A. Regression task – the development data and its distribution

Load the development data set D

> library(data.table)
> devel.D = fread("plr.dataset.D.development.csv", header=T)
> str(devel.D)

Find the number of different proteins in the data set

Analyze the distribution of ligand_distance

> summary(devel.D$ligand_distance)

Make a plot of empirical density

Hints

- use `cut(devel.D$ligand_distance)`
- set the intervals into which `ligand_distance` is to be cut by parameter `breaks`
B. Learning a Random Forest regression model

1. Prepare training and test subsets

```r
> proteins.devel = unique(devel.D$protein_id)
> length(proteins.devel)

> set.seed(1); test.index = sample(20, 3, rep=F)
> proteins.test = proteins.devel[test.index]
> proteins.test

> proteins.train = proteins.devel[-test.index]
> proteins.train

> data.train = as.data.frame(devel.D[protein_id %in% proteins.train])
> nrow(data.train)
> length(unique(data.train$protein_id))

# remove protein_id
> data.train = data.train[, -1]
> str(data.train)

> data.test = as.data.frame(devel.D[protein_id %in% proteins.test])
> nrow(data.test)
> length(unique(data.test$protein_id))
> data.test = data.test[, -1]
```

2. Fit a Random Forest model

```r
library(randomForest)

RF.ntree = 200                          # number of trees in the ensemble
RF.mtry  = 10                           # the default for regression is p/3

model.RF =
    randomForest(ligand_distance ~ ., data.train, ntree=RF.ntree, mtry=RF.mtry)

Try different parameter values.

Example of the output
> model.RF

Call:
    randomForest(formula = ligand_distance ~ ., data = data.train,
      ntree = RF.ntree, mtry = RF.mtry)

Type of random forest: regression
Number of trees: 100
No. of variables tried at each split: 10

Mean of squared residuals: 1.20056
    % Var explained: 42.26
```
3. Evaluate your model

Use `plot(model)` to show the out-of-bag estimate of MSE error

Now compare the plotted error with the training error and test error!
- Why are the values different?

```r
# prediction on the test set
> prediction.RF.test = predict(model.RF, data.test, type="response")
> MSE.test = mean( (data.test$ligand_distance - prediction.RF.test)^2 )

... and similarly using the training data set
```

Now do the cross-validation test and observe the variance of the test error. Then repeat whole cross-validation procedure several times (e.g. 10 times) with different random splits. You should always keep whole test proteins separated from the training ones!

Also, look at another plot – “pseudo R-squared”: $1 - \text{MSE/Var(y)}$. Available for regression only.

```r
> plot(model.RF$rsq)
```
4. Explore the variable importance

```r
> model.RF$importance
> varImpPlot(model.RF)
```