PrecisionFDA Phase I

Alex Harms

October 17, 2018

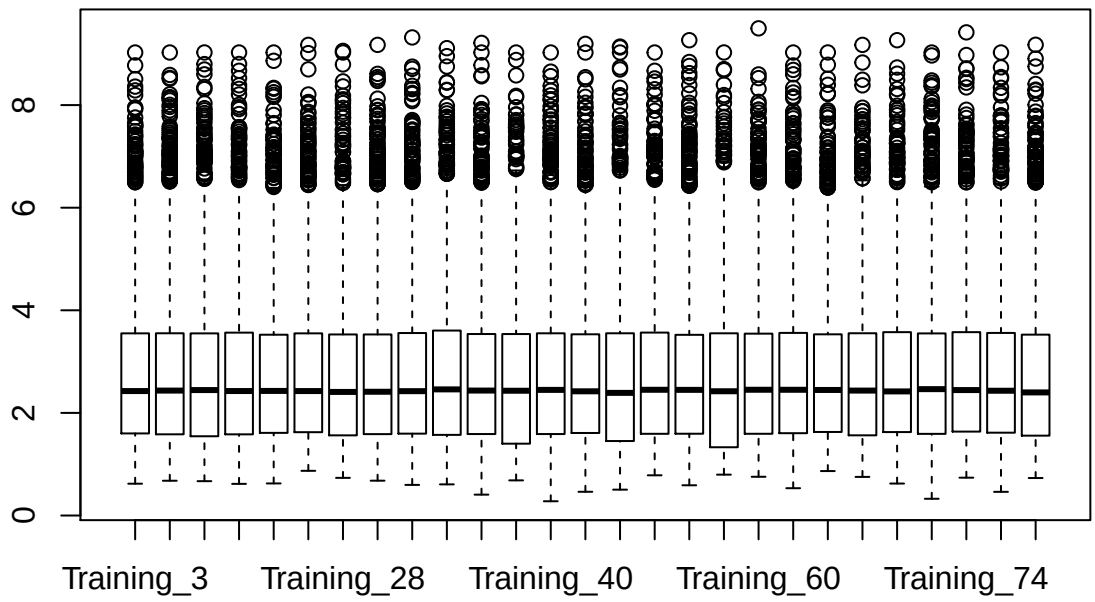
Part A: Exploratory Data Analysis

Will read in the data here.

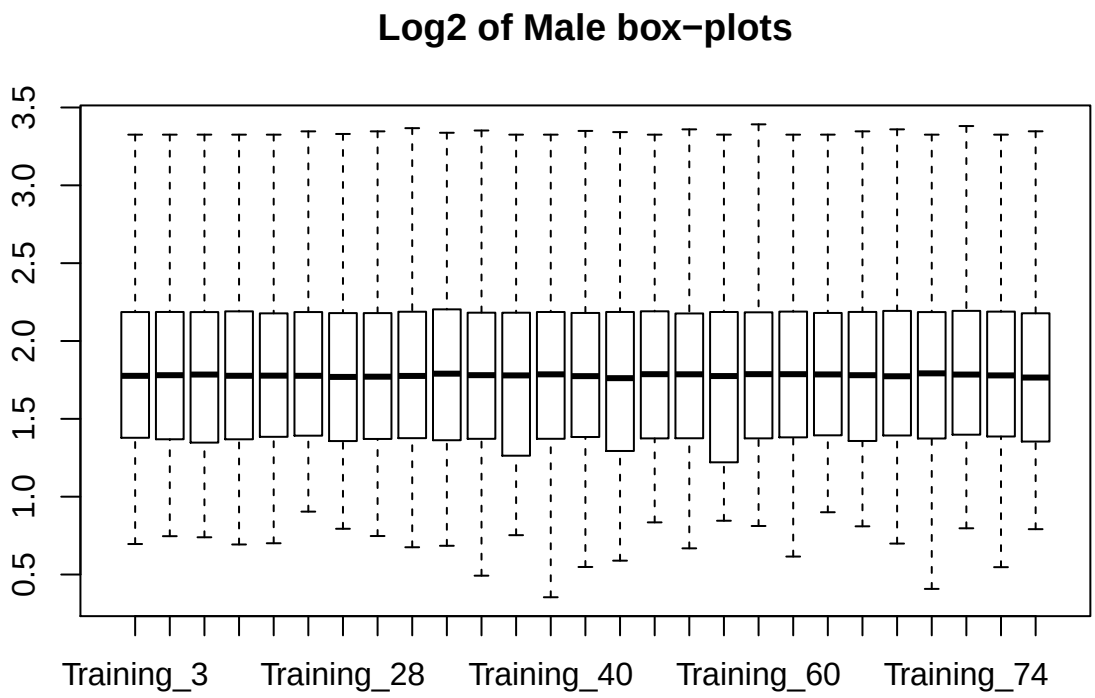
```
test_cli <- read.table("test_cli.tsv", row.names=1, header=T)
train_cli <- read.table("train_cli.tsv", row.names=1, header=T)
test_pro <- read.table("test_pro.tsv", row.names=1, header=T)
train_pro <- read.table("train_pro.tsv", row.names=1, header=T)
train_missmatch <-read.csv("sum_tab_1.csv",header = T, row.names = 1)
True_train <- cbind(train_cli,train_missmatch)
```

