lab 7

Abdullah Taman

2024-04-25

- Data Wrangling ///////// Now let's read the dataset.

```
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
df <- read.csv("E:/champions work/datasets/BrainCancerNA.csv", header = TRUE)
```

After that we check the number of columns and rows.

```
nrow(df)
```

```
## [1] 130
```

```
ncol(df)
```

```
## [1] 54547
```

Same number of rows mentioned in lab, but not same number of columns, the number in lab is bigger. Let's do some modifications on the table.

```
df <- subset(df, select = -1)
```

```r
sample_ids <- 834:963
rownames(df) <- sample_ids
```

```r
expression.data <- df[, -which(names(df) == "type")]
```

Now we do the required imputatio, replace the NA's with the mean of its column.

```r
df_filled <- apply(expression.data, 2, function(x) {
  mean_val <- mean(x, na.rm = TRUE)
  x[is.na(x)] <- mean_val
  return(x)
})

df_filled <- as.data.frame(df_filled)

anyNA(df_filled)
```

```
## [1] FALSE
```

PCA with Single Value Decomposition is performed like that: we send the samples as columns and attributes as rows, so we we'll transpose the data in passing it to prcomp.

```r
pca_result <- prcomp( (df_filled), scale. = TRUE, rank. = 130)
transformation_matrix <- pca_result$x
```

