Prediction of Adar Editing Levels from bpRNA Structure

load the feature matrix and split into training and test sets

```
set.seed(1)

#load the feature matrix
data=read.table("rf_features.txt",header=TRUE,sep='\t')
#remove values with no NaN in editing level
data=data[is.nan(data$ave_editing_level)==FALSE,]
head(data)
```

```
##   ave_editing_level edit_feat prev_feat next_feat mp1 mref1 malt1   mtype1
## 1              0.58         I         S         S  44     G     A mismatch
## 2              0.46         I         S         S  45     G     A mismatch
## 3              0.49         B         S         S  46     G     A mismatch
## 4              0.48         I         S         S  47     G     A mismatch
## 5              0.59         I         S         S  48     C     A mismatch
## 6              0.39         I         S         S  52     G     A mismatch
##   mfeat1 mfeat1_prev mfeat1_next adist1  mp2 mref2 malt2 mtype2 mfeat2
## 1      S           H           I      6 None  None  None   None   None
## 2      S           H           I      5 None  None  None   None   None
## 3      S           H           B      4 None  None  None   None   None
## 4      S           H           I      3 None  None  None   None   None
## 5      S           H           I      2 None  None  None   None   None
## 6      S           I           I     -2 None  None  None   None   None
##   mfeat2_prev mfeat2_next adist2
## 1        None        None   None
## 2        None        None   None
## 3        None        None   None
## 4        None        None   None
## 5        None        None   None
## 6        None        None   None
```

The feature values are:

- edit_feat – Single character value of structure type for the edited A at position 50. One of S,H,M,I,B,X,E
- prev_feat – The 5' structural feature upstream of edited A
- next_feat – The 3' structural feature downstream of edited A
- mp1 – Firt mutated position along the RNA sequence
- mref1 – Reference allele at first mutated position
- mtype1 – One of "mismatch","indel","wt"
- mfeat1 – Single character value of structure type for mutated base. One of S,H,M,I,B,X,E
- mfeat1_prev – The 5' structural feature upstream of mutated base.
- mfeat1_next – The 3' structural feature downstream of mutated base.
- adist1 – The distance (in sequence space) of the mutated base from the edited A.

Repeat for possible second mutation ('None' if only 1 mutation present in sequence):

- mp2
- mref2
- mtype2
- mfeat2
- mfeat2_prev
- mfeat2_next
- adist2

```r
#get training and test splits; use 80% train, 20% test split
train_indices=sample(nrow(data),0.8*nrow(data),replace=FALSE)
train_split=data[train_indices,]
test_split=data[-train_indices,]
```

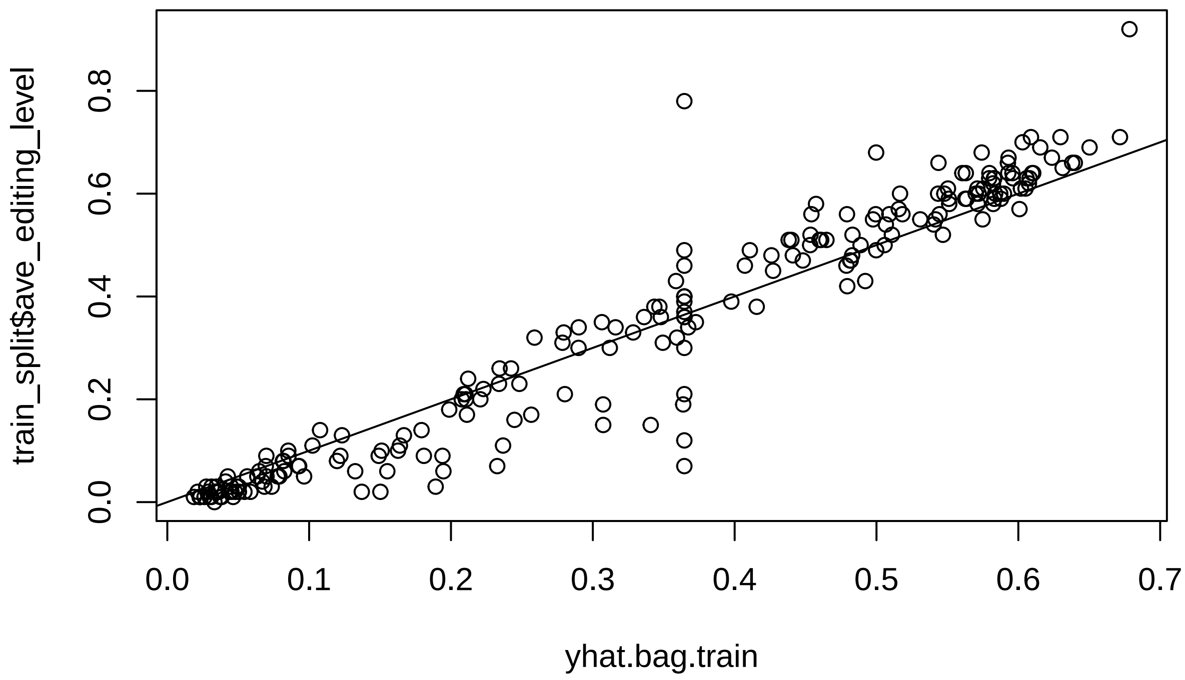
Bagging

```r
#train rf
bag.data=randomForest(ave_editing_level~.,data=data,subset=train_indices,mtry=19,importance=TRUE)
print(bag.data)
```

```
##
## Call:
##  randomForest(formula = ave_editing_level ~ ., data = data, mtry = 19,      importance = TRUE, subse
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 19
##
##           Mean of squared residuals: 0.02028694
##                     % Var explained: 66.33
```

