Secondary Investigation

Ethan Allavarpu (UID: 405287603)

10/28/2020

Transforming and Cleaning the Data

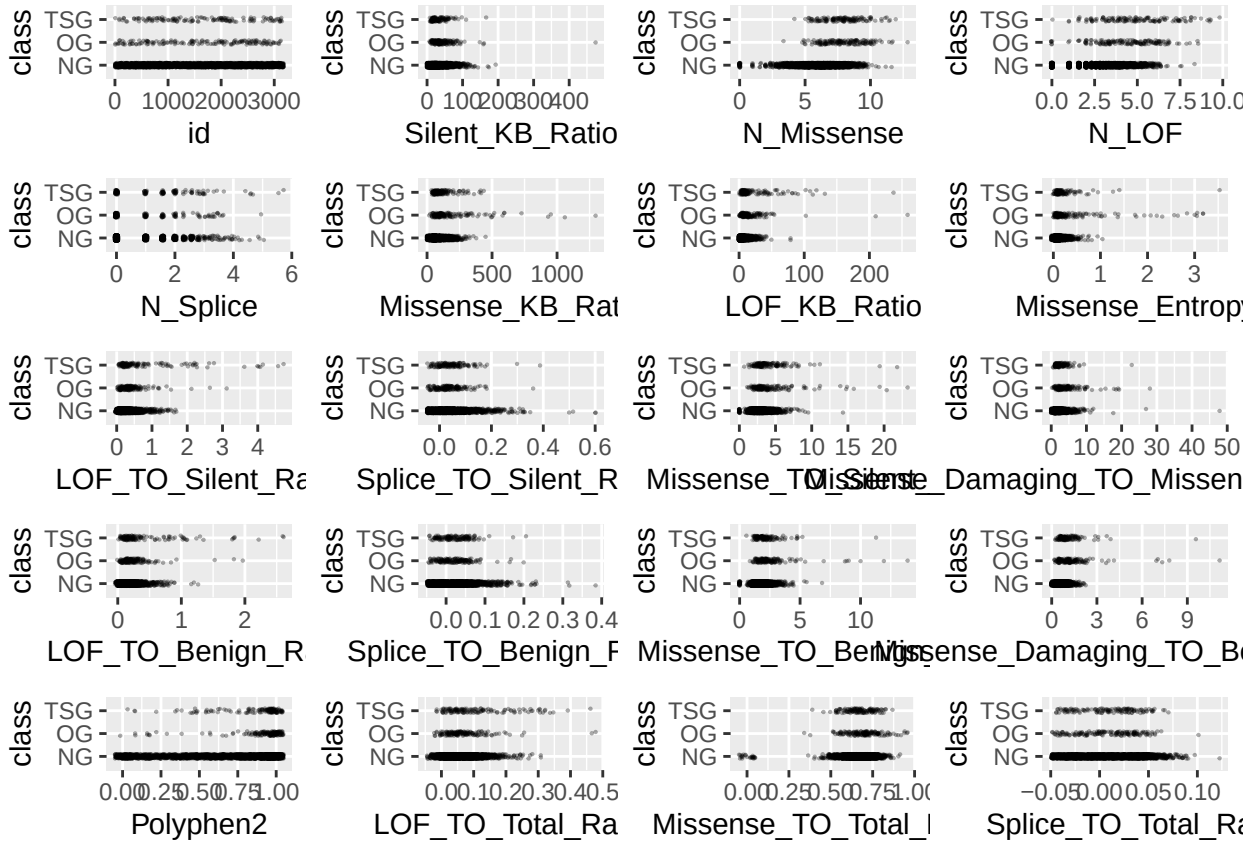
```
training <- read.csv("training.csv", stringsAsFactors = TRUE)
training$class <- factor(training$class)
levels(training$class) <- c("NG", "OG", "TSG")
outlier <- function(data) {
  low <- mean(data) - 3 * sd(data)
  high <- mean(data) + 3 * sd(data)
  which(data < low | data > high)
}
outlier_index <- sort(table(unlist(lapply(training[,-99], outlier))), decreasing = TRUE)
outlier_index[1:100]
```

```
##
##  915 1280 2918  517 1914 2182 3052 1173 2215 3049  259  740 1749 1979 2998  417
##   24   24   24   22   22   22   20   19   19   19   18   18   18   18   18   17
##  441  806 2297  422  635 1258 1570 2278 2518 2729   80  150 2694  169  276  341
##   17   17   17   16   16   16   16   16   16   16   15   15   15   14   14   14
## 1528 1556 1726 1809 1911 1955 2071 2624 2641 3120 3142   73  277  364  751 1244
##   14   14   14   14   14   14   14   14   14   14   14   13   13   13   13   13
## 1330 2329 2787  343 1138 1171 1188 1372 1460 2031 2251 2968 2983 3166  352  634
##   13   13   13   12   12   12   12   12   12   12   12   12   12   12   11   11
##  907  923 1096 1858 2636  588 1137 1317 1463 1561 1740 1991 2487 2540 2555 2621
##   11   11   11   11   11   10   10   10   10   10   10   10   10   10   10   10
## 2815 3029   74  144  657  789  857 1267 1610 1932 2022 2093 2142 2534 2666 2721
##   10   10    9    9    9    9    9    9    9    9    9    9    9    9    9    9
## 2848 2900 3027  155
##    9    9    9    8
```