1. Distribution by sample

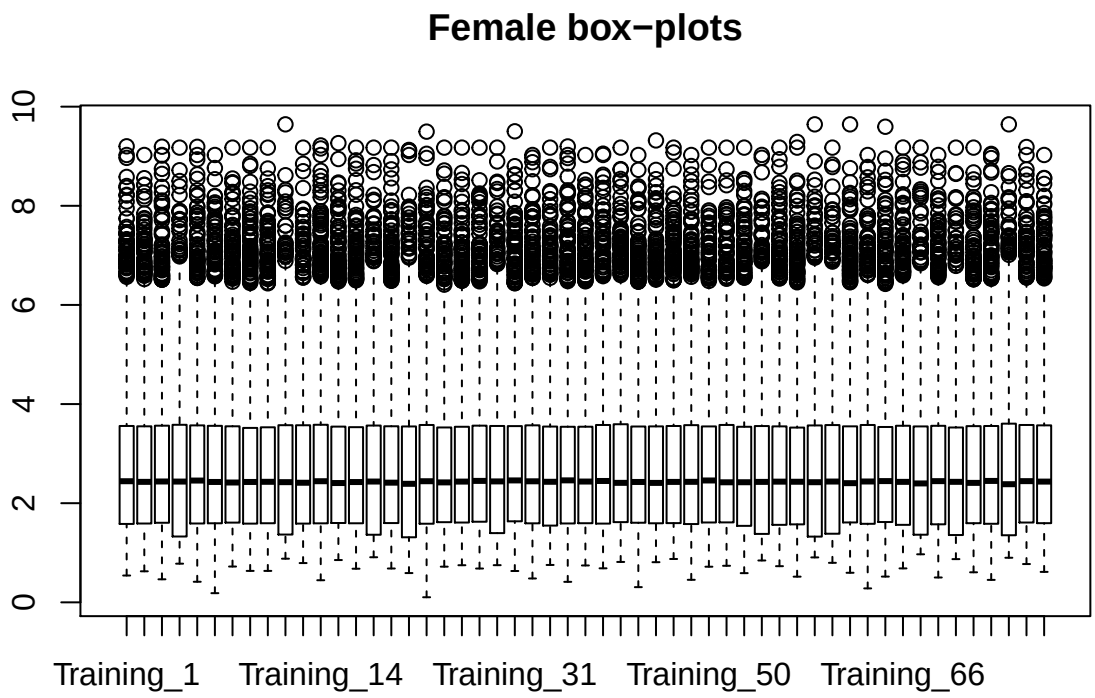
```
male_train_pro <- train_pro[,which(train_cli$gender=="Male")]
female_train_pro <- train_pro[,-which(train_cli$gender=="Male")]
High_train_pro <- train_pro[,which(train_cli$msi =="MSI-High")]
Low_train_pro <- train_pro[,-which(train_cli$msi =="MSI-High")]

male_log_train_pro <- log2(male_train_pro+1)
female_log_train_pro <- log2(female_train_pro+1)

High_log_train_pro <- log2(High_train_pro+1)
Low_log_train_pro <- log2(Low_train_pro+1)

boxplot(male_train_pro, main="Male box-plots")
```



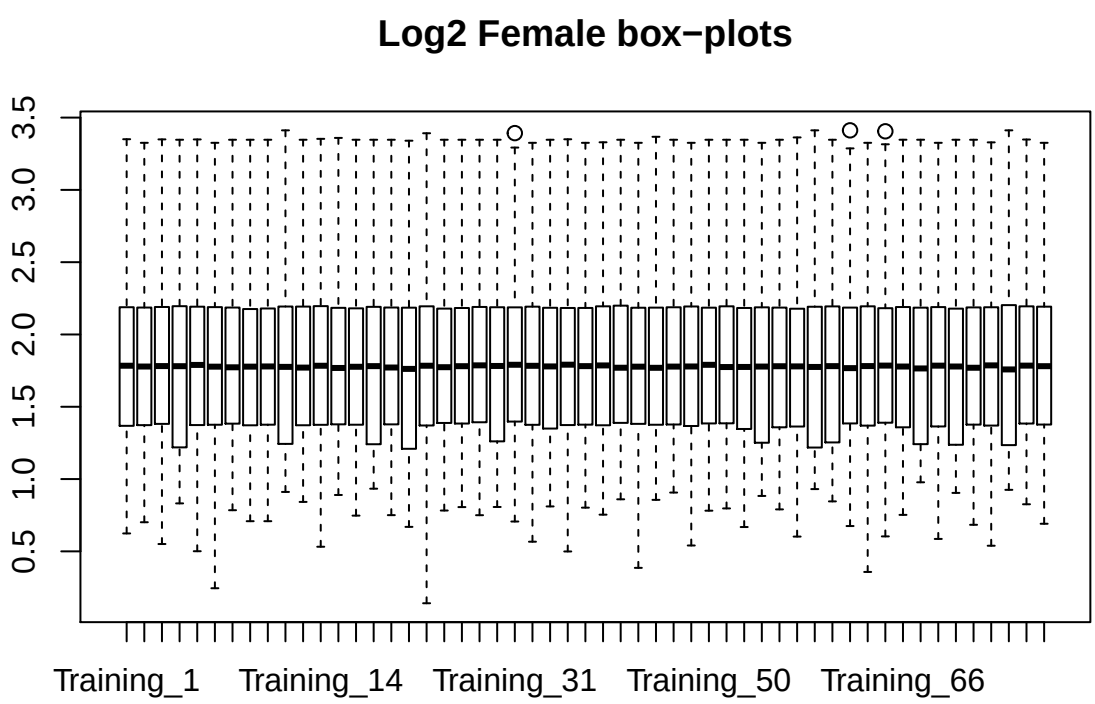

```
boxplot(male_log_train_pro, main="Log2 of Male box-plots")
```