Let's put the data in this new dimensionse resulting from PCA in this df as equired.

```r
pcs <- as.data.frame(transformation_matrix)
head(pcs)
```

```
##            PC1      PC2       PC3      PC4        PC5        PC6        PC7
## 834 -150.80739 17.36365 -73.88854 88.84069  -2.044181  80.860605 -49.804766
## 835  -77.96779 32.78616  80.00930 21.92558 -49.720095 -39.956394   5.920903
## 836  -65.79805 62.17283  22.21468 -8.42992 -19.864732 -32.901610  15.864421
## 837   96.54182 36.32262  32.18450 47.85215  -8.399707  24.538073  55.753085
## 838  -60.56244 51.96560  15.79173 51.70237  -7.309137   9.900024 -66.540747
## 839 -114.78732 28.79353 -29.14451 84.39713 -14.168681  36.455833 -27.660465
##            PC8        PC9        PC10        PC11       PC12       PC13
## 834 -53.909569  26.837253 -34.1335940  23.1019235  -4.828557  14.390373
## 835 -14.833076 -50.722622  26.9906044  24.5128608 -70.226205  12.696615
## 836  26.420319 -16.595095  -0.4256897 -40.9357967  39.787947  -6.873867
## 837 -28.018431 -19.846713  -5.5498149 -11.4186750 -38.254896 -10.593346
## 838  -7.672518 -12.434204 -44.0663373   0.6496405  16.085850 -15.030428
## 839 -34.375237  -2.857665 -31.4261601  -5.2235359  -8.197230  -1.809922
##           PC14       PC15       PC16        PC17       PC18       PC19
## 834 -14.106941 18.061973 -12.586609   1.7860382  -2.833805  -8.394914
## 835 -31.929356 10.985103   2.841245  -0.3439523  -2.399617 -14.005583
## 836  -7.173337 30.300016  13.963899  13.3893366  22.019105  18.467617
## 837  19.228036  4.850585 -40.204376 -17.3978978  -2.573198  25.531447
## 838   4.180227 20.716044  19.251569  -9.0580751 -10.593926  -2.132388
## 839 -29.926023  1.980824   7.542858   9.3250761 -11.916341   1.134308
```

```
##           PC20       PC21       PC22       PC23       PC24       PC25
## 834  22.0253592 -13.686795  12.191475  -8.845086   2.766531  -5.125628
## 835  -1.5115053  24.506609 -19.076837 -21.612079 -12.885895  17.212866
## 836 -21.2537527   9.699137   2.022580   2.013935  16.170380  -4.951233
## 837 -15.9543958 -29.725887   1.936472 -15.350087 -13.977524  -5.442972
## 838  -0.2281717 -13.058265 -27.325320  17.808956  22.112294 -19.350593
## 839 -10.4433001  -6.177557 -10.517318 -17.505246  10.341867   6.781381
##           PC26       PC27       PC28       PC29       PC30        PC31
## 834   6.006903 -22.658809   0.3533051 -18.213754  10.060466 -32.0992655
## 835 -12.889263   4.437536 -27.9614161 -18.181101  10.479514  11.6271424
## 836  -3.932372  -7.538936   9.6369235  -5.952092  -4.586767   0.7873402
## 837  -8.527385 -11.463136  20.5071070   6.636359   7.935941  20.8456454
## 838  19.890586  -1.677132   8.5530099 -22.531116  10.786835   3.2473034
## 839  -2.080116 -44.947619 -10.6018692  -8.389976 -12.053644  18.3352391
##           PC32      PC33       PC34       PC35      PC36       PC37       PC38
## 834  12.9843327 -6.329575   8.460334 -23.756503 15.741993 -12.643371  -2.338823
## 835 -23.7369902  7.657500  18.226033  32.325071 -4.210609   2.106550  36.136387
## 836   1.1157174 -3.225770  26.664026   2.494350 14.685352  -8.857070  -4.649083
## 837 -13.0999150 25.238651 -39.902215   5.701633  2.380709 -34.196069 -35.585867
## 838  -0.6947219 -5.600622 -29.965371  11.792482 15.651159  -0.457179  17.748459
## 839  18.5880945 -9.235975   9.617514  -7.413278 -7.179530  11.361187 -17.086627
##           PC39       PC40        PC41       PC42        PC43        PC44
## 834 -28.806223  11.221610  18.7796512  -9.983878   0.1583790   0.3840493
## 835 -24.075483 -12.734763 -32.0259937   8.908786 -47.4145498  -9.6203813
## 836   4.206550   8.576800  10.9721574   2.196887  -1.6428383  13.8531758
## 837  15.561149   6.700997  15.3442340 -26.202945 -13.0239076 -40.7492799
## 838   6.488239  13.797828  -6.5690058  -8.401893   7.5708882  -7.4410692
## 839  12.161678   1.500154   0.8050972  -6.809174   0.5178395  36.9995574
##           PC45       PC46      PC47        PC48       PC49       PC50
## 834   9.600162 -13.350265  4.989022   5.1783678   1.189853   1.377132
## 835  -9.422971  -5.506392 16.152663  12.1954024   2.787390 -23.367169
## 836  -2.002719  -7.875087  4.668298   1.4385478 -10.410520 -10.659127
## 837  43.375713   2.019791 11.291017  -0.1297186 -47.508602 -13.628434
## 838 -12.698985  -9.907316  6.954570  24.6719475   3.966980 -16.233894
## 839 -14.178977   5.224827  2.102027 -12.3270072   8.495911   5.306229
##           PC51       PC52       PC53       PC54       PC55       PC56
## 834  23.664189 -20.643238  -6.566169   1.874348   8.198715   9.348107
## 835  17.634337 -13.875980  16.987746 -18.499342  11.886945  12.668591
## 836  18.150757  -5.769517  11.475488  -1.368248  16.296031   3.604260
## 837 -13.854944  19.103357  35.338910   1.492650   7.684033 -19.736188
## 838   7.855293  -9.788105  10.719762  -9.160597 -49.907402 -12.681877
## 839  -2.238381  23.278596 -22.332310   6.769739   8.849882   7.946028
##           PC57       PC58       PC59      PC60      PC61      PC62      PC63
## 834  -1.2296656  -6.629987 -5.2715976 -5.482347 10.516692 13.387677 17.326483
## 835 -25.2871710   2.616697 30.9157710 -8.886653 34.752317 11.381297 -8.596270
## 836   0.3147172   1.490328  0.9943687 -6.428416 -3.406335 -9.917829 15.582710
## 837   7.0018424 -10.108469  0.9525161 -9.126861  5.939685 -7.176599 22.904716
## 838  11.0068232  12.338807  3.5533798  4.962681 -3.194021 -8.825645 -6.123349
## 839  24.6926632  12.922862 12.9524567 -4.205242 11.476068 -1.937324 16.732314
##           PC64       PC65        PC66       PC67       PC68       PC69
## 834  -7.741250  -6.404905 -2.43713988 -17.346772  -7.898694  13.224894
## 835  17.759479  12.391868 13.06760421  -8.212385  -1.043454 -22.420464
## 836  17.607565 -12.545523 13.22717251   2.368493  -1.417448   9.454997
## 837  -5.898056 -26.699270 15.34702747 -18.503588  28.489050 -26.869430
```

```
## 838 -12.239236  -8.009729  4.37989756 -22.698516 -13.170940   5.989141
## 839  -5.725376   9.553248  0.02263478  10.898100   8.067364 -42.571155
##           PC70       PC71       PC72       PC73         PC74       PC75
## 834   0.7545381  1.1471414  10.329388  -8.474933  -4.13489484  -2.722576
## 835   9.4890648  0.9986013 -25.909827   6.405501  17.34543313  -8.559445
## 836  -5.9471816  1.2366709  -2.108201  -4.163862   2.14332371   3.791568
## 837 -28.3706849 -1.3224086  -6.931029   7.563773   4.25222446  -6.386953
## 838 -17.4007232  1.2855737  21.881314  35.646446 -21.24981335 -11.921381
## 839  23.3296871 -4.1060696  -2.958172 -16.999744   0.05521442   3.417734
##           PC76         PC77       PC78       PC79       PC80        PC81
## 834 -10.382366  -0.02667448  -2.488715 15.4511717   8.146349   0.4316358
## 835 -20.309225   2.98491439 -12.074483  9.4305574   9.569446  -1.8350840
## 836  -3.596480   6.63347311  -1.379327  4.9757900  -2.165669  -6.2899221
## 837   2.090263 -12.59064377  -1.280248 -8.0022546  31.354111  -9.7367891
## 838   3.416929  30.57552923 -38.285179  0.1144863   5.775576   1.9578030
## 839  18.630337 -14.38416586 -24.445476  1.0604311 -15.016126 -12.0138550
##           PC82       PC83        PC84      PC85       PC86       PC87      PC88
## 834   2.609790  27.187645  -2.8147562 -5.897777 -31.038517 -14.163226 13.286563
## 835 -24.738869  -4.028263   8.2016004  9.322275 -16.754475 -13.827895  8.853022
## 836   1.732217  18.594769  -0.4271006  5.596116  19.596226   9.127532 14.592666
## 837   3.995361   0.413001  -0.9529035 -4.510601   6.379092   2.699511 -9.604090
## 838   3.603413   5.868322 -14.5571967  6.832155  12.236223 -16.067462 -2.887035
## 839  13.153819 -25.741292   2.2835313  7.716646  15.130235  18.886038 -8.299826
##           PC89       PC90      PC91        PC92       PC93        PC94
## 834   8.7270710  20.123283 17.915865 -30.2641129 -48.297673 -29.5585589
## 835  -3.0231015  19.751537 -5.131536   5.5757719   2.692149  -0.9021003
## 836   0.5136553   7.832297 -5.436534  20.3144495  11.976050  -8.3987930
## 837  -5.3638866  14.379362  5.094415   0.2248037   6.737838  -5.1463240
## 838 -19.8306940 -11.384527 -3.866830 -16.5870270   4.804414  -4.3693873
## 839  15.7058765  18.004900 -8.812580 -12.9397262  26.920750 -17.1759570
##           PC95        PC96       PC97       PC98       PC99     PC100
## 834 -19.3962048 -14.6640452  15.355554   7.907848  17.097715 21.822556
## 835  19.7862378  -4.2839744   1.888107  -6.821259   2.662317 -8.550970
## 836   0.5318593   3.8799885   8.214761  24.851695  -7.301481 -5.298795
## 837  -0.8516015  -0.8233787   4.865696  -7.568692  11.503548 12.459571
## 838  38.9812328  -2.0843143  12.010471   1.205710 -24.882939 -6.323834
## 839   8.2385659 -16.4229980 -14.144237 -17.087499  -2.192819 17.287854
##          PC101      PC102       PC103       PC104       PC105      PC106
## 834 -19.428434 -1.1004040  -3.9679513 -14.2798840  -7.4603145  1.5024683
## 835  -4.219983  6.3125070  -0.6043201  -2.8258321 -12.5570280  4.2098750
## 836   3.090961  1.5528441  -2.3103468  -2.9447272  -0.7557518  2.4858044
## 837   6.274273 -0.5254985  -7.4914554   0.5388545  -6.3374626 -0.7431780
## 838 -10.228906 -3.9531603  -3.3425016   5.1459092  17.9731483 -0.8525033
## 839 -12.475388 11.9056038 -23.5509801  -4.3136818  15.8265917 14.9137593
##          PC107       PC108      PC109     PC110      PC111        PC112
## 834  10.918284   3.7659158  3.1311408  2.548163 -3.5667490  -9.22304048
## 835  -0.823574  -7.9368117 12.4760575 -5.164992  2.0023183  -4.99630149
## 836   3.120926   3.0901362  7.1561299  1.322878 -0.4027744 -19.19549086
## 837  -3.026156   0.4504998  0.9725802 -2.530974 -0.2431999   1.38488429
## 838 -15.736756 -11.9930092 -6.5259424 -8.827180 -6.7492961   0.10867662
## 839  -4.458381  -6.3219111  6.7808556 -3.123858 -6.8731663   0.09457532
##         PC113      PC114     PC115       PC116      PC117      PC118      PC119
## 834 -3.061659  1.3352888  4.100523  0.04121697  5.8176786  -3.271895 -0.1782065
## 835 -8.792925 -5.8035459  2.373918  2.73461874  0.7468876  -1.821977  9.6342615
```

```
## 836 19.576277 21.9366034  2.875570 -9.69993016 49.1171025   7.441667 -6.6246223
## 837 -4.858507  2.5288074  4.593017 -0.16697151 -0.2589990   1.484417  3.4309284
## 838  2.716184  0.7974984 11.164738  3.47065741  2.8959307  -4.980408  1.8605294
## 839  2.328385  1.8504500 14.528673 -5.14138422 11.8359701 -20.012014 -8.8288315
##          PC120       PC121     PC122      PC123      PC124        PC125
## 834 -1.9539486  -0.7314507  2.463932  1.4301785 -0.9827257  1.336363838
## 835 -1.0285225   2.1229269 -3.666457 -0.8820472 -0.8848751  0.404173984
## 836 11.4633560  29.3483693  5.746483  6.6470327 26.1694434 16.522658712
## 837 -3.2271806  -1.8871468 -2.563296  0.4945899  0.6391271  0.006848787
## 838  8.7509267  -5.8833826 -8.709924 -1.6350019 -6.7274067  5.344573652
## 839  0.4605642 -10.0995098 -5.747430  0.7738560 -5.7082760 -2.048995415
##           PC126      PC127      PC128        PC129         PC130
## 834 -1.56445779   1.438197  -3.256786   2.77422382  4.207843e-13
## 835  3.91139734  -1.101504   3.651758   1.50017540 -8.271424e-14
## 836 22.77153879 -27.407709 -24.902828 -13.00637484  1.829670e-13
## 837  2.30807940  -1.030491   2.335024   0.80017840  4.872955e-14
## 838  0.06873257  -1.399920   2.030029  -0.05377979  9.757845e-14
## 839  8.74740562   1.026017   4.959046   3.58499431  1.681613e-13
```