Predictions on training split

```r
#get predictions on training & test data
yhat.bag.train=predict(bag.data,newdata=train_split)
yhat.bag.test=predict(bag.data,newdata=test_split)
plot(yhat.bag.train,train_split$ave_editing_level)
abline(0,1)
```

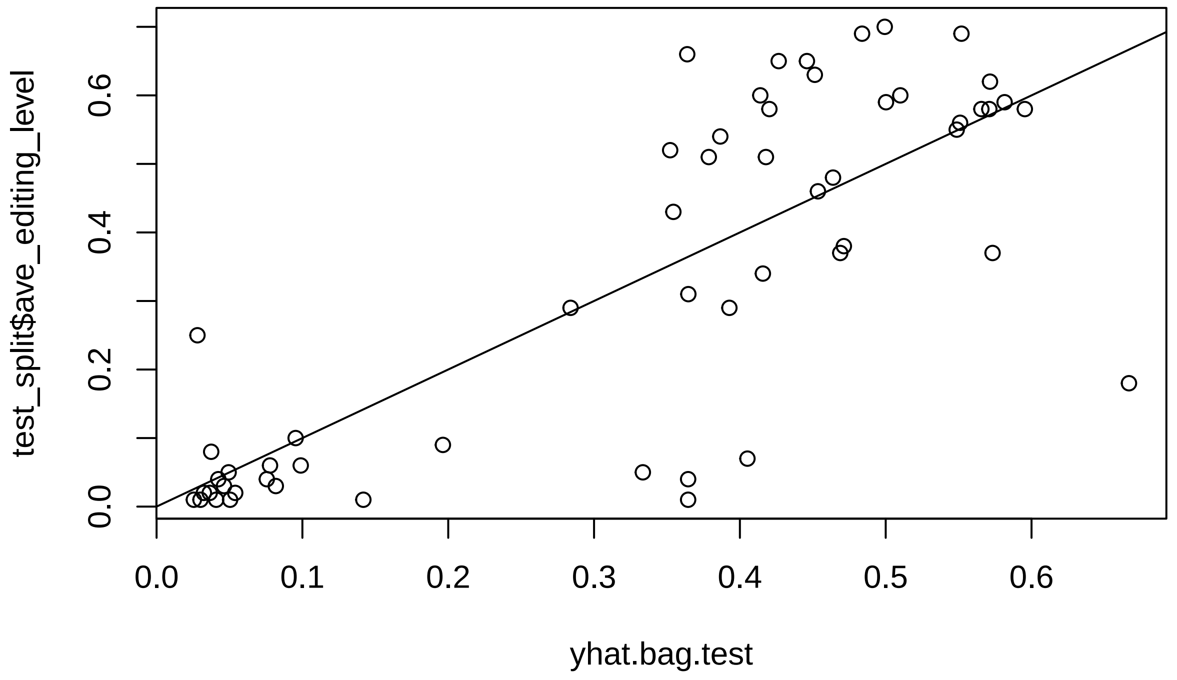


```
print(mean((yhat.bag.train-train_split$ave_editing_level)^2))
```

```
## [1] 0.00472248
```

Predictions on test split

```
plot(yhat.bag.test,test_split$ave_editing_level)
abline(0,1)
```



```
print(mean((yhat.bag.test-test_split$ave_editing_level)^2))
```