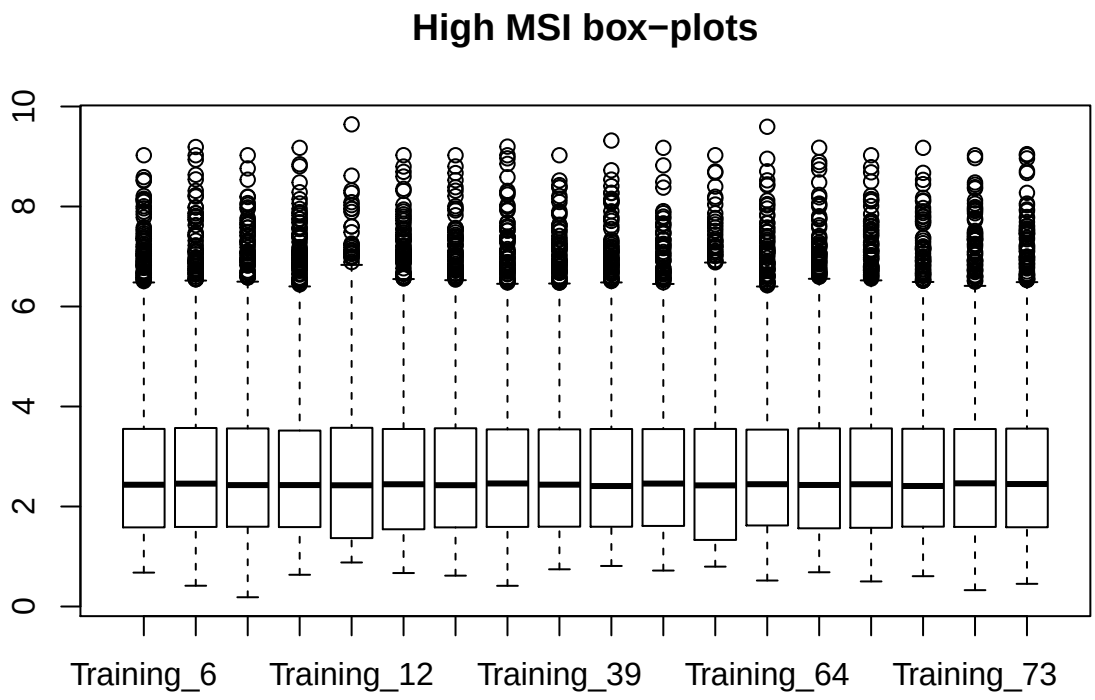

```
boxplot(female_train_pro, main="Female box-plots")
```



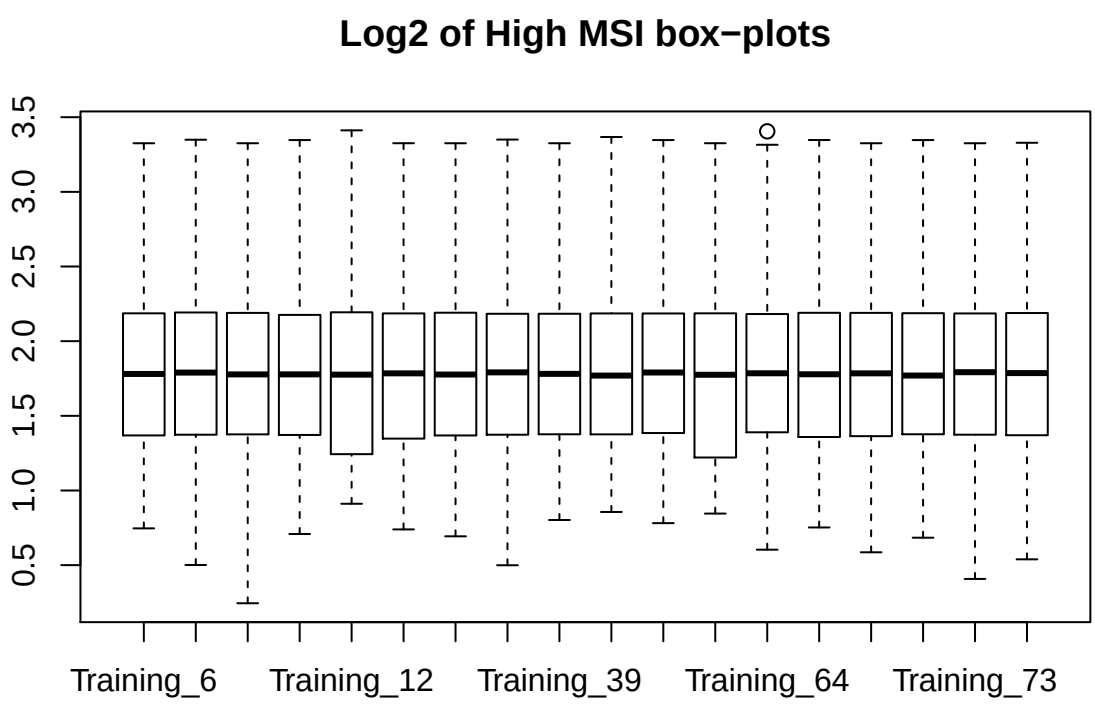

```
boxplot(female_log_train_pro, main="Log2 Female box-plots")
```