Now we isolate the type in a new dataframe because thw transofrmation we'll do on it tha will be needed.

```
type_column <- df[, "type", drop = FALSE]
```

```
merged_df <- pcs
merged_df$type <- type_column$type
```

Then we create this new dataframe new_df to put the first 3 pcs in it along with the type column to be able to plot the points.

```
new_df <- merged_df[, c(  "PC1", "PC2", "PC3", "type")]
```

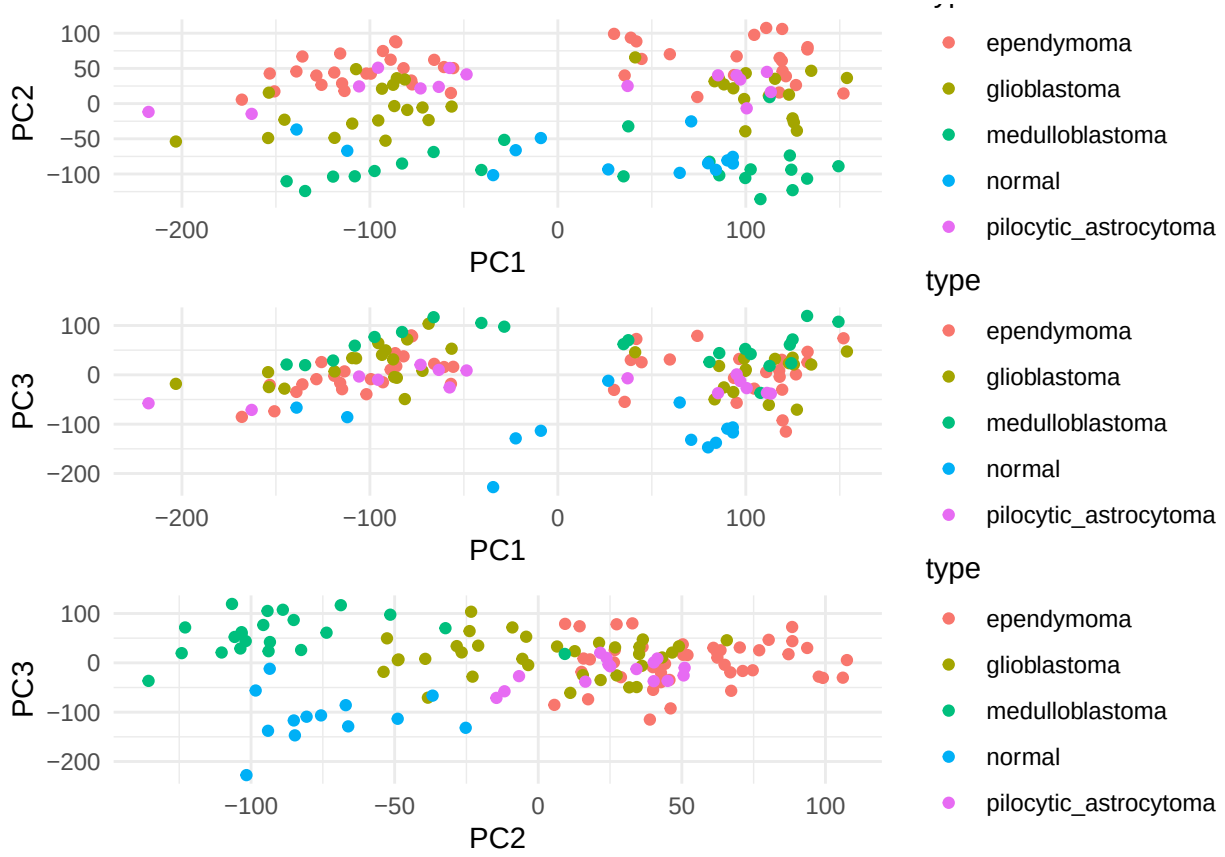
```
library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
# Plot PC1 vs PC2
plot_pc1_pc2 <- ggplot(new_df, aes(x = PC1, y = PC2, color = type)) +
  geom_point() +
  labs(x = "PC1", y = "PC2") +
  theme_minimal()

# Plot PC1 vs PC3
plot_pc1_pc3 <- ggplot(new_df, aes(x = PC1, y = PC3, color = type)) +
  geom_point() +
  labs(x = "PC1", y = "PC3") +
```

```
  theme_minimal()

# Plot PC2 vs PC3
plot_pc2_pc3 <- ggplot(new_df, aes(x = PC2, y = PC3, color = type)) +
  geom_point() +
  labs(x = "PC2", y = "PC3") +
  theme_minimal()

# Combine plots into a single figure
grid.arrange(plot_pc1_pc2, plot_pc1_pc3, plot_pc2_pc3, nrow = 3)
```