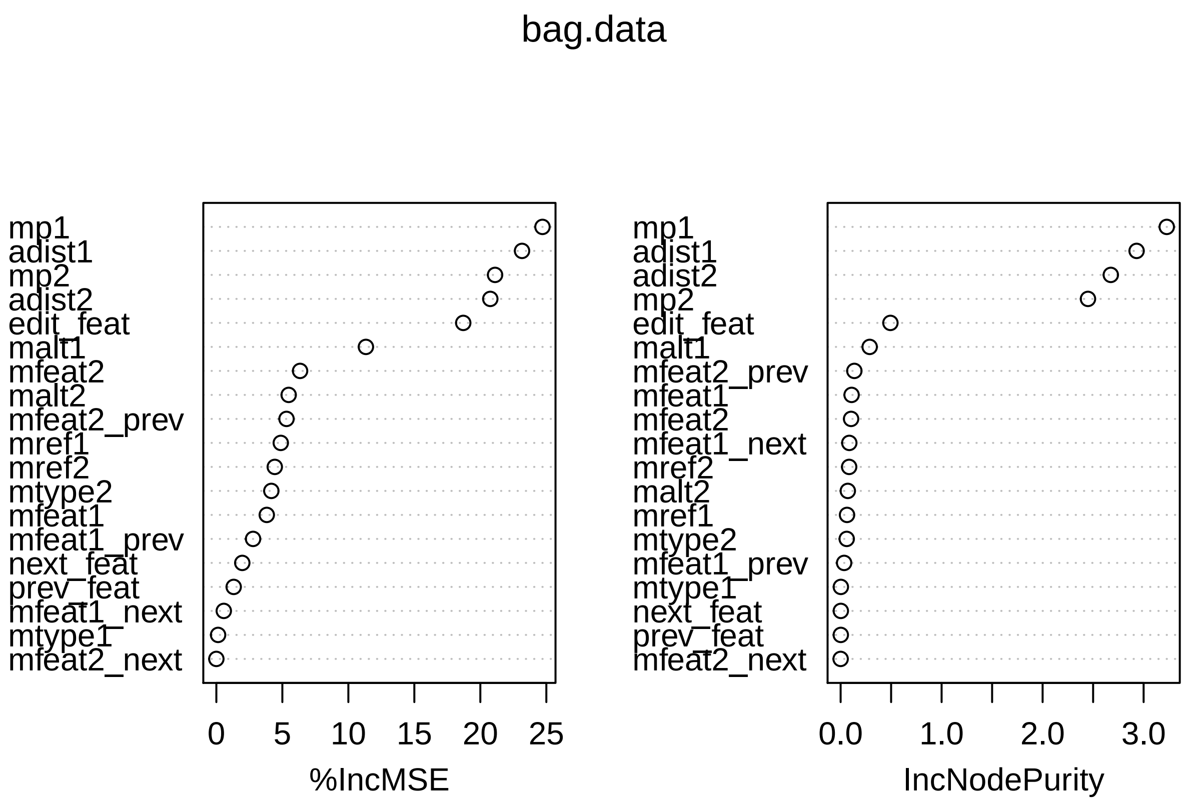
```
## [1] 0.02331793
```

Feature Importance

```
#get the feature importance
importance (bag.data )
```

```
##                %IncMSE IncNodePurity
## edit_feat   18.7039268   0.493035316
## prev_feat    1.3142513   0.001756823
## next_feat    1.9645418   0.002037126
## mp1         24.7061524   3.229132949
## mref1        4.8820341   0.063430900
## malt1       11.3305610   0.287789138
## mtype1       0.1320094   0.002121355
## mfeat1       3.8201350   0.108592032
## mfeat1_prev  2.7825562   0.034755064
## mfeat1_next  0.5648393   0.085860348
## adist1      23.1578734   2.930658413
## mp2         21.1147003   2.448996284
## mref2        4.4266836   0.084321687
## malt2        5.4808279   0.071266847
## mtype2       4.1661620   0.060279124
## mfeat2       6.3435142   0.103652048
## mfeat2_prev  5.3159174   0.135840224
## mfeat2_next  0.0000000   0.000000000
## adist2      20.7532765   2.675104494
```