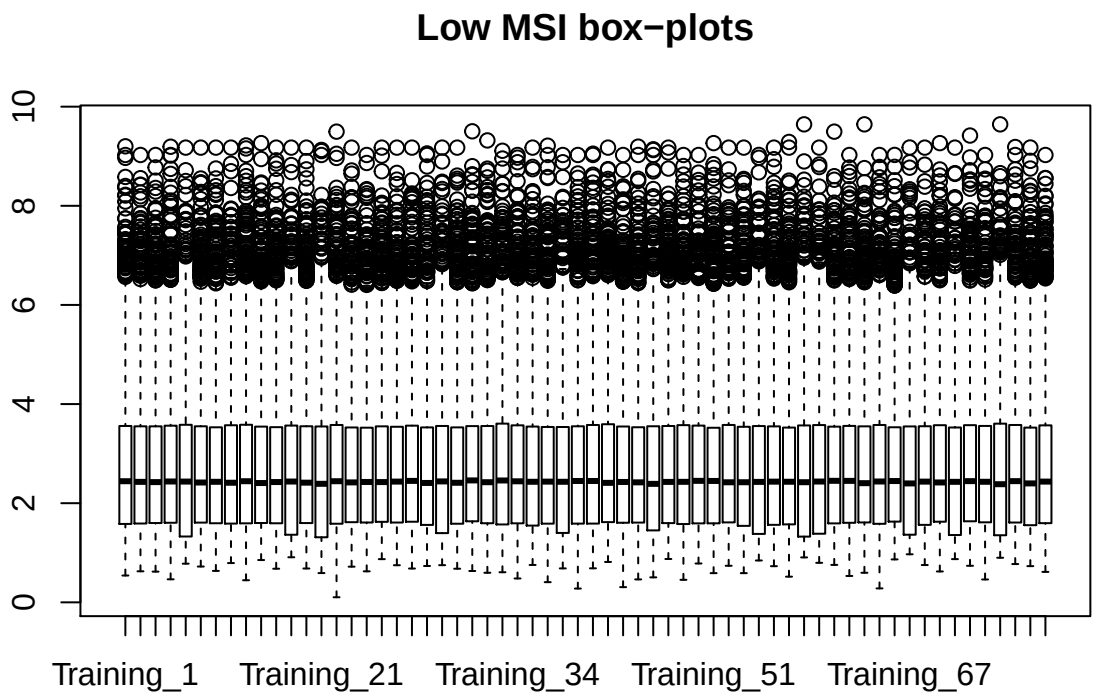

```
boxplot(High_train_pro, main="High MSI box-plots")
```



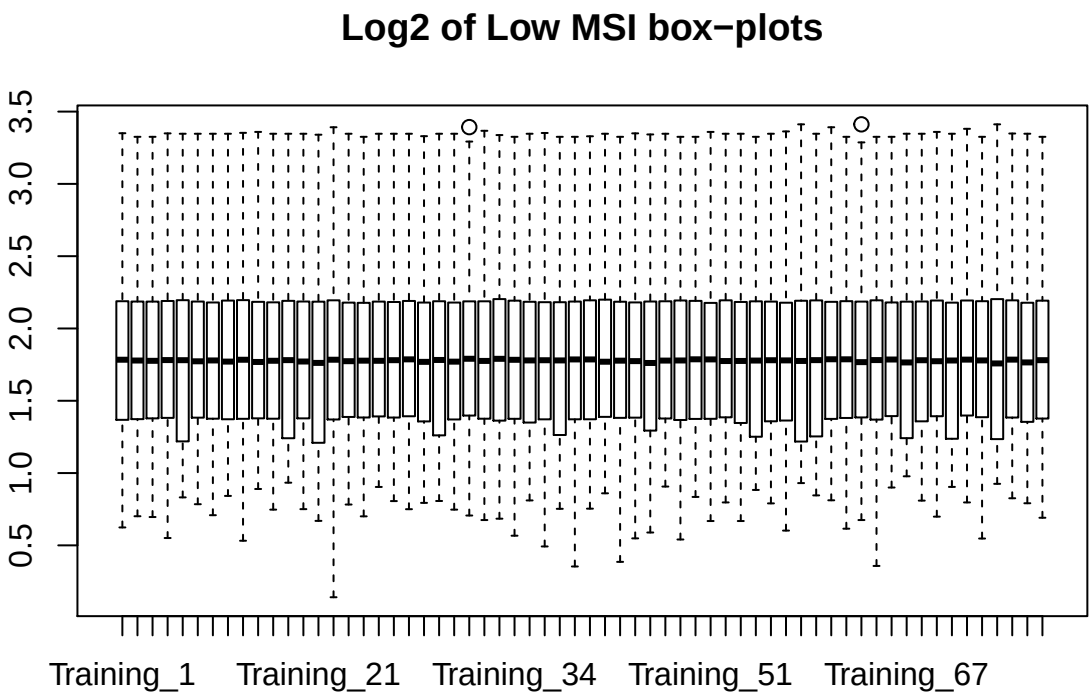

```
boxplot(High_log_train_pro, main="Log2 of High MSI box-plots")
```




```
boxplot(Low_train_pro, main="Low MSI box-plots")
```




```
boxplot(Low_log_train_pro, main="Log2 of Low MSI box-plots")
```