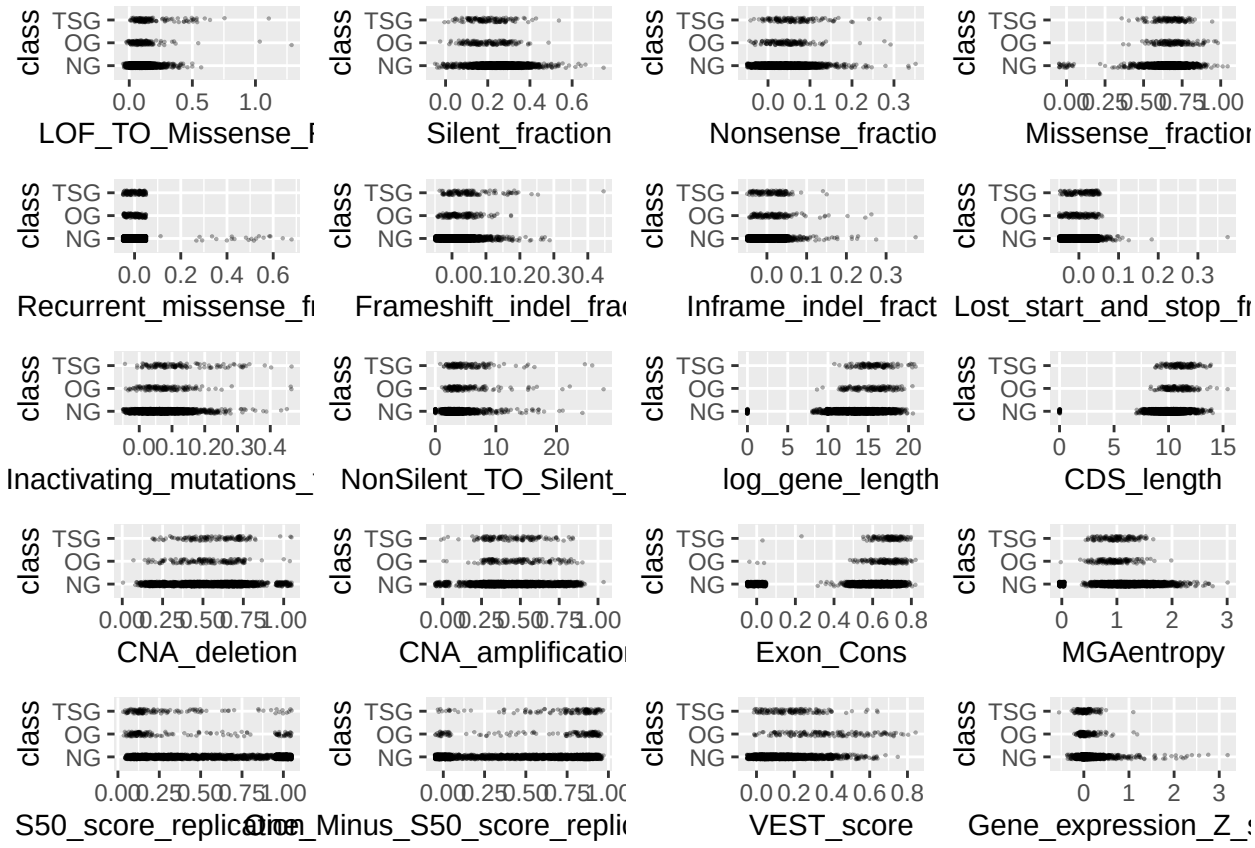
```
training <- training[-as.numeric(names(outlier_index)[1:50]),]
training <- training[-which(training$Missense_TO_Silent_Ratio > 100), ]
training <- training[-which(training$Missense_KB_Ratio > 2000), ]
training <- training[-which(training$LOF_TO_Silent_Ratio > 5), ]
training <- training[-which(training$Gene_expression_Z_score > 4), ]
```

```
library(ggplot2)
scatter <- function(var) {
  ggplot(training, aes_string(var, "class")) +
    geom_jitter(width = 0.05, height = 0.1, size = 0.1,
                colour = rgb(0, 0, 0, alpha = 1 / 3))
}
scat_plot <- lapply(names(training)[-99], scatter)
library(gridExtra)
```

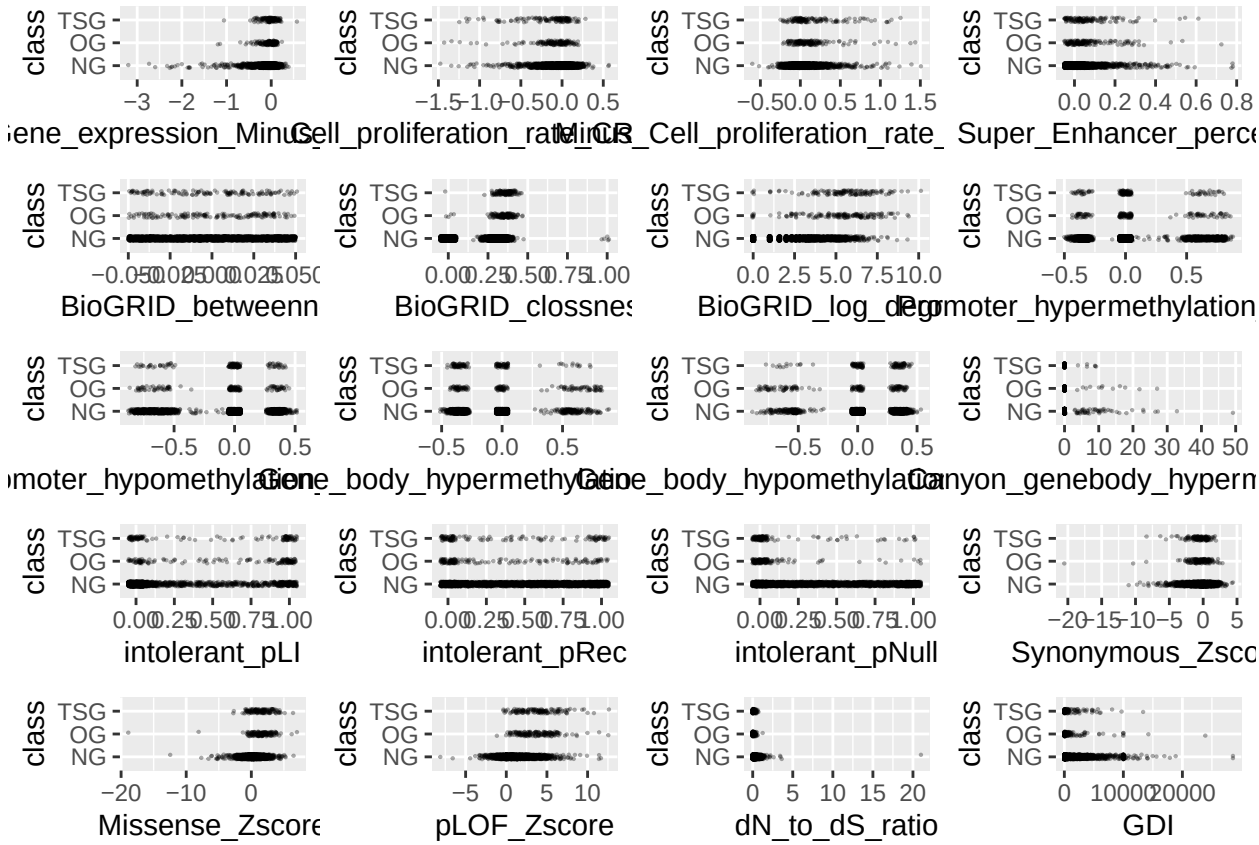
```
grid.arrange(grobs = scat_plot[1:20], ncol = 4)
```



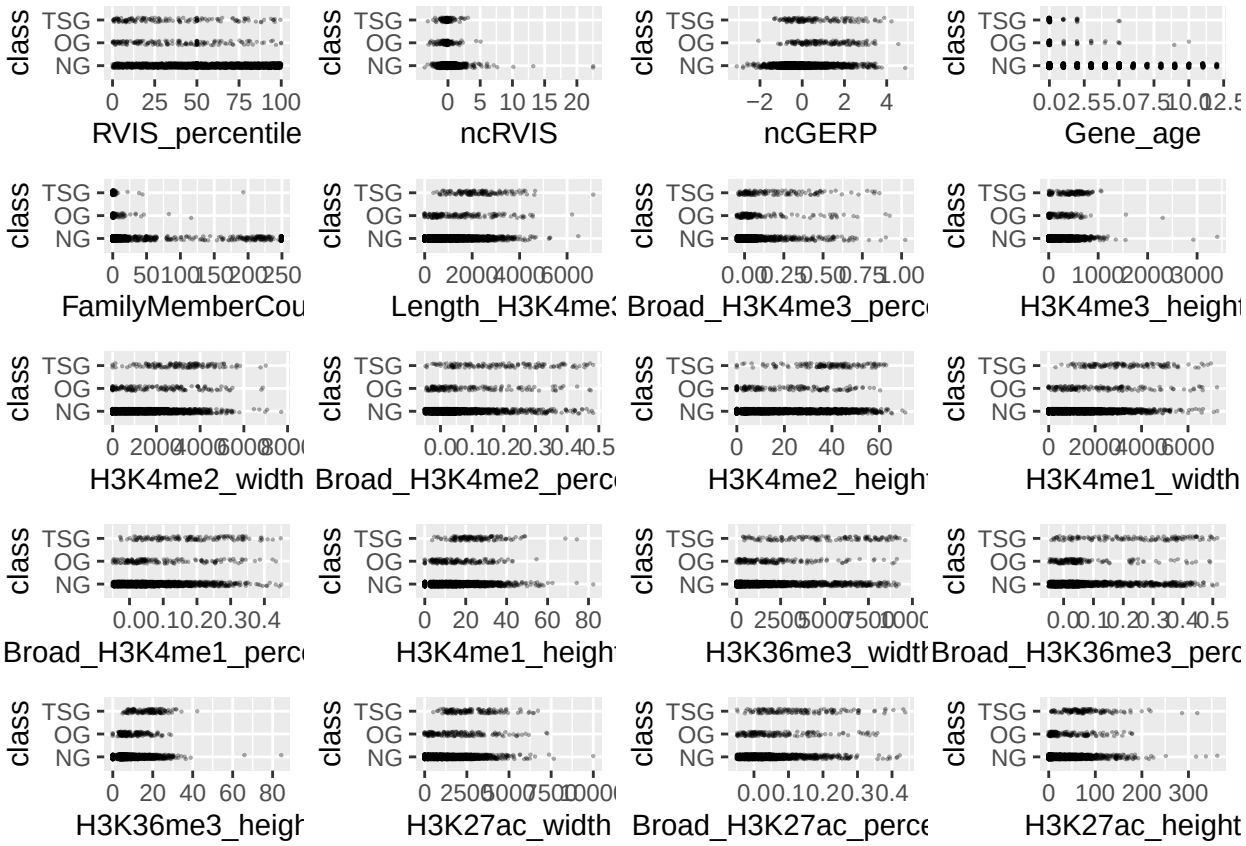

```
grid.arrange(grobs = scat_plot[21:40], ncol = 4)
```



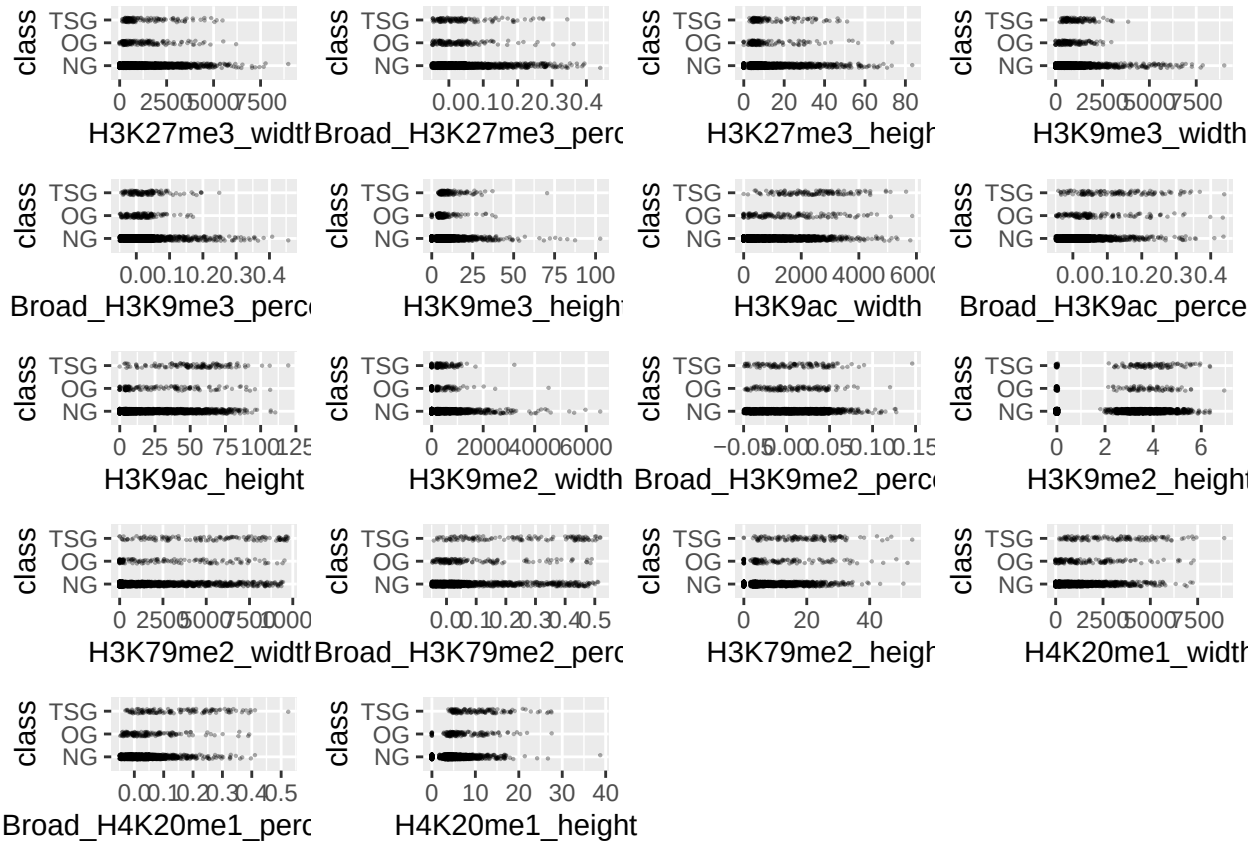

```
grid.arrange(grobs = scat_plot[41:60], ncol = 4)
```




```
grid.arrange(grobs = scat_plot[61:80], ncol = 4)
```




```
grid.arrange(grobs = scat_plot[81:98], ncol = 4)
```




```
sig <- logical(98)
names(sig) <- names(training)[-99]
k <- 1
diffs <- logical(98)
for (var in names(training)[-99]) {