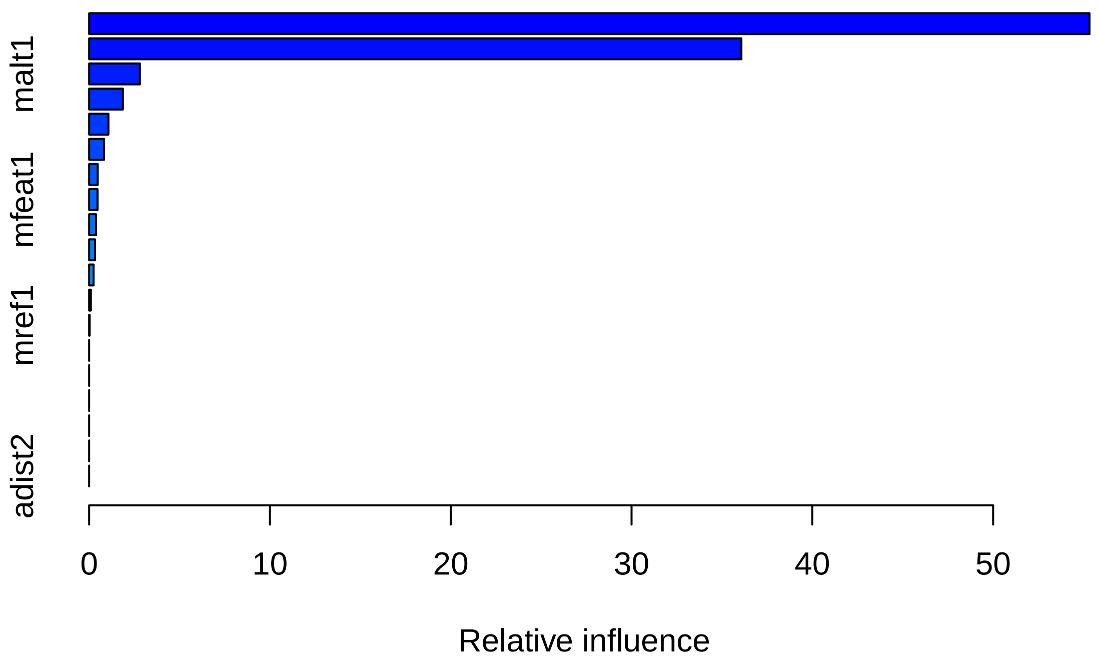
```
varImpPlot(bag.data)
```



The features of highest importance are distance from editing site, position of the first and second mutations.

Boosting

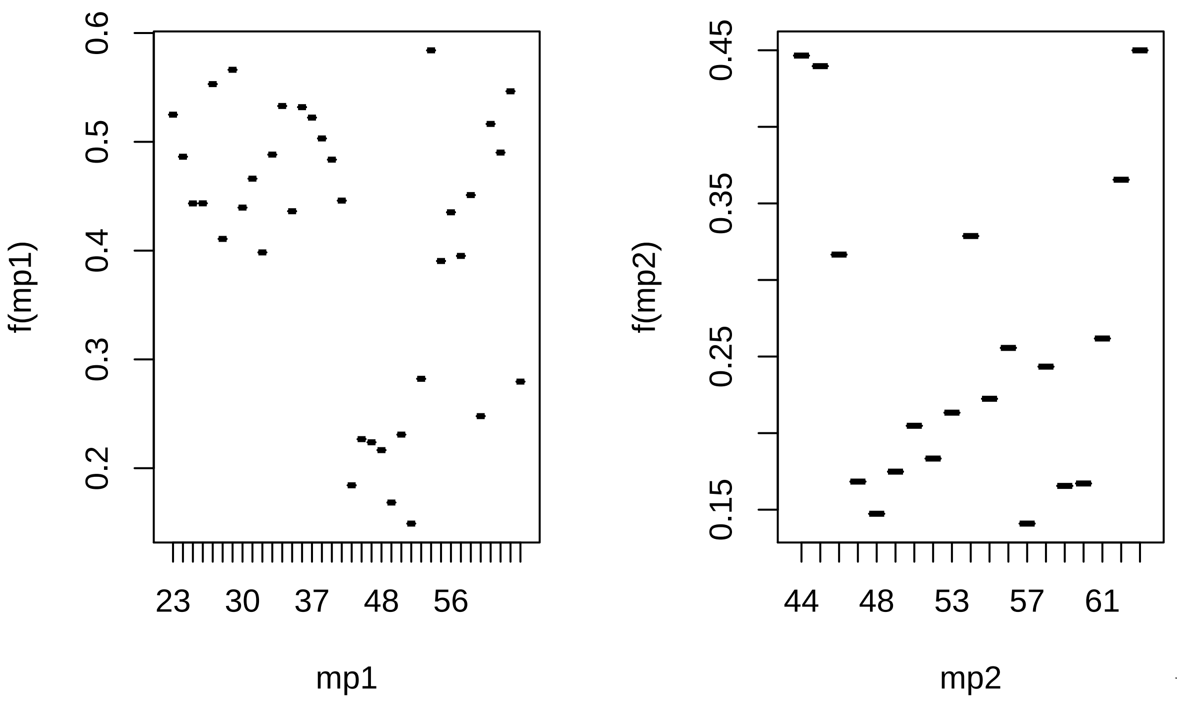
```
boost.data=gbm(ave_editing_level~.,data=train_split,distribution="gaussian",n.trees=5000,interaction.de
summary(boost.data)
```



```
##                      var     rel.inf
## mp1                  mp1 55.33374191
## mp2                  mp2 36.07809806
## malt1              malt1  2.80729944
## edit_feat      edit_feat  1.86652344
## mfeat2_prev mfeat2_prev  1.06718314
## mfeat1_next mfeat1_next  0.82839213
## mfeat2            mfeat2  0.47408074
## mfeat1            mfeat1  0.46410669
## mtype2            mtype2  0.38417724
## mref2              mref2  0.32889475
## malt2              malt2  0.24439359
## mfeat1_prev mfeat1_prev  0.09503127
## mref1              mref1  0.02807759
## prev_feat      prev_feat  0.00000000
## next_feat      next_feat  0.00000000
## mtype1            mtype1  0.00000000
## adist1            adist1  0.00000000
## mfeat2_next mfeat2_next  0.00000000
## adist2            adist2  0.00000000
```

The "mp1" and "mp2" features are the most important variables (these are the positions along the sequence of mutation 1 and 2). We produce partial dependence plots for these two variables. These plots illustrate the marginal effect of the mp1 and mp2 variables after integrating out the other variables.

```r
par(mfrow=c(1,2))
plot(boost.data,i="mp1")
plot(boost.data,i="mp2")
```



