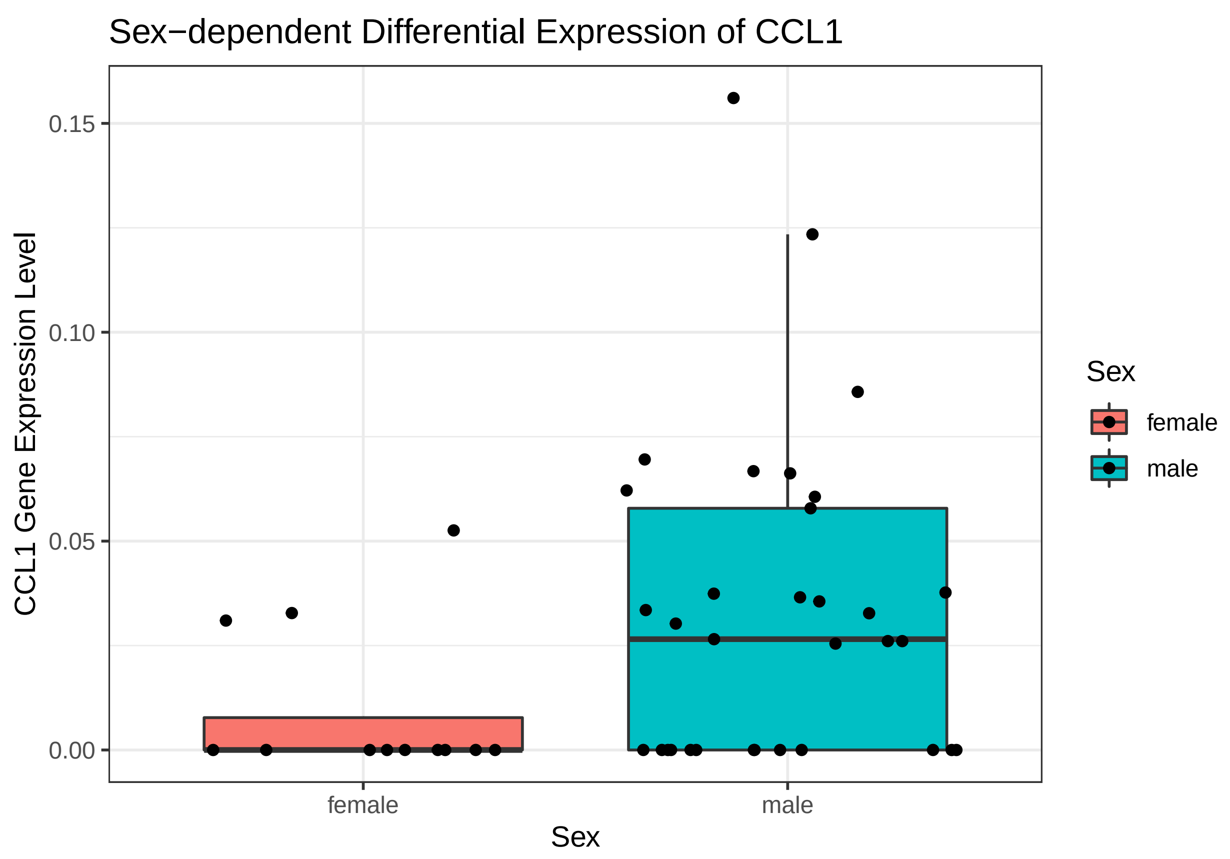


It is deeply satisfying to see our resulting heatmap–data is beautiful! Based on our previously determined number of clusters for both genes and samples, we see that samples of normal tissue have much lower expression levels of almost all genes except DEFA3, CCL1, IL24, AREG, and CXCL13 while these samples express higher levels of SH2D1B and LPA relative to GBM tissue samples. We also see that when clustering on samples (i.e., columns), the ME subtype and Solid Tissue Normal subtype are nicely clustered together in their respective clusters.