  model <- aov(training[[var]] ~ factor(training$class))
  sig[k] <- summary(model)[[1]][1, 5]
  diffs[k] <- all(TukeyHSD(model)$`factor(training$class)`[, 4] < 0.05)
  k <- k + 1
}
sort(sig[diffs])
```

```
##                   Broad_H4K20me1_percentage
##                                7.591509e-151
##                                   VEST_score
##                                2.046663e-124
##                             Missense_Entropy
##                                1.795853e-115
##                     Broad_H3K9ac_percentage
##                                2.070999e-114
##                              H3K79me2_height
##                                4.967193e-114
##                               H3K79me2_width
##                                1.613005e-113
##                   Broad_H3K79me2_percentage
##                                1.186192e-110
##                                intolerant_pLI
```

```
##                              2.386124e-110
##                    Broad_H3K4me2_percentage
##                              4.069382e-109
##           Missense_Damaging_TO_Benign_Ratio
##                              1.321433e-108
##                             H4K20me1_width
##                              3.085692e-108
##                   Broad_H3K36me3_percentage
##                              2.818836e-107
##                             H3K36me3_width
##                              2.371798e-106
##                            H4K20me1_height
##                              3.751861e-103
##                         LOF_TO_Silent_Ratio
##                              2.253534e-102
##                   Broad_H3K27ac_percentage
##                              4.421588e-100
##                   Broad_H3K4me1_percentage
##                               6.741241e-99
##                               LOF_KB_Ratio
##                               3.926170e-97
##                             Missense_KB_Ratio
##                               9.622464e-95
##                                      N_LOF
##                               2.566144e-91
##                              H3K4me1_width
##                               2.949853e-82
##                   Broad_H3K4me3_percentage
##                               1.019930e-80
##                         LOF_TO_Benign_Ratio
##                               2.996845e-79
##                             H3K36me3_height
##                               3.812963e-77
##                              H3K4me2_width
##                               3.284765e-76
##                               H3K9ac_width
##                               4.849792e-74
##                              H3K9ac_height
##                               6.225877e-68
##                              H3K27ac_width
##                               2.431524e-65
##                             