8. Appendix 1: Classification Code

tmp.html

AML classification using Dirichlet Processes

```
# /lustre/scratch117/casm/team154/cr8/DLBCL_study/annovar_transfer/reference/bsub_farm yeste
# /lustre/scratch117/casm/team154/cr8/DLBCL_study/annovar_transfer/reference/bsub_farm yeste
```

Code run on

```
options(markdown.HTML.header = "tmp.html")
system("hostname -f", intern=TRUE)
```

```
## [1] "bc-29-2-08.internal.sanger.ac.uk"
```

```
Sys.time()
```

```
## [1] "2017-08-27 17:03:27 BST"
```

```
getwd()
```

```
## [1] "/lustre/scratch117/casm/team154/cr8/DLBCL_study/annovar_transfer/full_study/synthesiz
```

using

```
library(knitr)
```

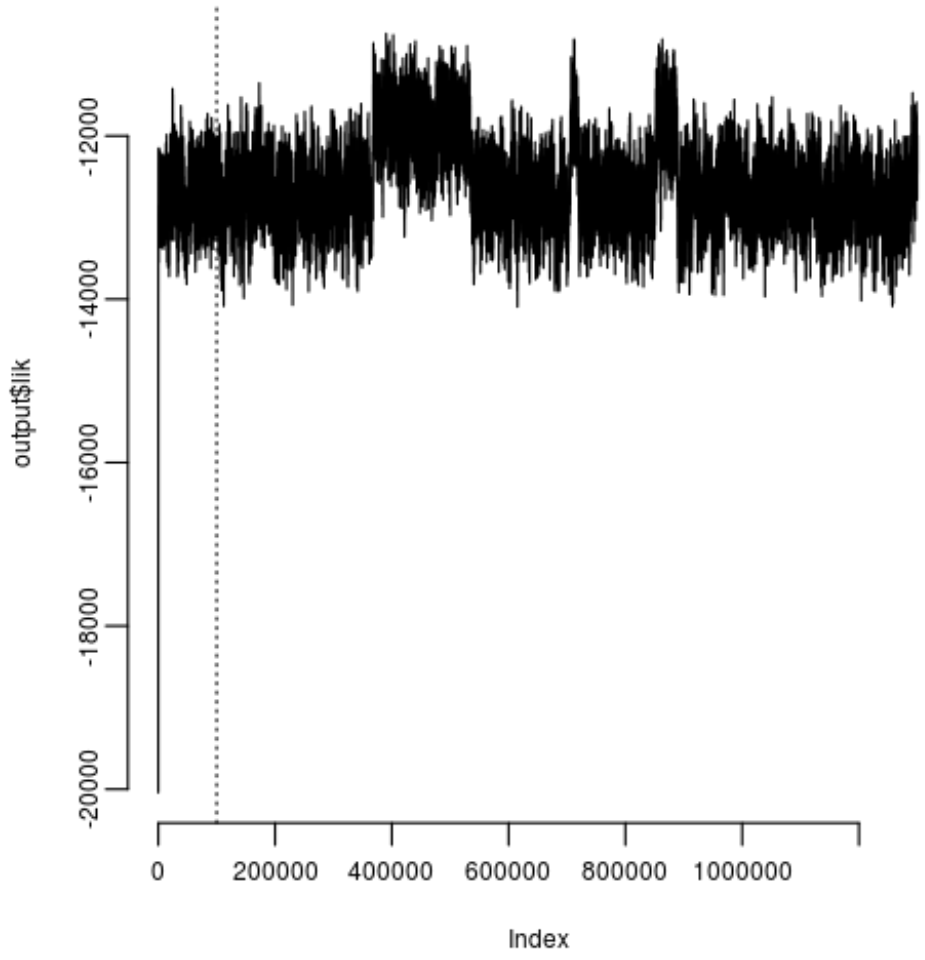
Libraries and data

```
source("/lustre/scratch117/casm/team154/cr8/DLBCL_study/cleaning_and_annotation/code/global_s
load_global_packages()

library(CoxHD) # library(devtools); install_github("mg14/CoxHD/CoxHD")
library(mg14) # library(devtools); install_github("mg14/mg14")
library(hdp)
library(lattice)
set1 <- brewer.pal(8, "Set1")

# If running from classification_workspace.Rdata instead of from scratch
load("classification_workspace.Rdata")
spin("/lustre/scratch117/casm/team154/cr8/DLBCL_study/cleaning_and_annotation/code/visualize_
```

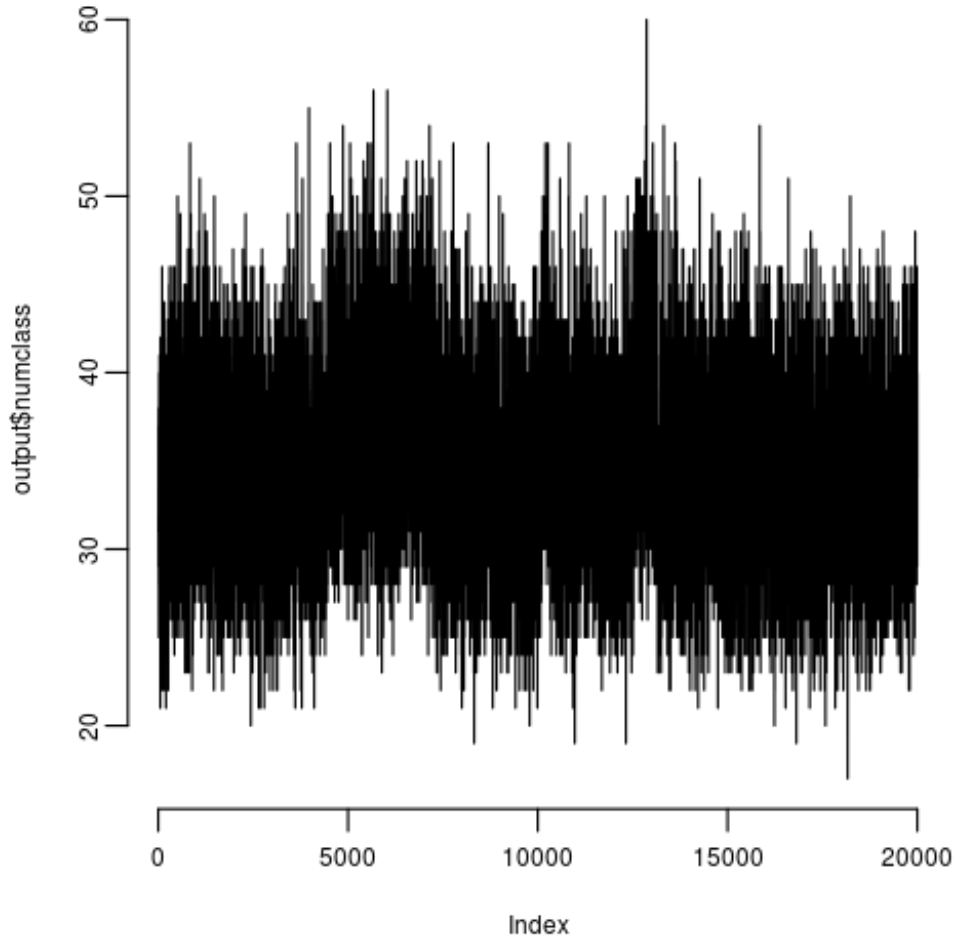
```
##
##
## processing file: /lustre/scratch117/casm/team154/cr8/DLBCL_study/cleaning_and_annotation/c
```

```
## Error in parse_block(g[-1], g[1], params.src): duplicate label 'run'
```

```
plot(output$lik, type='l'); abline(v=burnin, lty=3)
```




```
plot(output$numclass, type='l')
```



AML classes

```
posteriorMerged <- hdp_extract_signatures(output, prop.explained=0.99, cos.merge=0.95)
```

```
#posteriorMeans <- Reduce("+",posteriorMerged$sigs_qq)/length(posteriorMerged$sigs_qq)
posteriorSamples <- array(unlist(posteriorMerged$sigs_qq), dim=c(dim(posteriorMerged$sigs_qq[
rownames(posteriorSamples) <- colnames(genotypesImputed)
colnames(posteriorSamples) <- 1:ncol(posteriorSamples) -1
posteriorMeans <- rowMeans(posteriorSamples, dim=2)
posteriorQuantiles <- apply(posteriorSamples, 1:2, quantile, c(0.025,.5,0.975))
posteriorMode <- apply(posteriorSamples, 1:2, function(x) {t <- table(x); as.numeric(names(t)
kable(posteriorQuantiles[2,,], "html", table.attr = 'id="posteriorMedian"') # Posterior media
```

	0	1	2	3	4	5	6	7
SOCS1	0	0	0	0	80	0	1	0
FOXO1	0	77	0	0	0	0	0	0
BTG2	0	0	55	0	0	0	1	0
FAS	0	0	0	0	16	31	0	0
ARID1A	0	115	0	0	0	0	0	0
KMT2D	0	607	0	0	0	0	0	0
CREBBP	0	363	0	0	0	0	0	0
TNFRSF14	0	312	0	0	0	0	0	0
ID3	0	0	0	0	0	0	0	0

	0	1	2	3	4	5	6	7
ZFP36L1	0	0	0	0	4	0	4	0
ASXL2	0	0	0	0	0	0	3	0
NOTCH2	0	0	0	0	0	65	0	0
WT1	0	0	0	0	0	0	3	0
IRF4	0	0	23	0	0	0	0	0
CDKN2A	0	0	50	0	0	0	0	0
PTPRC	0	0	0	0	0	0	0	0
MYC	0	0	0	0	0	0	10	0
B2M	0	2	0	0	57	47	0	0
TP53	0	37	0	73	49	0	4	30
TBL1XR1	0	0	48	0	0	0	1	0
TET2	0	0	0	0	0	0	2	0
TNFAIP3	0	0	0	0	38	56	0	0
ASXL1	0	0	0	0	0	0	1	0
NFKBIE	0	0	0	0	39	0	0	0
NOTCH1	0	0	0	0	0	0	3	0
STAT3	0	0	0	0	0	0	0	0
IRF8	0	110	0	0	0	0	0	0
SMARCB1	0	0	0	0	0	0	13	0
BCL10	0	0	0	0	0	48	0	0
SRSF2	0	0	0	0	0	0	5	0
PDS5B	0	0	0	0	0	0	5	0
MEF2B	0	49	0	0	0	0	0	0
BTK	0	0	0	0	0	0	0	0
POU2AF1	0	34	0	0	0	0	0	0
ARID1B	0	42	0	0	0	0	0	0
MGA	0	0	0	0	0	0	1	0
CARD11	0	97	0	0	0	0	0	0
EP300	0	46	0	0	0	0	0	0
SPEN	0	0	0	0	0	35	3	0
ATRX	0	0	0	0	0	0	7	0
DNMT3A	0	0	0	0	0	0	0	0
NF1	0	0	18	0	0	0	1	0
PIK3R1	0	19	0	0	0	0	0	0
FBXO11	0	0	0	0	0	0	0	0
PRDM1	0	0	41	0	0	0	0	0
IKZF1	0	0	0	0	0	0	0	0
TRAF3	0	0	0	0	0	0	6	0
SMARCA4	0	0	0	0	0	0	0	0
PTEN	0	0	0	0	0	0	0	0
CDKN1B	0	0	0	0	0	0	8	0
BCORL1	0	0	0	0	0	0	4	0

MSH2	0	0	0	0	0	0	5	0
	0	1	2	3	4	5	6	7
MYD88	0	0	149	0	0	0	1	0
RB1	0	0	0	0	0	0	1	0
EZH2	0	290	0	0	0	0	0	0
ATM	0	0	0	0	0	0	6	0
BCL11A	0	0	0	0	0	0	1	0
BCOR	0	0	0	0	0	0	3	0
KMT2C	0	0	0	0	0	0	2	0
RASA2	0	0	0	0	0	0	8	0
CHD2	0	0	0	0	0	0	4	0
NFKB1	0	0	0	0	0	0	2	0
PHF6	0	0	0	0	0	0	0	0
MAP2K1	0	0	0	0	0	0	0	0
CASP8	0	0	0	0	0	0	10	0
CCND3	0	0	0	35	0	0	0	0
BCL7A	0	0	0	0	0	0	3	0
KDM6A	0	0	0	0	0	0	1	0
XPO1	0	0	0	0	0	0	0	0
PAX5	0	0	0	0	0	0	0	0
DDX3X	0	0	0	0	0	0	0	0
FAM46C	0	0	0	0	0	0	1	0
ARID2	0	0	0	0	0	0	4	0
CD79B	0	0	0	0	0	0	2	0
FBXW7	0	0	0	0	0	0	0	0
PTPN6	0	0	0	0	0	0	1	0
FAT1	0	0	0	0	0	0	13	0
MSH6	0	0	0	0	0	0	12	0
BIRC3	0	0	0	0	0	0	4	0
BLM	0	0	0	0	0	0	2	0
CD79A	0	0	0	0	0	0	2	0
PPM1D	0	0	0	0	0	0	1	0
BCL6	0	0	0	0	0	0	5	0
SETD2	0	0	0	0	0	0	6	0
HIST1H3B	0	0	0	0	0	0	3	0
CUX1	0	0	0	0	0	0	2	0
PMS2	0	0	0	0	0	0	5	0
CYLD	0	0	0	0	0	0	4	0
IRF1	0	0	0	0	0	0	0	0
CDKN2B	0	0	0	0	0	0	1	0
CBFB	0	0	0	0	0	0	1	0
CTCF	0	0	0	0	0	0	4	0
MTOR	0	0	0	0	0	0	5	0

PTPRD	0	0	0	0	0	0	2	0
STAT6	0	35	0	0	0	0	0	0
SGK1	0	0	0	0	26	0	0	0
BRAF	0	0	0	0	0	0	0	0
WHSC1	0	0	0	0	0	0	1	0
RHOA	0	0	0	0	0	0	0	0
KRAS	0	0	0	0	0	0	3	0
CXCR4	0	0	0	0	0	0	0	0
SF3B1	0	0	0	0	0	0	6	0
CD58	0	0	0	0	0	0	3	0
NF2	0	0	0	0	0	0	3	0
PIK3CA	0	0	0	0	0	0	2	0
U2AF1	0	0	0	0	0	0	3	0
IDH2	0	0	0	0	0	0	1	0
NRAS	0	0	0	0	0	0	4	0
ARAF	0	0	0	0	0	0	2	0
JAK3	0	0	0	0	0	0	4	0
CDKN2C	0	0	0	0	0	0	1	0
IDH1	0	0	0	0	0	0	1	0
MLH1	0	0	0	0	0	0	2	0
GNAS	0	0	0	0	0	0	1	0
JAK2	0	0	0	0	0	0	1	0
TCF3	0	0	0	0	0	0	0	0
KLF2	0	0	4	0	0	23	1	0
TERT	0	0	0	0	0	0	1	0

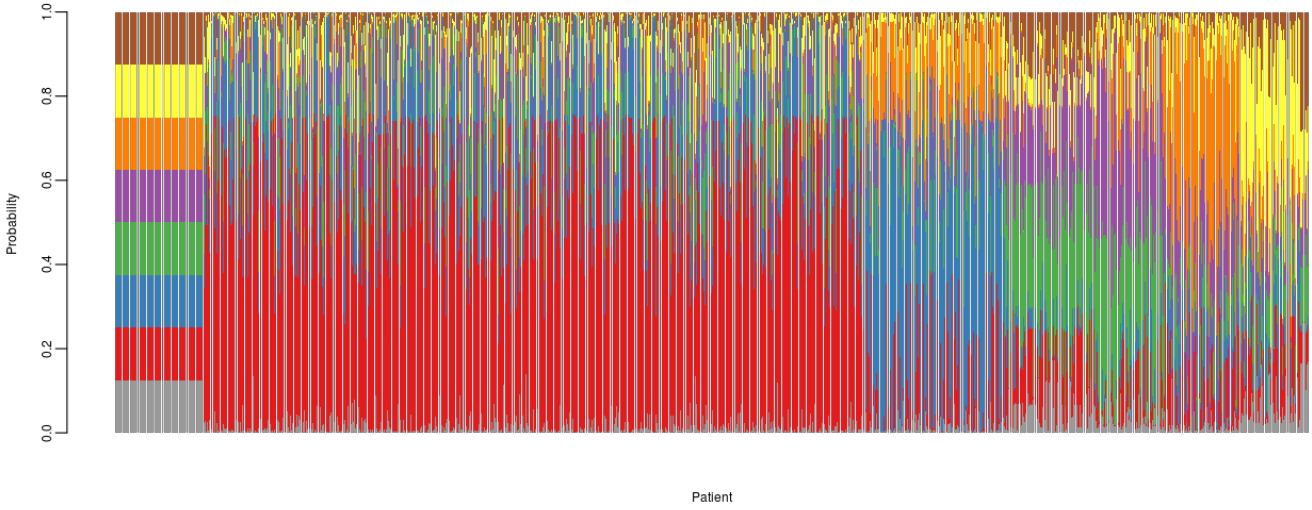
Most prevalent lesions

```
genes <- apply(posteriorMeans, 2, function(x) paste(ifelse(x>10,rownames(posteriorMeans),"")[c
genes <- gsub(";+$","",genes)
genes
```

```
##                                     0                                     1
##                          "TET2;TP53" "KMT2D;CREBBP;TNFRSF14;EZH2;ARID1A"
##                                     2                                     3
##  "KMT2D;MYD88;CREBBP;TNFRSF14;EZH2"          "TP53;SOCS1;TET2;B2M;CCND3"
##                                     4                                     5
##       "SOCS1;B2M;TP53;TET2;TNFAIP3"    "TNFAIP3;NOTCH2;B2M;BCL10;MYD88"
##                                     6                                     7
##                     "TP53;MYC;FAT1"                   "TP53;TET2;KMT2D"
```

