Cheat sheet for MLR

Libraries that are useful this task

```
library(caret) #machine learinign workflow
library(leaps) #library for stepwise regression
library(MASS) #contains some important basic linear regression tools
library(caTools)
library(tidyverse) #helps us to write concise code
```

Importing the data

```
dataset <-  read_delim('Cao2015_SI.csv',
                         delim = ",", //check your system delimiter
                         col_names = TRUE,
                         trim_ws = TRUE)
```

Data purification

Are all parameters required (e.g. name, smiles)? Use

```
dplyr::select( … , … , … )
```

to select the parameters that are going to the model development or

```
dplyr::select(-c( … , … , …))
```

to remove variables that are not relevant.

Data pre-processing

```
#observe the correlation between variables

#NB! Remove predictable variable from the dataset before or rename

correlationMatrix <- cor(dataset)

# find attributes that are highly corrected (ideally >0.75)

highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.75)

dataset2 <- dataset2 %>%

  dplyr::select(-highlyCorrelated)
```

Building stepwise regression automatically

```
# Fit the full model
full.model <- lm(RT ~., data = training_set)
# Stepwise regression model
step.model <- stepAIC(full.model, direction = "both",
                      trace = FALSE)
summary(…)
```

Preparing the training and test set with 80/20 ratio

```
set.seed(123)
```

Lets create the split parameter with values:

```
split <- sample.split(dataset$..., SplitRatio = …)
```

And now split the data to two sets: training and test and remove the split from the dataset at the same time

```
training_set <- subset(dataset, split == TRUE)
```

```
test_set <- subset(dataset, split == FALSE)
```

Fitting Multiple Linear Regression to the Training set

```
regressor = lm(formula = RT ~ .,
               data = …)
#assessing the model
```

```
summary(regressor)
```

Use the regression to predict the retention times

```
training_set <- training_set %>%
  mutate( … = predict(regressor, newdata = … ))
```

Visualising the fit

```
#lets have a look how did the prediction work out for training set
```

```
ggplot(data = … ) +
  geom_point(mapping = aes(x = … , y = … ))
```

Training with automated cross-validation from package *caret*

```
fitControl <- trainControl(
  method = "repeatedcv",
  number = …, //number of resampling iterations
  repeats = …) //the number of complete sets of folds to compute

regressor <- train(RT ~ ., data = dataset,
                 method = "glmStepAIC",
                 trControl = fitControl,
                 verbose = FALSE)
```