Length_H3K4me3
##                               7.944077e-63
##                                     ncGERP
##                               2.844423e-62
##                    Missense_TO_Benign_Ratio
##                               4.090738e-62
##                              H3K27ac_height
##                               1.724476e-60
## Missense_Damaging_TO_Missense_Benign_Ratio
##                               1.685814e-57
##                              H3K4me3_height
##                               2.070353e-52
##                              H3K4me2_height
```

```
##                                9.196675e-51
##                                    N_Splice
##                                1.258008e-48
##                           LOF_TO_Total_Ratio
##                                2.656330e-48
##                              H3K4me1_height
##                                1.000074e-43
##                         LOF_TO_Missense_Ratio
##                                3.449479e-43
##                    Frameshift_indel_fraction
##                                2.654203e-34
##              Inactivating_mutations_fraction
##                                8.751694e-32
##           Cell_proliferation_rate_CRISPR_KD
##                                3.540506e-24
##     Minus_Cell_proliferation_rate_CRISPR_KD
##                                3.540506e-24
##                       Inframe_indel_fraction
##                                3.671900e-13
```

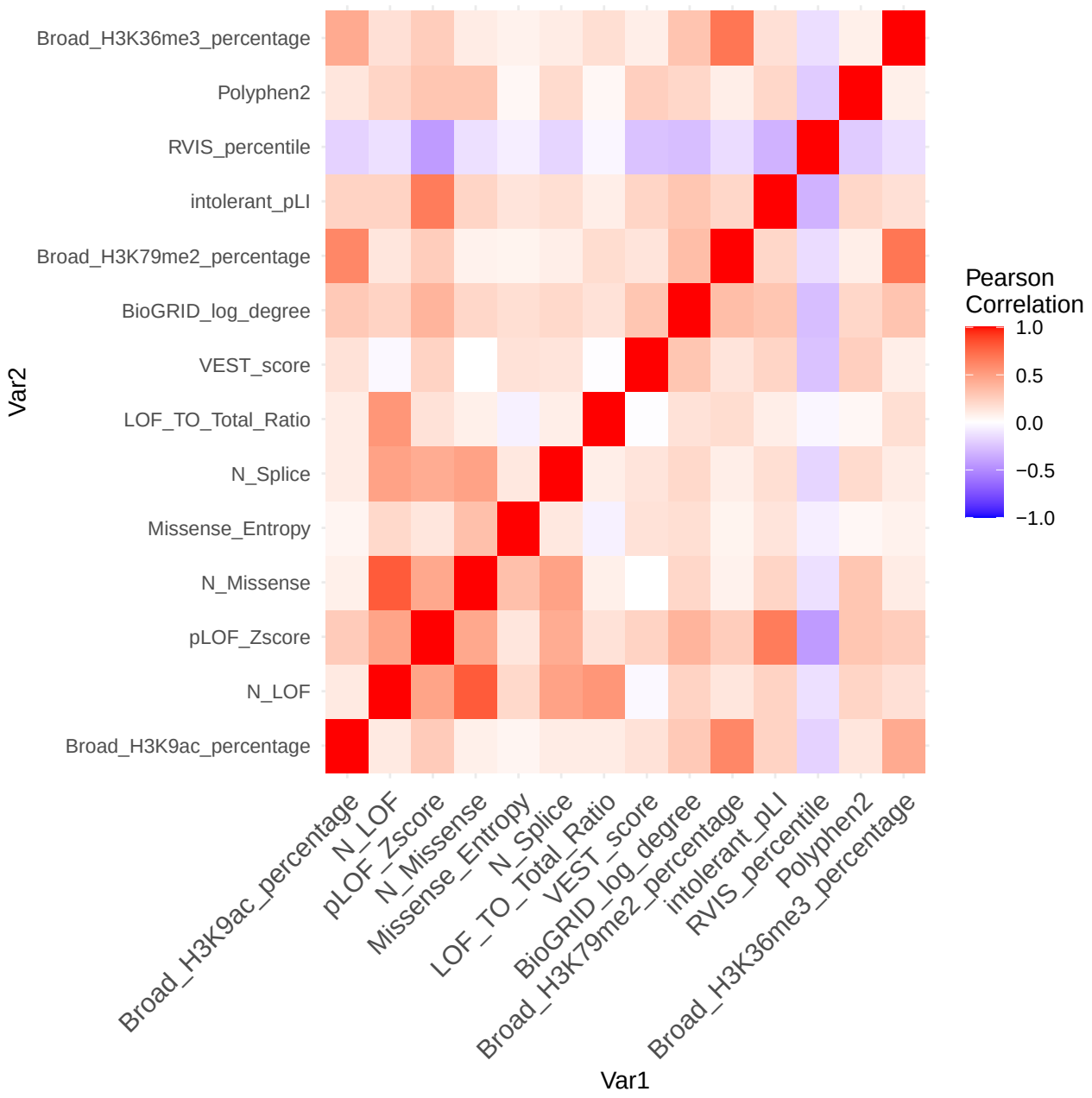
```
score <- function (conf_mat) {
  print(sum(diag(conf_mat) * c(1, 20, 20)))
  print(sum(diag(conf_mat) * c(1, 20, 20)) / sum(apply(conf_mat, 2, sum) * c(1, 20, 20)))
}

classify <- function(probs) {
  if (any(probs[2:3] > 0.05)) {
    subset <- probs[2:3]
    output <- which(subset == max(subset))
    if (length(output) > 1) {
        output <- sample(1:2, 1)
    }
  } else {
    output <- 0
  }
  output
}
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
vars <- training %>% select(Broad_H3K9ac_percentage, N_LOF, pLOF_Zscore, N_Missense, Missense_Entropy,
                            N_Splice, LOF_TO_Total_Ratio, VEST_score, BioGRID_log_degree, Broad_H3K79me
```

```
                            intolerant_pLI, RVIS_percentile,
                            Polyphen2, Broad_H3K36me3_percentage, class)
vars$class <- factor(vars$class)
levels(vars$class) <- c("NG", "OG", "TSG")
cor_mtx = round(cor(vars[, names(vars) != "class"]), 2)
library(reshape2)
#reshape it
melted_cor_mtx <- melt(cor_mtx)

#draw the heatmap
cor_heatmap = ggplot(data = melted_cor_mtx, aes(x=Var1, y=Var2, fill=value)) + geom_tile()
cor_heatmap = cor_heatmap +
scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1,1), space =
theme_minimal() +
theme(axis.text.x = element_text(angle = 45, vjust = 1, size = 12, hjust = 1))

cor_heatmap