WOW! We could see in the PC1 vs PC2 plot that there is some clusters identefied, the normal people are down in the grid, then the more we go up, we see the samples of genes. We could see that pc2 is resluting in a real discrimination between deifferent samples, so we'll correct for it. Now we'll identify and replace outliers with NA

```
df_no_outliers <- df_filled

for (col in colnames(df_no_outliers)) {
  col_mean <- mean(df_no_outliers[, col], na.rm = TRUE)
  col_sd <- sd(df_no_outliers[, col], na.rm = TRUE)

  outliers <- which(df_no_outliers[, col] > col_mean + 3 * col_sd |
                    df_no_outliers[, col] < col_mean - 3 * col_sd)

  df_no_outliers[outliers, col] <- NA
```

```
}
```

Imputation Again!

```
for (col in colnames(df_no_outliers)) {
  col_mean <- mean(df_no_outliers[, col], na.rm = TRUE)
  df_no_outliers[, col][is.na(df_no_outliers[, col])] <- col_mean
}
```

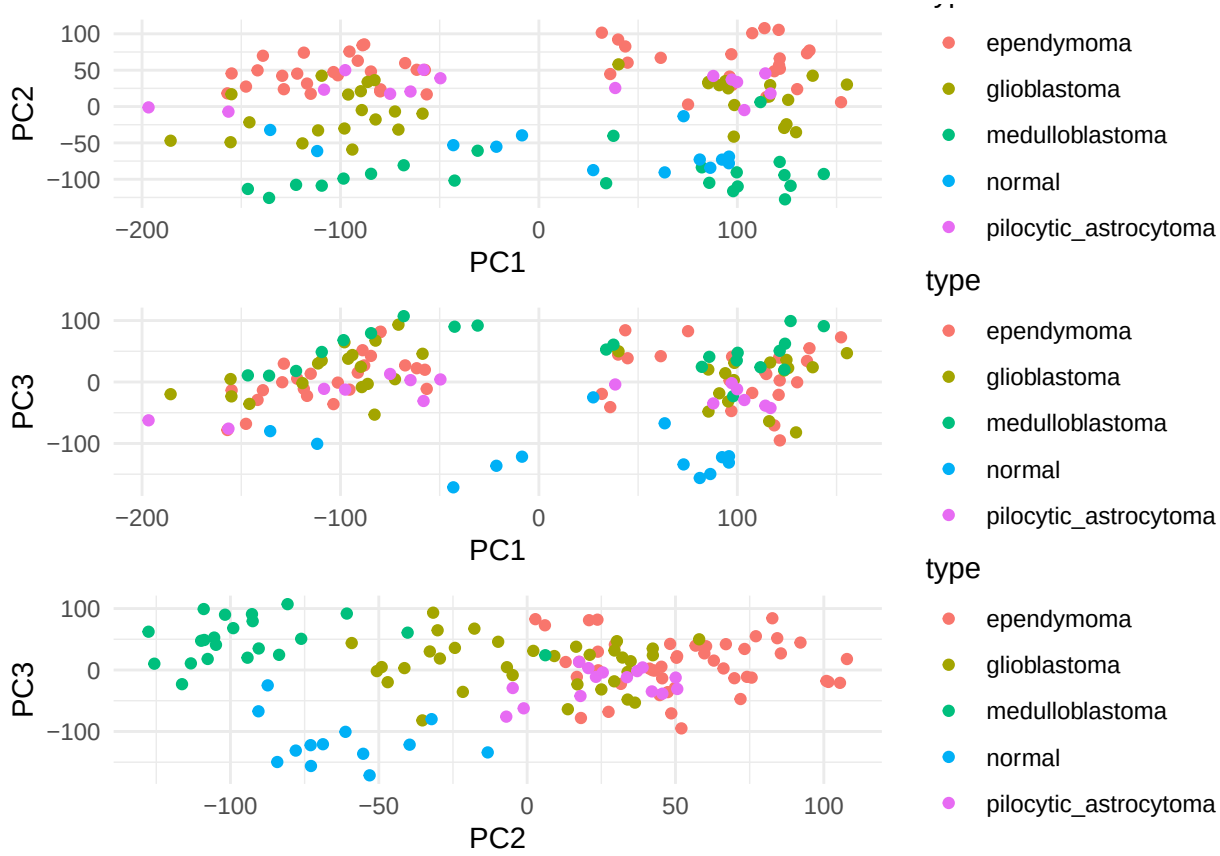
We'll do the normalized between arrays from limma library.

```
library(limma)

normalized_df <- normalizeBetweenArrays(df_no_outliers, method = "quantile")
```

Now let's perfrom the PCA once more.

```
pca_result <- prcomp( (normalized_df), scale. = TRUE)

transformation_matrix <- pca_result$x
```

```
pcs <- as.data.frame(transformation_matrix)

colnames(pcs) <- paste0("PC", 1:ncol(pcs))

 merged_df <- pcs
merged_df$type <-  type_column$type

plot_pc1_pc2 <- ggplot(merged_df, aes(x = PC1, y = PC2, color = type)) +
  geom_point() +
  labs(x = "PC1", y = "PC2") +
  theme_minimal()

plot_pc1_pc3 <- ggplot(merged_df, aes(x = PC1, y = PC3, color = type)) +
  geom_point() +
  labs(x = "PC1", y = "PC3") +
  theme_minimal()

plot_pc2_pc3 <- ggplot(merged_df, aes(x = PC2, y = PC3, color = type)) +
  geom_point() +
  labs(x = "PC2", y = "PC3") +
  theme_minimal()

grid.arrange(plot_pc1_pc2, plot_pc1_pc3, plot_pc2_pc3, nrow = 3)
```



Same result as above, so a decision is made, we'll correct for pc2. Now let's do the embedding part for the glm.

```
type_column$embedding <- ifelse(type_column$type == 'normal', 0, 1)
```