We use the boosted model to predict on the test data:

```r
yhat.boost=predict(boost.data,newdata=test_split,n.trees=5000)
mean((yhat.boost-test_split$ave_editing_level)^2)
```

```
## [1] 0.02125189
```

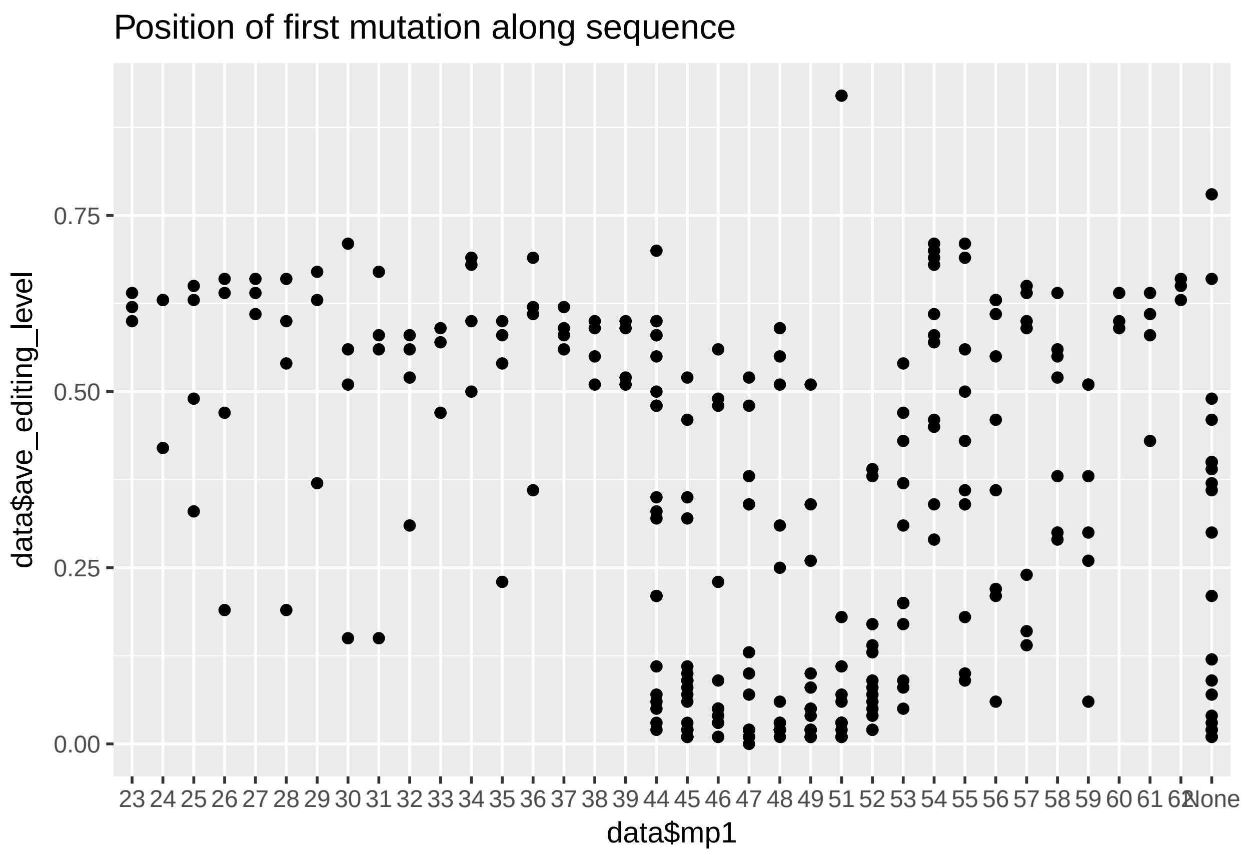
Experiment with the boosting shrinkage parameter (increase to 0.2 from default of 0.001)

```
for(shrinkage_val in c(0.005, 0.01, 0.05, 0.1, 0.2))
{
print(shrinkage_val)
boost.data=gbm(ave_editing_level~.,
               data=train_split,
               distribution="gaussian",
               n.trees=5000,
               interaction.depth=4,
               shrinkage=shrinkage_val,verbose=F)
yhat.boost=predict(boost.data,newdata=test_split,n.trees=5000)
print(mean((yhat.boost-test_split$ave_editing_level)^2))
}
```