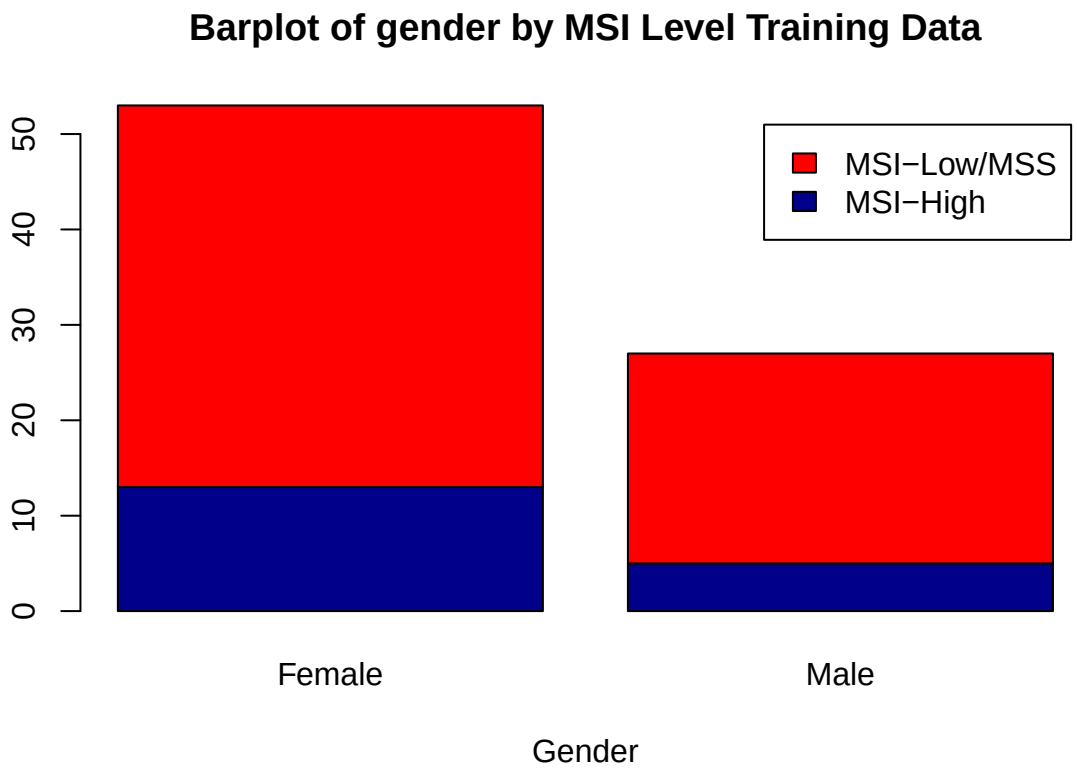


2. Correlation between gender and MSI

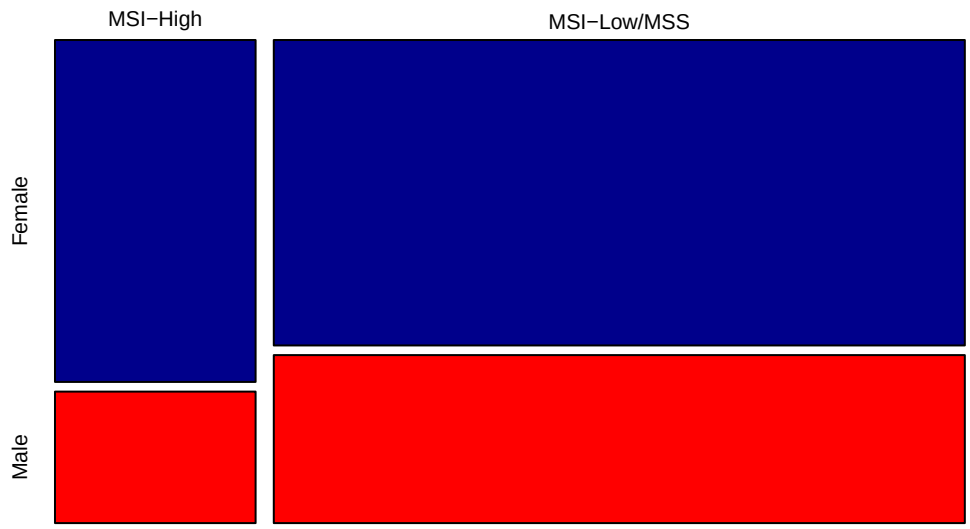
```
#Trainning data
dat <- table(train_cli$msi,train_cli$gender)
dat
```

```
##
##                Female Male
##   MSI-High         13    5
##   MSI-Low/MSS      40   22
```

```
barplot(dat, main="Barplot of gender by MSI Level Training Data",
        xlab="Gender", col=c("darkblue","red"),
        legend = rownames(dat))
```




```
plot(dat, col=c("darkblue","red"), main = "Training Data")
```




```
chisq.test(dat)
```