Now read the important genes

```
# Read the text file with column names
column_names <- read.table("E:\\champions work\\datasets\\top_5000.txt", header = FALSE)

column_names <- as.vector(column_names$V1)
```

```
missing_cols <- setdiff(column_names, colnames(normalized_df))
missing_cols
```

```
## [1] "X1553499_s_at" "X1553530_a_at" "X1553474_at"   "X1553449_at"
## [5] "X1553447_at"   "X1553569_at"   "X1553436_at"   "X1553424_at"
## [9] "X1553508_at"
```

There are columns that doesn't exist in the original dataframe, we shall leave it. We'll intersection between the given genes and genes we have.

```
existing_cols <- intersect(column_names, colnames(normalized_df))
new_df <- normalized_df[, existing_cols]
```

Let's complete the new_df that we'll use in our logistic regression.

```
new_df <- cbind(new_df, PC2 = pcs$PC2)
new_df <- cbind(new_df, class = type_column$embedding)
```

```
new_df <- data.frame(new_df)
```

Now we'll perform the logistic regression on the 5000 gene correcting for the pc2.

```
library(stats)



pvalue_df <- data.frame(Gene = character(), P_Value = numeric(), stringsAsFactors = FALSE)
for (i in 1:(ncol(new_df)-2)) {
  formula <- paste("class ~ ", names(new_df)[i], " + PC2", sep="")
  model <- glm(formula, data = new_df, family = binomial)
  p_value <- summary(model)$coefficients[2, 4]
  pvalue_df <- rbind(pvalue_df, data.frame(Gene = names(new_df)[i], P_Value = p_value))
}
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
significant_genes <- pvalue_df[pvalue_df$P_Value < 0.05, ]
```

Nw we draw the heatmap using gplot library.

```
library(gplots)
```

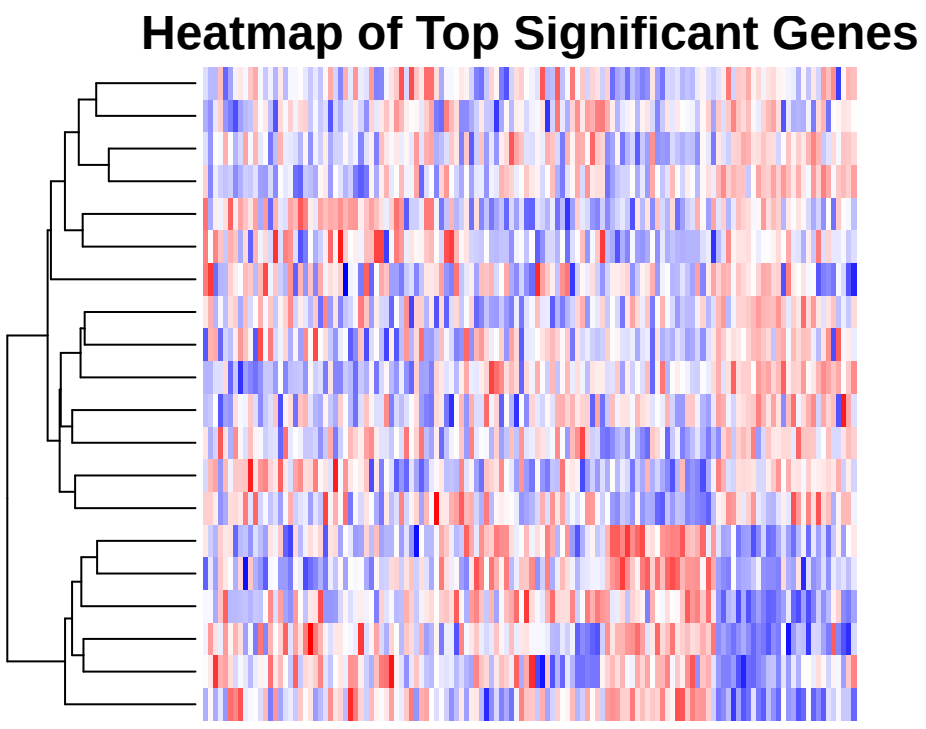
```
## Warning: package 'gplots' was built under R version 4.3.3
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
# Assuming `significant_genes` contains your gene data, and `new_df` is your expression data

# Select top significant genes
top_significant_genes <- head(significant_genes[order(significant_genes$P_Value), ], 20)$Gene
```

```r
# Extract expression data for top genes
top_genes_expression <- new_df[, top_significant_genes]

# Transpose the expression data
top_genes_expression <- t(top_genes_expression)

# Set row names as gene names
rownames(top_genes_expression) <- top_significant_genes

# Create the heatmap
heatmap(as.matrix(top_genes_expression),
        Colv = NA, scale = "column",
        labRow = FALSE, labCol = FALSE,
        col = colorRampPalette(c("blue", "white", "red"))(100),
        main = "Heatmap of Top Significant Genes",
        xlab = "Samples", ylab = "")  # No y-axis label

# Print gene names beside rows
for (i in 1:nrow(top_genes_expression)) {
  text(x = ncol(top_genes_expression) + 0.5, y = i,
       labels = rownames(top_genes_expression)[i], adj = c(1, 0.5), cex = 0.8)
}
```



Now we're trying to draw the volcano plot. We'll calculate the logcf, we'll get the average of each gene in the normal "cnotrol" and sick case, then from these numbers we calculate the logcf which =