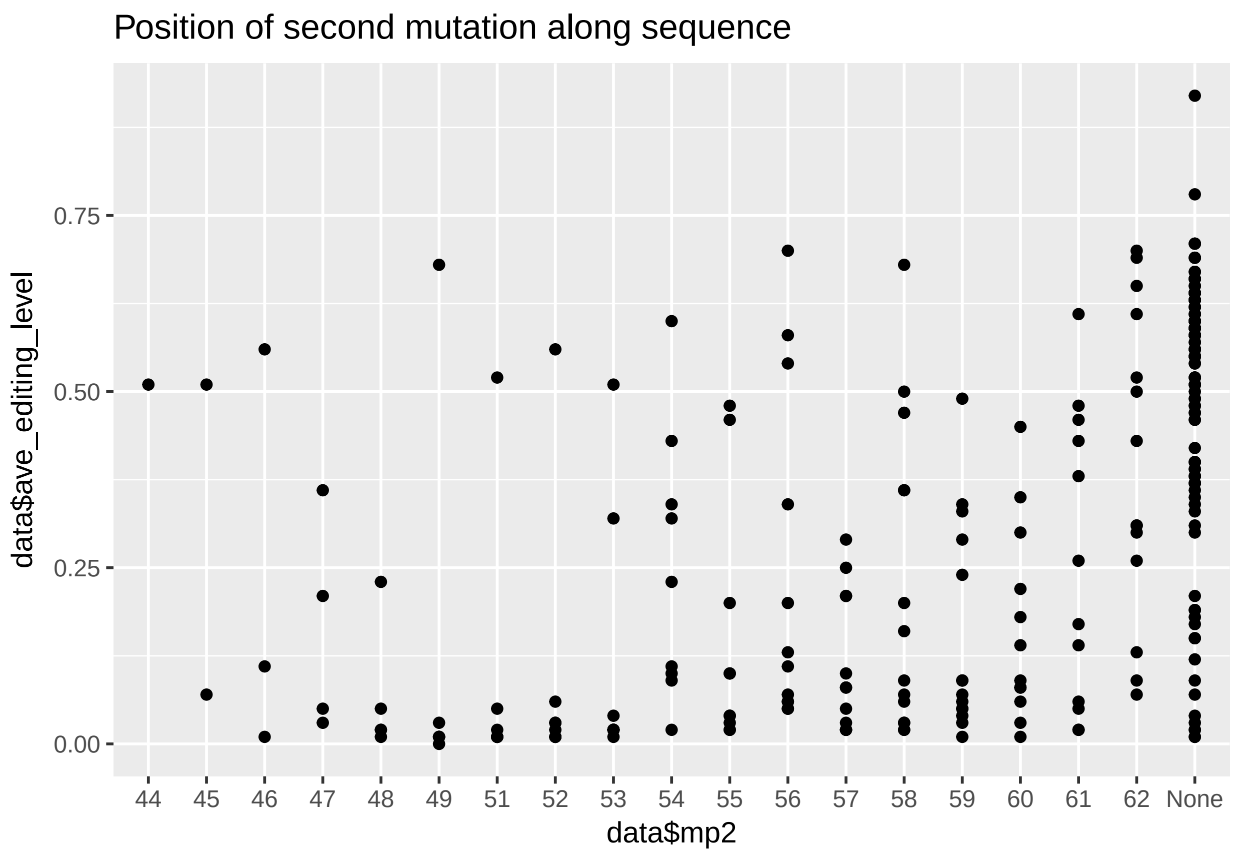
```
## [1] 0.005
## [1] 0.02435091
## [1] 0.01
## [1] 0.02757491
## [1] 0.05
## [1] 0.0335663
## [1] 0.1
## [1] 0.03733136
## [1] 0.2
## [1] 0.04807122
```