Assignment from posterior samples

```
library(RColorBrewer)
col <- c(brewer.pal(9,"Set1")[c(9,1:8)], brewer.pal(8,"Dark2"))
posteriorProbability <- apply(sapply(posteriorMerged$sigs_nd_by_dp, colMeans)[,-1],2,function
o <- order(apply(posteriorProbability,2,which.max))
barplot(posteriorProbability[,o], col=col, border=NA, ylab="Probability", xlab="Patient")
```




```
data.frame(Prob=rowMeans(posteriorProbability), genes)
```

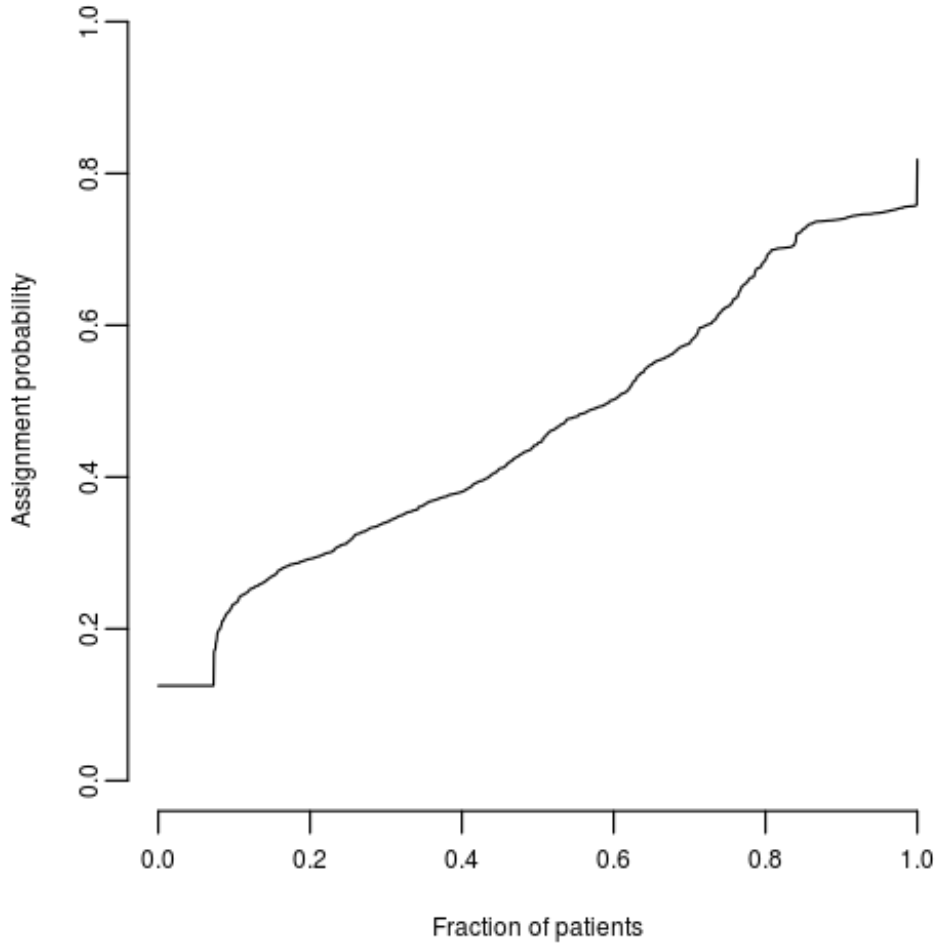
```
##         Prob                              genes
## 0 0.03245091                          TET2;TP53
## 1 0.35612484 KMT2D;CREBBP;TNFRSF14;EZH2;ARID1A
## 2 0.18731913  KMT2D;MYD88;CREBBP;TNFRSF14;EZH2
## 3 0.11346064         TP53;SOCS1;TET2;B2M;CCND3
## 4 0.10614709       SOCS1;B2M;TP53;TET2;TNFAIP3
## 5 0.08139741    TNFAIP3;NOTCH2;B2M;BCL10;MYD88
## 6 0.07549752                     TP53;MYC;FAT1
## 7 0.04760247                  TP53;TET2;KMT2D
```

Classes

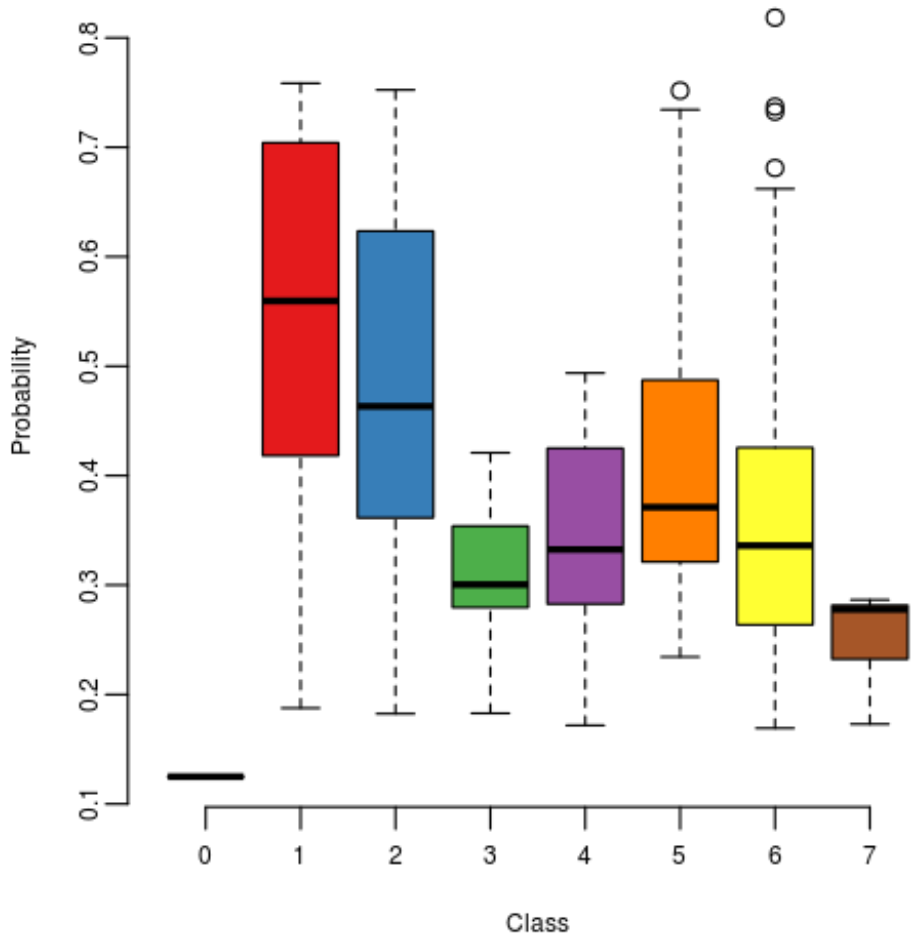
```
dpClass <- factor(apply(posteriorProbability, 2, which.max)-1)
table(dpClass)
```

```
## dpClass
##   0   1   2   3   4   5   6   7
## 118 890 190 126  90 102  80  11
```

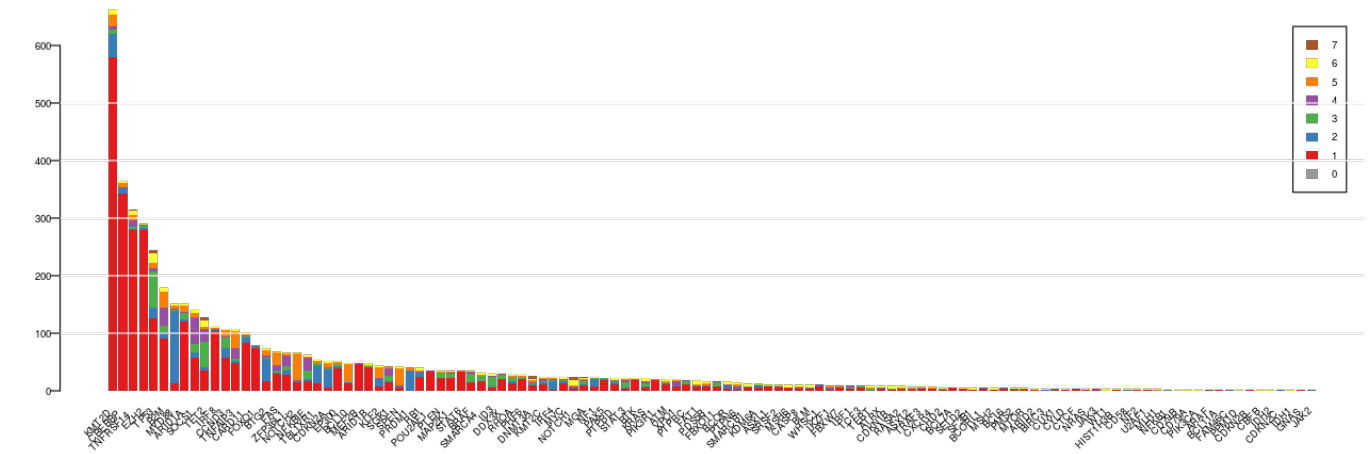
```
plot(seq(0,1,l=ncol(posteriorProbability)),sort(apply(posteriorProbability,2,max)), type='l',
```




```
boxplot(apply(posteriorProbability,2,max) ~ dpClass, col=col, ylab="Probability", xlab="Class
```

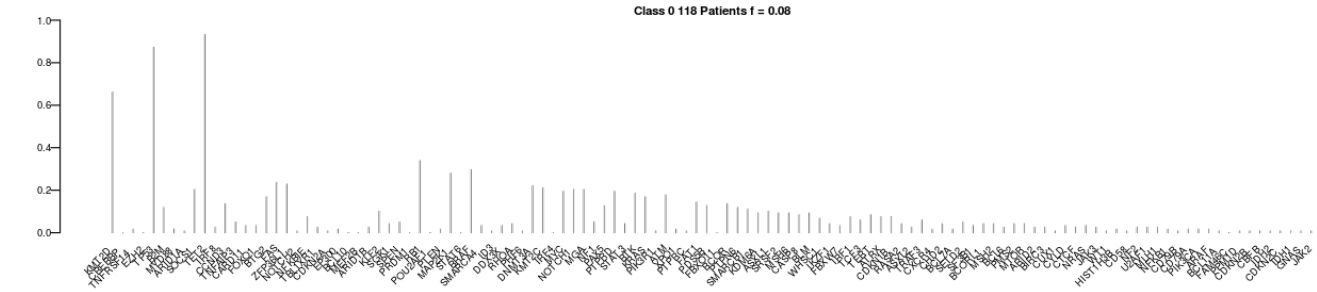


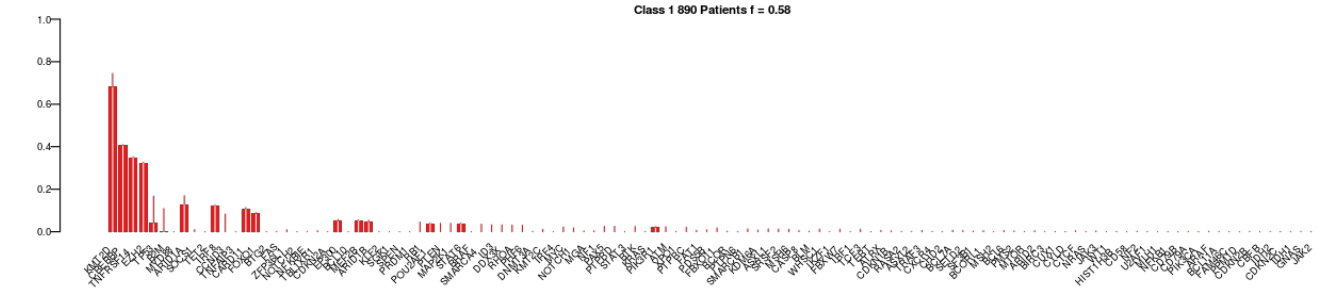

```
par(mar=c(6,3,1,1)+.1, cex=.8)
o <- order(colSums(genotypesImputed), decreasing=TRUE)
driverPrevalence <- t(sapply(split(as.data.frame(as.matrix(genotypesImputed)), dpClass), colS
b <- barplot(driverPrevalence, col=col, las=2, legend=TRUE, border=NA, args.legend=list(borde
abline(h=seq(100,500,100), col="white")
rotatedLabel(b, labels=colnames(genotypesImputed)[o])
```

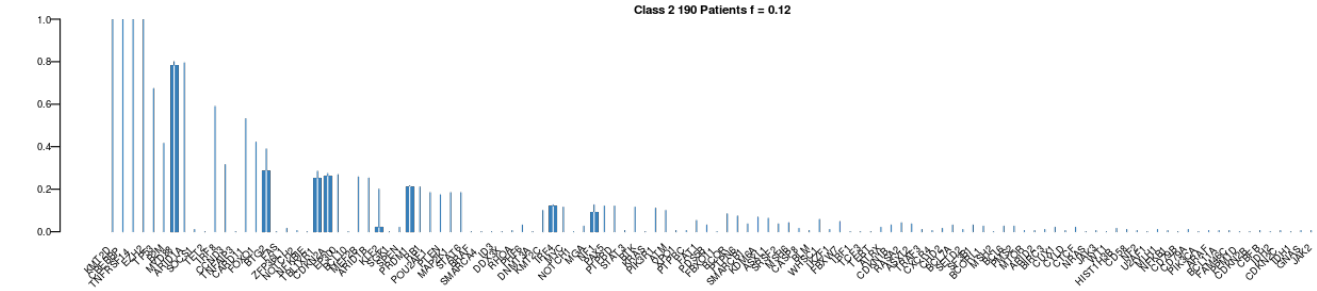


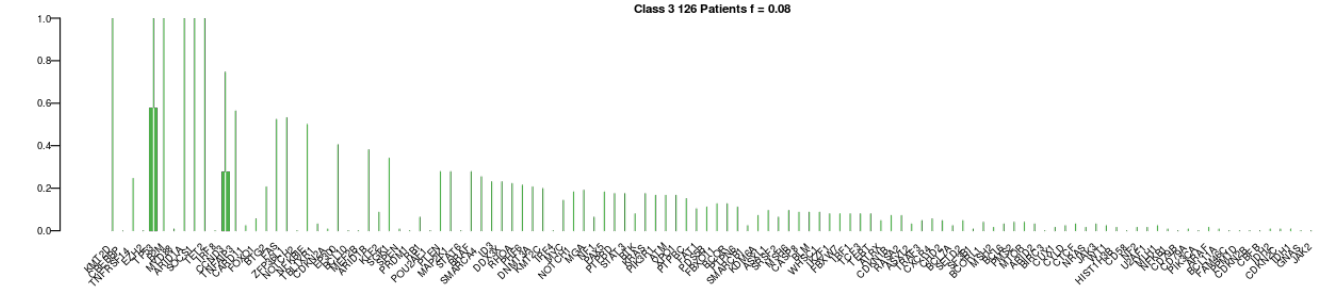
Driver signatures

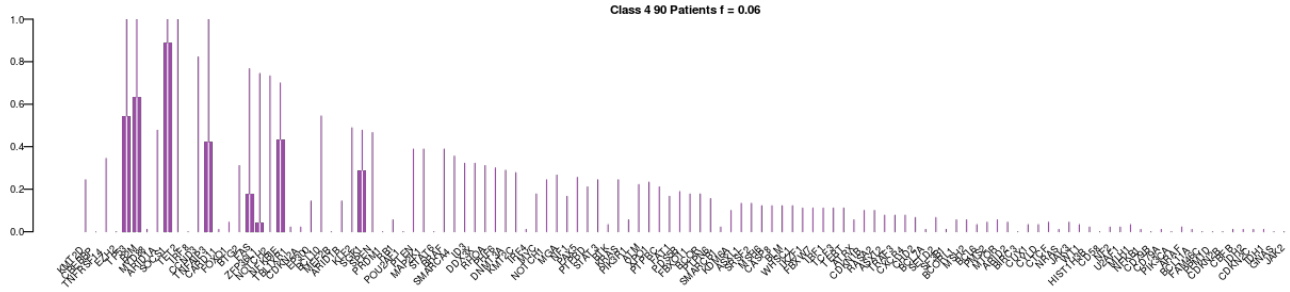
```
par(mar=c(6,3,1,1)+.1, cex=.8)
t <- table(dpClass)
i <- 0; for(c in levels(dpClass)){i <- 1+i
b <- barplot(posteriorQuantiles[2,o,c]/t[i], col=col[i], las=2, legend=FALSE, border=NA,  name
segments(b, posteriorQuantiles[1,o,c]/t[i], b, posteriorQuantiles[2,o,c]/t[i], col="white")
segments(b, posteriorQuantiles[2,o,c]/t[i], b, posteriorQuantiles[3,o,c]/t[i], col=col[i])
rotatedLabel(b, labels=colnames(genotypesImputed)[o])
}
```