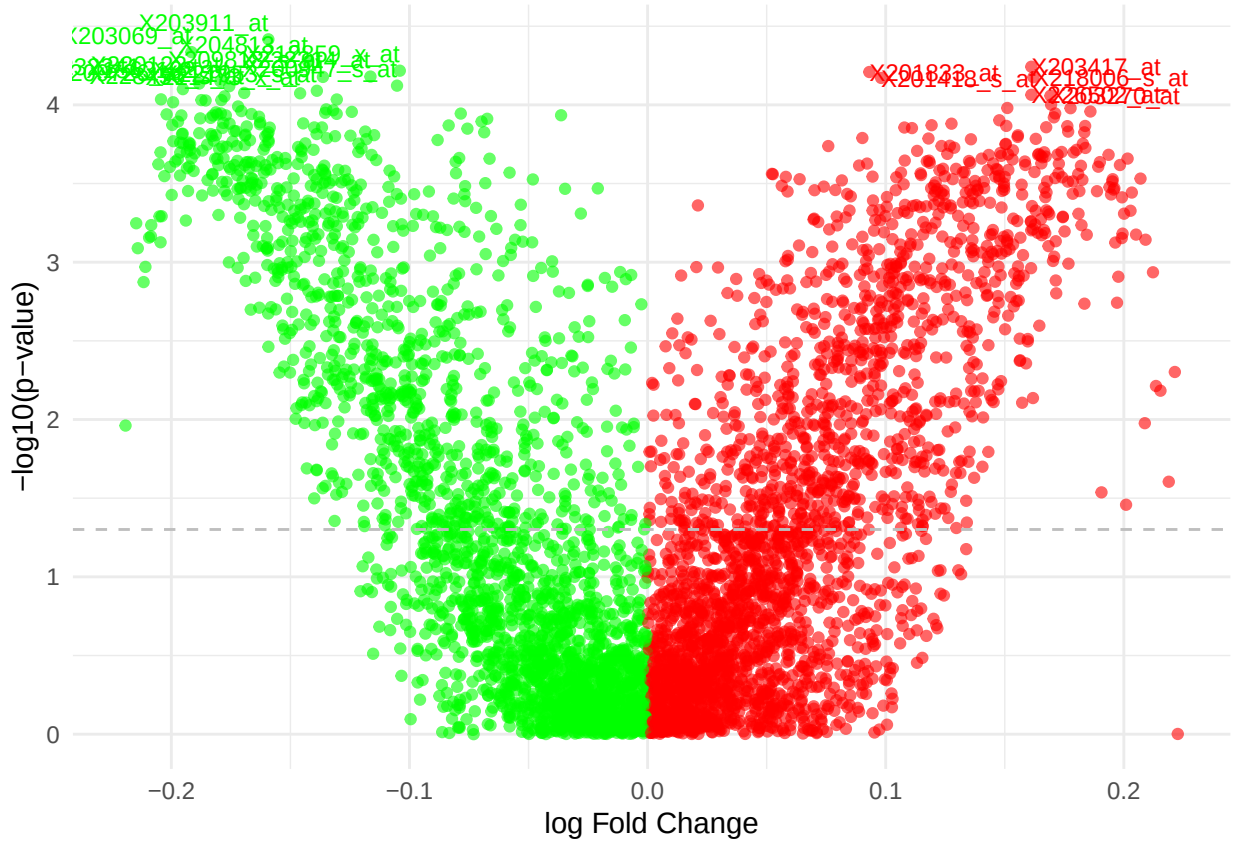
log(avg_in_illness)/log(avg_in_control). We followed the following prcocdure

```
gene_cols <- names(new_df)[1:(ncol(new_df) - 2)]

average_expression <- data.frame(Gene = character(), Cancerous = numeric(), Normal = numeric(), strings

for (gene in gene_cols) {
  cancerous_avg <- mean(new_df[new_df$class == 1, gene], na.rm = TRUE)
  normal_avg <- mean(new_df[new_df$class == 0, gene], na.rm = TRUE)
  average_expression <- rbind(average_expression, data.frame(Gene = gene, Cancerous = cancerous_avg, No
}

colnames(average_expression) <- c("Gene", "Cancerous", "Normal")
```

```
average_expression$logFC <- log2(average_expression$Cancerous / average_expression$Normal)
average_expression$p_Val <- pvalue_df$P_Value
```

Now let's idetify the downregulated and upregulated regions. Then we plot the volcano.

```
library(ggplot2)
top_significant_genes <- average_expression[order(average_expression$p_Val), "Gene"][1:20]

average_expression$Regulation <- ifelse(average_expression$logFC > 0, "Upregulated", "Downregulated")
average_expression$Regulation <- factor(average_expression$Regulation, levels = c("Upregulated", "Downr

ggplot(average_expression, aes(x = logFC, y = -log10(p_Val), color = Regulation)) +
  geom_point(alpha = 0.6) +
  scale_color_manual(values = c("Upregulated" = "red", "Downregulated" = "green")) +
  geom_text(data = subset(average_expression, Gene %in% top_significant_genes),
            aes(label = Gene), vjust = ifelse(average_expression$logFC[average_expression$Gene %in% top
            hjust = ifelse(average_expression$logFC[average_expression$Gene %in% top_significant_genes]
            size = 3) +
  theme_minimal() +
  labs(x = "log Fold Change", y = "-log10(p-value)") +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "gray") +
  theme(legend.position = "none")
```



We used the help of these two links. https://kasperdanielhansen.github.io/genbioconductor/html/limma.html https://biostatsquid.com/volcano-plots-r-tutorial/ Now lt's get the top 20.

```
top_significant_genes
```

```
##  [1] "X203911_at"   "X203069_at"   "X204813_at"   "X203417_at"   "X212859_x_at"
##  [6] "X201833_at"   "X228314_at"   "X209818_s_at" "X218006_s_at" "X201418_s_at"
## [11] "X220122_at"   "X232341_x_at" "X200947_s_at" "X1563182_at"  "X209726_at"
## [16] "X221953_s_at" "X228456_s_at" "X221473_x_at" "X226502_at"   "X203270_at"
```

We used david tools to do the conversion Here is the result:

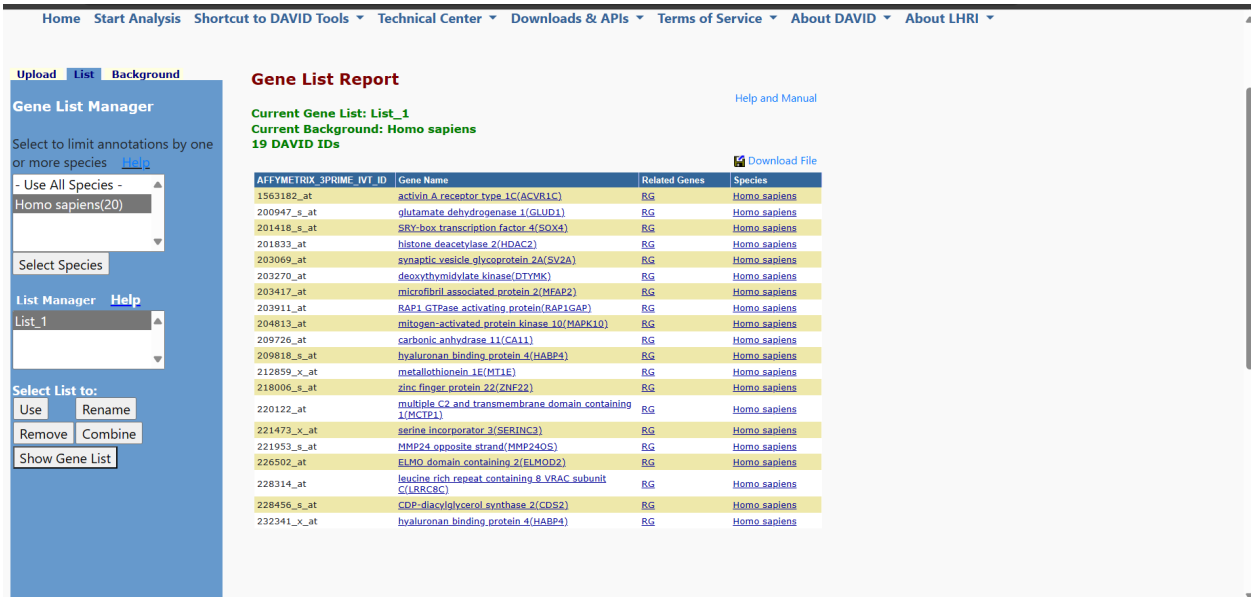
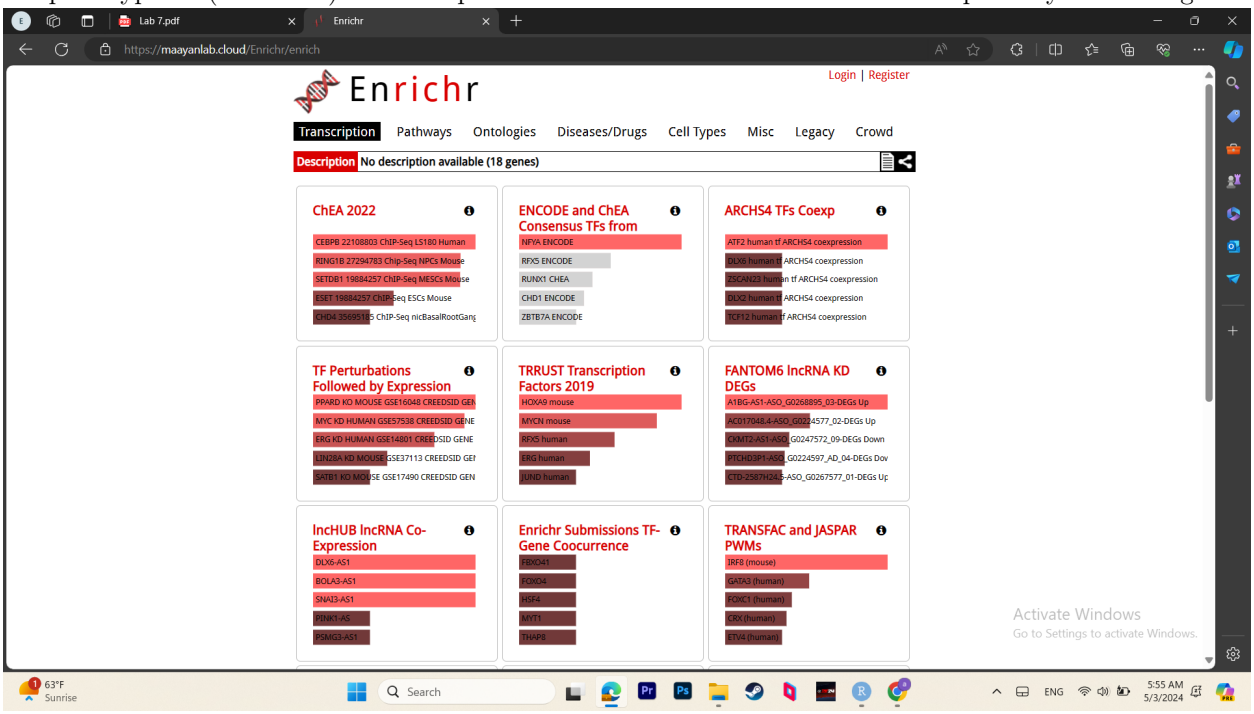


Figure 1: Caption for the image

```
AFFYMETRIX_3PRIME_IVT_ID        Name    Species
221953_s_at     MMP24 opposite strand(MMP24OS)  Homo sapiens
209726_at       carbonic anhydrase 11(CA11)     Homo sapiens
204813_at       mitogen-activated protein kinase 10(MAPK10)     Homo sapiens
201418_s_at     SRY-box transcription factor 4(SOX4)    Homo sapiens
226502_at       ELMO domain containing 2(ELMOD2)        Homo sapiens
228314_at       leucine rich repeat containing 8 VRAC subunit C(LRRC8C) Homo sapiens
221473_x_at     serine incorporator 3(SERINC3)  Homo sapiens
203069_at       synaptic vesicle glycoprotein 2A(SV2A)  Homo sapiens
212859_x_at     metallothionein 1E(MT1E)        Homo sapiens
203417_at       microfibril associated protein 2(MFAP2) Homo sapiens
209818_s_at     hyaluronan binding protein 4(HABP4)     Homo sapiens
232341_x_at     hyaluronan binding protein 4(HABP4)     Homo sapiens
203270_at       deoxythymidylate kinase(DTYMK)  Homo sapiens
218006_s_at     zinc finger protein 22(ZNF22)   Homo sapiens
203911_at       RAP1 GTPase activating protein(RAP1GAP) Homo sapiens
228456_s_at     CDP-diacylglycerol synthase 2(CDS2)     Homo sapiens
200947_s_at     glutamate dehydrogenase 1(GLUD1)        Homo sapiens
201833_at       histone deacetylase 2(HDAC2)    Homo sapiens
220122_at       multiple C2 and transmembrane domain containing 1(MCTP1)        Homo sapiens
1563182_at      activin A receptor type 1C(ACVR1C)      Homo sapiens
```

Here is the names of the genes. 221953_s_at MMP24 opposite strand(MMP24OS) Homo sapiens 209726_at carbonic anhydrase 11(CA11) Homo sapiens 204813_at mitogen-activated protein kinase 10(MAPK10) Homo sapiens 201418_s_at SRY-box transcription factor 4(SOX4) Homo sapiens 226502_at ELMO domain containing 2(ELMOD2) Homo sapiens 228314_at leucine rich repeat containing 8 VRAC subunit C(LRRC8C) Homo sapiens 221473_x_at serine incorporator 3(SERINC3) Homo sapiens 203069_at synaptic vesicle glycoprotein 2A(SV2A) Homo sapiens 212859_x_at metallothionein 1E(MT1E) Homo sapiens 203417_at microfibril associated protein 2(MFAP2) Homo sapiens 209818_s_at hyaluronan binding protein 4(HABP4) Homo sapiens 232341_x_at hyaluronan binding protein 4(HABP4) Homo sapiens 203270_at deoxythymidylate kinase(DTYMK) Homo sapiens 218006_s_at zinc finger protein 22(ZNF22) Homo sapiens 203911_at RAP1 GTPase activating protein(RAP1GAP) Homo sapiens 228456_s_at CDP-diacylglycerol synthase 2(CDS2) Homo sapiens 200947_s_at glutamate dehydrogenase 1(GLUD1) Homo sapiens 201833_at histone deacetylase 2(HDAC2) Homo sapiens 220122_at

multiple C2 and transmembrane domain containing 1(MCTP1) Homo sapiens 1563182_at activin A receptor type 1C(ACVR1C) Homo sapiens Now we'll use enrich r to find the pathway of these genes.