Feature values vs editing levels

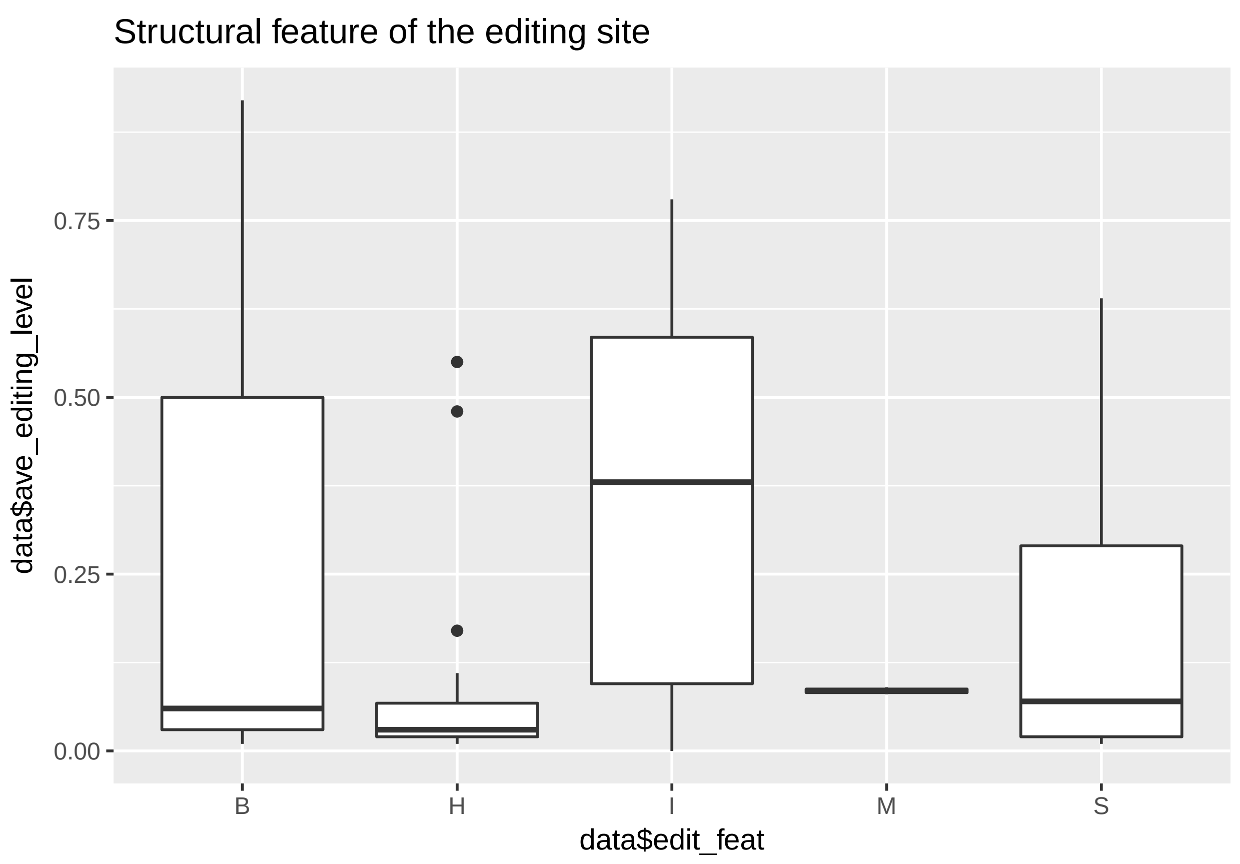
```
p1=ggplot(data=data,aes(x=data$mp1,y=data$ave_editing_level))+
  geom_point()+
  ggtitle("Position of first mutation along sequence")
p1
```



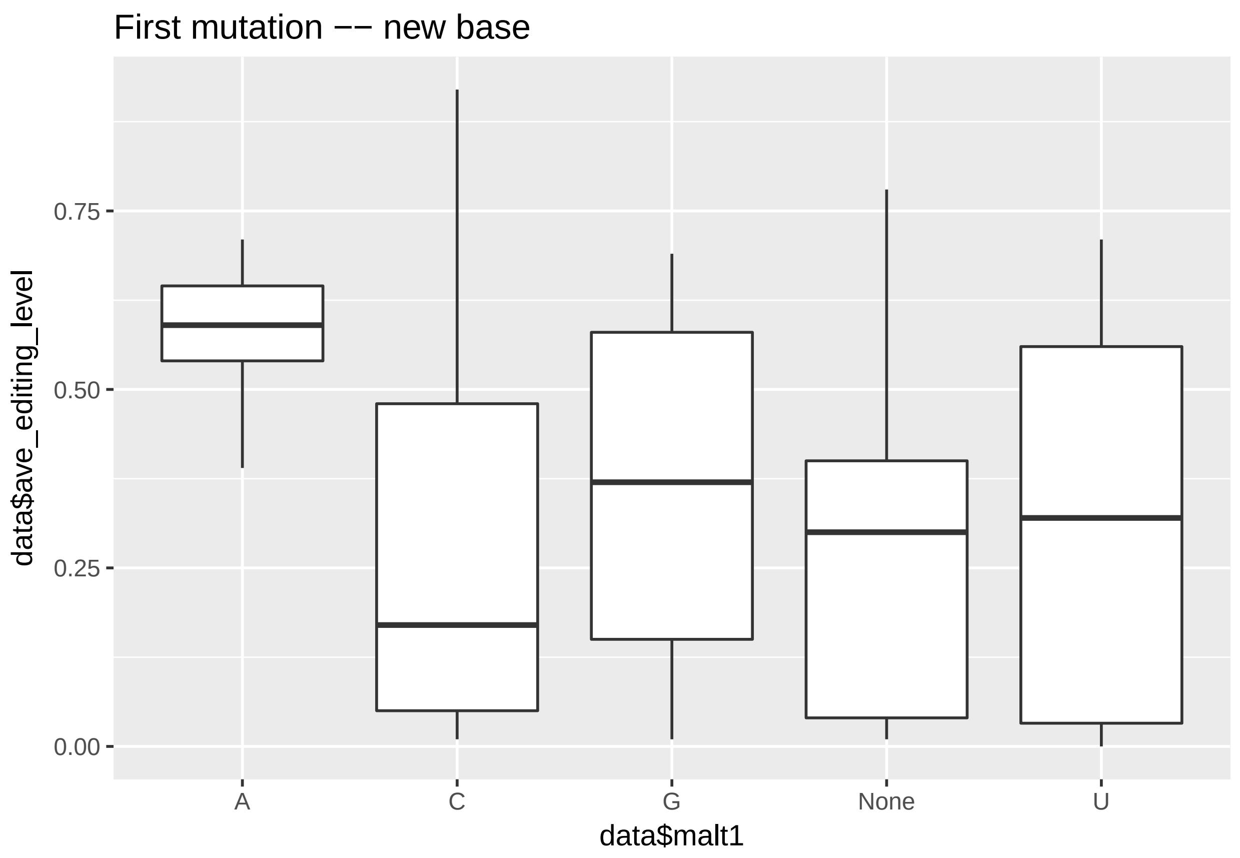
```r
p2=ggplot(data=data,aes(x=data$mp2,y=data$ave_editing_level))+
  geom_point()+
  ggtitle("Position of second mutation along sequence")
p2
```