```




```
library(dplyr)
set.seed(nrow(training) +345)
vars <- training %>% select(Broad_H3K9ac_percentage, N_LOF, pLOF_Zscore, Missense_Entropy,
                            N_Splice, LOF_TO_Total_Ratio, VEST_score, BioGRID_log_degree, Broad_H3K79me
                            Gene_age,
                            Polyphen2, Broad_H3K36me3_percentage, class)
library(caret)
```

```
## Loading required package: lattice
```

```
vars_test <- createDataPartition(vars$class, p = 0.7,
                                 list = FALSE)
vars_train <- vars[vars_test, ]
vars_test <- vars[-vars_test, ]
```

```
train_cont <- trainControl(method = "cv", number = 5, classProbs = TRUE, savePredictions = TRUE)


knn_ft <- train(class ~ ., data = vars_train, method = "knn", preProc = c("center", "scale"),
                trControl = train_cont, tuneGrid = expand.grid(k = seq(from = 1, to = 25, by = 5)))
for (k in seq(from = 1, to = 25, by = 5)) {
  preds <- predict(knn_ft, newdata = vars_test, type = "prob")
  knn_mod <- table("pred"=unlist(apply(preds, 1, classify)), "obs" = vars_test$class)
  print(knn_mod)
  score(knn_mod)
}
```

```
##     obs
## pred  NG  OG TSG
##    0 734   7   8
##    1  61  27   6
##    2  54  11  25
## [1] 1774
## [1] 0.701463
##     obs
## pred  NG  OG TSG
##    0 734   7   8
##    1  57  27   6
##    2  58  11  25
## [1] 1774
## [1] 0.701463
##     obs
## pred  NG  OG TSG
##    0 734   