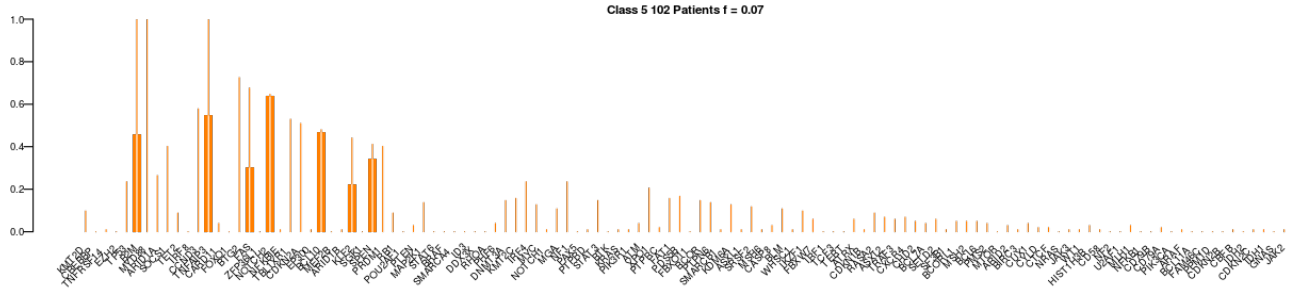


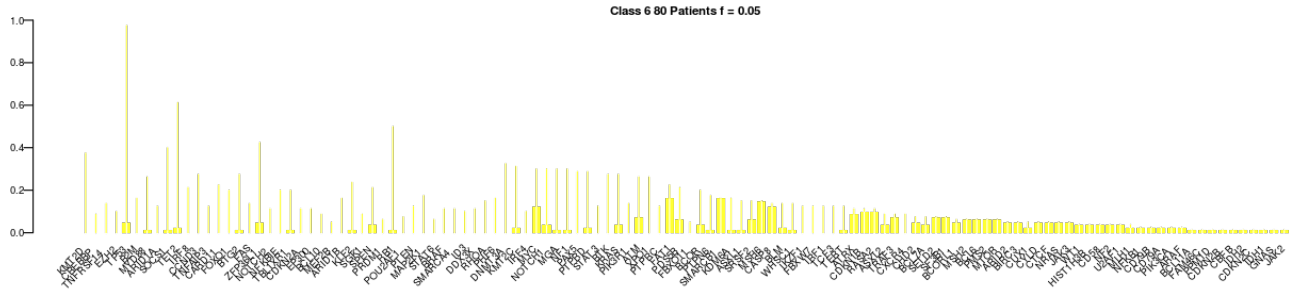


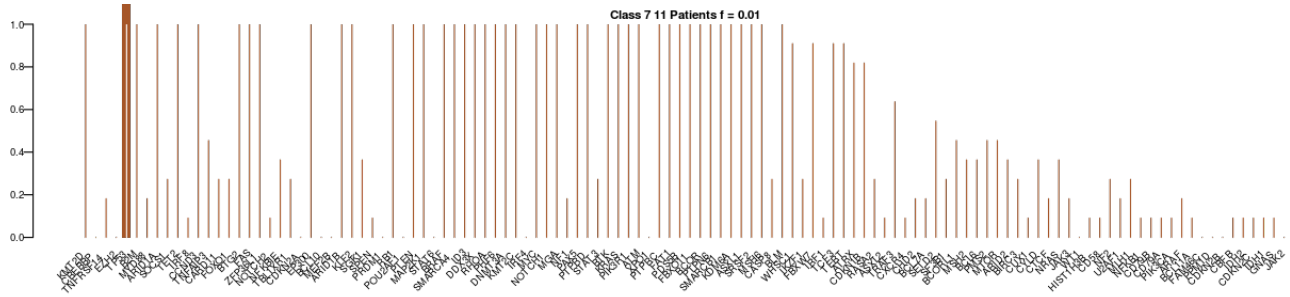






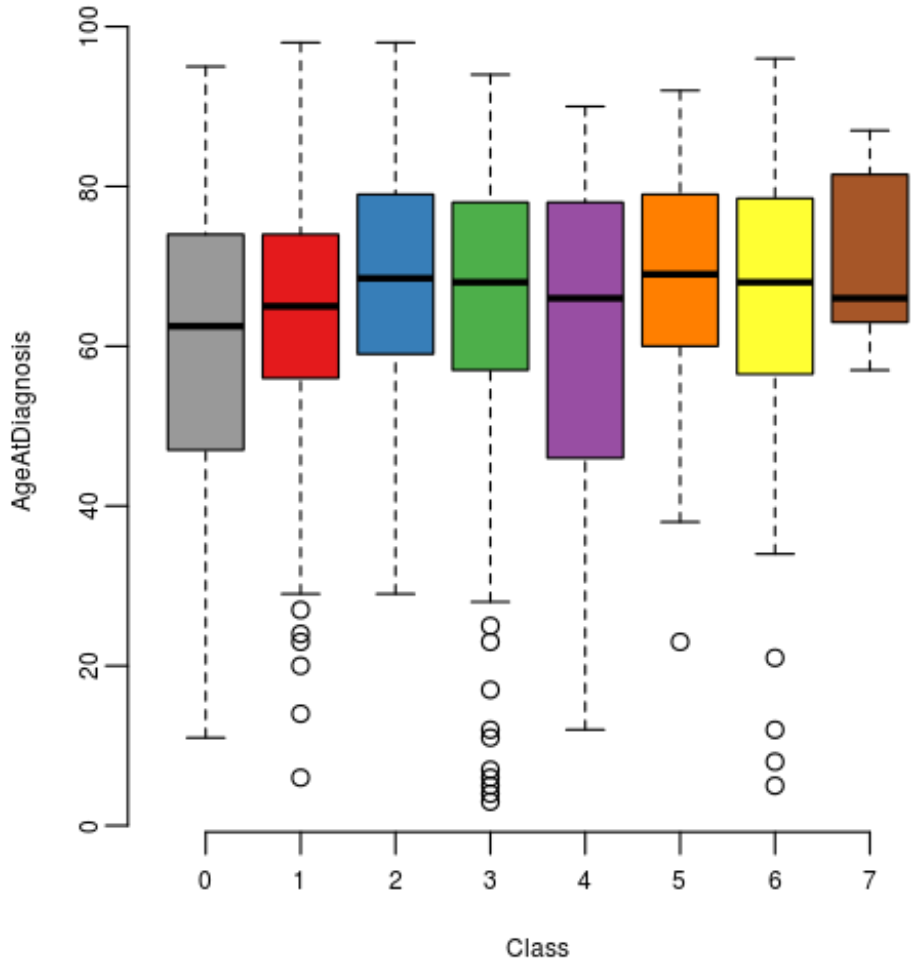




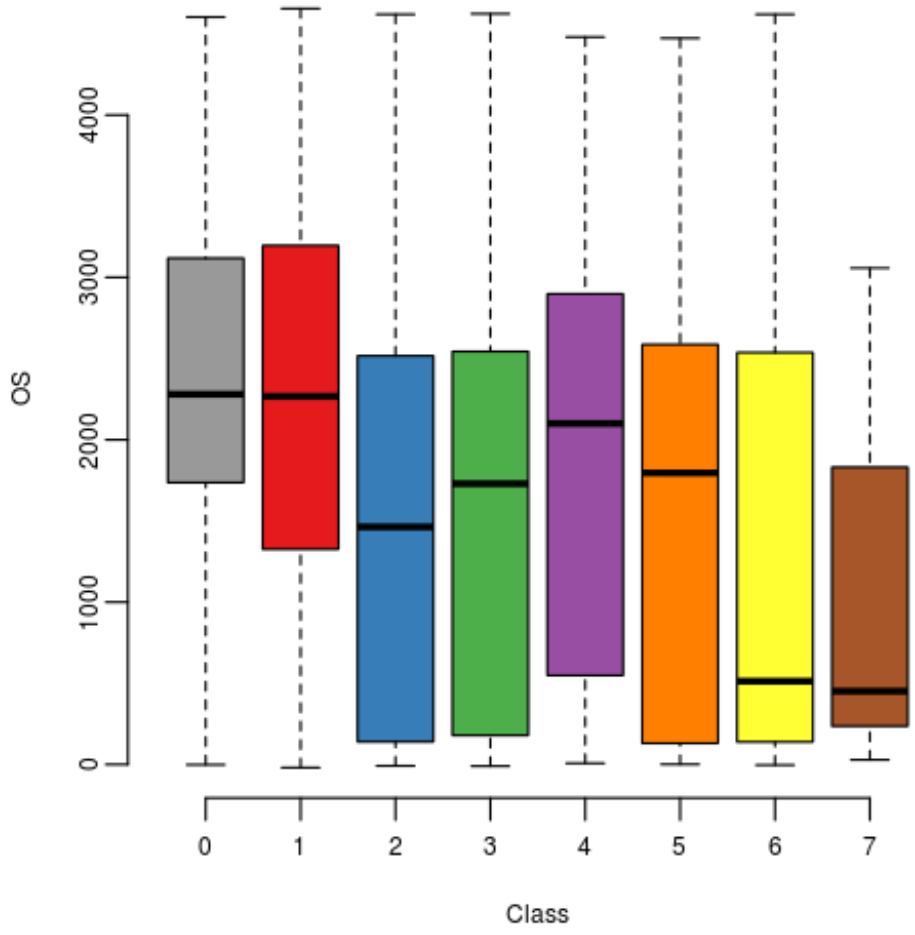


Clinical associations

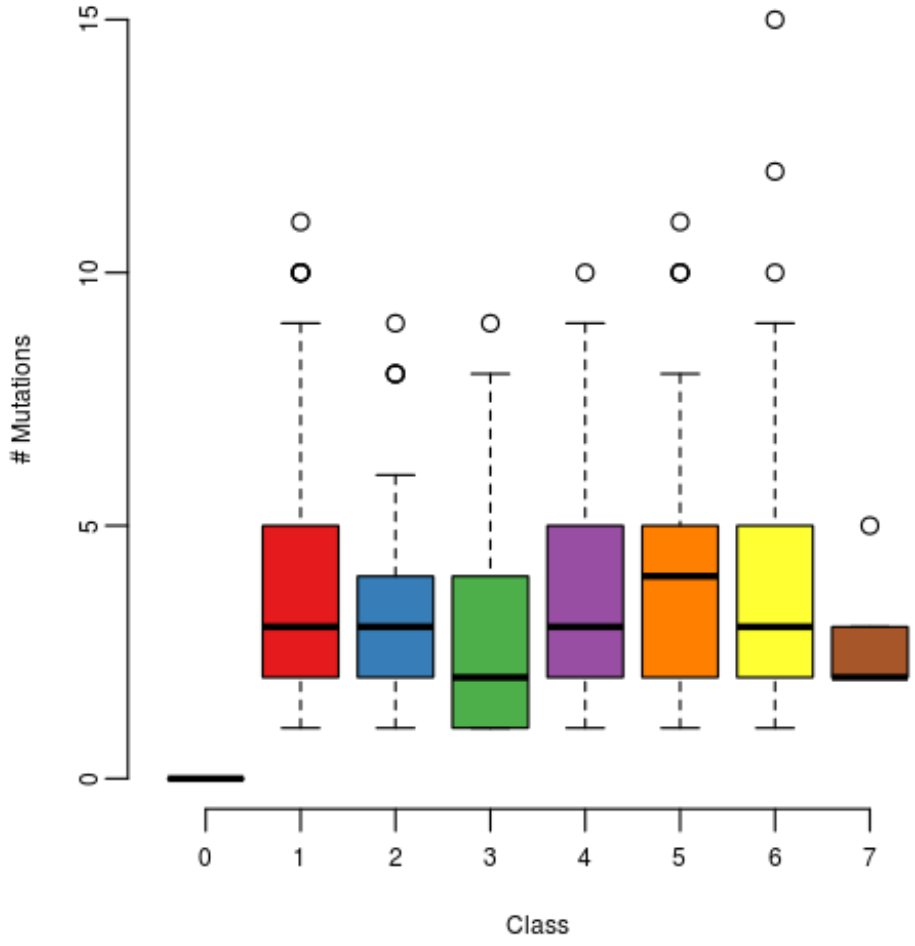
```
# Numerical
boxplot(clinical_information$AgeAtDiagnosis ~ factor(dpClass), xlab = "Class", ylab = "AgeAtD
```



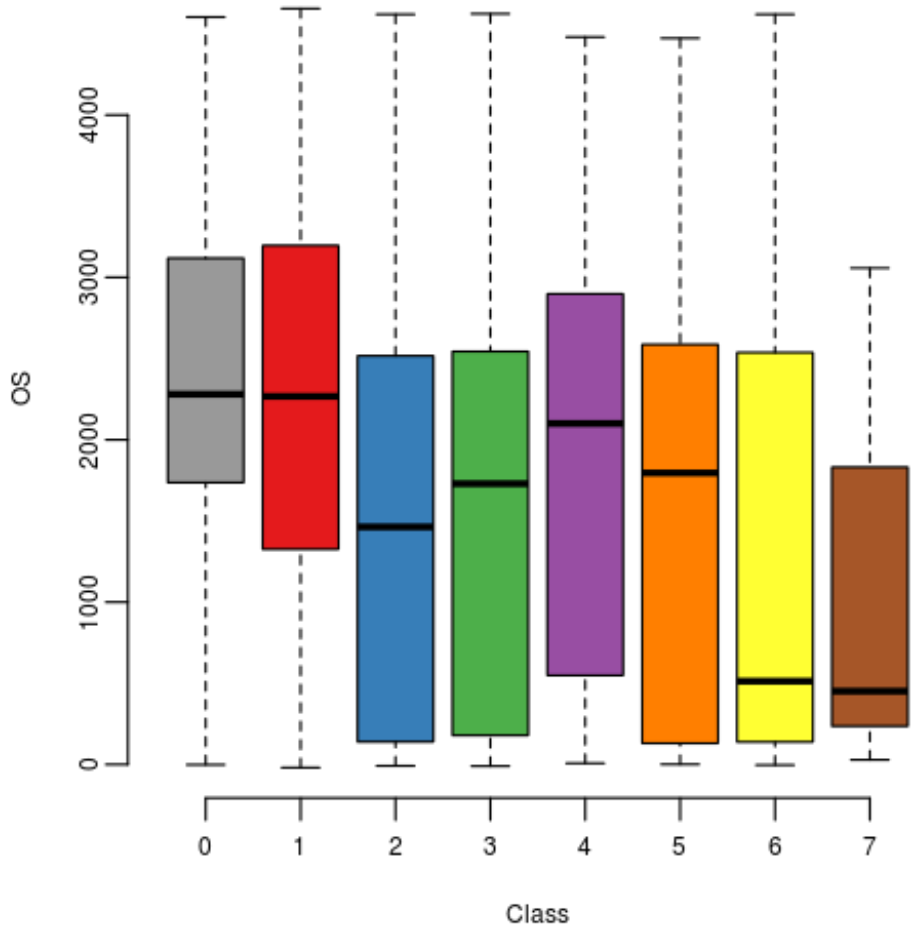

```
boxplot(clinical_information$OS ~ factor(dpClass), xlab = "Class", ylab = "OS", col=col)
```



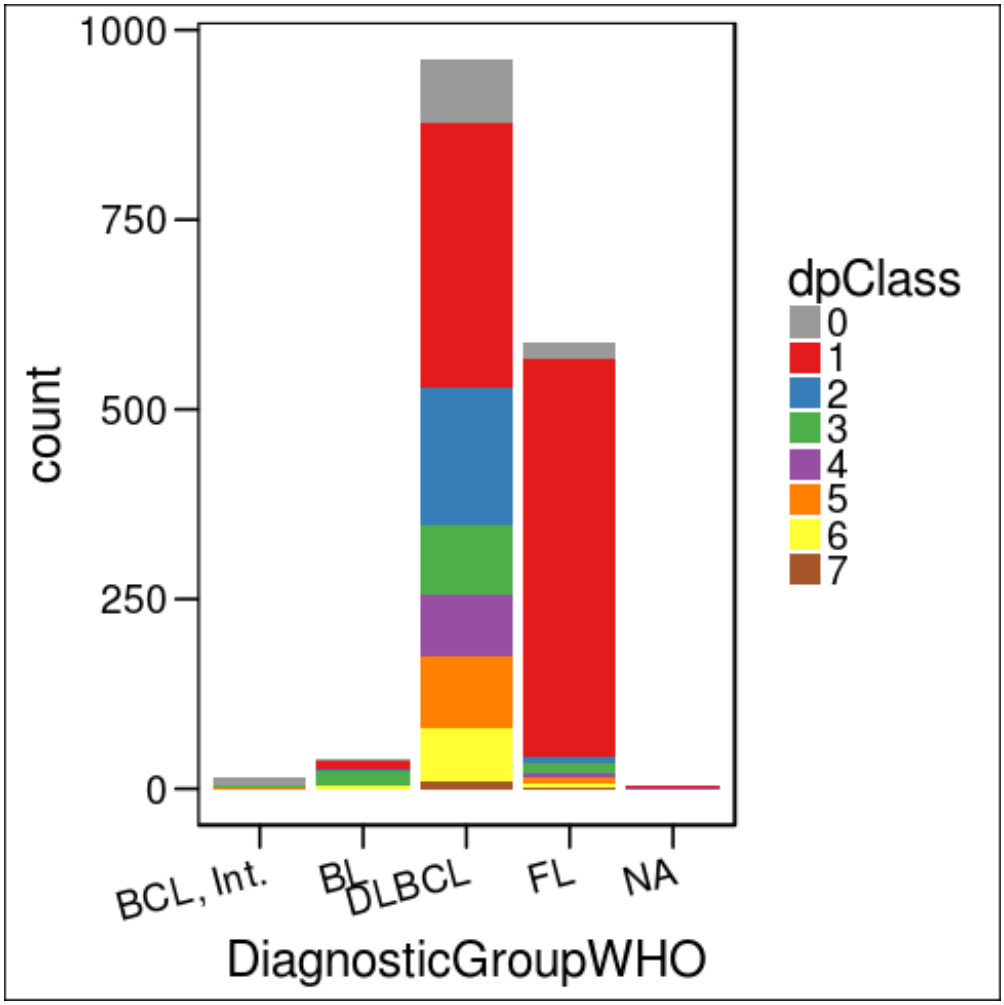

```
boxplot(rowSums(genotypesImputed) ~ factor(dpClass), xlab="Class",ylab="# Mutations", col=col
```