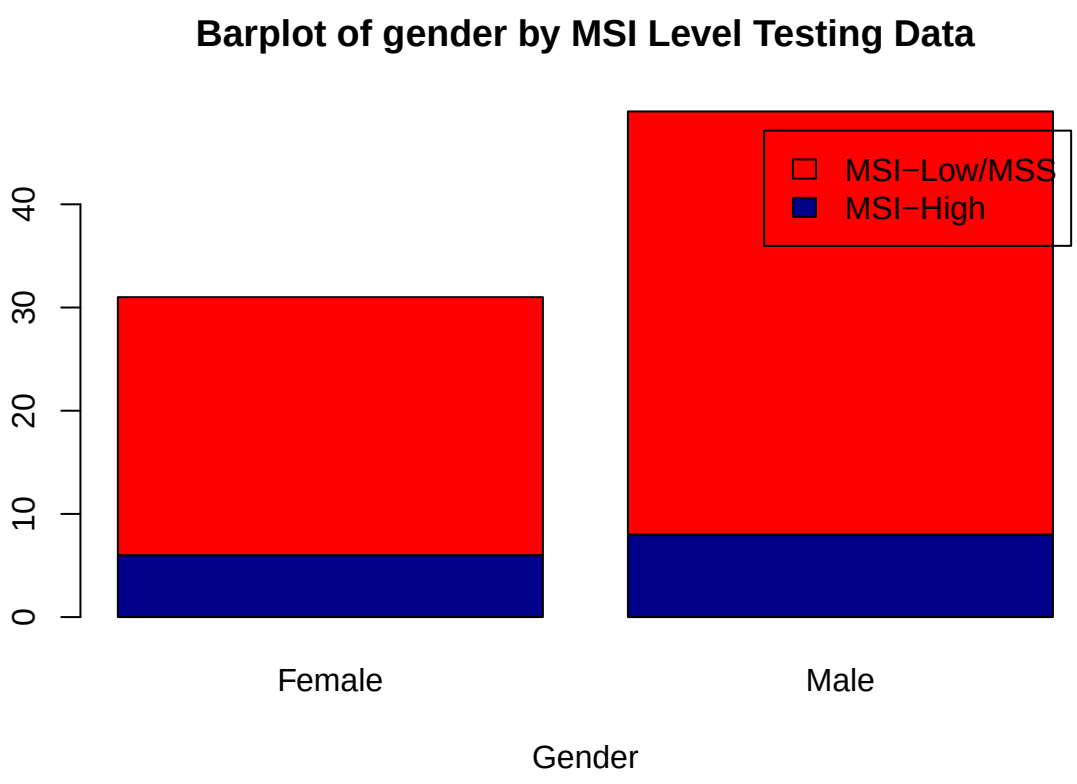
```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dat
## X-squared = 0.106, df = 1, p-value = 0.7447
```

Note that the Chi-Square test has a p-value greater than 0.05. Thus it is not significant so we fail to reject the two groups being different.

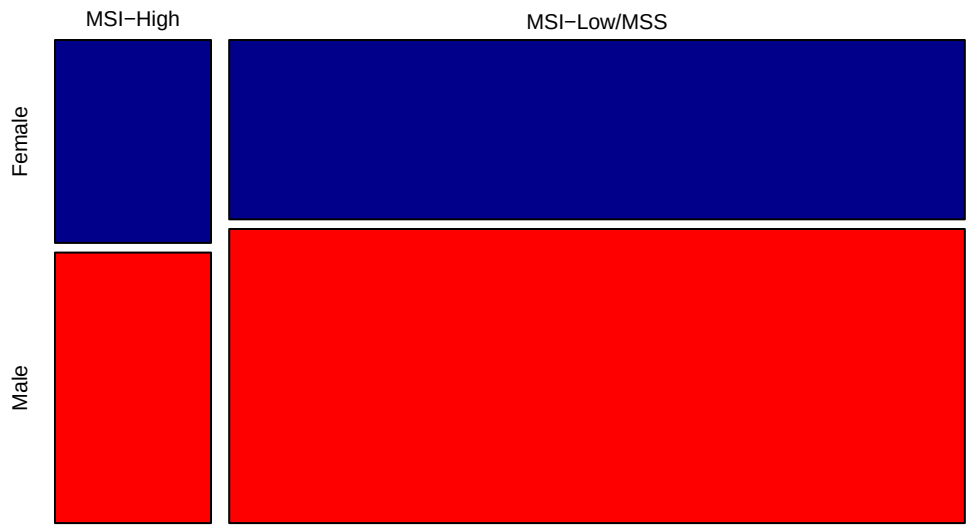
```
#Trainning data
dat2 <- table(test_cli$msi,test_cli$gender)
dat2
```

```
##
##               Female Male
##   MSI-High         6    8
##   MSI-Low/MSS     25   41
```

```
barplot(dat2, main="Barplot of gender by MSI Level Testing Data",
        xlab="Gender", col=c("darkblue","red"),
        legend = rownames(dat))
```




```
plot(dat2, col=c("darkblue","red"), main = "Testing Data")
```

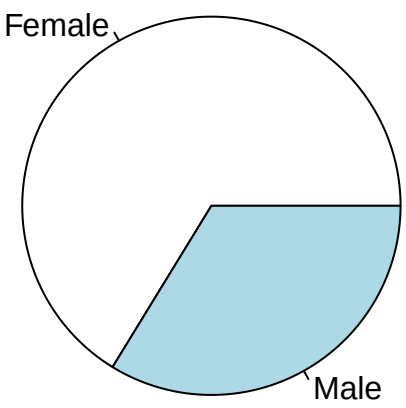



```
chisq.test(dat2)
```

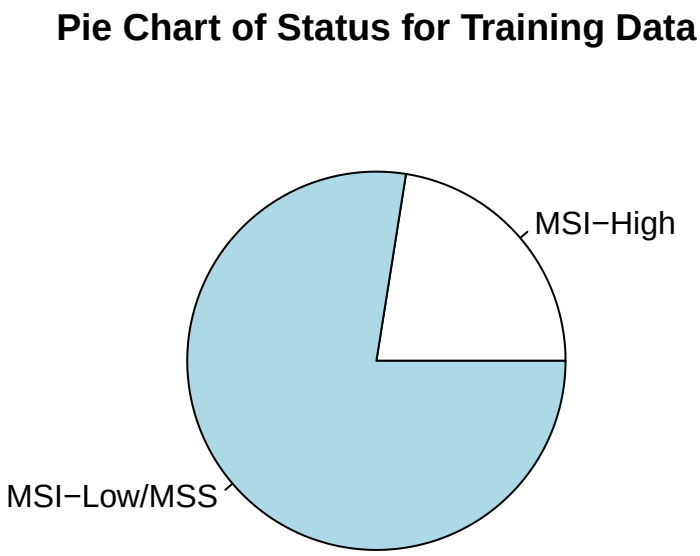
```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dat2
## X-squared = 0.0020519, df = 1, p-value = 0.9639
```