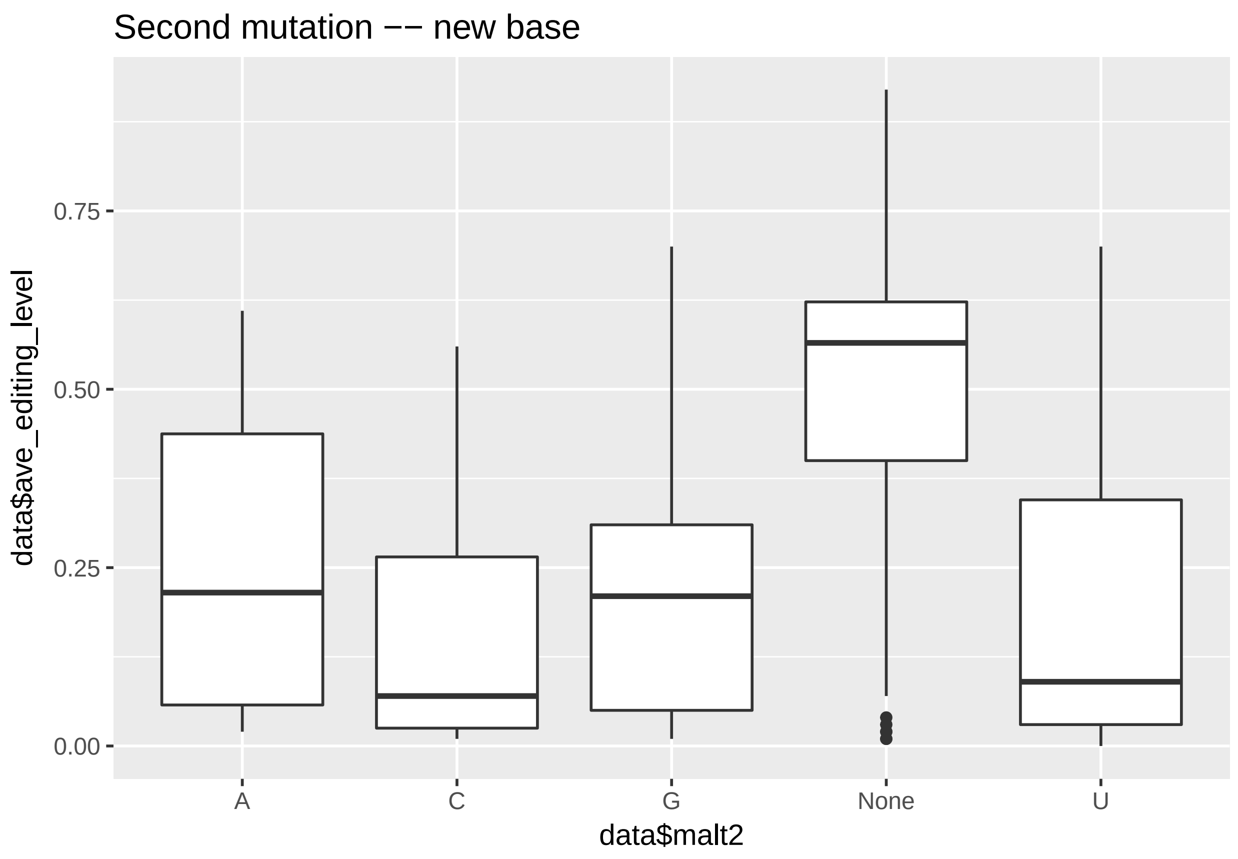
```
p3=ggplot(data=data,aes(x=data$edit_feat,y=data$ave_editing_level))+
  geom_boxplot()+
  ggtitle("Structural feature of the editing site")
p3
```



```
p4=ggplot(data=data,aes(x=data$malt1,y=data$ave_editing_level))+
  geom_boxplot() +
  ggtitle("First mutation -- new base ")
p4
```



```
p5=ggplot(data=data,aes(x=data$malt2,y=data$ave_editing_level))+
  geom_boxplot() +
  ggtitle("Second mutation -- new base")
p5
```



Second mutation −− new base
data$ave_editing_level
0.00
0.25
0.50
0.75
A
C
G
None
U
data$malt2