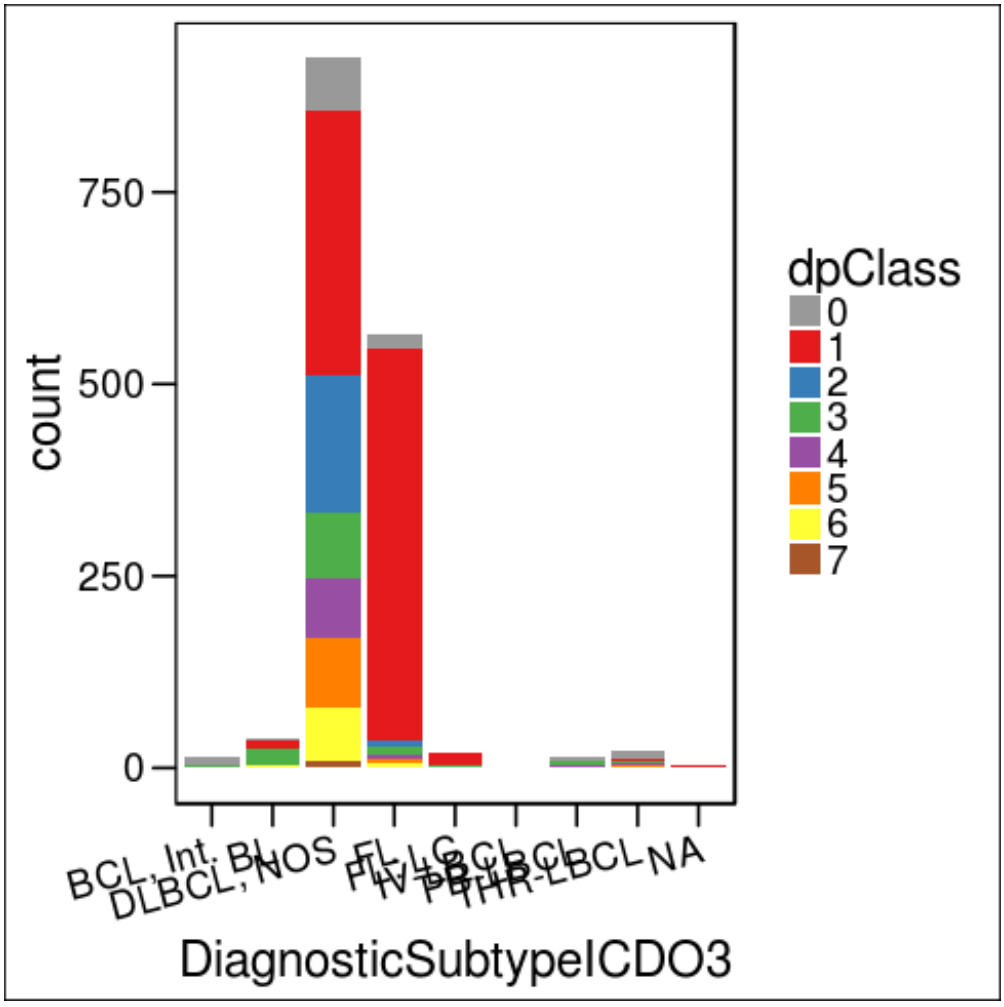

```
boxplot(clinical_information$OS ~ factor(dpClass), xlab = "Class", ylab = "OS", col=col)
```



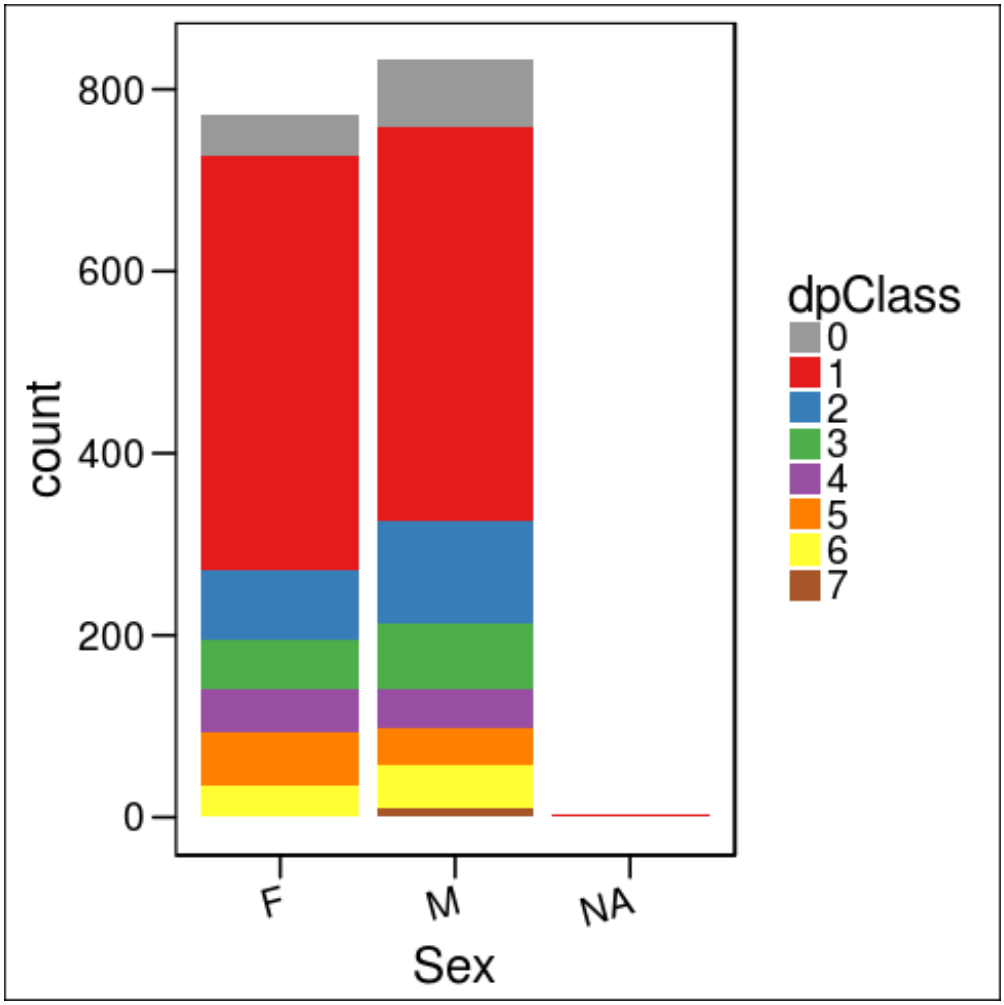

```
# Categorical
categorical_df <- cbind(clinical_information, dpClass = factor(dpClass))
# First, see results incorporating total counts
ggplot(categorical_df, aes(x = DiagnosticGroupWHO, fill = dpClass)) + geom_bar() + scale_fill
```



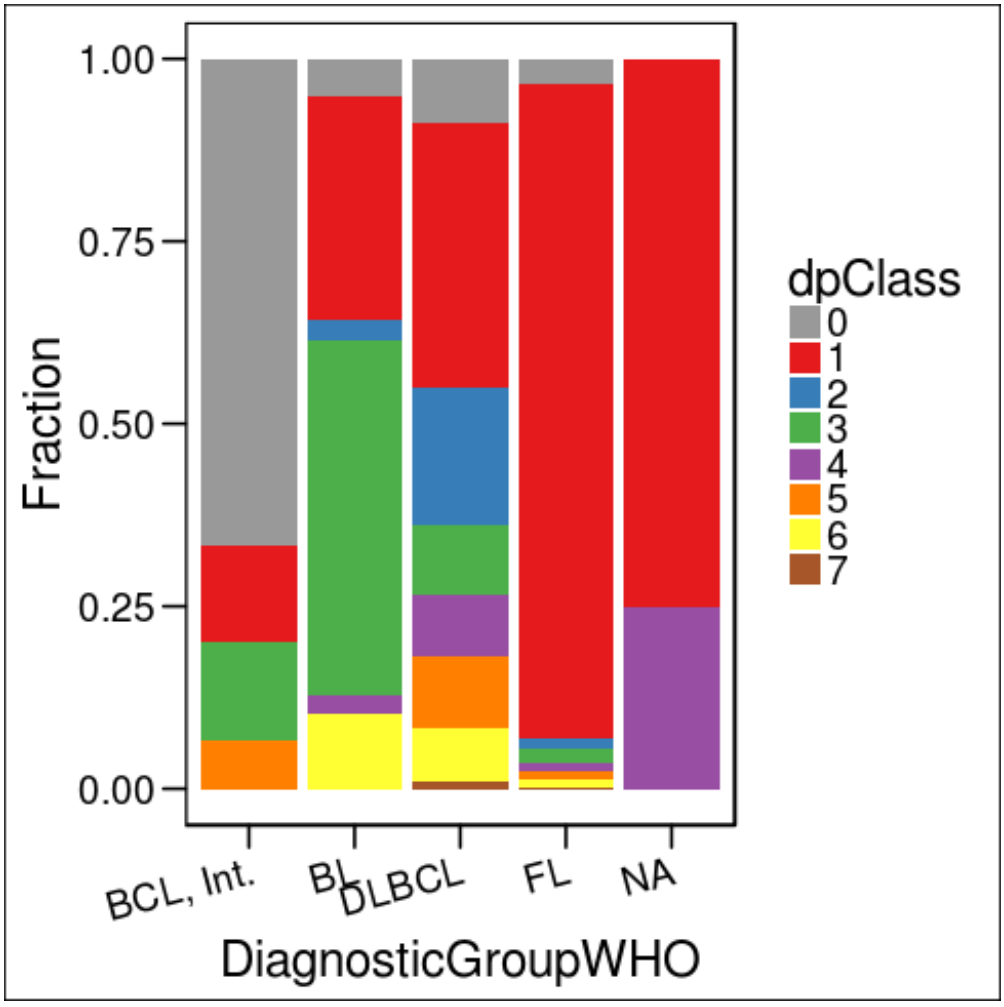

```
ggplot(categorical_df, aes(x = DiagnosticSubtypeICDO3, fill = dpClass)) + geom_bar() + scale_
```



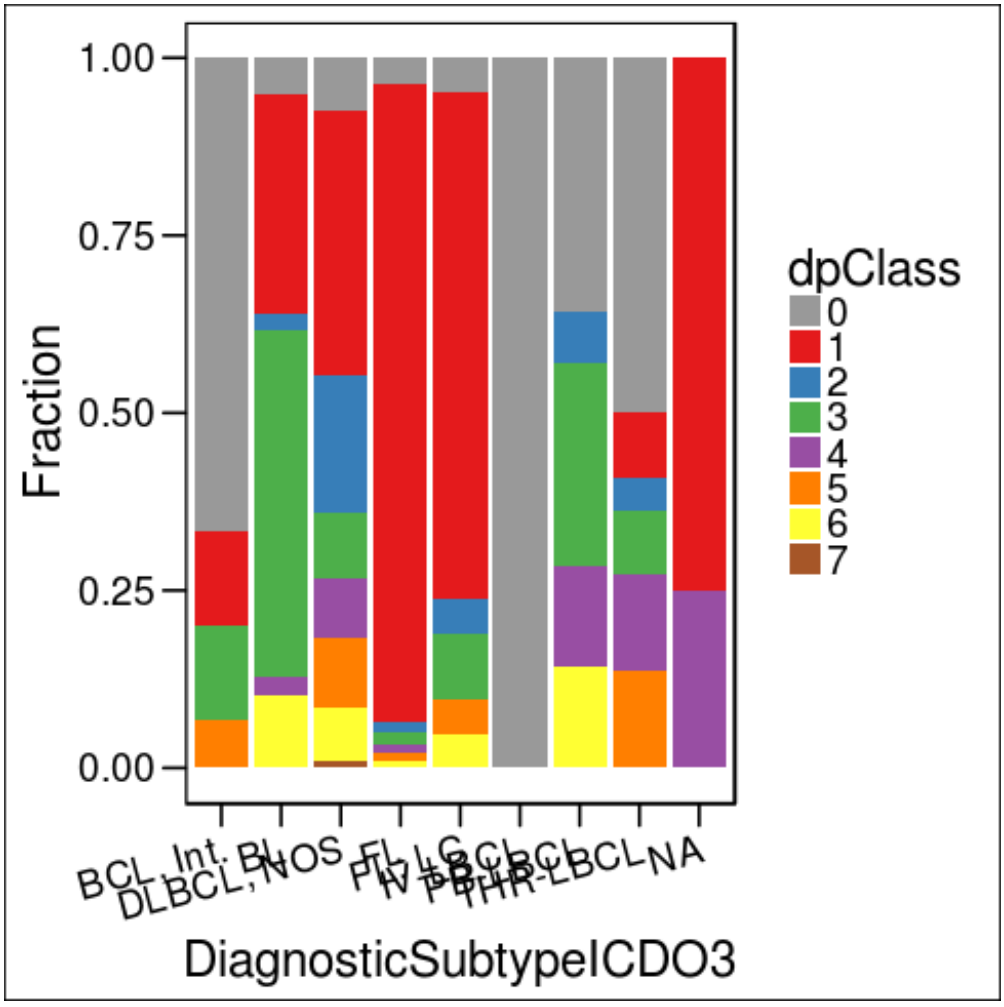

```
ggplot(categorical_df, aes(x = Sex, fill = dpClass)) + geom_bar() + scale_fill_manual(values
```



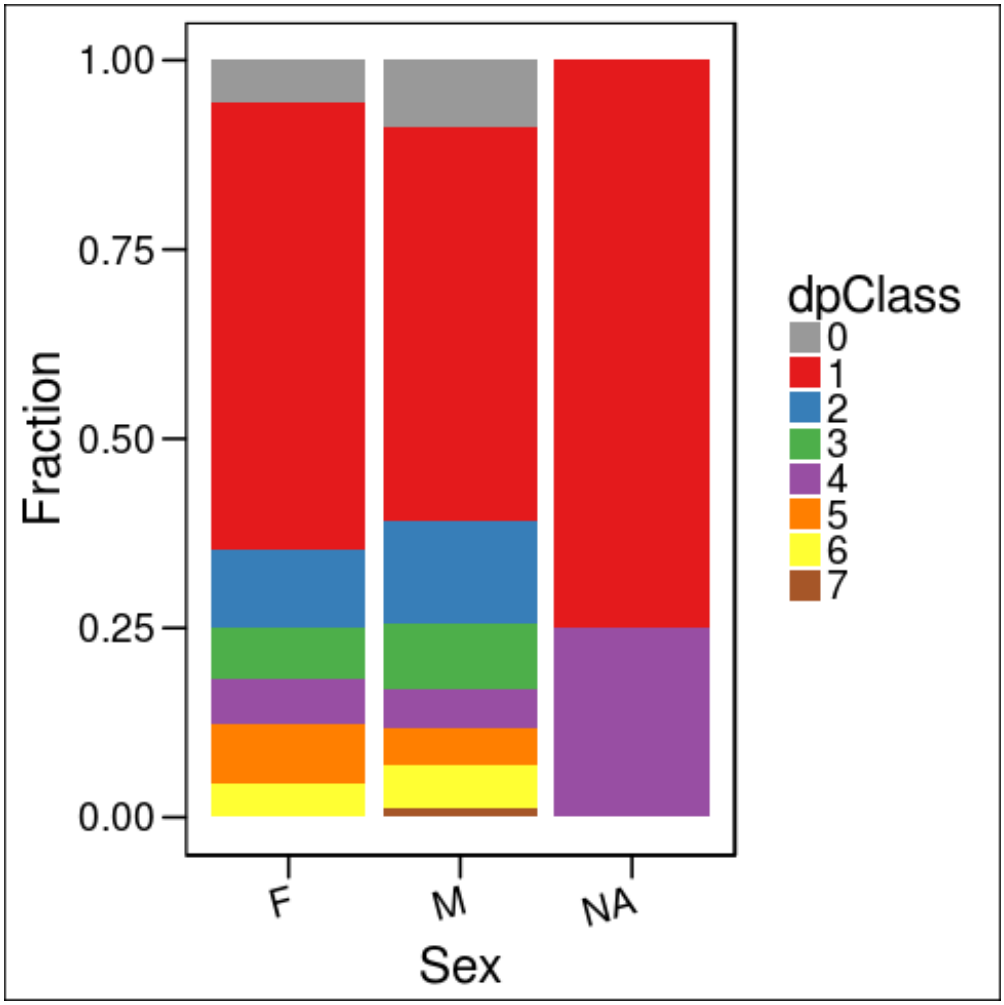

```
# Next, see results just looking at percentages
x_labels_rotation_angle <- 15
ggplot(categorical_df, aes(x = DiagnosticGroupWHO, fill = dpClass)) + geom_bar(position="fill
```




```
ggplot(categorical_df, aes(x = DiagnosticSubtypeICDO3, fill = dpClass)) + geom_bar(position="
```