3. Additional Exploratory Plots

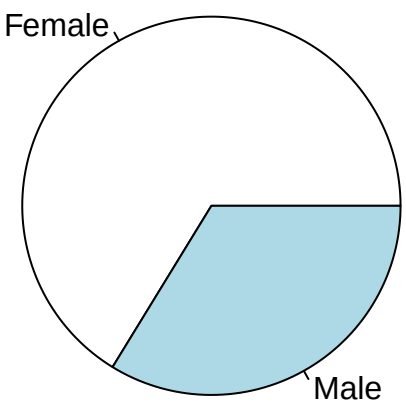
```
gender_tab <- table(train_cli$gender)
pie(gender_tab, main = "Pie Chart of Gender for Training Data")
```



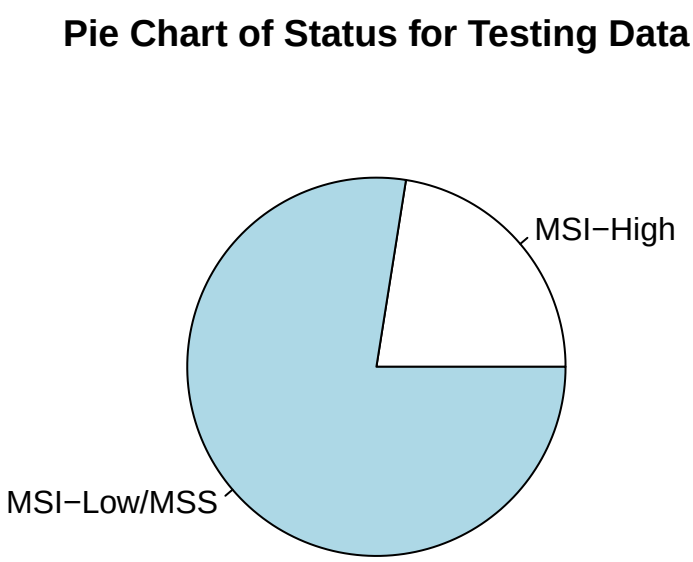

```
status_tab <- table(train_cli$msi)
pie(status_tab, main = "Pie Chart of Status for Training Data")
```




```
gender_tab <- table(train_cli$gender)
pie(gender_tab, main = "Pie Chart of Gender for Testing Data")
```




```
status_tab <- table(train_cli$msi)
pie(status_tab, main = "Pie Chart of Status for Testing Data")
```



4. Observations

Noticed that there are a lot of NA values in the data. Also distributioni of the observations appears a little skewed and looks better after a log transformation. Additionally I took a look at the classification table for the train and test. It is interesting to note that there are quite a few more Females in the trainning data and the reverse is true for the testing data. However the High and Low status approximately the proportion in both samples.

Part B: Predict Gender

1. Build first model

a). Read in data and preprocess (normalization, imputation, or digitalization)

The data was read in previously. Will rearrange it here though.

```
Male_train_pro <- as.data.frame(t(male_log_train_pro))
Male_train_pro$gender <- "Male"

Female_train_pro <- as.data.frame(t(female_log_train_pro))
Female_train_pro$gender <- "Female"

gender.t_train_pro <- rbind(Male_train_pro,Female_train_pro)
```

```
gender.t_train_pro$gender <- as.factor(gender.t_train_pro$gender)

gender.t_train_pro[is.na(gender.t_train_pro)] <- -100
```

b). Feature selection, if needed. (highly recommended as data is noisy)

```
#Use the two lines below to download BioLite Packages
# source("https://bioconductor.org/biocLite.R")
# biocLite("biomaRt")

library("biomaRt")
ensembl = useMart("ensembl", dataset = "hsapiens_gene_ensembl")
geneInfo = getBM(attributes = c( "hgnc_symbol", "chromosome_name"),
                 mart = ensembl)
#Since I am interested in gender differences in this part I am selecting only the genes expressed in th
Y_genes <- geneInfo[geneInfo[,2]=="Y",]
proteins <- colnames(gender.t_train_pro)
in.both <- c(intersect(proteins,Y_genes$hgnc_symbol),"gender")

Gender.mytest <- gender.t_train_pro[, names(gender.t_train_pro) %in% in.both]
```

c). Build one model as proof of concept (just make it work)

```
library(randomForest)
```

d). Evaluate model using leave-one-out cross validation (LOOCV). Compute error rate.

```
n.fold <- as.vector(c(1,2,4,5,8,10,20))
error <- vector()
Gender.mytest <- Gender.mytest[ order(row.names(Gender.mytest)), ]
True_train <- True_train[order(row.names(True_train)),]
missmatch <- vector()
x <- vector()
a <- vector()
j <- nrow(Gender.mytest)
for(i in 1:j){
    set.seed(123)
    temp.test <- Gender.mytest[-i,]
    temp.correct <- Gender.mytest[i,which(colnames(Gender.mytest)=="gender")]
    rf <- randomForest(gender~., data = temp.test,ntree=1000)
    temp.pred <- predict(rf,Gender.mytest[i,])
    x <- temp.pred==temp.correct
    missmatch[i] <- ifelse(x==TRUE,0,1)
    a[i] <- True_train$mismatch[i]==missmatch[i]
  }
error <- 1-(sum(a)/nrow(Gender.mytest))
error
```

```
## [1] 0.125
```

The error rate is 12.5% by LOOCV.