7   8
##    1  58  28   6
##    2  57  10  25
## [1] 1794
## [1] 0.7093713
##     obs
## pred  NG  OG TSG
##    0 734   7   8
##    1  56  27   5
##    2  59  11  26
## [1] 1794
## [1] 0.7093713
##     obs
## pred  NG  OG TSG
##    0 734   7   8
##    1  56  28   4
##    2  59  10  27
## [1] 1834
## [1] 0.7251878
```

```
train_cont <- trainControl(method = "cv", number = 5, classProbs = TRUE, savePredictions = TRUE)
qda_ft <- train(class ~ ., data = vars_train, method = "qda", preProc = c("center", "scale"),
                trControl = train_cont)
preds <- predict(qda_ft, newdata = vars_test, type = "prob")

qda_mod <- table("pred"=apply(preds, 1, classify), "obs" = vars_test$class)
```

```
qda_mod
```

```
##     obs
## pred  NG  OG TSG
##    0 749   5   5
##    1  19  30   4
##    2  81  10  30
```

```
score(qda_mod)
```

```
## [1] 1949
## [1] 0.7706603
```

```
train_cont <- trainControl(method = "cv", number = 5, classProbs = TRUE, savePredictions = TRUE)
lda_ft <- train(class ~ ., data = vars_train, method = "lda", preProc = c("center", "scale"),
                trControl = train_cont)
preds <- predict(lda_ft, newdata = vars_test, type = "prob")

lda_mod <- table("pred"=apply(preds, 1, classify), "obs" = vars_test$class)
lda_mod
```

```
##     obs
## pred  NG  OG TSG
##    0 781   7   7
##    1  21  32   4
##    2  47   6  28
```

```
score(lda_mod)
```

```
## [1] 1981
## [1] 0.7833136
```