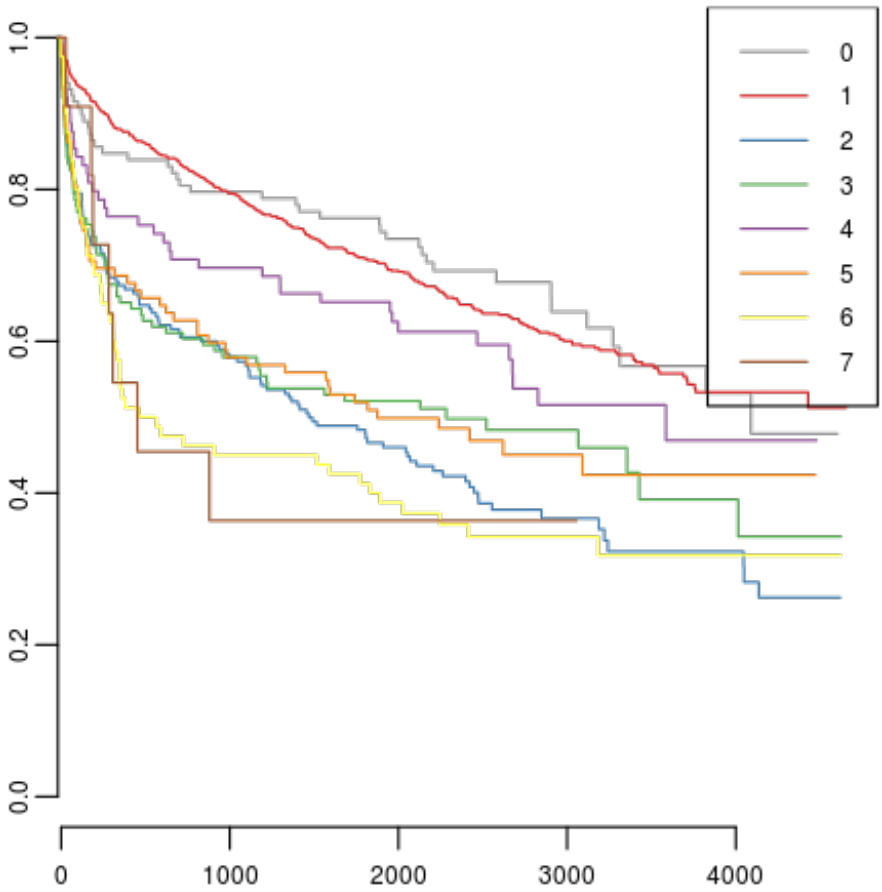

```
ggplot(categorical_df, aes(x = Sex, fill = dpClass)) + geom_bar(position="fill") + scale_fill
```



Survival

Simple coxph

```
# Actual survival code
os <- Surv(time = clinical_information$OS, event = clinical_information$SurvivalStatus)

plot(survfit(os ~ dpClass), col=col)
legend("topright", legend = levels(dpClass), col=col, lty=1)
```




```
kable(summary(survfit(os ~ dpClass))$table)
```

	records	n.max	n.start	events	*rmean	*se(rmean)	median	0.95LCL	0.95UCL
dpClass=0	118	118	118	43	3179.909	173.27050	4091.0	3271	NA
dpClass=1	887	887	887	346	3136.705	62.45958	NA	3756	NA
dpClass=2	190	190	190	121	2079.030	141.87390	1470.0	1108	2399
dpClass=3	126	126	126	67	2327.578	189.45547	2283.0	957	NA
dpClass=4	89	89	89	40	2789.888	215.21273	3585.0	2466	NA
dpClass=5	102	102	102	55	2378.981	207.44751	1876.0	982	NA
dpClass=6	80	80	80	53	1870.553	225.21687	513.5	318	2023
dpClass=7	11	11	11	7	1887.727	623.81602	452.0	283	NA

```
summary(coxph(os ~ dpClass))
```

```
## Call:
## coxph(formula = os ~ dpClass)
##
##   n= 1603, number of events= 732
##    (4 observations deleted due to missingness)
##
##             coef exp(coef) se(coef)     z Pr(>|z|)
## dpClass1 0.05994   1.06178  0.16171 0.371 0.710867
## dpClass2 0.82397   2.27952  0.17764 4.638 3.51e-06 ***
## dpClass3 0.64791   1.91154  0.19548 3.314 0.000918 ***
## dpClass4 0.31127   1.36515  0.21969 1.417 0.156533
## dpClass5 0.64184   1.89998  0.20364 3.152 0.001623 **
## dpClass6 0.94815   2.58094  0.20542 4.616 3.92e-06 ***
## dpClass7 1.01213   2.75146  0.40796 2.481 0.013102 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          exp(coef) exp(-coef) lower .95 upper .95
## dpClass1     1.062     0.9418    0.7734     1.458
## dpClass2     2.280     0.4387    1.6093     3.229
## dpClass3     1.912     0.5231    1.3031     2.804
## dpClass4     1.365     0.7325    0.8875     2.100
## dpClass5     1.900     0.5263    1.2747     2.832
## dpClass6     2.581     0.3875    1.7255     3.860
## dpClass7     2.751     0.3634    1.2368     6.121
##
## Concordance= 0.598  (se = 0.01 )
## Rsquare= 0.053   (max possible= 0.998 )
## Likelihood ratio test= 87.52  on 7 df,   p=4.441e-16
## Wald test            = 93.49  on 7 df,   p=0
## Score (logrank) test = 97.93  on 7 df,   p=0
```

#' Risk variance #+ RFX, cache=TRUE library(CoxHD) dataFrameOsTD <- dataFrame[tplSplitOs,] dataFrameOsTD[which(tplIndexOs), grep("TPL", colnames(dataFrameOsTD), value=TRUE)] <- 0 ## Set pre-tpl variables to zero mainGroups <- grep("[A-Z][a-z]+[A-Z]",levels(groups), invert=TRUE, value=TRUE) mainIdx <- groups %in% mainGroups osTDIdx <- !grepl("TPL_efs", colnames(dataFrame)) mainIdxOsTD <- mainIdx & osTDIdx whichRFXOsTDGG <- which((colSums(dataFrame)>=8 | mainIdxOsTD) & osTDIdx & groups %in% c(mainGroups,"GeneGene")) # ie, > 0.5%

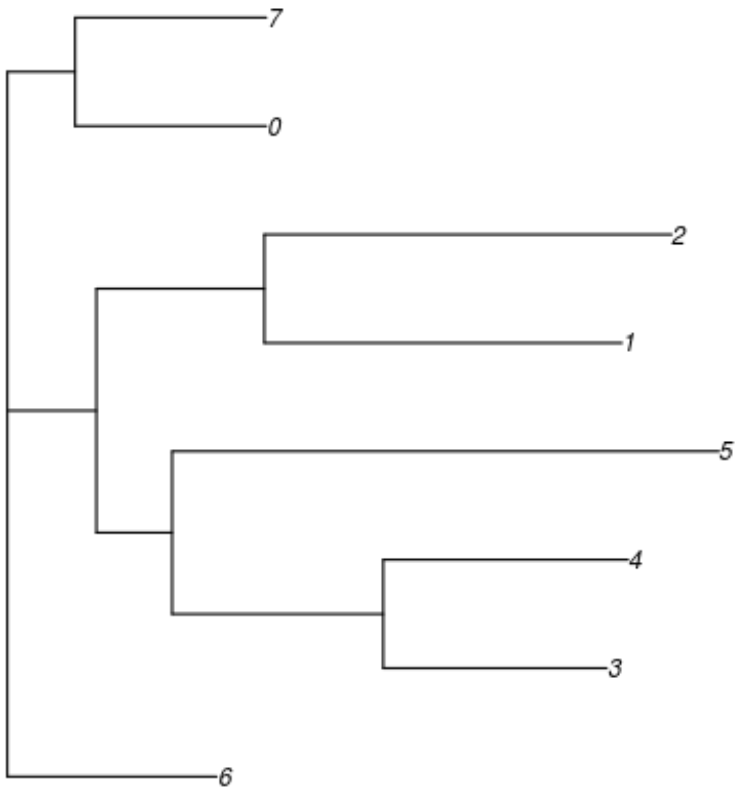
coxRFXFitOsTDGGc <- CoxRFX(dataFrameOsTD[,whichRFXOsTDGG], osTD, groups[whichRFXOsTDGG], which.mu=mainGroups) ## allow only the main groups to have mean different from zero.. coxRFXFitOsTDGGc

d <- cbind(dataFrameOsTD[,whichRFXOsTDGG],DP=t(posteriorProbability)[tplSplitOs,-1]) coxRFXFitOsTDGGcDP <- CoxRFX(d, osTD, c(as.character(groups[whichRFXOsTDGG]),rep("DP", nlevels(dpClass)-1)), which.mu=mainGroups) ## allow only the main groups to have mean different from zero.. coxRFXFitOsTDGGcDP

PlotVarianceComponents(coxRFXFitOsTDGGcDP, col=col) round(cov(PartialRisk(coxRFXFitOsTDGGcDP)),2)

Phylogeny

```
library(ape)
plot(nj(dist(t(posteriorMeans/(rep(rowSums(posteriorProbability), each=nrow(posteriorMeans)))
```

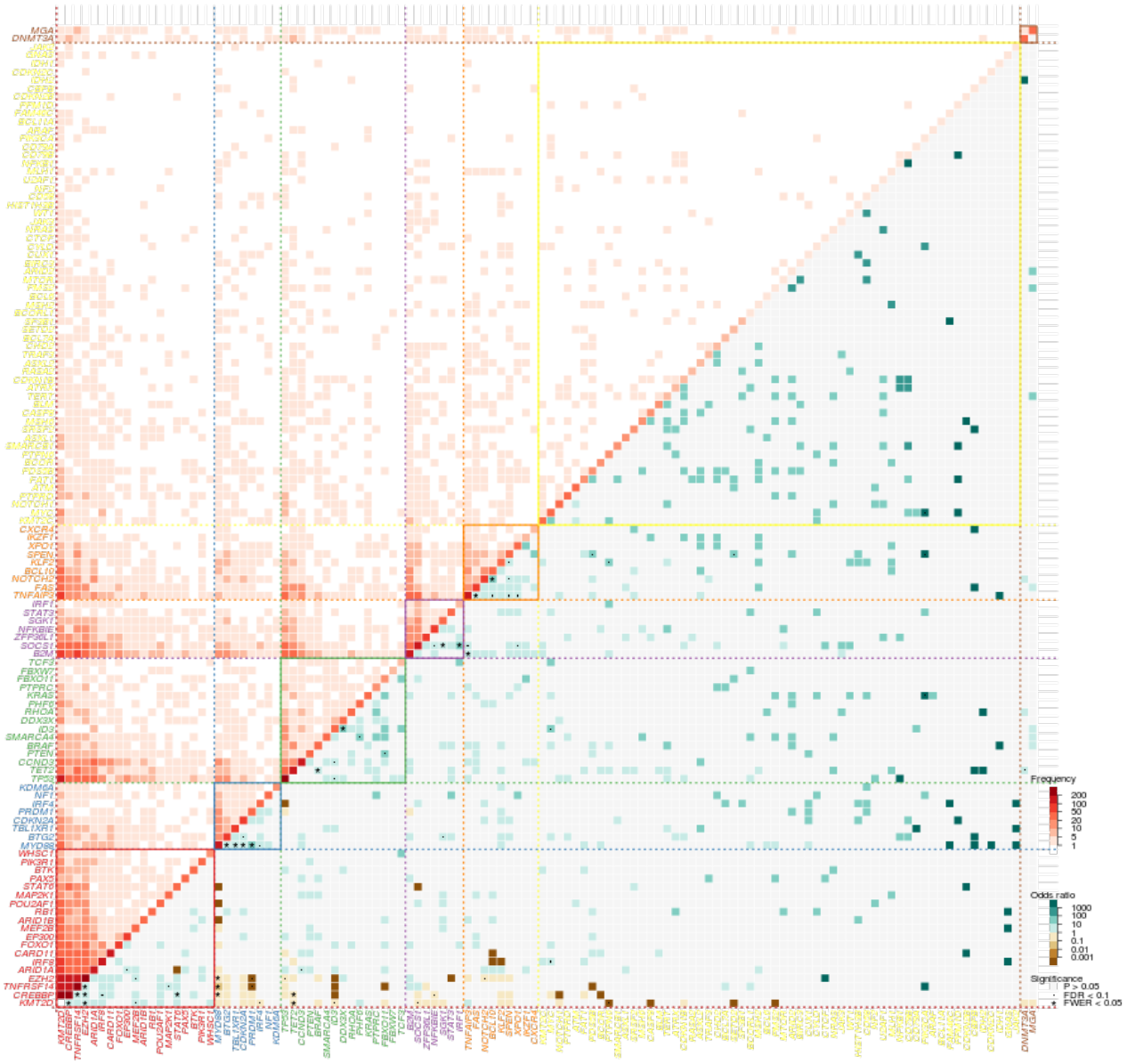


Gene:Gene interactions

Population based

```
geneToClass <- factor(apply(posteriorMeans, 1,which.max) -1, levels = as.numeric(colnames(pos
getOdds <- function(x) {
  f <- sapply(1:ncol(x),
              function(i) sapply(1:ncol(x),
                                 function(j) {
                                   if(j <= i) return(c(NA,NA))
                                   f<- try(fisher.test(x[,i], x[,j]), silent=TRUE)
                                   if(class(f)=="try-error") c(0,NA)
                                   else if(f$estimate>1) c(-log10(f$p.val),f$estimate)
                                   else c(log10(f$p.val), f$estimate)}
              ),
              simplify="array")
  for(i in 1:2)
    f[i,,][upper.tri(f[i,,])] <- t(f[i,,])[upper.tri(f[i,,])]
  return(f)
}
f <- getOdds(genotypesImputed)
logPInt <- f[1,,]
odds <- f[2,,]
pairs <- sapply(1:ncol(genotypesImputed), function(i) colMeans(genotypesImputed * genotypesIm
diag(logPInt) <- 0
diag(odds) <- 1
colnames(odds) <- rownames(odds) <- colnames(logPInt) <- rownames(logPInt) <- colnames(genoty
odds[odds<1e-3] = 1e-4
odds[odds>1e3] = 1e4
```

```
odds[10^-abs(logPInt) > 0.1] = 1
logOdds=log10(odds)
diag(logPInt) <- NA
par(bty="n", mgp = c(2,.5,0), mar=c(4,4,4,4)+.1, las=2, tcl=-.33)
ix = TRUE#colnames(interactions) %in% colnames(all_genotypes)
o = order(geneToClass, -colSums(genotypesImputed))
M <-  matrix( NA, ncol=ncol(odds), nrow=nrow(odds))
M[lower.tri(M)] <- cut(logOdds[o,o][lower.tri(M)], breaks = c(-4:0-.Machine$double.eps,0:4), i
M[upper.tri(M, diag=TRUE)] <- as.numeric(cut(pairs[o,o][upper.tri(M, diag=TRUE)]*nrow(genotype
image(x=1:ncol(logPInt), y=1:nrow(logPInt), M, col=c(brewer.pal(9,"BrBG"), c("white",brewer.pa
l <- colnames(logPInt)[o]
mtext(side=1, at=1:ncol(logPInt), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=col[gene
mtext(side=2, at=1:ncol(logPInt), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=col[gene
abline(h=0:ncol(logPInt)+.5, col="white", lwd=.5)
abline(v=0:ncol(logPInt)+.5, col="white", lwd=.5)
P <- 10^-abs(logPInt[o,o])
P[upper.tri(P)] <- NA
w = arrayInd(which(p.adjust(P, method="BH") < .1), rep(nrow(logPInt),2))
points(w, pch=".", col="black")
w = arrayInd(which(p.adjust(P) < .05), rep(nrow(logPInt),2))
points(w, pch="*", col="black")
image(y = 1:9 +18, x=rep(ncol(logPInt),2)+c(2,3), z=matrix(c(1:8), nrow=1), col=c("white",brew
axis(side = 4, at = seq(1,7) + 19, cex.axis=.66, tcl=-.15, label=c(1,5,10,20,50,100,200), las=
mtext(side=4, at=28, "Frequency", las=2, line=-1,cex=.66)
image(y = 1:8 +5, x=rep(ncol(logPInt),2)+c(2,3), z=matrix(c(1:8), nrow=1), col=brewer.pal(8,"B
axis(side = 4, at = seq(1,7) + 5.5, cex.axis=.66, tcl=-.15, label=10^seq(-3,3), las=1, lwd=.5)
mtext(side=4, at=14, "Odds ratio", las=2, line=-1,cex=.66)
mtext(side=4, at=4, "Significance", las=2, line=-1,cex=.66)
points(x=rep(ncol(logPInt),2)+2.5, y=1:2, pch=c("*","."))
image(x=rep(ncol(logPInt),2)+c(2,3), y=(2:3) +0.5, z=matrix(1), col=brewer.pal(3,"BrBG"), add=
mtext(side=4, at=3:1, c("P > 0.05", "FDR < 0.1", "FWER < 0.05"), cex=.66, line=0.2)
t <- c(0,table(geneToClass))
s <- cumsum(t)+.5
abline(h=s[-length(s)], col=col, lty=3)
abline(v=s[-length(s)], col=col, lty=3)
rect(s[-1],s[-1], s[-length(s)], s[-length(s)], border=col)
```