2. Build alternative model

a). Read in data and preprocess (normalization, imputation, or digitalization)

Data was already done in previous sections.

b). Feature selection, if needed. (highly recommended as data is noisy)

Reducing the data the same way as before.

c). Build one model as proof of concept (just make it work)

```
library(e1071)
#Not correct data but shows this works
svm.model <- svm(gender~., data = temp.test)
head(predict(svm.model, temp.test))
```

```
##  Training_1 Training_10 Training_11 Training_12 Training_13 Training_14
##      Female      Female      Female      Female      Female      Female
## Levels: Female Male
```

d). Evaluate model using leave-one-out cross validation (LOOCV). Compute error rate.

```
a2 <- vector()
j2 <- nrow(gender.t_train_pro)
missmatch2 <- vector()
#m <- colSums(gender.t_train_pro[,-4119])
#gender.t_train_pro2 <- gender.t_train_pro[,-which(m < -7000)]
for(i in 1:j2){
    set.seed(321)
    temp.test2 <- Gender.mytest[-i,]#gender.t_train_pro2[-i,]
    temp.correct2 <- Gender.mytest[i,which(colnames(Gender.mytest)=="gender")]# gender.t_train_pro2[i,w
    svm.model2 <- svm(gender~., data = temp.test2)
    temp.pred2 <- predict(svm.model2,Gender.mytest[i,])
    x2 <- temp.pred2==temp.correct2
    missmatch2[i] <- ifelse(x2==TRUE,0,1)
    a2[i] <- True_train$mismatch[i]==missmatch2[i]
  }
error2 <- 1-(sum(a2)/j)
error2
```

```
## [1] 0.1125
```

This error rate is about 11.25%. This is less than the Random Forest model.

3. Final prediction.

a). Choose and train best model. As the training dataset contains mislabeled samples, you may want to exclude a small number of training samples that are predicted wrong in the LOOCV.

Best model was SVM. Thus we will proceed with that method. Will removed missmatched data from SVM method and labeled missmatched.

```
remove <- missmatch2 + True_train$mismatch
Gender.mytrain.final <- Gender.mytest[-which(remove>0),]

Final.Model <- svm(gender~., data = Gender.mytrain.final)
```

b) Make prediction on test data.

```
male_test_pro <- test_pro[,which(test_cli$gender=="Male")]
female_test_pro <- test_pro[,-which(test_cli$gender=="Male")]

male_log_test_pro <- log2(male_test_pro+1)
female_log_test_pro <- log2(female_test_pro+1)

Male_test_pro <- as.data.frame(t(male_log_test_pro))
Male_test_pro$gender <- "Male"

Female_test_pro <- as.data.frame(t(female_log_test_pro))
Female_test_pro$gender <- "Female"

gender.t_test_pro <- rbind(Male_test_pro,Female_test_pro)
gender.t_test_pro$gender <- as.factor(gender.t_test_pro$gender)

gender.t_test_pro[is.na(gender.t_test_pro)] <- -100

gender.t_test_pro <- gender.t_test_pro[order(row.names(gender.t_test_pro)), ]


Predictions <- predict(Final.Model, gender.t_test_pro)

Final <- as.data.frame(cbind(gender.t_test_pro$gender, Predictions))
Final$mislabeled <- ifelse(Final[,1] == Final[,2],0, 1)

Mislabled <- cbind(row.names(Final), Final$mislabeled)
write.csv(Mislabled,"Test_Mislabeled.csv",col.names = c("sample","mislabeled"))
```

The file created called Test_Mislabeled.csv contains which samples were mislabeled by the model technique used.

Part C. Reproducibility check

Part D. Predict microsatellite instability (MSI) status in cancer

Follow the same steps for Part B and submit 4 files. The two CSV files should be named ???MSI_mismatch_training.csv??? and ???MSI_mismatch_testing.csv???.

1. Build first model

a). Read in data and preprocess (normalization, imputation, or digitalization)

b). Feature selection, if needed. (highly recommended as data is noisy)

c). Build one model as proof of concept (just make it work)

d). Evaluate model using leave-one-out cross validation (LOOCV). Compute error rate.

2. Build alternative model

a). Read in data and preprocess (normalization, imputation, or digitalization)

b). Feature selection, if needed. (highly recommended as data is noisy)

c). Build one model as proof of concept (just make it work)

d). Evaluate model using leave-one-out cross validation (LOOCV). Compute error rate.

3. Final prediction.

a). Choose and train best model. As the training dataset contains mislabeled samples, you may want to exclude a small number of training samples that are predicted wrong in the LOOCV.

b) Make prediction on test data.

Part E. Combine results of gender and MSI status predictions

1. Compare LOOCV results from gender model and MSI model to see if the same training samples are mislabeled for both gender and MSI status.

2. Combine predictions results of both gender and MSI status models and generate one file with mislabeled test samples. Use this format and name this file: final_mismatch.csv