Let's dive into the KEGG 2021.

Amazing! We found out that these genes are in the pathway of microRNA's in cnacer~ So the analysis was successful.

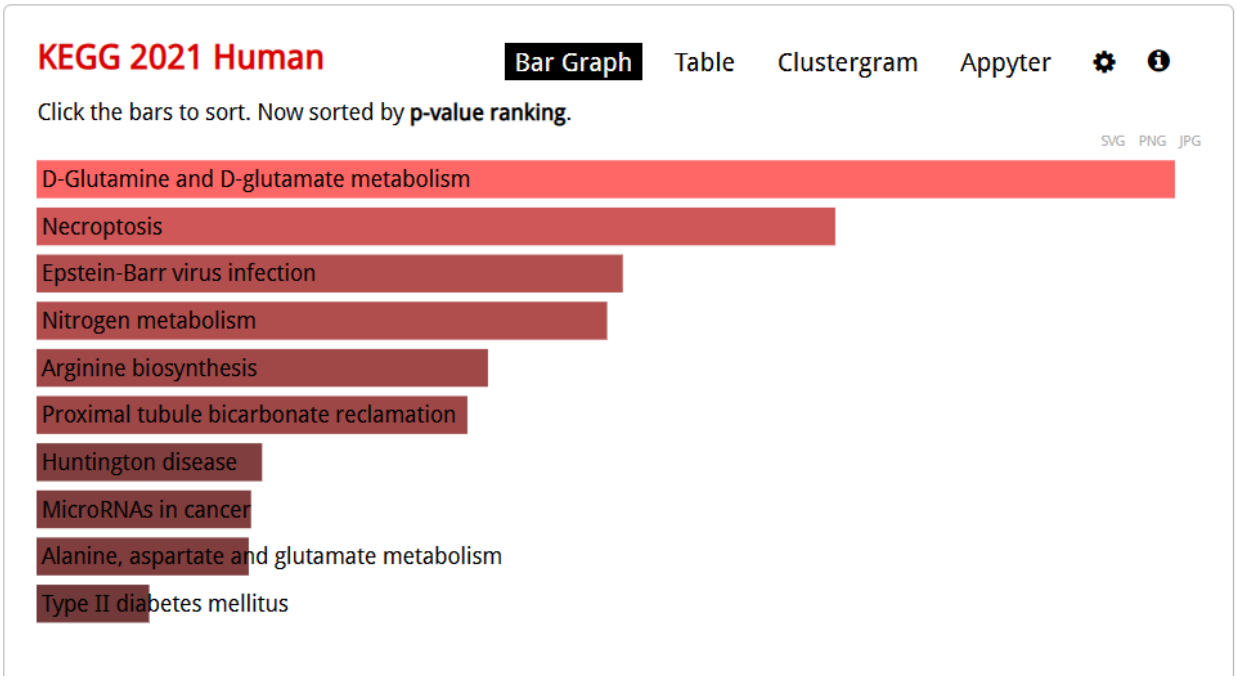


Figure 2: Caption for the image

Thank you ! <3

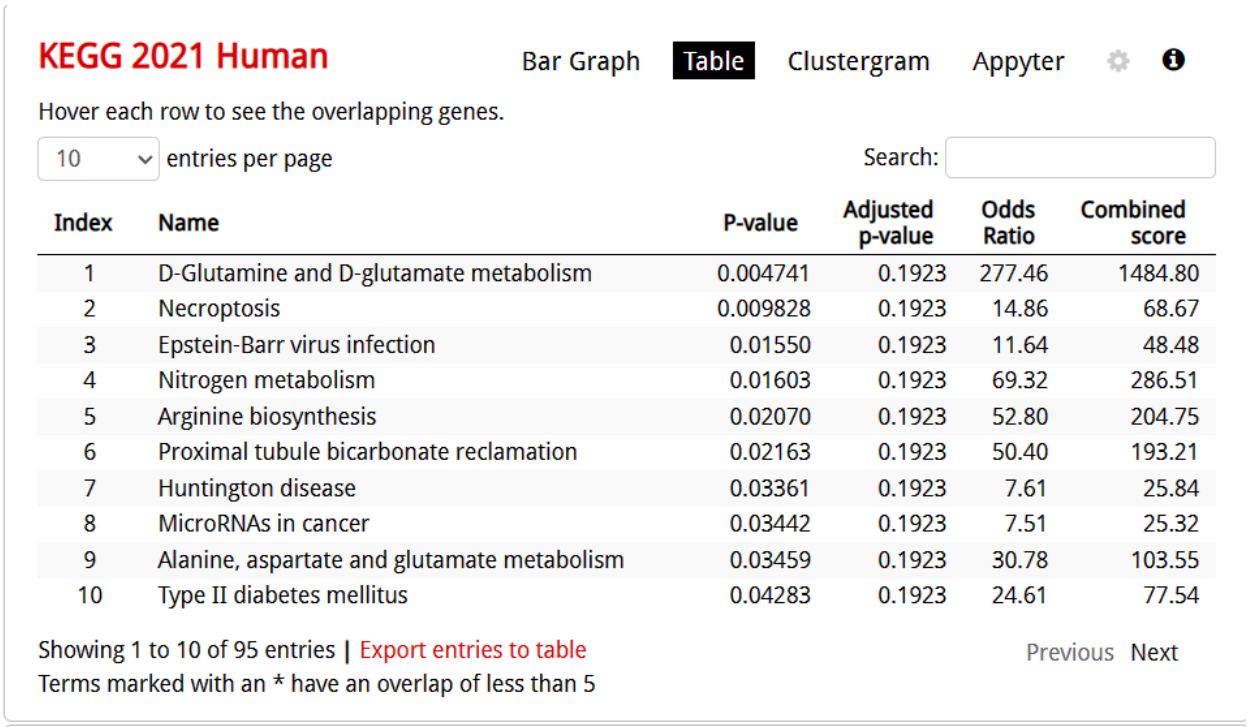


Figure 3: Caption for the image