Expected heatmap

```
set.seed(42)
t <- table(dpClass)
pp <- t(t(posteriorMeans)/as.numeric(t))
expectedOdds <- sapply(colnames(genotypesImputed), function(j){
  sapply(colnames(genotypesImputed), function(i){
    if(i==j) return(0)
    P <- Reduce("+",lapply(seq_along(t), function(k) {t[k] * (pp[i,k] * c(1,-1) + c(0,1)) %o%
    #res <- round(log10(M[1,1]*M[2,2]/M[1,2]/M[2,1]))
    #if( sum(M[,1]) * sum(M[1,])/sum(M) < 5 & res < 0) res <- 0
    #return(res)
    M <- matrix(rmultinom(1,sum(t), P), ncol=2)
    f <- fisher.test(round(M))
    res <- pmin(pmax(round(log10(f$estimate)),-4),4)
    if(f$p.value > 0.05)
      res <- 0
    return(res)
  })
})
```

```
## Error in rmultinom(1, sum(t), P): negative probability
```

```
par(bty="n", mgp = c(2,.5,0), mar=c(4,4,4,4)+.1, las=2, tcl=-.33)
o = order(geneToClass, -colSums(genotypesImputed))
image(x=1:ncol(expectedOdds), y=1:nrow(expectedOdds), expectedOdds[o,o], col=brewer.pal(9,"Br
```

```
## Error in ncol(expectedOdds): object 'expectedOdds' not found
```

```
l <- colnames(expectedOdds)[o]
```

```
## Error in is.data.frame(x): object 'expectedOdds' not found
```

```
mtext(side=1, at=1:ncol(expectedOdds), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=co
```

```
## Error in ncol(expectedOdds): object 'expectedOdds' not found
```

```
mtext(side=2, at=1:ncol(expectedOdds), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=co
```

```
## Error in ncol(expectedOdds): object 'expectedOdds' not found
```

```
abline(h=0:ncol(expectedOdds)+.5, col="white", lwd=.5)
```

```
## Error in ncol(expectedOdds): object 'expectedOdds' not found
```

```
abline(v=0:ncol(expectedOdds)+.5, col="white", lwd=.5)
```

```
## Error in ncol(expectedOdds): object 'expectedOdds' not found
```

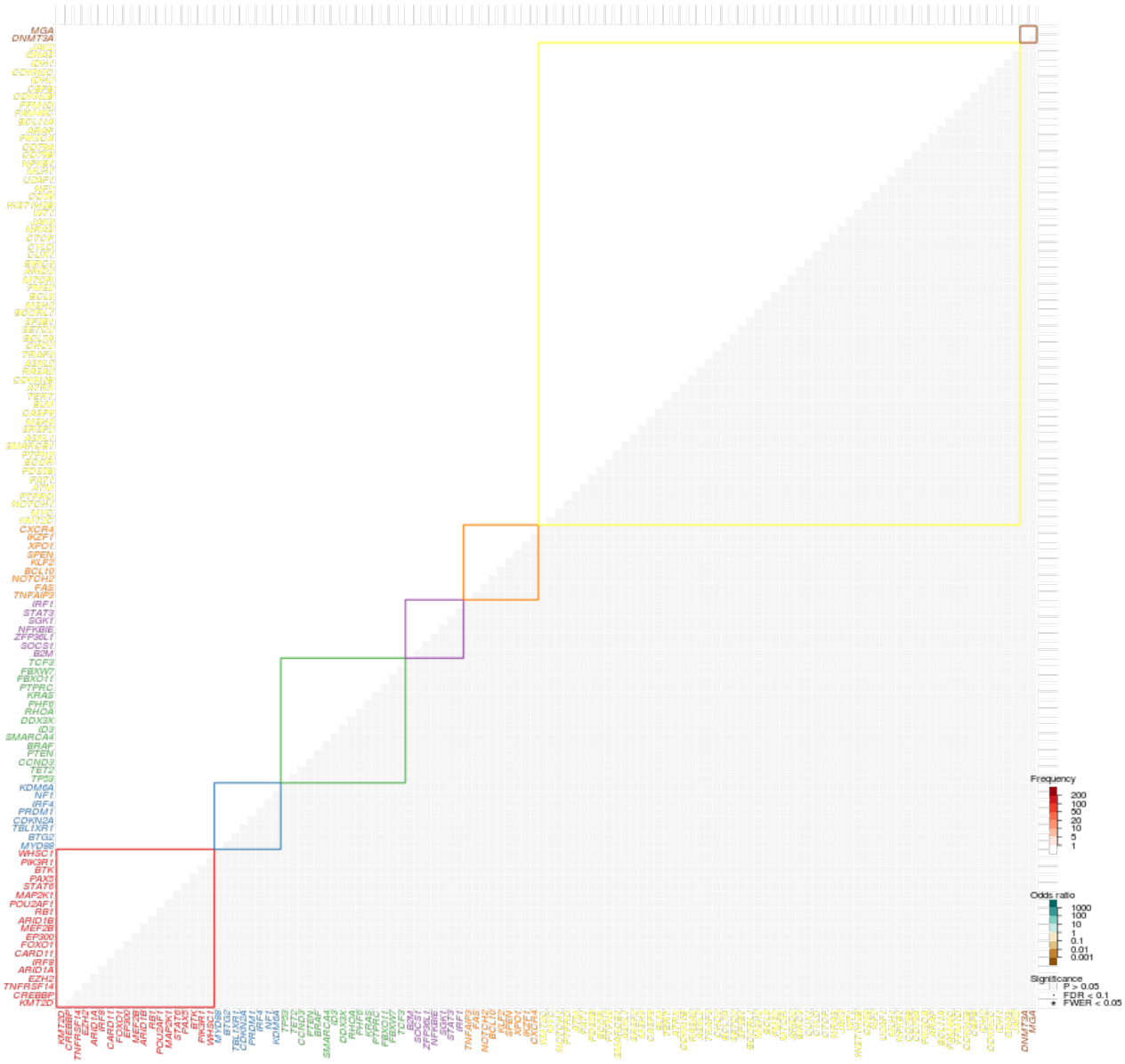
```
t <- c(0,table(geneToClass))
s <- cumsum(t)+.5
rect(s[-1],s[-1], s[-length(s)], s[-length(s)], border=col)
```

```
## Error in rect(s[-1], s[-1], s[-length(s)], s[-length(s)], border = col): plot.new has not
```

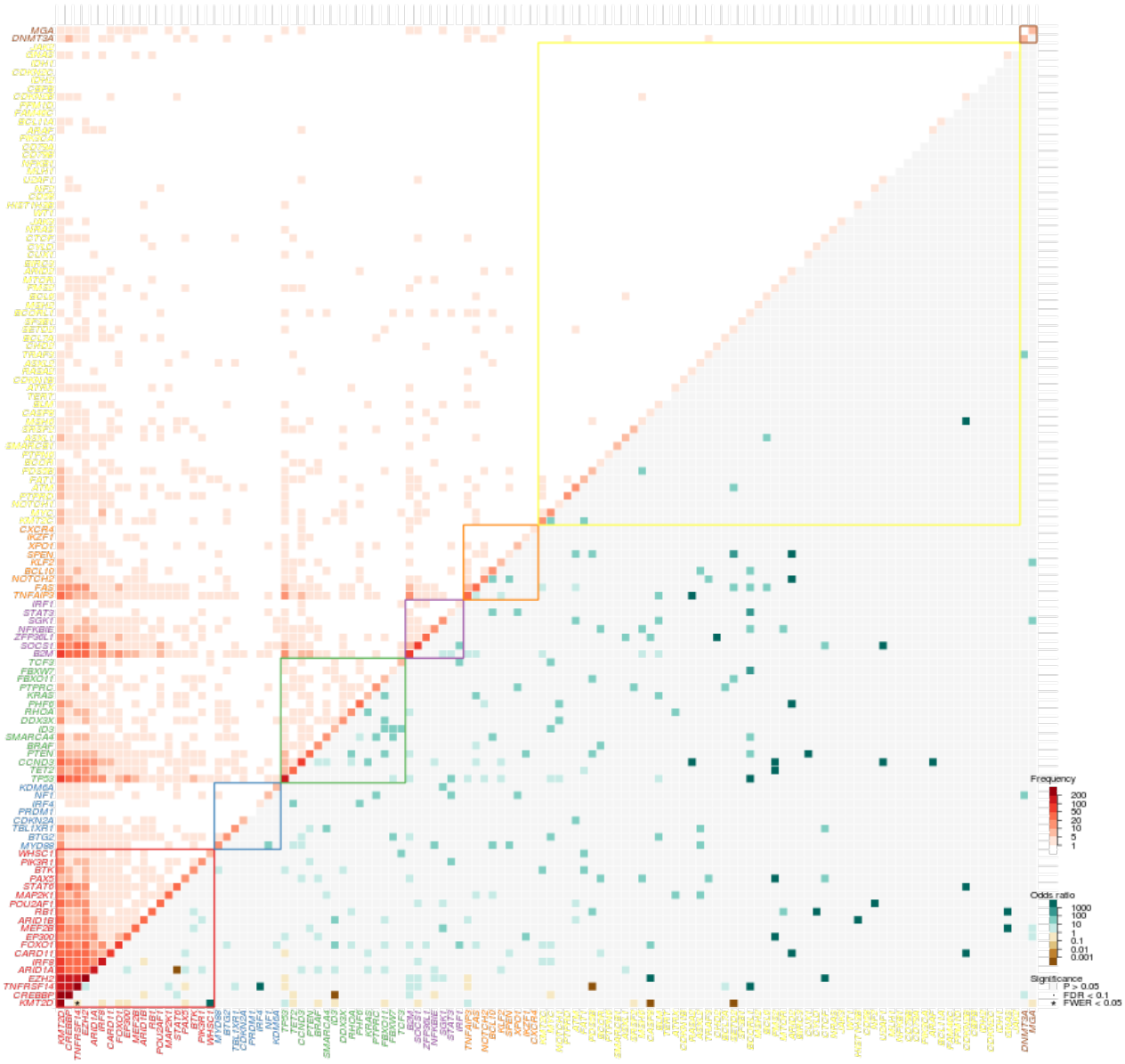
Per class

```r
for(cls in levels(dpClass)){
  w <- dpClass == cls
  f <- getOdds(genotypesImputed[w,])
  logPInt <- f[1,,]
  odds <- f[2,,]
  pairs <- sapply(1:ncol(genotypesImputed), function(i) colMeans(genotypesImputed[w,] * genot
  diag(logPInt) <- 0
  diag(odds) <- 1
  colnames(odds) <- rownames(odds) <- colnames(logPInt) <- rownames(logPInt) <- colnames(geno
  odds[odds<1e-3] = 1e-4
  odds[odds>1e3] = 1e4

  odds[10^-abs(logPInt) > 0.1] = 1
  logOdds=log10(odds)
  diag(logPInt) <- NA
  par(bty="n", mgp = c(2,.5,0), mar=c(4,4,4,4)+.1, las=2, tcl=-.33)
  ix = TRUE#colnames(interactions) %in% colnames(all_genotypes)
  o = order(geneToClass, -colSums(genotypesImputed))
  M <-  matrix( NA, ncol=ncol(odds), nrow=nrow(odds))
  M[lower.tri(M)] <- cut(logOdds[o,o][lower.tri(M)], breaks = c(-4:0-.Machine$double.eps,0:4),
  M[upper.tri(M, diag=TRUE)] <- as.numeric(cut(pairs[o,o][upper.tri(M, diag=TRUE)]*nrow(genoty
  image(x=1:ncol(logPInt), y=1:nrow(logPInt), M, col=c(brewer.pal(9,"BrBG"), c("white",brewer
  l <- colnames(logPInt)[o]
  mtext(side=1, at=1:ncol(logPInt), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=col[ge
  mtext(side=2, at=1:ncol(logPInt), l, cex=.66, font=ifelse(grepl("^[A-Z]",l),3,1), col=col[ge
  abline(h=0:ncol(logPInt)+.5, col="white", lwd=.5)
  abline(v=0:ncol(logPInt)+.5, col="white", lwd=.5)
  P <- 10^-abs(logPInt[o,o])
  P[upper.tri(P)] <- NA
  w = arrayInd(which(p.adjust(P, method="BH") < .1), rep(nrow(logPInt),2))
  points(w, pch=".", col="black")
  w = arrayInd(which(p.adjust(P) < .05), rep(nrow(logPInt),2))
  points(w, pch="*", col="black")
  image(y = 1:9 +18, x=rep(ncol(logPInt),2)+c(2,3), z=matrix(c(1:8), nrow=1), col=c("white",br
  axis(side = 4, at = seq(1,7) + 19, cex.axis=.66, tcl=-.15, label=c(1,5,10,20,50,100,200), la
  mtext(side=4, at=28, "Frequency", las=2, line=-1,cex=.66)
  image(y = 1:8 +5, x=rep(ncol(logPInt),2)+c(2,3), z=matrix(c(1:8), nrow=1), col=brewer.pal(8,
  axis(side = 4, at = seq(1,7) + 5.5, cex.axis=.66, tcl=-.15, label=10^seq(-3,3), las=1, lwd=.
  mtext(side=4, at=14, "Odds ratio", las=2, line=-1,cex=.66)
  mtext(side=4, at=4, "Significance", las=2, line=-1,cex=.66)
  points(x=rep(ncol(logPInt),2)+2.5, y=1:2, pch=c("*","."))
  image(x=rep(ncol(logPInt),2)+c(2,3), y=(2:3) +0.5, z=matrix(1), col=brewer.pal(3,"BrBG"), ad
  mtext(side=4, at=3:1, c("P > 0.05", "FDR < 0.1", "FWER < 0.05"), cex=.66, line=0.2)
  t <- c(0,table(geneToClass))
  s <- cumsum(t)+.5
  rect(s[-1],s[-1], s[-length(s)], s[-length(s)], border=col)
  title(main=paste0("Class ",cls,": ", genes[cls]))
}
```