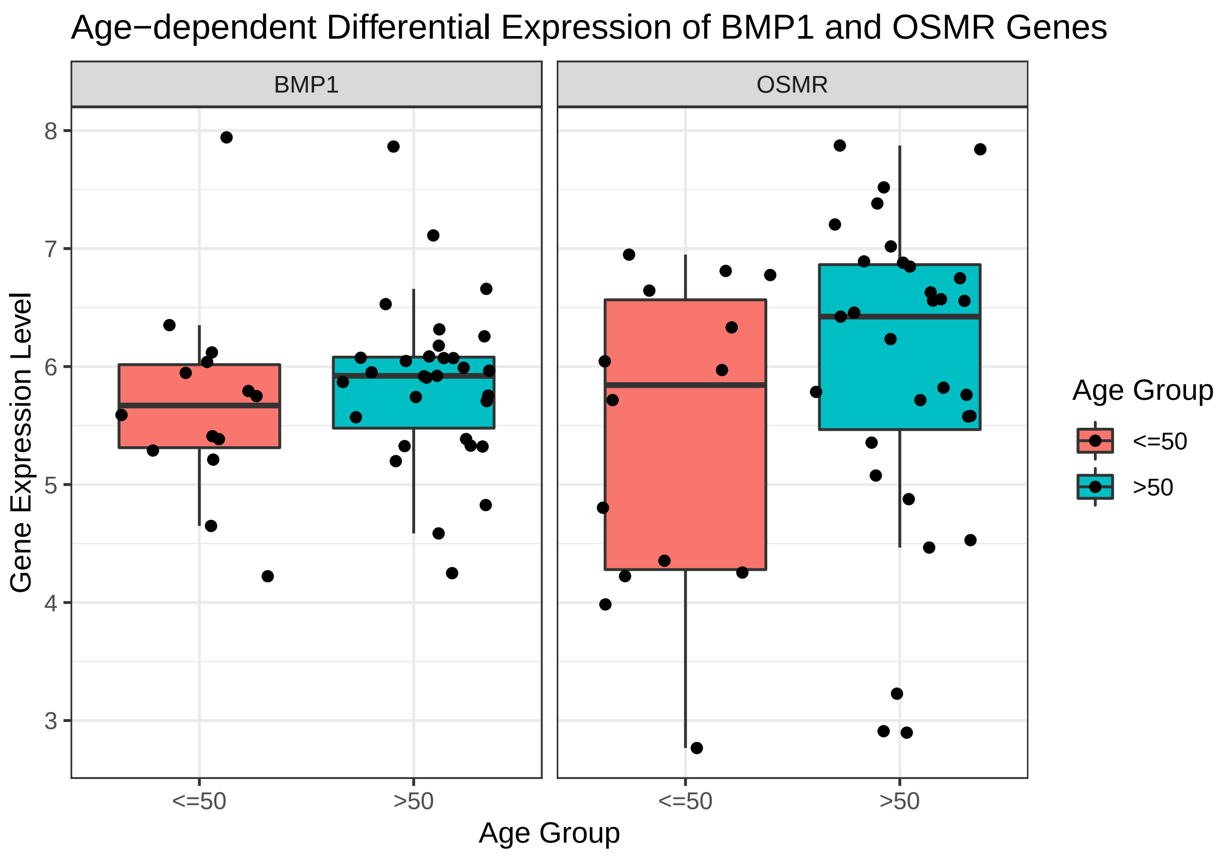
Verification of Observed Trends Identified by Liang et al. (2020)

Finally, we attempt to replicate the observed trends identified by Liang et al. Based on their results, CCL1 is expressed at higher levels in males relative to females while BMP1 and OSMR are expressed at higher levels in those with ages >50yo. We will accomplish this by creating box plots of the respective observations.

```
# make a copy of the counts matrix with gene name as the row names and merge in
# sample metadata for plotting
gene_sample = GBMMatrix_norm_filter %>%
  # merge gene metadata containing only the gene names by the row names
  merge(gene_metadata %>% select(gene_name), by = "row.names") %>%
  column_to_rownames("gene_name") %>% # make the gene name the row names
  # remove the original row names that was forced into a column when forcing matrix
  # into dataframe
  select(-Row.names) %>%
  as.matrix() %>% # convert dataframe to matrix to transpose the data
  t() %>% # transpose the data
  merge(sample_metadata, by = "row.names") %>% # merge sample metadata by row names
  # fix the row names because the original row names were made into a column from
  # forcing matrix into dataframe during merging
  column_to_rownames("Row.names")

# make box plot of CCL1 expression levels between genders
gene_sample %>%
  filter(!is.na(gender)) %>% # filter out rows with NAs as gender
  ggplot(aes(x = gender, y = CCL1, fill = gender)) +
  # remove the outliers in this layer to avoid double-plotting
  geom_boxplot(outlier.shape = NA) +
  geom_jitter() + # add jittered data points on top of the box plots
  labs(x = "Sex", y = "CCL1 Gene Expression Level", fill = "Sex",
       title = "Sex-dependent Differential Expression of CCL1") +
  theme_bw()
```



```r
# make box plot of BMP1 and OSMR gene expression levels between age groups faceted
# by the gene
gene_sample %>%
  # keep only the columns we will use, otherwise pivot_longer() doesn't like it
  select(BMP1, OSMR, age_group) %>%
  pivot_longer(cols = c(BMP1, OSMR)) %>% # convert to long data format for faceted plots
  filter(age_group != "Unknown") %>% # filter out any samples with unknown age
  ggplot(aes(x = age_group, y = value, fill = age_group)) +
  # remove the outliers in this layer to avoid double-plotting
  geom_boxplot(outlier.shape = NA) +
  geom_jitter() + # add jittered data points on top of the box plots
  facet_wrap(~name) + # facet by the gene names
  labs(x = "Age Group", y = "Gene Expression Level", fill = "Age Group",
       title = "Age-dependent Differential Expression of BMP1 and OSMR Genes") +
  theme_bw()
```



Based on the above plots, we have verified that males express higher levels of CCL1 than females and both BMP1 and OSMR are expressed at higher levels in those with ages greater than 50yo, supporting the findings of Liang et al.