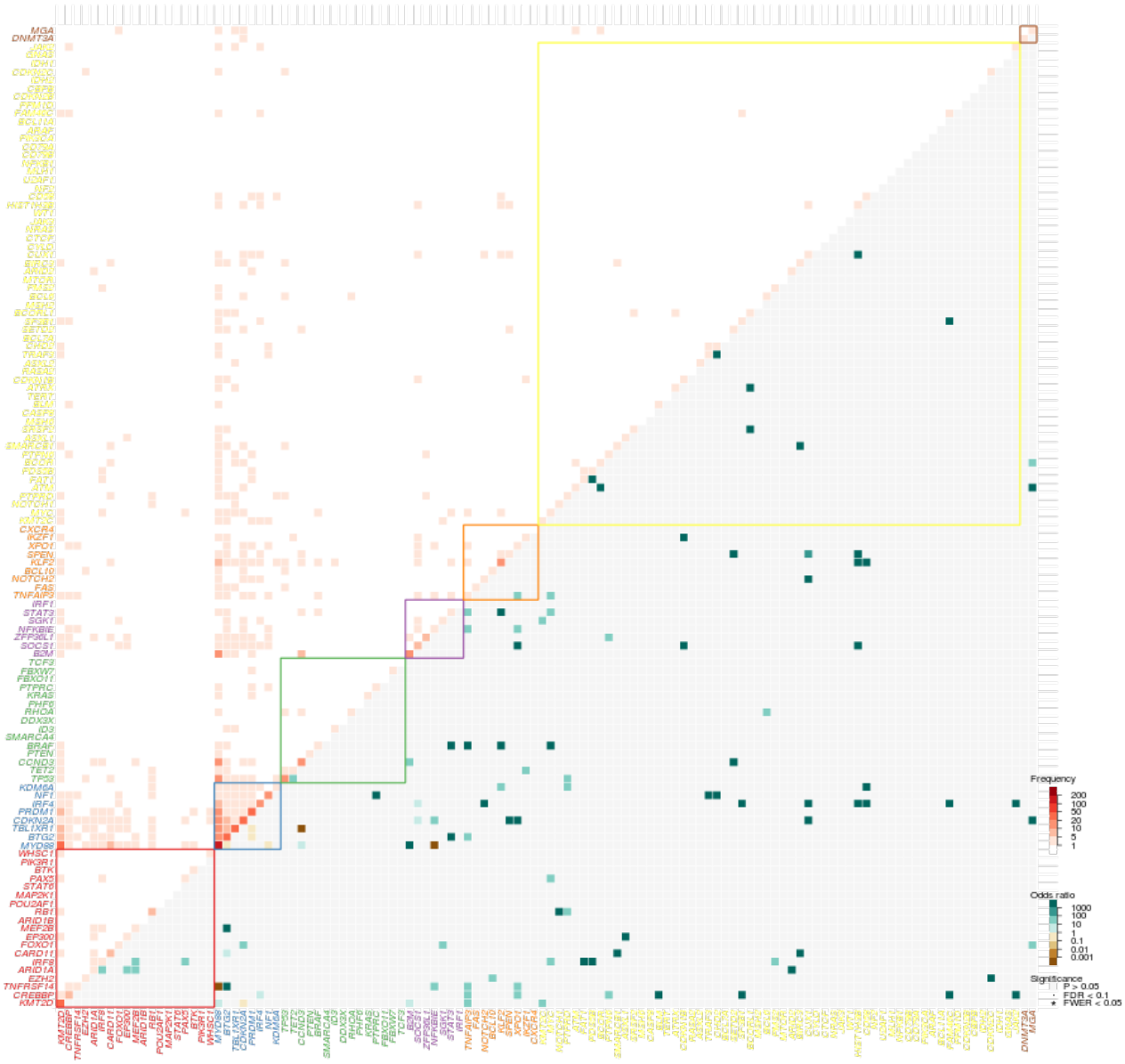
Class 0: TET2;TP53



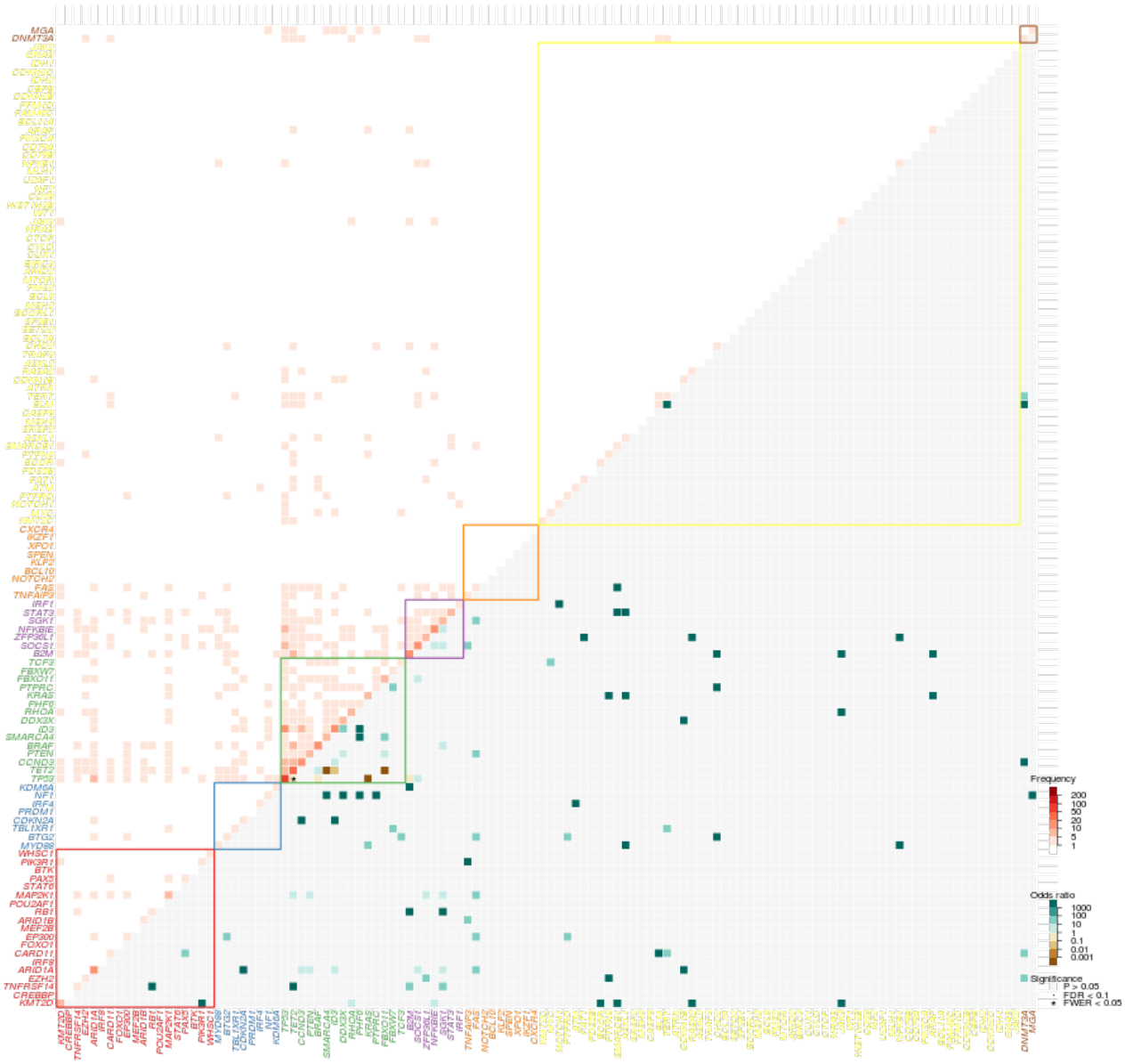
Class 1: KMT2D;CREBBP;TNFRSF14;EZH2;ARID1A



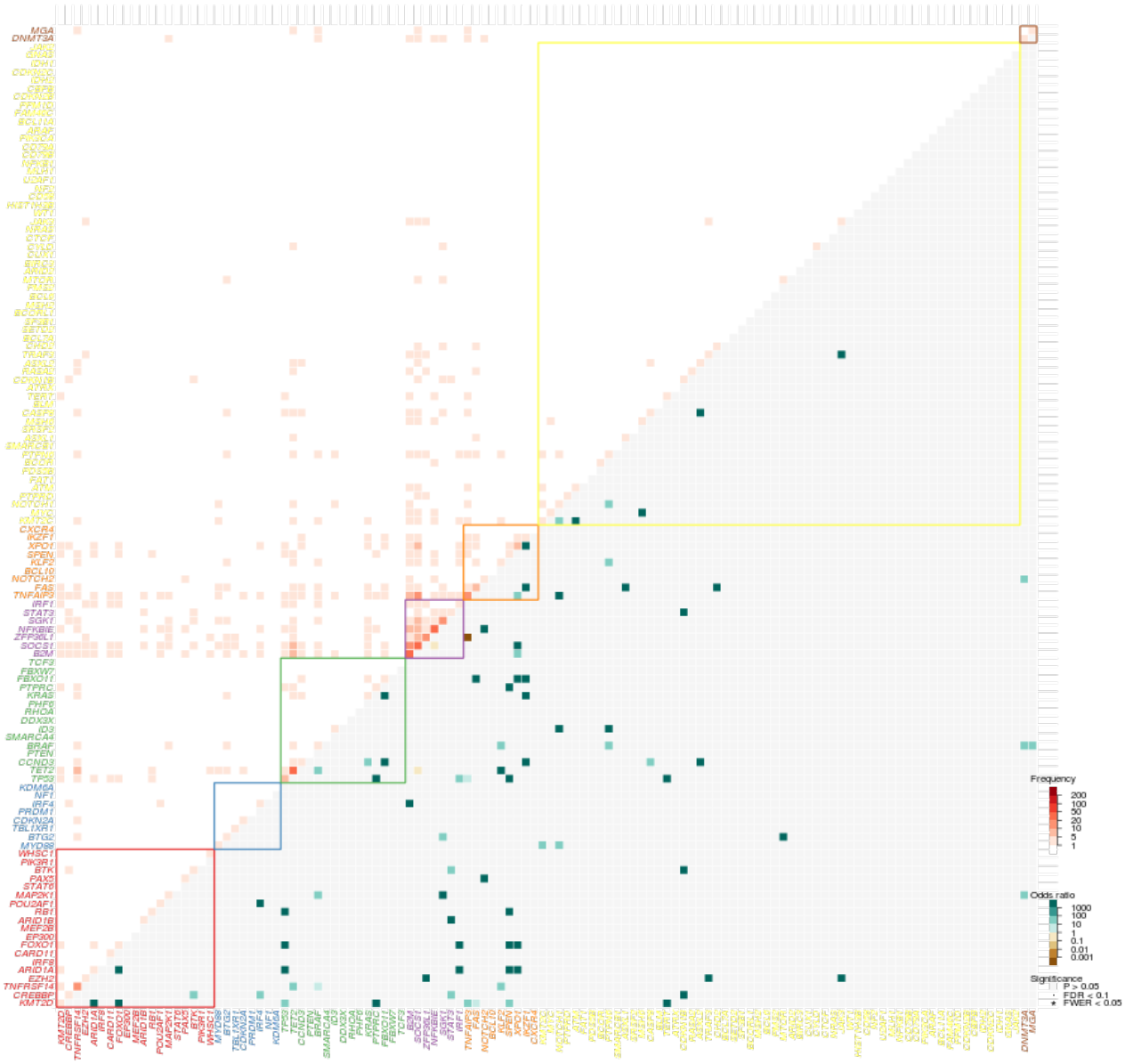
Class 2: KMT2D;MYD88;CREBBP;TNFRSF14;EZH2



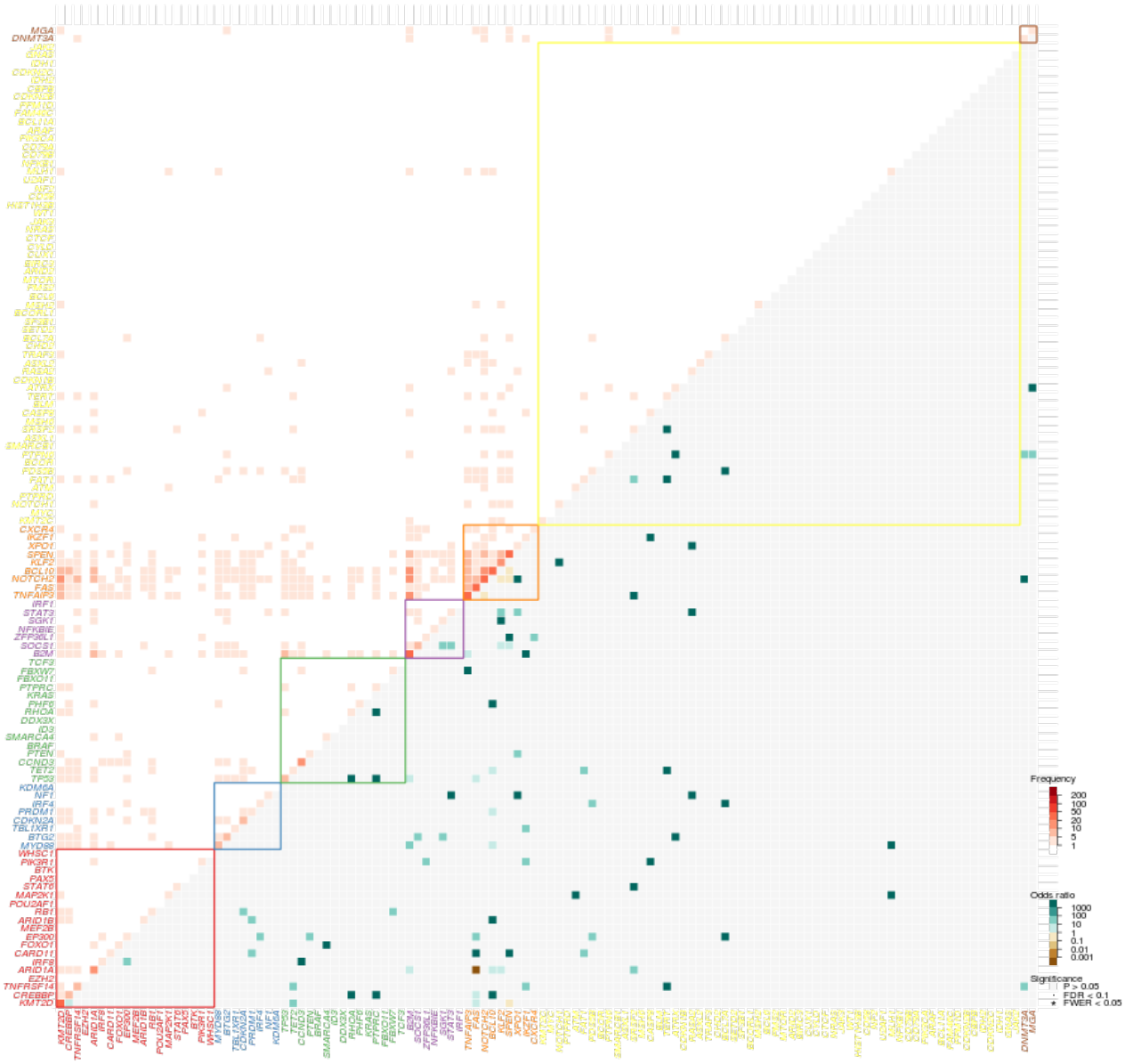
Class 3: TP53;SOCS1;TET2;B2M;CCND3



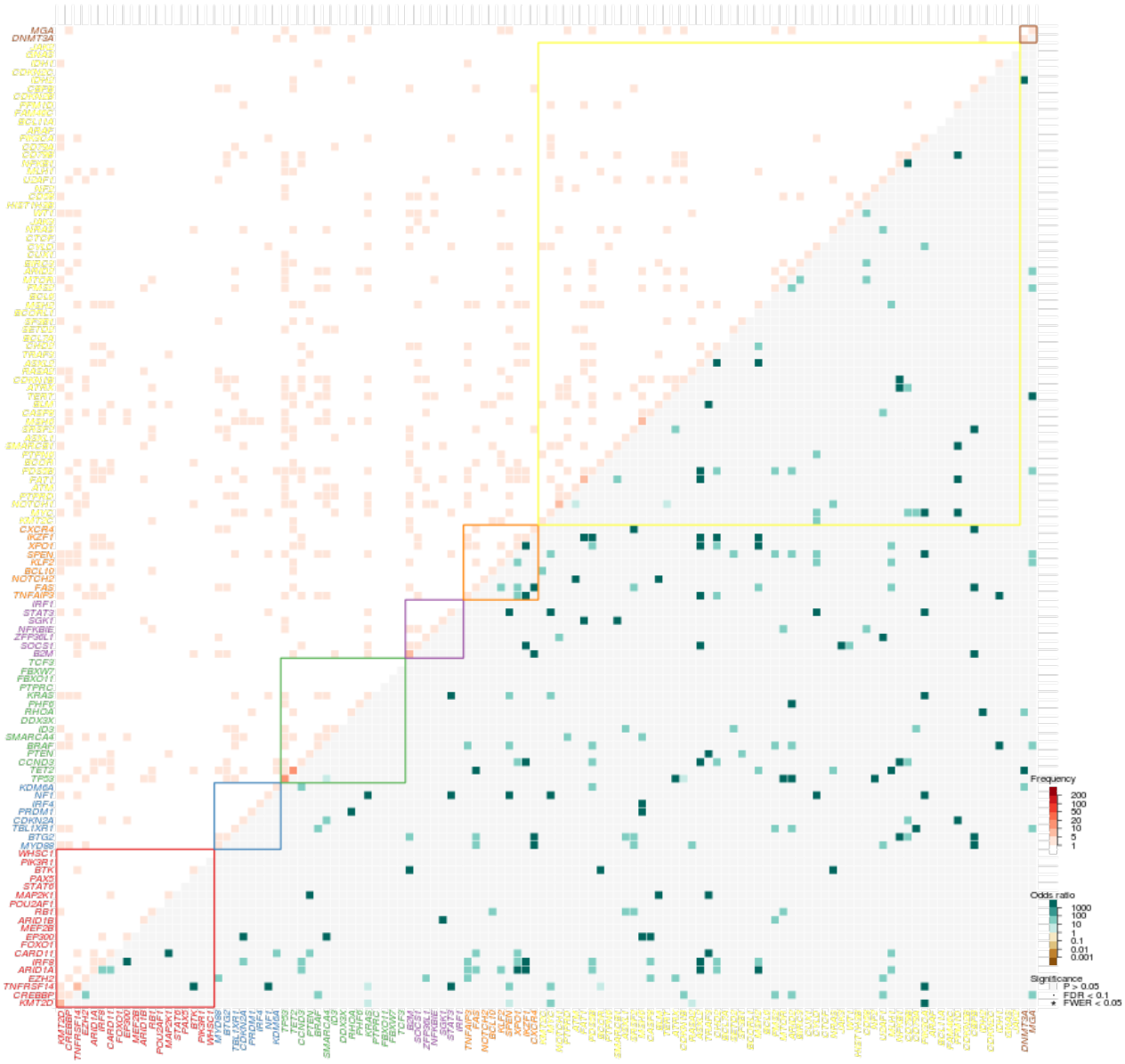
Class 4: SOCS1;B2M;TP53;TET2;TNFAIP3

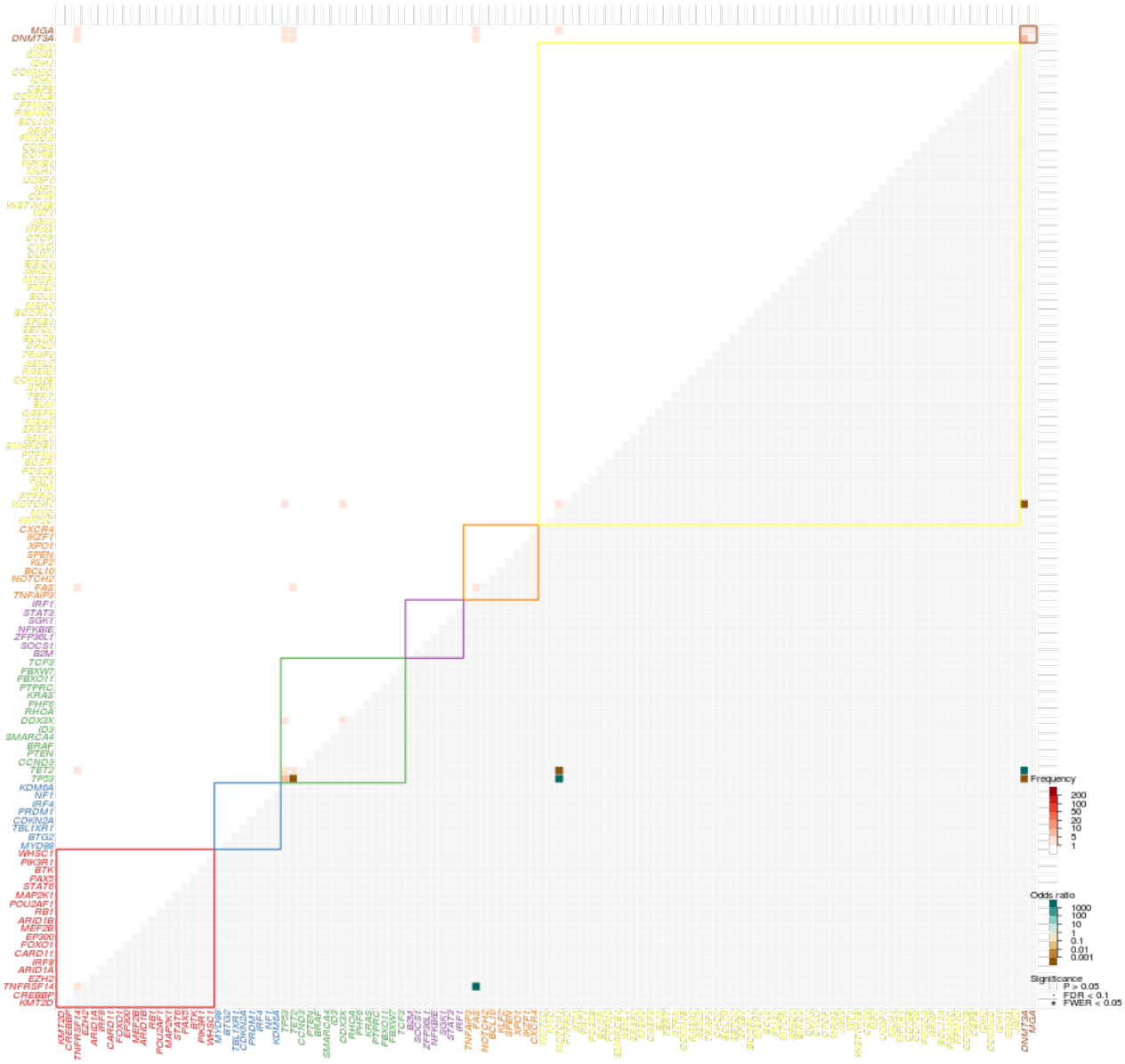


Class 5: TNFAIP3;NOTCH2;B2M;BCL10;MYD88



Class 6: TP53;MYC;FAT1





Alternatively: Naive Bayes assignment

```
sigBayes <- function(genotype, sigs){
  dSig <- function(sig,genotype){
    dmultinom(genotype, prob=sig/sum(sig))
  }
  lik <- apply(sigs,2, dSig, genotype)
  lik/sum(lik)
}

naiveBayes <- t(apply(genotypesImputed,1,sigBayes,posteriorMeans))
```

Save

```
# names(dpClass) <- clinicalData$PDID
# save(dpClass, posteriorMeans, posteriorQuantiles, posteriorProbability, file='dpClass.RData
```

#' ## Curated classification c <- read.table("../data/reduced_classes.txt", header=TRUE, sep="\t") curatedClass <- c$ReductionClass names(curatedClass) <- c$Sample rm©

library(clue) t <- table(dpClass, curatedClass) s <- solve_LSAP(t, maximum=TRUE) t[,c(s, setdiff(1:ncol(t),s))]

#' ### Associations

X <- as.matrix(MakeInteger(curatedClass)) Z <- as.matrix(genotypesImputed) Y <- as.matrix(dataFrame[groups %in% c("Clinical","Demographics")])

cv.glm <- function(x,y, fold=5, family="gaussian"){ cvIdx <- sample(1:nrow(x)%% fold +1) p <- numeric(length(y)) for(i in 1:fold){ p[cvIdx==i] <- predict(glm(y ~ ., data=x, subset=cvIdx!=i, family=family), newdata=x[cvIdx==i,]) } if(family=="gaussian"){ m <- mean((p-y)$^{2)}$ s <- sd(sapply(1:fold, function(i) mean((p-y)[cvIdx==i]^{2})))/sqrt(fold) return(c(avg=m, sd=s)) } else if(family=="binomial"){ m <- performance(prediction(p, y), "auc")@y.values[[1]] s <- sd(sapply(1:fold, function(i) performance(prediction(p[cvIdx==i], y[cvIdx==i]), "auc")@y.values[[1]]))/sqrt(fold) return(c(avg=m, sd=s)) } }

#' #### White counts #+ wbc_box y <- log(clinicalData$wbc) boxplot(y ~ factor(curatedClass), ylab="log wbc", las=2) #+ wbc_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="White counts")

#' #### Bone marrow blasts #+ BM_box y <- car::logit(clinicalData$BM_Blasts/100) boxplot(y ~ factor(curatedClass), ylab="logit BM blasts", las=2) #+ BM_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="BM blasts")

#' #### PB blasts #+ PB_box y <- car::logit(clinicalData$PB_Blasts/100) boxplot(y ~ factor(curatedClass), ylab="logit PB blasts", las=2) #+ PB_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="PB blasts")

#' #### Age #+ Age_box y <- clinicalData$AOD boxplot(y ~ factor(curatedClass), ylab="Age", las=2) #+ Age_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Age")

#' #### LDH #+ LDH_box y <- log(clinicalData$LDH) boxplot(y ~ factor(curatedClass), ylab="log LDH", las=2) #+

LDH_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="LDH")

#' #### Platelets #+ platelets_box y <- log(clinicalData$platelet) boxplot(y ~ factor(curatedClass), ylab="log platelets", las=2) #+ platelets_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Platelets")

#' #### HB #+ HB_box y <- log(clinicalData$HB) boxplot(y ~ factor(curatedClass), ylab="log HB", las=2) #+ HB_bar, fig.width=1.5 set.seed(42) mse0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), na.omit(y)) g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), type="mse", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) mse1 <- c(min(g$cvm), g$cvsd[which.min(g$cvm)]) v <- var(y, use='c') a <- (v-c(subtypes=mse0[1], `subtypes+genomics`=min(mse1[1])))/v*100 barplot(rbind(a,100-a), ylab="Explained variance (%)", names=rep("",2), ylim=c(0,100)) -> b segments(b,(v-c(mse0[1]-mse0[2],mse1[1]-mse1[2]))*100/v,b,(v-c(mse0[1]+mse0[2], mse1[1]+mse1[2]))*100/v) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="HB")

oddsPlot <- function(t){ plot(x=rep(1:2, ncol(t)), t[], ylab="Number of cases", xlab= names(dimnames(t))[1], xaxt="n", log='y') mtext(side=1,at=c(1,2), text=paste0(rownames(t), ", n=", rowSums(t)), las=1, pch=16) segments(1,t[1,],2, t[2,]) p <- sapply(1:ncol(t), function(i) {f <- fisher.test(cbind(rowSums(t[,-i]), t[,i])); c(p.value=f$p.value, OR=f$estimate[1], f$conf.int)}) mtext(side=4, at=t[2,], text=paste0(colnames(t), ", OR=", format(p[2,],digits=1), "(", apply(round(p[3:4,],2),2,paste, collapse="-"),")", sig2star(p[1,]))) }

#' #### Splenomegaly #+ Splenomegaly_bar, fig.width=1.5 library(ROCR) set.seed(42) y <- clinicalData$Splenomegaly table(Splenomegaly=y,factor(curatedClass)) auc0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), y[!is.na(y)], family="binomial") g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), family='binomial',type="auc", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) auc1 <- c(max(g$cvm), g$cvsd[which.max(g$cvm)]) a <- c(subtypes=auc0[1], `subtypes+genomics`=auc1[1])*100 barplot(rbind(a,150-a)-50, ylab="AUC (%)", main="Splenomegaly", offset=50, names=rep("",2), ylim=c(50,100)) segments(b,c(auc0[1]-auc0[2],auc1[1]-auc1[2])*100,b,c(auc0[1]+auc0[2], auc1[1]+auc1[2])*100) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Splenomegaly")

#' #### Gender #+ Gender_bar, fig.width=1.5 set.seed(42) y <- clinicalData$gender -1 table(Gender=factor(y, labels=c('male','female')),factor(curatedClass)) auc0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), y[!is.na(y)], family="binomial") g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), family='binomial',type="auc", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) auc1 <- c(max(g$cvm), g$cvsd[which.max(g$cvm)]) a <- c(subtypes=auc0[1], `subtypes+genomics`=auc1[1])*100 barplot(rbind(a,150-a)-50, ylab="AUC (%)", offset=50, names=rep("",2), ylim=c(50,100)) segments(b,c(auc0[1]-auc0[2],auc1[1]-auc1[2])*100,b,c(auc0[1]+auc0[2], auc1[1]+auc1[2])*100) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Gender")

#' #### CR #+ CR_bar, fig.width=1.5 set.seed(42) y <- !is.na(clinicalData$CR_date) y[is.na(clinicalData$CR_date) & clinicalData$OS==0] <- NA table(CR=y,factor(curatedClass)) auc0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), y[!is.na(y)], family="binomial") g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), family='binomial',type="auc", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) auc1 <- c(max(g$cvm), g$cvsd[which.max(g$cvm)]) a <- c(subtypes=auc0[1], `subtypes+genomics`=auc1[1])*100 barplot(rbind(a,150-a)-50, ylab="AUC (%)", offset=50, names=rep("",2), ylim=c(50,100)) segments(b,c(auc0[1]-auc0[2],auc1[1]-auc1[2])*100,b,c(auc0[1]+auc0[2], auc1[1]+auc1[2])*100) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Complete remission")

#' #### OS #+ OS_bar, fig.width=1.5 set.seed(42) y <- os[1:1540,2] y[os[1:1540,1] < 3 * 365 & os[1:1540,2]==0] <- NA table(OS=factor(y,labels=c("alive","dead")),factor(curatedClass)) auc0 <- cv.glm(as.data.frame(X[!is.na(y),-1]), y[!is.na(y)], family="binomial") g <- cv.glmnet(cbind(X,Z)[!is.na(y),], na.omit(y), family='binomial',type="auc", alpha=1, penalty.factor=c(rep(0,ncol(X)), rep(1,ncol(Z)))) auc1 <- c(max(g$cvm), g$cvsd[which.max(g$cvm)]) a <- c(subtypes=auc0[1], `subtypes+genomics`=auc1[1])*100 barplot(rbind(a,150-a)-50, ylab="AUC (%)", offset=50, names=rep("",2), ylim=c(50,100)) segments(b,c(auc0[1]-auc0[2],auc1[1]-auc1[2])*100,b,c(auc0[1]+auc0[2], auc1[1]+auc1[2])*100) rotatedLabel(b, labels=c("subtypes","subtypes+genomics")) title(main="Overall survival at 3yr")

Session

```
devtools::session_info()
```

```
## Session info -------------------------------------------------------------
```

```
##  setting  value
##  version  R version 3.3.3 (2017-03-06)
##  system   x86_64, linux-gnu
##  ui       X11
##  language (EN)
##  collate  en_GB.UTF-8
##  tz       Europe/London
##  date     2017-08-29
```

```
## Packages -----------------------------------------------------------------
```

```
##  package       * version  date       source
##  AnnotationDbi   1.36.2   2017-05-15 Bioconductor
##  ape           * 4.1      2017-02-14 CRAN (R 3.3.3)
##  assertthat      0.2.0    2017-04-11 CRAN (R 3.3.3)
##  base          * 3.3.3    2017-03-15 local
##  Biobase         2.34.0   2017-05-15 Bioconductor
##  BiocGenerics    0.20.0   2017-04-24 Bioconductor
##  biomaRt       * 2.30.0   2017-05-15 Bioconductor
##  bitops          1.0-6    2013-08-17 CRAN (R 3.3.3)
##  cellranger      1.1.0    2016-07-27 CRAN (R 3.3.3)
##  codetools       0.2-15   2016-10-05 CRAN (R 3.3.3)
##  colorspace      1.3-2    2016-12-14 CRAN (R 3.3.3)
##  CoxHD         * 0.0.61   2017-07-04 Github (mg14/CoxHD@d295566)
##  data.table    * 1.10.4   2017-02-01 CRAN (R 3.3.3)
##  datasets      * 3.3.3    2017-03-15 local
##  DBI             0.6-1    2017-04-01 CRAN (R 3.3.3)
##  devtools        1.13.2   2017-06-02 CRAN (R 3.3.3)
##  digest          0.6.12   2017-01-27 CRAN (R 3.3.3)
##  dplyr         * 0.5.0    2016-06-24 CRAN (R 3.3.3)
##  dtplyr        * 0.0.2    2017-04-21 CRAN (R 3.3.3)
##  evaluate        0.10.1   2017-06-24 cran (@0.10.1)
##  foreach         1.4.3    2015-10-13 cran (@1.4.3)
##  ggplot2       * 2.2.1    2016-12-30 CRAN (R 3.3.3)
##  ggrepel       * 0.6.5    2016-11-24 CRAN (R 3.3.3)
##  ggthemes      * 3.4.0    2017-02-19 CRAN (R 3.3.3)
```

```
##  glmnet          2.0-10   2017-05-06 cran (@2.0-10)
##  graphics      * 3.3.3    2017-03-15 local
##  grDevices     * 3.3.3    2017-03-15 local
##  grid          * 3.3.3    2017-03-15 local
##  gridExtra     * 2.2.1    2016-02-29 CRAN (R 3.3.3)
##  gsubfn        * 0.6-7    2017-04-13 Github (ggrothendieck/gsubfn@d2ef6c4)
##  gtable          0.2.0    2016-02-26 CRAN (R 3.3.3)
##  hdp           * 0.0.1    2017-07-19 Github (nicolaroberts/hdp@506f381)
##  highr           0.6      2016-05-09 cran (@0.6)
##  hms             0.3      2016-11-22 CRAN (R 3.3.3)
##  IRanges         2.8.2    2017-04-24 Bioconductor
##  iterators       1.0.8    2015-10-13 cran (@1.0.8)
##  knitr         * 1.16     2017-05-18 cran (@1.16)
##  labeling        0.3      2014-08-23 CRAN (R 3.3.3)
##  lattice       * 0.20-35  2017-03-25 CRAN (R 3.3.3)
##  lazyeval        0.2.0    2016-06-12 CRAN (R 3.3.3)
##  lsa             0.73.1   2015-05-08 cran (@0.73.1)
##  magrittr        1.5      2014-11-22 CRAN (R 3.3.3)
##  MASS            7.3-47   2017-04-21 CRAN (R 3.3.3)
##  Matrix          1.2-10   2017-04-28 CRAN (R 3.3.3)
##  memoise         1.1.0    2017-04-21 CRAN (R 3.3.3)
##  methods       * 3.3.3    2017-03-15 local
##  mg14          * 0.0.5    2017-07-04 Github (mg14/mg14@a8b4ba8)
##  mice            2.30     2017-02-18 cran (@2.30)
##  munsell         0.4.3    2016-02-13 CRAN (R 3.3.3)
##  mvtnorm         1.0-6    2017-03-02 cran (@1.0-6)
##  nlme            3.1-131  2017-02-06 CRAN (R 3.3.3)
##  nnet            7.3-12   2016-02-02 CRAN (R 3.3.3)
##  parallel      * 3.3.3    2017-03-15 local
##  plyr            1.8.4    2016-06-08 CRAN (R 3.3.3)
##  proto         * 1.0.0    2016-10-29 cran (@1.0.0)
##  R6              2.2.2    2017-06-17 cran (@2.2.2)
##  RColorBrewer  * 1.1-2    2014-12-07 CRAN (R 3.3.3)
##  Rcpp            0.12.11  2017-05-22 cran (@0.12.11)
##  RCurl           1.95-4.8 2016-03-01 CRAN (R 3.3.3)
##  readr         * 1.1.0    2017-03-22 CRAN (R 3.3.3)
##  readxl        * 1.0.0    2017-04-18 CRAN (R 3.3.3)
##  rpart           4.1-11   2017-04-21 CRAN (R 3.3.3)
##  RSQLite         1.1-2    2017-01-08 CRAN (R 3.3.3)
##  S4Vectors       0.12.2   2017-04-24 Bioconductor
##  scales        * 0.4.1    2016-11-09 CRAN (R 3.3.3)
##  SnowballC       0.5.1    2014-08-09 cran (@0.5.1)
##  splines         3.3.3    2017-03-15 local
##  stats         * 3.3.3    2017-03-15 local
##  stats4          3.3.3    2017-03-15 local
##  stringi         1.1.5    2017-04-07 CRAN (R 3.3.3)
##  stringr       * 1.2.0    2017-02-18 CRAN (R 3.3.3)
##  survival      * 2.41-3   2017-04-04 CRAN (R 3.3.3)
##  tcltk           3.3.3    2017-03-15 local
##  tibble          1.3.0    2017-04-01 CRAN (R 3.3.3)
##  tidyr         * 0.6.1    2017-01-10 CRAN (R 3.3.3)
```

```
##  tools           3.3.3    2017-03-15 local
##  utils         * 3.3.3    2017-03-15 local
##  withr           1.0.2    2016-06-20 CRAN (R 3.3.3)
##  XML             3.98-1.7 2017-05-03 CRAN (R 3.3.3)
```